aws

ユーザーガイド

EventBridge スケジューラ

EventBridge スケジューラ: ユーザーガイド

Table of Contents

Amazon EventBridge Scheduler とは

Amazon EventBridge Scheduler はサーバーレススケジューラで、1 つの中央マネージドサービスからタスクを作成、実行、管理できます。スケーラビリティの高いス EventBridge ケジューラを使用すると、270 を超える AWS サービスと 6,000 を超えるAPIオペレーションを呼び出すことができる数百万のタスクをスケジュールできます。インフラストラクチャをプロビジョニングして管理したり、複数の サービスと統合したりすることなく、 EventBridge Scheduler はスケジュールを大規模に配信し、メンテナンスコストを削減する機能を提供します。

EventBridge スケジューラは、ダウンストリームターゲットの可用性に基づいてスケジュールを調整する組み込みメカニズムを使用して、タスクを確実に配信します。ス EventBridge ケジューラを使用すると、繰り返しパターンの cron 式と rate 式を使用してスケジュールを作成したり、1 回限りの呼び出しを設定したりできます。配信の時間枠を柔軟に設定したり、再試行制限を定義したり、失敗したトリガーの最大保持時間を設定したりできます。

トピック

ス EventBridge ケジューラの主な機能

EventBridge スケジューラには、ターゲットの設定やスケジュールのスケーリングに使用できる以下の主要な機能があります。

- テンプレート化されたターゲット – ス EventBridge ケジューラは、Amazon 、Amazon SQS、LambdaSNS、および を使用して一般的なAPIオペレーションを実行するためのテンプレート化されたターゲットをサポートします EventBridge。事前定義されたターゲットを使用すると、ス EventBridge ケジューラコンソール、ス EventBridge ケジューラ 、または を使用してSDKスケジュールをすばやく設定できます AWS CLI。
- ユニバーサルターゲット – ス EventBridge ケジューラは、270 を超える AWS サービスとスケジュールに基づく 6,000 を超えるAPIオペレーションをターゲットとするカスタマイズされたトリガーを作成するために使用できるユニバーサルターゲットパラメータ (UTP) を提供します。ではUTP、ス EventBridge ケジューラコンソール、ス EventBridge ケジューラ 、または を使用してSDK、カスタマイズされたトリガーを設定できます AWS CLI。

- 柔軟な時間枠 – ス EventBridge ケジューラは柔軟な時間枠をサポートしているため、スケジュールを分散し、ターゲットの正確なスケジュールされた呼び出しを必要としないユースケースのトリガーの信頼性を向上させることができます。
- 再試行 — ス EventBridge ケジューラはターゲットに at-least-once イベント配信を提供します。つまり、ターゲットからの応答で少なくとも 1 つの配信が成功します。ス EventBridge ケジューラを使用すると、失敗したタスクのスケジュールの再試行回数を設定できます。 EventBridge スケジューラは、失敗したタスクを遅延して再試行し、スケジュールの信頼性を向上させ、ターゲットが使用可能であることを確認します。

スケ EventBridge ジューラへのアクセス

ス EventBridge ケジューラは、 EventBridge コンソール、スケ EventBridge ジューラ SDK、 AWS CLI、またはス EventBridge ケジューラ を直接使用して使用できますAPI。

Amazon EventBridge Scheduler のセットアップ

ス EventBridge ケジューラを使用する前に、次のステップを完了する必要があります。

トピック

にサインアップする AWS

がない場合は AWS アカウント、次のステップを実行して作成します。

にサインアップするには AWS アカウント

1. https://portal.aws.amazon.com/billing/サインアップ を開きます。
2. オンラインの手順に従います。

   サインアップ手順の一環として、通話呼び出しを受け取り、電話キーパッドで検証コードを入力するように求められます。

   にサインアップすると AWS アカウント、 AWS アカウントのルートユーザーが作成されます。ルートユーザーには、アカウントのすべての AWS サービス とリソースへのアクセス権があります。セキュリティのベストプラクティスとして、ユーザーに管理アクセスを割り当て、ルートユーザーのみを使用してルートユーザーアクセスが必要なタスクを実行してください。

IAM ユーザーを作成する

管理者ユーザーを作成するには、以下のいずれかのオプションを選択します。

管理者を管理する方法を 1 つ選択します	目的	方法	以下の操作も可能
IAM Identity Center 内 (推奨)	短期の認証情報を使用して AWSにアクセスします。 これはセキュリティのベストプラクティスと一致しています。ベストプラクティスの詳細については、「 ユーザーガイド」の「 のセキュリティのベストプラクティス IAMIAM」を参照してください。	AWS IAM Identity Center ユーザーガイドの「開始方法」の手順に従います。	ユーザーガイド の を使用する AWS CLI ように を設定 AWS IAM Identity Centerして、プログラムによるアクセスを設定します。 AWS Command Line Interface
で IAM (非推奨)	長期認証情報を使用して AWSにアクセスする。	「 ユーザーガイドIAM」の「最初の管理者ユーザーとユーザーグループの作成IAM」の手順に従います。	「 ユーザーガイド」の IAM「ユーザーのアクセスキーを管理する」でプログラムによるアクセスを設定します。 IAM

マネージドポリシーを使用する

前のステップでは、 AWS リソースにアクセスするための認証情報を持つ IAM ユーザーを設定します。ほとんどの場合、ス EventBridge ケジューラを安全に使用するには、ス EventBridge ケジューラを使用するために必要なアクセス許可のみを持つ個別のユーザー、グループ、またはロールを作成することをお勧めします。 EventBridge スケジューラは、一般的なユースケースで次の マネージドポリシーをサポートしています。

- the section called “AmazonEventBridgeSchedulerFullAccess” – コンソールと を使用してス EventBridge ケジューラへのフルアクセスを許可しますAPI。
- the section called “AmazonEventBridgeSchedulerReadOnlyAccess” – ス EventBridge ケジューラへの読み取り専用アクセスを許可します。

これらの管理ポリシーは、前のステップでAdministratorAccessポリシーをアタッチしたのと同じ方法でプリンIAMシパルにアタッチできます。ID ベースのIAMポリシーを使用してス EventBridge ケジューラへのアクセスを管理する方法の詳細については、「」を参照してくださいthe section called “アイデンティティベースのポリシーを使用する”。

実行ロールを設定する

実行ロールは、ユーザーに代わって他の とやり取りするためにス EventBridge ケジューラが引き受ける IAMロール AWS サービス です。このロールにアクセス許可ポリシーをアタッチして、ターゲットを呼び出すアクセス許可をス EventBridge ケジューラに付与します。

コンソールを使用して新しいスケジュールを作成するときに、新しい実行ロールを作成することもできます。コンソールを使用する場合、ス EventBridge ケジューラは選択したターゲットに基づいてアクセス許可を持つロールをユーザーに代わって作成します。ス EventBridge ケジューラがユーザーに代わってロールを作成する場合、ロールの信頼ポリシーには、ユーザーに代わってロールを引き受けることができるプリンシパルを制限する条件キーが含まれます。これにより、混乱した代理のセキュリティ問題を防ぐことができます。

次の手順では、新しい実行ロールを作成する方法と、ターゲットを呼び出すためのアクセス許可をス EventBridge ケジューラに付与する方法について説明します。このトピックでは、一般的なテンプレート化されたターゲットの権限について説明します。他のターゲットに権限を追加する方法については、「the section called “テンプレート化されたターゲットの使用”」を参照してください。

を使用して実行ロールを作成するには AWS CLI

1. 次の継承ロールJSONポリシーをコピーし、 としてローカルに保存しますScheduler-Execution-Role.json。この信頼ポリシーにより、ス EventBridge ケジューラはユーザーに代わってロールを引き受けることができます。

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

```
            "Effect": "Allow",
            "Principal": {
                "Service": "scheduler.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

> **Important**
>
> 実稼働環境で実行ロールを設定する場合は、混乱した代理の問題を防ぐための追加の保護手段を導入することをお勧めします。詳細およびポリシーの例については、「the section called “混乱した代理の防止”」を参照してください。

2. AWS Command Line Interface（AWS CLI) から、次のコマンドを入力して新しいロールを作成します。*SchedulerExecutionRole* をこのロールに割り当てる名前に置き換えます。

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

成功すると、 次の出力が表示されます。

```
{
    "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. ス EventBridge ケジューラがターゲットを呼び出すことを許可する新しいポリシーを作成するには、次のいずれかの一般的なターゲットを選択します。アクセスJSON許可ポリシーをコピーし、ローカルに.jsonファイルとして保存します。

   Amazon SQS – SendMessage

   以下により、ス EventBridge ケジューラはアカウント内のすべての Amazon SQSキューで sqs:SendMessageアクションを呼び出すことができます。

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

   Amazon SNS – Publish

   以下では、ス EventBridge ケジューラがアカウント内のすべての Amazon SNSトピックで sns:Publishアクションを呼び出すことができます。

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

```
}
```

Lambda – Invoke

以下では、ス EventBridge ケジューラがアカウント内のすべての Lambda 関数に対して `lambda:InvokeFunction`アクションを呼び出すことを許可します。

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

4. 次のコマンドを実行して、新しいアクセス許可ポリシーを作成します。*PolicyName* をこのポリシーに割り当てる名前に置き換えます。

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

成功すると、次の出力が表示されます。ポリシー を書き留めますARN。次のステップARNでこれを使用して、ポリシーを実行ロールにアタッチします。

```
{
    "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. 次のコマンドを実行して、ポリシーを実行ロールにアタッチします。を、前のステップで作成したポリシーARNの *your-policy-arn*に置き換えます。*SchedulerExecutionRole* を実行ロールの名前に置き換えます。

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

attach-role-policy オペレーションはコマンドラインにレスポンスを返しません。

ターゲットをセットアップする

ス EventBridge ケジューラスケジュールを作成する前に、スケジュールが呼び出すターゲットが少なくとも 1 つ必要です。既存のリソースを使用するか、新しい AWS リソースを作成できます。次の手順は、 を使用して新しい標準 Amazon SQSキューを作成する方法を示しています AWS CloudFormation。

新しい Amazon SQSキューを作成するには

1. 次のJSON AWS CloudFormation テンプレートをコピーし、 としてローカルに保存します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. から AWS CLI次のコマンドを実行して、Scheduler-Target-SQS.jsonテンプレートから AWS CloudFormation スタックを作成します。

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

成功すると、次の出力が表示されます。

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

3. 次のコマンドを実行して、 AWS CloudFormation スタックの概要情報を表示します。この情報には、スタックの状態とテンプレートで指定されている出力が含まれます。

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

成功すると、コマンドは Amazon SQSキューを作成し、次の出力を返します。

```
{
```

```
    "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"
            }
        }
    ]
}
```

このガイドの後半では、 の値を使用してQueueARN、ス EventBridge ケジューラのターゲットとしてキューを設定します。

次のステップ

セットアップステップが完了したら、開始方法ガイドを使用して最初のス EventBridge ケジューラスケジューラを作成し、ターゲットを呼び出します。

ス EventBridge ケジューラの開始方法

このトピックでは、新しいス EventBridge ケジューラスケジュールの作成について説明します。ス EventBridge ケジューラコンソール、 (AWS CLI)、 AWS Command Line Interface または AWS SDKs を使用して、テンプレート化された Amazon SQSターゲットでスケジュールを作成します。次に、ロギングを設定し、再試行回数を設定し、失敗したタスクの最大保持時間を設定します。スケジュールを作成したら、スケジュールがターゲットを正常に呼び出し、ターゲットキューにメッセージを送信することを確認します。

> **Note**
>
> このガイドに従うには、「」で説明されている最低限必要なアクセス許可を持つIAMユーザーを設定することをお勧めしますthe section called “アイデンティティベースのポリシーを使用する”。ユーザーを作成して設定したら、次のコマンドを実行してアクセス認証情報を設定します。 AWS CLIを設定するには、アクセスキー ID と シークレットアクセスキーが必要です。
>
> ```
> $ 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
> ```
>
> 認証情報を設定するさまざまな方法について詳しくは、「バージョン 2 用AWS Command Line Interface ユーザーガイド」の「構成設定と優先順位」を参照してください。

トピック

前提条件

このセクションの手順を開始する前に、次を実行します。

- 「設定」で説明されている各タスクを実行します。

ス EventBridge ケジューラコンソールを使用してスケジュールを作成する

コンソールを使用して新しいスケジュールを作成するには

1. にサインインし AWS Management Console、次のリンクを選択して EventBridge コンソールのス EventBridge ケジューラセクションを開きます: https://us-west-2.console.aws.amazon.com/scheduler/home?region=us-west-2#home

   > **Note**
   >
   > の AWS Management Consoleリージョンセレクタ AWS リージョン を使用して を切り替えることができます。

2. [スケジュール] ページで、[スケジュールを作成] を選択します。
3. [スケジュールの詳細を指定] ページの [スケジュールの名前と説明] セクションで、次を実行します。

   a. [スケジュール名] で、スケジュールの名前を入力します。例えば、**`MyTestSchedule`**

   b. [説明 - オプション] で、スケジュールの説明を入力します。例えば、**`My first schedule`** と指定します。

   c. [スケジュールグループ] で、ドロップダウンオプションからスケジュールグループを選択します。スケジュールグループをまだ作成していない場合は、スケジュールの `default` グループを選択できます。新しいスケジュールグループを作成するには、コンソールの説明にある [独自のスケジュールを作成] リンクを選択します。スケジュールグループを使用して、スケジュールのグループにタグを追加します。

4. [スケジュールのパターン] セクションで、次の操作を行います。

   a. [頻度] で、以下のいずれかのパターンオプションを選択します。設定オプションは、選択したパターンによって変わります。

- [1 回限りのスケジュール] – 1 回限りのスケジュールは、指定した日時に 1 回だけターゲットを呼び出します。

  [日付と時刻] には、有効な日付を YYYY/MM/DD 形式で入力します。次に、24 時間の hh:mm 形式でタイムスタンプを指定します。最後に、ドロップダウンオプションからタイムゾーンを選択します。

- [繰り返しのスケジュール] – 繰り返しのスケジュールは、cron 式または rate 式を使用して指定したレートでターゲットを呼び出します。

  Cron ベースのスケジュールを選択して、cron式を使用してスケジュールを設定します。rate 式を使用するには、レートベースのスケジュールを選択し、値 に正の数を入力し、ドロップダウンオプションから単位を選択します。

  cron 式と rate 式の詳細については、「スケジュールタイプ」を参照してください。

b. [柔軟な時間枠] で、[オフ] を選択してオプションをオフにするか、ドロップダウンリストから事前定義された時間枠のいずれかを選択します。例えば、[15 分] を選択し、1 時間に 1 回ターゲットを呼び出す繰り返しのスケジュールを設定した場合、スケジュールは毎時の開始後 15 分以内に実行されます。

5. 前のステップで [繰り返しのスケジュール] を選択した場合は、[時間枠] セクションでタイムゾーンを指定し、必要に応じてスケジュールの開始日時と終了日時を設定します。開始日のない繰り返しのスケジュールは、作成されて利用可能になるとすぐに開始されます。終了日のない繰り返しのスケジュールは、そのターゲットを無期限に呼び出し続けます。
6. [Next (次へ)] を選択します。
7. [ターゲットを選択] ページで、次の操作を行います。

   a. テンプレート化されたターゲットを選択し、ターゲット を選択しますAPI。この例では、Amazon SQS **SendMessage** テンプレート化されたターゲットを選択します。

   b. SendMessage セクションのSQSキュー で、ドロップダウンリストから ARNなどの既存の Amazon `arn:aws:sqs:`*`us-west-2`*`:`*`123456789012`*`:`*`TestQueue`* SQSキューを選択します。新しいキューを作成するには、新しいSQSキューの作成を選択して Amazon SQSコンソールに移動します。キューの作成が完了したら、ス EventBridge ケジューラコンソールに戻り、ドロップダウンを更新します。新しいキューARNが表示され、選択することができます。

c. ターゲット には、ス EventBridge ケジューラがターゲットに配信するペイロードを入力します。この例では、次のメッセージをターゲットキューに送信します: **`Hello, it's EventBridge Scheduler.`**

8. [次へ] を選択し、[設定 - オプション] ページで次の操作を行います。

9.

a. [スケジュールの状態] セクションの [スケジュールを有効にする] で、スイッチを使って機能のオンとオフを切り替えます。デフォルトでは、ス EventBridge ケジューラはスケジュールを有効にします。

b. スケジュール完了後のアクションセクションで、スケジュールの完了後にス EventBridge ケジューラが実行するアクションを設定します。

- スケジュールを自動的に削除するDELETE場合は、 を選択します。1 回限りのスケジュールの場合は、スケジュールがターゲットを一度呼び出した後に削除が実行されます。繰り返しのスケジュールの場合は、スケジュールが最後に予定されていた呼び出しの後に削除が実行されます。自動削除の詳細については、「the section called “スケジュール完了後の削除”」を参照してください。
- スケジュールの完了後にス EventBridge ケジューラがアクションを実行しない場合は、を選択するか、値を選択NONEしないでください。

c. 「再試行ポリシーとデッドレターキュー (DLQ) 」セクションの「再試行ポリシー 」で、「再試行」をオンにして、スケジュールの再試行ポリシーを設定します。再試行ポリシーでは、スケジュールがターゲットの呼び出しに失敗した場合、ス EventBridge ケジューラはスケジュールを再実行します。設定されている場合は、スケジュールの最大保持時間と再試行を設定する必要があります。

d. イベントの最大経過時間 - オプションで、ス EventBridge ケジューラが未処理のイベントを保持する必要がある最大時間 (s) と最小 (s) を入力します。

> **Note**
>
> 最大値は 24 時間です。

e. 最大再試行回数 には、ターゲットがエラーを返した場合にス EventBridge ケジューラがスケジュールを再試行する最大回数を入力します。

> **Note**
>
> 再試行の最大値は 185 です。

f. デッドレターキュー (DLQ) では、次のオプションから選択します。

- なし — を設定しない場合は、このオプションを選択しますDLQ。
- アカウントの AWS Amazon SQSキューを としてDLQ選択する – このオプションを選択し、ARNドロップダウンリストからキューを選択し、スケジュールを作成するキューDLQ AWS アカウント と同じキューを設定します。
- 他の AWS アカウントの Amazon SQSキューを としてDLQ指定します。このオプションを選択し、キューが別の にある場合はDLQ、 として設定されたキューARNの を入力します AWS アカウント。このオプションを使用するには、キューARNの正確な を入力する必要があります。

g. 暗号化セクションで、暗号化設定をカスタマイズ (アドバンスト) を選択して、カスタマーマネージドKMSキーを使用してターゲット入力を暗号化します。このオプションを選択した場合は、既存のKMSキーを入力するARNか、 AWS KMSキーの作成を選択して AWS KMS コンソールに移動します。ス EventBridge ケジューラが保管中のデータを暗号化する方法の詳細については、「」を参照してくださいthe section called “保管中の暗号化”。

h. [アクセス許可] で [既存のロールを使用] を選択し、セットアップ手順中に作成したロールをドロップダウンリストから選択します。IAM コンソールに移動を選択して新しいロールを作成することもできます。

ス EventBridge ケジューラで新しい実行ロールを作成する場合は、代わりにこのスケジュールの新しいロールを作成するを選択します。その後、[ロール名] で名前を入力します。このオプションを選択すると、ス EventBridge ケジューラはテンプレート化されたターゲットに必要なアクセス許可をロールに追加します。

10. [Next (次へ)] を選択します。
11. [スケジュールの確認と作成] ページで、スケジュールの詳細を確認します。各セクションで、そのステップに戻って詳細を編集するには、[編集] を選択します。
12. [スケジュールを作成] を選択して、新しいスケジュールの作成を完了します。[スケジュール] ページで、新規および既存のスケジュールのリストを表示できます。[ステータス] 列で、新しいスケジュールが [有効] になっていることを確認します。
13. スケジュールが Amazon SQSターゲットを呼び出すことを確認するには、Amazon SQSコンソールを開き、以下を実行します。

a. [キュー] リストからターゲットキューを選択します。

b. [メッセージの送信と受信] を選択します。

c. [メッセージの送信と受信] ページの [メッセージの受信] で [メッセージのポーリング] を選択し、スケジュールによってターゲットキューに送信されたテストメッセージを取得します。

を使用してスケジュールを作成する AWS CLI

次の例は、 AWS CLI コマンドを使用して`create-schedule`、テンプレート化された Amazon SQS ターゲットでス EventBridge ケジューラスケジュールを作成する方法を示しています。次のパラメータのプレースホルダ値を自分自身の情報へと置き換えます。

- --name – スケジュールの名前を入力します。
- RoleArn – スケジュールに関連付ける実行ロールARNの を入力します。
- Arn – ターゲットARNの を入力します。この場合、ターゲットは Amazon SQSキューです。
- 入力 – ス EventBridge ケジューラがターゲットキューに配信するメッセージを入力します。

```
$ 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"}'
```

ス EventBridge ケジューラを使用してスケジュールを作成する SDKs

次の例では、ス EventBridge ケジューラを使用してSDKs、テンプレート化された Amazon SQSターゲットでス EventBridge ケジューラスケジュールを作成します。

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");
    }
}
```

次のステップ

- コンソール、またはス EventBridge ケジューラ を使用してスケジュールを管理する方法の詳細については AWS CLI、SDK「」を参照してくださいスケジュールの管理。
- テンプレート化されたターゲットの設定方法とユニバーサルターゲットパラメータの使用方法の詳細については、「ターゲットの管理」を参照してください。
- ス EventBridge ケジューラのデータ型とAPIオペレーションの詳細については、スケEventBridge ジューラAPIリファレンス を参照してください。

スケ EventBridge ジューラのスケジュールタイプ

次のトピックでは、Amazon EventBridge Scheduler がサポートするさまざまなスケジュールタイプと、ス EventBridge ケジューラが夏時間をどのように処理するか、およびさまざまなタイムゾーンでスケジュールする方法について説明します。スケジュールを設定する際には、レートベース、cron ベース、および 1 回限りのスケジュールの 3 つのスケジュールタイプから選択できます。

レートベースのスケジュールと cron ベースのスケジュールはどちらも繰り返しのスケジュールです。各定期的なスケジュールタイプは、設定するスケジュールタイプのスケジュール式を使用して設定し、ス EventBridge ケジューラが式を評価するタイムゾーンを指定します。

1 回限りのスケジュールは、ターゲットを 1 回だけ呼び出すスケジュールです。1 回限りのスケジュールを設定するには、ス EventBridge ケジューラがスケジュールを評価する時刻、日付、タイムゾーンを指定します。

> **Note**
>
> ス EventBridge ケジューラのすべてのスケジュールタイプは、60 秒の精度でターゲットを呼び出します。つまり、スケジュールを で実行するように設定した場合`1:00`、柔軟な時間枠が設定されていないと仮定して`1:00:59`、 `1:00:00` と APIの間でターゲットが呼び出されます。

以下のセクションでは、定期的なスケジュールタイプごとにスケジュール式を設定する方法と、ス EventBridge ケジューラで 1 回限りのスケジュールを設定する方法について説明します。

トピック

レートベースのスケジュール

レートベースのスケジュールは、スケジュールに指定した開始日の後に開始され、スケジュールの終了日までユーザーが定義した標準レートで実行されます。一般的な繰り返しのスケジュールのユースケースのほとんどは、レートベースのスケジュールを使用して設定できます。例えば、15 分ごとに 1 回、2 時間に 1 回、または 5 日に 1 回ターゲットを呼び出すスケジュールが必要な場合は、レートベースのスケジュールを使用してこれを実現できます。レートベースのスケジュールは、rate 式を使用して設定します。

レートベースのスケジュールでは、`StartDate` プロパティを使用してスケジュールが最初に出現する日を設定します。レートベースのスケジュールに `StartDate` を指定しない場合、スケジュールはただちにターゲットを呼び出します。

rate 式には次のように 2 つの必須フィールドがあり、空白で区切られます。

構文

```
rate(value unit)
```

value

正数。

単位

スケジュールでターゲットを呼び出したい時間単位。

有効な入力: `minutes` | `hours` | `days`

例

次の例は、 コマンドで AWS CLI `create-schedule` rate 式を使用してレートベースのスケジュールを設定する方法を示しています。この例では、5 分ごとに実行されるスケジュールを作成し、テンプレート化された`SqsParameters`ターゲットタイプを使用して Amazon SQSキューにメッセージを配信します。

この例では `--start-date` パラメータの値を設定していないため、スケジュールを作成してアクティブ化した直後に、ターゲットの呼び出しが開始されます。

```
$ 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 ベースのスケジュール

cron 式は、選択した特定の時間に実行されるきめ細かな定期的なスケジュールを作成します。ス EventBridge ケジューラは、世界協定時 (UTC) またはスケジュールの作成時に指定したタイムゾーンでの cron ベースのスケジュールの設定をサポートします。cron ベースのスケジュールでは、スケジュールを実行するタイミングと頻度をより細かく制御できます。ス EventBridge ケジューラのレート式のいずれかではサポートされていないカスタマイズされた繰り返しスケジュールが必要な場合は、cron ベースのスケジュールを使用します。例えば、毎月第 1 月曜日の午前 8 時 (PST) に実行する PST 毎月第 1 月曜日の 。cron ベースのスケジュールは、cron 式を使用して設定します。

cron 式は、以下に示すように、分、時間、月 day-of-month、、 day-of-weekおよびオプションの 1 つのフィールドである年という空白で区切られた 5 つの必須フィールドで構成されます。

構文

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

フィールド	値	ワイルドカード
分	0-59	, - * /
時間	0-23	, - * /
Day-of-month	1-31	, - * ? / L W
月	1 ~ 12 または JAN-DEC	, - * /
Day-of-week	1 ~ 7 または SUN-SAT	, - * ? L #
年	1970-2199	, - * /

ワイルドカード

- , (カンマ) のワイルドカードには、追加の値が含まれます。月フィールドでは、 には 1 月 JANFEB、MAR2 月、3 月が含まれます。
- - (ダッシュ) のワイルドカードは、範囲を指定します。日フィールドの「1-15」は、指定した月の 1 日から 15 日を含みます。
- [*] (アスタリスク) のワイルドカードには、フィールドのすべての値が含まれます。[時間] フィールドの * には すべての時間が含まれます。D フィールドay-of-month と D ay-of-week フィールドの両方で * を使用することはできません。一方に使用する場合は、もう一方に [?] を使用する必要があります。
- [/] (スラッシュ) ワイルドカードで増分を指定します。分フィールドで、「1/10」と入力して、その時間の最初の分から始めて、10 分毎を指定できます (11 分、21 分、31 分など)。
- ? (疑問符) ワイルドカードは任意を意味します。D ay-of-month フィールドに 7 と入力し、曜日が許容できる場合は、Day-of-week フィールドに ？ と入力できます。
- D ay-of-month または D ay-of-week フィールドの L ワイルドカードは、月または週の最終日を指定します。
- D ay-of-month フィールドの**W**ワイルドカードは平日を指定します。D ay-of-month フィールドで、月の 3 日目に最も近い曜日**3W**を指定します。
- D ay-of-week フィールドの # ワイルドカードは、1 か月内の指定された曜日の特定のインスタンスを指定します。例えば、3#2 は、月の第 2 火曜日を示します。3 は週の 3 番目の日 (火曜日) を示し、2 は月のそのタイプの 2 番目の日を示します。

> **Note**
>
> 「#」文字を使用する場合は、 day-of-week フィールドで定義できる式は 1 つだけです。例えば、"3#1,6#3" は 2 つの式として解釈されるため、無効です。

例

次の例は、 コマンドで AWS CLI `create-schedule` cron 式を使用して cron ベースのスケジュールを設定する方法を示しています。この例では、2022 年から 2023 年の各月の最終金曜日の午前 10:15 UTC+0 に実行されるスケジュールを作成し、テンプレート化された`SqsParameters`ターゲットタイプを使用して Amazon SQSキューにメッセージを配信します。

```
$ 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"}'
```

1 回限りのスケジュール

1 回限りのスケジュールでは、有効な日付を使用して指定した日付と時刻に 1 回だけターゲットが呼び出され、timestamp. EventBridge Scheduler は、協定世界時 (UTC) 、またはスケジュールの作成時に指定したタイムゾーンでのスケジューリングをサポートします。

> **Note**
>
> 1 回限りのスケジュールは、実行とターゲットの呼び出しが完了した後も、アカウントクォータに対してカウントされます。1 回限りのスケジュールは、実行が完了した後に削除することをおすすめします。

1 回限りのスケジュールは at 式を使用して設定します。at 式は、以下に示すように、スケジュールを呼び出す日時で構成されます。EventBridge スケジューラがスケジュールを呼び出す日時で構成されます。

構文

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

1 回限りのスケジュールを設定すると、スケジューラは EventBridge を無視StartDateし、スケジュールに EndDate を指定します。

例

次の例は、 コマンドで AWS CLI create-schedule 式で を使用して 1 回限りのスケジュールを設定する方法を示しています。この例では、2022 年 11 月 20 日午後 1 時 UTC8 分に 1 回実行されるスケジュールを作成し、テンプレート化されたSqsParametersターゲットタイプを使用してAmazon SQSキューにメッセージを配信します。

```
$ 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"}'
```

ス EventBridge ケジューラのタイムゾーン

EventBridge スケジューラは、指定した任意のタイムゾーンでの cron ベースのスケジュールと 1 回限りのスケジュールの設定をサポートします。ス EventBridge ケジューラは、Internet Assigned Numbers Authority () によって管理されているタイムゾーンデータベースを使用しますIANA。

では AWS CLI、 `--schedule-expression-timezone`パラメータを使用してス EventBridge ケジューラがスケジュールを評価するタイムゾーンを設定できます。例えば、次のコマンドは、毎日午前 8 時 30 分にアメリカ/ニューヨークでテンプレート化された Amazon SQSSendMessageターゲットを呼び出す cron ベースのスケジュールを作成します。

```
$ 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"}'
```

ス EventBridge ケジューラの夏時間

EventBridge スケジューラは、夏時間に合わせてスケジュールを自動的に調整します。春に時間が進むときに、cron 式が存在しない日時に該当すると、スケジュールの呼び出しはスキップされます。秋に時間が戻ったとき、スケジュールは 1 回だけ実行され、その呼び出しは繰り返されません。次の呼び出しは、通常、指定された日時に行われます。

EventBridge スケジューラは、スケジュールの作成時に指定したタイムゾーンに応じてスケジュールを調整します。アメリカ/ニューヨークでスケジュールを設定した場合、そのタイムゾーンの時刻が変更されるとスケジュールが調整されます。一方、アメリカ/ロサンゼルスのスケジュールは、西海岸で時刻が変更されると 3 時間後に調整されます。

`rate(1 days)` など、単位として `days` を使用するレートベースのスケジュールの場合、`days` は時計上の 24 時間を表します。つまり、夏時間によって 1 日が 23 時間に短縮されたり、25 時間に延長されたりしても、 EventBridge スケジューラはスケジュールの最後の呼び出しから 24 時間後にレート式を評価します。

> **Note**
>
> 地域の規則や規制により、一部のタイムゾーンでは夏時間が適用されません。夏時間が適用されないタイムゾーンでスケジュールを作成した場合、ス EventBridge ケジューラはスケジュールを調整しません。夏時間の調整は、協定世界時 () のスケジュールには適用されませんUTC。

例

アメリカ/ロサンゼルスで次の cron 式を使用してスケジュールを作成するシナリオを考えてみましょう: `cron(30 2 * * ? *)` このスケジュールは、指定されたタイムゾーンで毎日午前 2 時 30 分に実行されます。

- Spring-forward – Spring の時刻が午前 1 時 59 分から午前 3 時まで早まると、ス EventBridge ケジューラはその日にスケジュール呼び出しをスキップし、翌日にスケジュールの通常の実行を再開します。
- フォールバック — 時刻が午前 2 時 59 分から午前 2 時までフォールで逆算すると、 EventBridge スケジューラはシフトが発生する前の午前 2 時 30 分にスケジュールを 1 回だけ実行しますが、時刻シフト後の午前 2 時 30 分にスケジュール呼び出しを再度実行しません。

ス EventBridge ケジューラでのスケジュールの管理

スケジュールは、Amazon EventBridge Scheduler を使用して作成、設定、管理する主なリソースです。

すべてのスケジュールには、スケジュールが実行されるタイミングと頻度を決定するスケジュール式があります。ス EventBridge ケジューラは、rate、cron、および 1 回限りのスケジュールの 3 つのタイプのスケジュールをサポートします。さまざまなスケジュールタイプの詳細については、「スケジュールタイプ」を参照してください。

スケジュールを作成するときは、そのスケジュールが呼び出すターゲットを設定します。ターゲットは、スケジュールを実行するたびにス EventBridge ケジューラがユーザーに代わって呼び出すAPIオペレーションです。 EventBridge スケジューラは、2 種類のターゲットをサポートしています。テンプレート化されたターゲットは、サービスのコアグループ全体で共通のAPIオペレーションを呼び出し、ユニバーサルターゲットパラメータ (UTP) は、270 を超えるサービスで 6,000 を超えるオペレーションを呼び出すために使用できます。ターゲットの設定の詳細については、「ターゲットの管理」を参照してください。

再試行ポリシー とデッドレターキュー (DLQ) の 2 つの主要なメカニズムを使用して、ス EventBridge ケジューラがイベントをターゲットに正常に配信できない場合にスケジュールが失敗を処理する方法を設定します。再試行ポリシーは、ス EventBridge ケジューラが失敗したイベントを再試行する必要がある回数と、未処理のイベントを保持する期間を決定します。DLQ は、再試行ポリシーを使い果たした後、失敗したイベントを に配信するために使用する標準の Amazon SQS キュース EventBridge ケジューラです。を使用してDLQ、スケジュールまたはそのダウンストリームターゲットに関する問題のトラブルシューティングを行うことができます。詳細については、「the section called “の設定 DLQ”」を参照してください。

このセクションでは、コンソール、、 AWS CLI およびス EventBridge ケジューラ を使用してス EventBridge ケジューラのスケジュールを管理する例を示しますSDKs。

トピック

- ス EventBridge ケジューラのスケジュール状態の変更
- ス EventBridge ケジューラでの柔軟な時間枠の設定
- ス EventBridge ケジューラでスケジュールのデッドレターキューを設定する
- スケ EventBridge ジューラでスケジュールを削除する
- 次のステップ

ス EventBridge ケジューラのスケジュール状態の変更

ス EventBridge ケジューラのスケジュールには、有効と無効の 2 つの状態があります。次の例では、UpdateSchedule を使用して、5 分ごとに起動して Lambda ターゲットを呼び出すスケジュールを無効にします。

を使用する場合はUpdateSchedule、必要なパラメータをすべて指定する必要があります。スケ EventBridge ジューラは、スケジュールを指定した情報に置き換えます。以前に設定したパラメータを指定しないと、デフォルトの null に設定されます。

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"
}
```

次の例では、Python SDKと UpdateScheduleオペレーションを使用して、テンプレート化されたターゲットSQSを使用して Amazon をターゲットとするスケジュールを無効にします。

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')
```

ス EventBridge ケジューラでの柔軟な時間枠の設定

柔軟な時間枠でスケジュールを設定すると、ス EventBridge ケジューラは設定した時間枠内にターゲットを呼び出します。これは、ターゲットの呼び出しを正確にスケジュールする必要がない場合に便利です。柔軟な時間枠を設定すると、ターゲットの呼び出しが分散されるため、スケジュールの信頼性が向上します。

例えば、1 時間ごとに実行されるスケジュールに 15 分の柔軟な時間枠を設定すると、スケジュールされた時間から 15 分以内にターゲットを呼び出します。次の AWS CLI、 および ス EventBridge ケジューラSDKの例では、 UpdateScheduleを使用して、1 時間に 1 回実行されるスケジュールに 15 分の柔軟な時間枠を設定します。

> **Note**
>
> 柔軟な時間枠を設定するかどうかを指定する必要があります。このオプションを設定しない場合は、OFF を指定します。値を FLEXIBLE に設定した場合は、スケジュールを実行する最大時間枠を指定する必要があります。

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)
```

ス EventBridge ケジューラでスケジュールのデッドレターキューを設定する

Amazon EventBridge スケジューラは、Amazon Simple Queue Service を使用してデッドレターキュー (DLQ) をサポートします。スケジュールがターゲットの呼び出しに失敗すると、ス EventBridge ケジューラは、呼び出しの詳細とターゲットから受信したレスポンスを含むJSONペイロードを、指定した Amazon SQS標準キューに配信します。

次のトピックでは、これをデッドレターイベント JSONと呼びます。デッドレターイベントを使用すると、スケジュールやターゲットに関する問題をトラブルシューティングできます。スケジュールに再試行ポリシーを設定すると、ス EventBridge ケジューラは設定した最大再試行回数を使い果たしたデッドレターイベントを配信します。

以下のトピックでは、スケジュールDLQの として Amazon SQSキューを設定し、ス EventBridge ケジューラが Amazon にメッセージを配信するために必要なアクセス許可を設定しSQS、 からデッドレターイベントを受信する方法について説明しますDLQ。

トピック

- Amazon SQSキューを作成する
- 実行ロールのアクセス許可を設定する
- デッドレターキューを指定する
- デッドレターイベントの取得

Amazon SQSキューを作成する

スケジュールDLQに を設定する前に、標準の Amazon SQSキューを作成する必要があります。Amazon SQSコンソールを使用してキューを作成する手順については、「Amazon Simple Queue Service デベロッパーガイドSQS」の「Amazon キューの作成」を参照してください。

> **Note**
>
> EventBridge スケジューラは、スケジュールの としてのFIFOキューの使用をサポートしていませんDLQ。

次の AWS CLI コマンドを使用して、標準キューを作成します。

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

成功すると、出力に QueueURL が表示されます。

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

キューを作成したら、キュー を書き留めますARN。ス EventBridge ケジューラのスケジュールDLQ に を指定するARN場合は、 が必要です。キューは Amazon SQSコンソールARNで、または get-queue-attributes AWS CLI コマンドを使用して見つけることができます。

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

成功すると、ARN出力にキューが表示されます。

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

次のセクションでは、スケジュール実行ロールに必要なアクセス許可を追加して、ス EventBridge ケジューラがデッドレターイベントを Amazon に配信できるようにしますSQS。

実行ロールのアクセス許可を設定する

スケジューラ EventBridge がデッドレターイベントを Amazon に配信できるようにするにはSQS、スケジュール実行ロールに次のアクセス許可ポリシーが必要です。スケジュール実行ロールに新しいアクセス権限ポリシーをアタッチする方法の詳細については、「実行ロールの設定」を参照してください。

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

> **Note**
>
> スケジューラ EventBridge を使用して Amazon SQSAPIターゲットを呼び出す場合、スケジュール実行ロールに必要なアクセス許可が既にアタッチされている可能性があります。

次のセクションでは、スケジューラ EventBridge コンソールを使用して、スケジュールDLQに を指定します。

デッドレターキューを指定する

を指定するにはDLQ、スケジューラ EventBridge コンソールまたは AWS CLI を使用して既存のスケジュールを更新するか、新しいスケジュールを作成します。

Console

コンソールDLQを使用して を指定するには

1. にサインインし AWS Management Console、次のリンクを選択して EventBridge 、conosle のス EventBridge ケジューラセクションを開きます。https://console.aws.amazon.com/scheduler/ホーム
2. ス EventBridge ケジューラコンソールで、新しいスケジュールを作成するか、編集するスケジュールのリストから既存のスケジュールを選択します。
3. 「設定」ページの「デッドレターキュー (DLQ) 」で、次のいずれかを実行します。
   - AWS アカウント内の Amazon SQSキューを としてDLQ選択 を選択し、ドロップダウンリストから ARNのキューDLQを選択します。
   - 他の AWS アカウントの Amazon SQSキューを として指定 を選択しDLQ、 のキュー ARNを入力しますDLQ。別の AWS アカウントでキューを選択すると、ス EventBridge ケジューラコンソールはキューをドロップダウンリストARNsに表示できなくなります。
4. 選択内容を確認し、スケジュールの作成またはスケジュールの保存を選択して、 の設定を完了しますDLQ。
5. (オプション) スケジュールDLQの詳細を表示するには、リストからスケジュールの名前を選択し、スケジュールの詳細ページのデッドレターキュータブを選択します。

AWS CLI

を使用して既存のスケジュールを更新するには AWS CLI

- `update-schedule` コマンドを使用してスケジュールを更新します。以前に作成した Amazon SQSキューを として指定しますDLQ。必要な Amazon アクセスSQS許可ARNを アタッチしたIAMロールを実行ロールとして指定します。他のすべてのプレースホルダー値を、ユーザー自身の情報に置き換えます。

```
$ 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"}'
```

DLQ を使用して で新しいスケジュールを作成するには AWS CLI

- スケジュールを作成するには、create-schedule コマンドを使用します。すべてのプレースホルダー値を、ユーザー自身の情報に置き換えます。

```
$ 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"}'
```

次のセクションでは、 を使用して AWS CLI からデッドレターイベントを受信しますDLQ。

デッドレターイベントの取得

以下に示すように receive-message コマンドを使用して、 からデッドレターイベントを取得しますDLQ。--max-number-of-messages 属性を使用して、取得するメッセージの数を設定できます。

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

成功すると、次のような出力が表示されます。

```
{
    "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"
                }
            }
        }
```

```
    ]
}
```

デッドレターイベントの以下の属性に注目しておくと、ターゲットの呼び出しが失敗した原因として考えられるものの特定とトラブルシューティングに役立ちます。

- **ERROR_CODE** – ス EventBridge ケジューラがターゲットのサービス から受け取るエラーコードが含まれますAPI。前の例では、Amazon によって返されるエラーコードは SQSですAWS.SimpleQueueService.NonExistentQueue。ス EventBridge ケジューラの問題によりスケジュールがターゲットの呼び出しに失敗した場合、代わりに というエラーコードが表示されますAWS.Scheduler.InternalServerError。
- **ERROR_MESSAGE** – ス EventBridge ケジューラがターゲットのサービス から受け取るエラーメッセージが含まれますAPI。前の例では、Amazon から返されるエラーメッセージSQSは ですThe specified queue does not exist for this wsdl version。ス EventBridge ケジューラの問題が原因でスケジュールが失敗した場合、代わりに というエラーメッセージが表示されますUnexpected error occurred while processing the request。
- **TARGET_ARN** - スケジュールが呼び出すターゲットARNの を、次のサービスARN形式で指定します: arn:aws:scheduler:::aws-sdk:*service*:*apiAction*。
- **EXHAUSTED_RETRY_CONDITION** – イベントが に配信された理由を示しますDLQ。この属性は、ターゲットからのエラーAPIが再試行可能なエラーであり、永続的なエラーではない場合に表示されます。属性には、スケジュールに設定した最大再試行回数を超えたDLQ後にスMaximumRetryAttempts EventBridge ケジューラがそれを に送信した場合、またはイベントがスケジュールに設定した最大期間より古くMaximumEventAgeInSeconds、まだ配信に失敗している場合、 の値を含めることができます。

前の例では、エラーコードとエラーメッセージに基づいて、スケジュールに指定したターゲットキューが存在しないと判断できます。

スケ EventBridge ジューラでスケジュールを削除する

スケジュールを削除するには、自動削除を設定するか、個々のスケジュールを手動で削除します。以下のトピックでは、両方の方法でスケジュールを削除する方法と、場面によってどちらの方法を選択した方がよいかについて説明します。

トピック

- スケジュール完了後の削除

- 手動削除

スケジュール完了後の削除

EventBridge スケジューラでスケジュールリソースを個別に管理する必要がない場合は、スケジュールの完了後に自動削除を設定します。一度に数千のスケジュールを作成し、必要に応じてスケジュール数を柔軟にスケールアップする必要がある用途では、自動削除を設定することで、指定したリージョンのスケジュール数に対するアカウントクォータに達しないようにすることができます。

スケジュールの自動削除を設定すると、ス EventBridge ケジューラは最後のターゲット呼び出し後にスケジュールを削除します。1 回限りのスケジュールの場合は、スケジュールがターゲットを一度呼び出した後に削除が実行されます。rate 式または cron 式を使用して設定した繰り返しのスケジュールの場合、スケジュールは最後の呼び出し後に削除されます。繰り返しのスケジュールの最後の呼び出しは、指定した `EndDate` に最も近い呼び出しです。自動削除でスケジュールを設定しても、の値を指定しない場合`EndDate`、ス EventBridge ケジューラはスケジュールを自動的に削除しません。

スケジュールを最初に作成するときに自動削除を設定したり、既存のスケジュールの設定を更新したりできます。次のステップでは、既存のスケジュールの自動削除を設定する方法について説明します。

AWS Management Console

1. でス EventBridge ケジューラコンソールを開きますhttps://console.aws.amazon.com/scheduler/。
2. スケジュールされたリストから、編集するスケジュールを選択し、[編集] を選択します。
3. 左のナビゲーションリストから、[設定] を選択します。
4. スケジュール完了後のアクションセクションDELETEで、ドロップダウンリストから を選択し、変更を保存します。

AWS CLI

1. 新しいプロンプトウィンドウを開きます。
2. update-schedule AWS CLI コマンドを使用して、次に示す既存のスケジュールを更新します。このコマンドは `--action-after-completion` を `DELETE` に設定します。この例では、ターゲット設定を JSON ファイルにローカルで定義していることを前提としています。

スケジュールを更新するには、ターゲットのほか、既存のスケジュールに設定したいその他のスケジュールパラメータを指定する必要があります。

これは 1 時間に 1 回の頻度で呼び出しを行う繰り返しのスケジュールです。そのため、`--action-after-completion` パラメータを設定する際に終了日を指定します。

```
$ 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"}' \
```

手動削除

スケジュールが不要になった場合には、DeleteSchedule オペレーションを使用して削除することができます。

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")
```

次のステップ

- Lambda と Step Functions のテンプレート化されたターゲットを設定する方法の詳細と、ユニバーサルターゲットパラメータの使用方法については、「ターゲットの管理」を参照してください。
- ス EventBridge ケジューラのデータ型とAPIオペレーションの詳細については、スケEventBridgeジューラAPIリファレンス を参照してください。

ス EventBridge ケジューラでのスケジュールグループの管理

スケジュールグループは、スケジュールの整理に使用する Amazon EventBridge Scheduler リソースです。

にはスdefaultケジューラグループが付属 AWS アカウント しています。新しいスケジュールは、default グループ、または自分で作成して管理するスケジュールグループに関連付けることができます。に最大 500 のスケジュールグループを作成できます AWS アカウント。ス EventBridge ケジューラでは、タグ を適用して、個々のスケジュールではなくスケジュールグループを整理します。

タグとは、ユーザーが定義する大文字と小文字を区別するキーと値で構成されるラベルです。タグを作成して、目的、所有者、環境などの基準に基づいてスケジュールを分類できます。例えば、次のタグを使用して、スケジュールが属する環境を特定できます: environment:*production*

> **Important**
>
> 個人を特定できる情報 (PII) やその他の機密情報や機密情報をタグに追加しないでください。タグには、請求を含む多くの AWS のサービスからアクセスできます。タグは、プライベートデータや機密データに使用することを意図していません。

スケジュールグループには、 ACTIVEと https://docs.aws.amazon.com/scheduler/latest/APIReference/API_GetScheduleGroup.html#scheduler-GetScheduleGroup-response-Stateの 2 つの状態がありますDELETING。

最初にグループを作成すると、デフォルトでは ACTIVE になっています。スケジュールは ACTIVE グループに追加できます。グループを削除すると、ス EventBridge ケジューラが関連するスケジュールの削除を完了するDELETINGまで、状態は に変わります。ス EventBridge ケジューラがグループ内のスケジュールを削除すると、そのグループはアカウントで使用できなくなります。

以下のトピックでは、スケジュールグループを作成し、タグを適用します。また、スケジュールをグループに関連付けることもできます。最後に、グループを削除します。

トピック

- ス EventBridge ケジューラでスケジュールグループを作成する
- ス EventBridge ケジューラでスケジュールグループを削除する

- 関連リソース

ス EventBridge ケジューラでスケジュールグループを作成する

スケジュールグループとタグ付けを使用して、共通の目的を共有するスケジュールや同じ環境に属するスケジュールを整理します。次の手順では、新しいスケジュールグループを作成し、タグを使用してラベルを付けます。次に、新しいスケジュールをそのグループに関連付けます。

> **Note**
>
> グループを作成すると、そのグループからスケジュールを削除したり、そのスケジュールを別のグループに関連付けることはできません。スケジュールをグループに関連付けることができるのは、最初にスケジュールを作成したときだけです。

ステップ 1: 新しいスケジュールグループを作成する

以下のトピックでは、新しいスケジュールグループを作成して次のタグを使用してラベルを付ける方法について説明します: `environment:development`

AWS Management Console

を使用して新しいグループを作成するには AWS Management Console

1. にサインイン AWS Management Console し、 で Amazon EventBridge コンソールを開きますhttps://console.aws.amazon.com/events/。
2. 左側のナビゲーションペインで、[スケジュールグループ] をクリックします。
3. [スケジュールグループ] ページで、[スケジュールグループを作成] を選択します。
4. [スケジュールグループ詳細] セクションの [名前] に、グループの名前を入力します。例えば、**`TestGroup`** と指定します。
5. [タグ] セクションで、次の操作を行います。

   a. [新しいタグを追加] をクリックします。

   b. [キー] には、このキーに割り当てる名前を入力します。このチュートリアルでは、このスケジュールグループが属する環境にラベルを付けるには、**`environment`** と入力します。

c. [値 - オプション] には、このキーに割り当てる値を入力します。このチュートリアルでは、環境キーの値 **development** を入力します。

> **Note**
>
> グループを作成した後に、タグを追加できます。

6. [スケジュールグループを作成] を選択して終了します。新しいグループが [スケジュールグループ] リストに表示されます。
7. (オプション) グループを編集したりタグを管理したりするには、新しいグループのチェックボックスを選択して [編集] を選択します。

> **Note**
>
> `default` スケジュールグループは編集できません。

AWS CLI

を使用して新しいグループを作成するには AWS CLI

1. 新しいコマンドプロンプトウィンドウを開きます。
2. AWS Command Line Interface （ AWS CLI) から、次の`create-schedule-group`コマンドを入力して新しいグループを作成します。このコマンドは、1 つのタグ `environment:development` を使用してグループを作成します。このタグまたは類似のタグ付けシステムを使用して、スケジュールグループが属する環境に応じてラベルを付けることができます。

   スケジュール名とタグのキーと値を自分自身の情報に置き換えます。

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

デフォルトでは、新しいグループの状態は `ACTIVE` になります。これで、作成した新しいグループに新しいスケジュールを関連付けることができます。

ステップ 2: スケジュールをグループに関連付ける

次の手順を使用して、前のステップで作成したグループに新しいスケジュールを関連付けます。

AWS Management Console

を使用してスケジュールをグループに関連付けるには AWS Management Console

1. にサインイン AWS Management Console し、 で Amazon EventBridge コンソールを開きますhttps://console.aws.amazon.com/events/。
2. 左のナビゲーションペインで、[スケジュール] を選択します。
3. [スケジュール] テーブルから [スケジュールを作成] を選択し、新しいスケジュールを作成します。
4. [スケジュールの詳細を指定] ページの [スケジュールグループ] で、ドロップダウンリストから新しいグループの名前を選択します。例えば、TestGroup を選択します。
5. スケジュールのパターン、ターゲット、設定を指定し、[スケジュールの確認と保存] ページで選択内容を確認します。新しいスケジュールの設定の詳細については、「使用開始」を参照してください。
6. スケジュールを終了して保存するには、[スケジュールを保存] を選択します。

AWS CLI

を使用してスケジュールをグループに関連付けるには AWS CLI

1. 新しいコマンドプロンプトウィンドウを開きます。
2. AWS Command Line Interface （ AWS CLI) から、次のコマンドを入力しますcreate-schedule。これにより、スケジュールが作成され、前のステップの sqs-test-schedule という名前のグループに関連付けられます。このスケジュールでは、テンプレート化された Amazon SQS ターゲットタイプを使用して SendMessageオペレーションを呼び出します。スケジュール名、ターゲット、およびグループ名を自分自身の情報に置き換えます。

```
$ 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"}'
```

これで、新しいスケジュールが TestGroup スケジュールグループに関連付けられました。

ス EventBridge ケジューラでスケジュールグループを削除する

以下では、 AWS Management Console と を使用してスケジュールグループを削除する方法について説明します AWS Command Line Interface。グループを削除すると、ス EventBridge ケジューラがグループ内のすべてのスケジュールを削除するまで、そのグループは DELETING状態になります。ス EventBridge ケジューラがグループ内のスケジュールを削除すると、そのグループはアカウントで使用できなくなります。

> **Note**
>
> グループを作成すると、そのグループからスケジュールを削除したり、そのスケジュールを別のグループに関連付けることはできません。スケジュールをグループに関連付けることができるのは、最初にスケジュールを作成したときだけです。

AWS Management Console

を使用してグループを削除するには AWS Management Console

1. にサインイン AWS Management Console し、 で Amazon EventBridge コンソールを開きますhttps://console.aws.amazon.com/events/。
2. 左のナビゲーションペインで、[スケジュールグループ] を選択します。
3. スケジュールグループページで、現在の 内の既存のグループのリストから AWS リージョン、削除するグループを見つけます。探しているグループが表示されない場合は、別の を選択します AWS リージョン。

   > **Note**
   >
   > default グループを削除したり編集したりすることはできません。

4. 削除するグループのチェックボックスをオンにします。
5. [削除] を選択します。
6. [スケジュールグループの削除] ダイアログボックスで、グループの名前を入力して選択を確定し、[削除] を選択します。

7. [スケジュールグループ] リストの [ステータス] 列が変わり、グループが削除中であることが示されます。ス EventBridge ケジューラがグループに関連付けられているすべてのスケジュールを削除するまで、グループはこの状態のままになります。
8. リストを更新してグループが削除されたことを確認するには、[更新] アイコンを選択します。

AWS CLI

を使用してグループを削除するには AWS CLI

1. 新しいコマンドプロンプトウィンドウを開きます。
2. AWS Command Line Interface （ AWS CLI) から次のdelete-schedule-groupコマンドを入力して、スケジュールグループを削除します。--name の値を自分自身の情報に置き換えます。

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

成功した場合、この AWS CLI オペレーションはレスポンスを返しません。

3. グループが DELETING の状態にあることを確認するには、以下の get-schedule-group コマンドを実行します。

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

成功すると、次のような出力が表示されます。

```
{
    "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 スケジューラは、グループに関連付けられたスケジュールを削除した後にグループを削除します。get-schedule-group をもう一度実行すると、次の ResourceNotFoundException 応答が返されます。

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

関連リソース

スケジュールグループの詳細については、以下のリソースを参照してください。

- CreateScheduleGroup スEventBridge ケジューラAPIリファレンスの オペレーション。
- DeleteScheduleGroup スEventBridge ケジューラAPIリファレンスの オペレーション。

ス EventBridge ケジューラでのターゲットの管理

以下のトピックでは、ス EventBridge ケジューラでテンプレート化されたターゲットとユニバーサルターゲットを使用する方法について説明し、ス EventBridge ケジューラのユニバーサルターゲットパラメータを使用して設定できるサポートされている AWS サービスのリストを提供します。

テンプレート化されたターゲットは、Amazon 、LambdaSQS、Step Functions などのコア AWS サービスのグループ全体で共通のAPIオペレーションのセットです。例えば、関数 を指定して Lambda の呼び出しAPIオペレーションをターゲットにしたりARN、ターゲットARNのキューで Amazon SQSの `SendMessage`オペレーションをターゲットにしたりできます。

ユニバーサルターゲットはカスタマイズ可能なパラメータのセットで、多くの AWS サービスに対してより広範なAPIオペレーションのセットを呼び出すことができます。例えば、ス EventBridge ケジューラのユニバーサルターゲットパラメータ (UTP) を使用して、 `CreateQueue`オペレーションを使用して新しい Amazon SQSキューを作成できます。

テンプレート化されたターゲットまたはユニバーサルターゲットを設定するには、ターゲットとして設定した API オペレーションを呼び出すアクセス許可がスケジュールに必要です。そのためには、スケジュールの実行ロールに必要なアクセス許可をアタッチします。例えば、Amazon SQSの `SendMessage`オペレーションをターゲットにするには、実行ロールに `sqs:SendMessage` アクションを実行するアクセス許可が付与されます。ほとんどの場合、ターゲットサービスがサポートする AWS マネージドポリシーを使用して必要なアクセス権限を追加できます。ただし、独自のカスタマー管理ポリシーを作成したり、実行ロールにアタッチされた既存のポリシーにインライン権限を追加したりすることもできます。以下のトピックでは、テンプレート化されたターゲットタイプとユニバーサルターゲットタイプの両方にアクセス権限を追加する例を示しています。

スケジュールの実行ロールのセットアップについては、「the section called “実行ロールを設定する”」を参照してください。

トピック

ス EventBridge ケジューラでのテンプレート化されたターゲットの使用

テンプレート化されたターゲットは、Amazon 、SQSLambda、Step Functions などのコア AWS サービスのグループ全体で共通するAPI一連のオペレーションです。例えば、関数 を指定して Lambda の InvokeオペレーションをターゲットにしたりARN、キュー を使用して Amazon SQSの SendMessageオペレーションをターゲットにしたりできますARN。テンプレート化されたターゲットを設定するには、ターゲットAPIオペレーションを実行するためのアクセス許可をスケジュールの実行ロールに付与する必要もあります。

AWS CLI またはス EventBridge ケジューラ のいずれかを使用して、プログラムでテンプレート化されたターゲットを設定するにはSDKs、実行ロールARNの 、ターゲットリソースARNの 、ス EventBridge ケジューラがターゲットに配信するオプションの入力、および一部のテンプレート化されたターゲットに対して、そのターゲットに追加の設定オプションを含む一意のパラメータセットを指定する必要があります。テンプレート化されたターゲットリソースARNに を指定すると、ス EventBridge ケジューラはそのサービスでサポートされているAPIオペレーションを呼び出すことを自動的に想定します。ス EventBridge ケジューラでサービスの別のAPIオペレーションをターゲットにする場合は、ターゲットをユニバーサルターゲット として設定する必要があります。

以下は、ス EventBridge ケジューラがサポートするすべてのテンプレート化されたターゲットの完全なリストであり、該当する場合は、各ターゲットの関連するパラメータの一意のセットです。各パラメータセットのリンクを選択すると、スEventBridge ケジューラAPIリファレンスの必須およびオプションのフィールドが表示されます。

- CodeBuild – StartBuild
- CodePipeline – StartPipelineExecution
- Amazon ECS – RunTask
  - パラメータ: EcsParameters
- EventBridge – PutEvents
  - パラメータ: EventBridgeParameters
- Amazon Inspector – StartAssessmentRun
- Kinesis – PutRecord
  - パラメータ: KinesisParameters
- Firehose – PutRecord
- Lambda – Invoke

- SageMaker – StartPipelineExecution
  - パラメータ: SageMakerPipelineParameters
- Amazon SNS – Publish
- Amazon SQS – SendMessage
  - パラメータ: SqsParameters
- Step Functions – StartExecution

次の例を使用して、さまざまなテンプレート化されたターゲットを設定する方法と、記述された各ターゲットに必要なIAMアクセス許可について説明します。

Amazon SQS **SendMessage**

Example 実行ロールのアクセス権限ポリシー

```
{
    "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 実行ロールのアクセス権限ポリシー

```
{
    "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 実行ロールのアクセス権限ポリシー

```
{
    "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");
    }
}
```

ス EventBridge ケジューラでのユニバーサルターゲットの使用

ユニバーサルターゲットはカスタマイズ可能なパラメータのセットで、多くの AWS サービスに対してより広範なAPIオペレーションのセットを呼び出すことができます。例えば、ユニバーサルターゲットパラメータ (UTP) を使用して、 `CreateQueue`オペレーションを使用して新しい Amazon SQSキューを作成できます。

、またはス EventBridge ケジューラ のいずれかを使用して AWS CLIスケジュールのユニバーサルターゲットを設定するにはSDKs、次の情報を指定する必要があります。

- RoleArn – ターゲットに使用するARN実行ロールの 。指定する実行ロールには、スケジュールでターゲットにするAPIオペレーションを呼び出すアクセス許可が必要です。
- Arn – ターゲットにするAPIオペレーションARNを含む完全なサービス を の形式で指定しますarn:aws:scheduler:::aws-sdk:*service*:*apiAction*。

  例えば、Amazon の場合SQS、指定するサービス名は ですarn:aws:scheduler:::aws-sdk:**sqs:sendMessage**。
- 入力 – ス EventBridge ケジューラがターゲット に送信するリクエストパラメータでJSON指定する正しい形式API。でJSON設定した のパラメータと形状Inputは、スケジュールが呼び出すサービスによって決まりAPIます。この情報を確認するには、ターゲットにするサービスのAPIリファレンスを参照してください。

サポートされていないアクション

EventBridge スケジューラは、次のプレフィックスのリストで始まる一般的なGETオペレーションなどの読み取り専用APIアクションをサポートしていません。

```
get
describe
list
poll
receive
```

```
search
scan
query
select
read
lookup
discover
validate
batchGet
batchDescribe
batchRead
transactGet
adminGet
adminList
testMigration
retrieve
testConnection
translateDocument
isAuthorized
invokeModel
```

例えば、 GetQueueUrlAPIアクションARNのサービスは になりますarn:aws:scheduler:::aws-sdk:sqs:**getQueueURL**。API アクションは get プレフィックスで始まるため、ス EventBridge ケジューラはこのターゲットをサポートしていません。同様に、Amazon MQ アクションListBrokersはプレフィックス で始まるため、ターゲットとしてサポートされていませんlist。

ユニバーサルターゲットの使用例

スケジュールInputフィールドで渡すパラメータは、API呼び出すサービスが受け入れるリクエストパラメータによって異なります。例えば、Lambda をターゲットにするにはInvoke、AWS Lambda APIリファレンス にリストされているパラメータを設定できます。これには、Lambda 関数に渡すことができるオプションのJSONペイロードが含まれます。

さまざまな に設定できるパラメータを確認するにはAPIs、そのサービスのAPIリファレンスを参照してください。Lambda と同様にInvoke、一部の はURIパラメータとリクエストボディペイロードAPIsを受け入れます。このような場合は、スケジュール でURIパスパラメータとJSONペイロードを指定しますInput。

次の例は、ユニバーサルターゲットを使用して Lambda、Amazon 、SQSおよび Step Functions で一般的なAPIオペレーションを呼び出す方法を示しています。

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");
    }
}
```

ス EventBridge ケジューラでのコンテキスト属性の追加

ターゲットに渡すペイロードで次のキーワードを使用して、スケジュールに関するメタデータを収集します。スケジュールがターゲットを呼び出すと、ス EventBridge ケジューラは各キーワードをそれぞれの値に置き換えます。

- **<aws.scheduler.schedule-arn>** – スケジュールARNの 。
- **<aws.scheduler.scheduled-time>** — スケジュールがターゲットを呼び出すために指定した時間 (例: 2022-03-22T18:59:43Z)。
- **<aws.scheduler.execution-id>** – ターゲットの呼び出しが試行されるたびにス EventBridge ケジューラが割り当てる一意の IDd32c5kddcf5bb8c3。例えば、。
- **<aws.scheduler.attempt-number>** — 現在の呼び出しの試行回数を識別するカウンター (例: 1)。

この例では、5 分ごとに起動し、Amazon SQSSendMessageオペレーションをユニバーサルターゲットとして呼び出すスケジュールを作成します。メッセージ本文には schedule-time の値が含まれています。

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)
```

次のステップ

ス EventBridge ケジューラのデータ型とAPIオペレーションの詳細については、「スケEventBridge ジューラAPIリファレンス」を参照してください。

Amazon EventBridge Scheduler のセキュリティ

のクラウドセキュリティが最優先事項 AWS です。お客様は AWS 、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャからメリットを得られます。

セキュリティは、 AWS とユーザーの間で共有される責任です。責任共有モデルでは、これをクラウドのセキュリティおよびクラウド内のセキュリティと説明しています。

- クラウドのセキュリティ — AWS は、 で AWS サービスを実行するインフラストラクチャを保護する責任を担います AWS クラウド。また、 は、安全に使用できるサービス AWS も提供します。コンプライアンスAWS プログラムコンプライアンスプログラム の一環として、サードパーティーの監査者は定期的にセキュリティの有効性をテストおよび検証。Amazon EventBridge Scheduler に適用されるコンプライアンスプログラムの詳細については、「コンプライアンスプログラムAWS による対象範囲内のサービスコンプライアンスプログラム」を参照してください。
- クラウドのセキュリティ — お客様の責任は、使用する AWS サービスによって決まります。また、お客様は、データの機密性、会社の要件、適用される法律や規制など、その他の要因についても責任を負います。

このドキュメントは、スケ EventBridge ジューラの使用時に責任共有モデルを適用する方法を理解するのに役立ちます。以下のトピックでは、セキュリティおよびコンプライアンスの目的を達成するようにス EventBridge ケジューラを設定する方法について説明します。また、ス EventBridge ケジューラリソースのモニタリングや保護に役立つ他の AWS のサービスの使用方法についても説明します。

トピック

- Amazon EventBridge Scheduler へのアクセスの管理
- Amazon EventBridge Scheduler でのデータ保護
- Amazon EventBridge Scheduler のコンプライアンス検証
- Amazon EventBridge Scheduler の耐障害性
- Amazon EventBridge Scheduler のインフラストラクチャセキュリティ

Amazon EventBridge Scheduler へのアクセスの管理

AWS Identity and Access Management （ IAM) は、管理者が AWS リソースへのアクセスを安全に制御 AWS サービス するのに役立つ です。IAM 管理者は、誰を認証 (サインイン) し、誰にス

EventBridge ケジューラリソースの使用を承認する (アクセス許可を付与する) かを制御します。IAM は追加料金なしで AWS サービス 使用できる です。

トピック

対象者

AWS Identity and Access Management （ IAM) の使用方法は、スケ EventBridge ジューラで行う作業によって異なります。

サービスユーザー – ジョブを実行するためにス EventBridge ケジューラサービスを使用する場合、管理者から必要な認証情報とアクセス許可が与えられます。さらに多くのス EventBridge ケジューラ機能を使用して作業を行う場合は、追加のアクセス許可が必要になることがあります。アクセスの管理方法を理解しておくと、管理者に適切な許可をリクエストするうえで役立ちます。ス EventBridge ケジューラの機能にアクセスできない場合は、「」を参照してくださいAmazon EventBridge Scheduler のアイデンティティとアクセスのトラブルシューティング。

サービス管理者 – 社内のス EventBridge ケジューラリソースを担当している場合は、通常、スケ EventBridge ジューラへのフルアクセスがあります。サービスユーザーがどのス EventBridge ケジューラ機能やリソースにアクセスするかを決めるのは管理者の仕事です。次に、サービスユーザーのアクセス許可を変更するリクエストをIAM管理者に送信する必要があります。このページの情報を確認して、 の基本概念を理解してくださいIAM。会社がス EventBridge ケジューラIAMで を使用する方法の詳細については、「」を参照してくださいス EventBridge ケジューラと の連携方法 IAM。

IAM 管理者 – IAM管理者は、ス EventBridge ケジューラへのアクセスを管理するポリシーの作成方法の詳細について確認する場合があります。で使用できるス EventBridge ケジューラのアイデンティティベースのポリシーの例を表示するにはIAM、「」を参照してくださいス EventBridge ケジューラでのアイデンティティベースのポリシーの使用。

アイデンティティを使用した認証

認証とは、ID 認証情報 AWS を使用して にサインインする方法です。として、IAMユーザーとして AWS アカウントのルートユーザー、または IAMロールを引き受けることによって認証 (にサインイン AWS) される必要があります。

ID ソースを介して提供された認証情報を使用して、フェデレーティッド ID AWS として にサインインできます。 AWS IAM Identity Center (IAM Identity Center) ユーザー、会社のシングルサインオン認証、Google または Facebook の認証情報は、フェデレーティッド ID の例です。フェデレーティッド ID としてサインインすると、管理者は以前に IAMロールを使用して ID フェデレーションをセットアップしていました。フェデレーション AWS を使用して にアクセスすると、間接的にロールを引き受けることになります。

ユーザーのタイプに応じて、 AWS Management Console または AWS アクセスポータルにサインインできます。へのサインインの詳細については AWS、「 ユーザーガイド」の「 へのサインイン AWS アカウント方法AWS サインイン 」を参照してください。

AWS プログラムで にアクセスする場合、 はソフトウェア開発キット (SDK) とコマンドラインインターフェイス (CLI) AWS を提供し、認証情報を使用してリクエストに暗号で署名します。 AWS ツールを使用しない場合は、リクエストに自分で署名する必要があります。推奨される方法を使用してリクエストを自分で署名する方法の詳細については、「 IAMユーザーガイド」のAWS API「リクエストの署名」を参照してください。

使用する認証方法を問わず、追加セキュリティ情報の提供をリクエストされる場合もあります。例えば、 AWS では、アカウントのセキュリティを高めるために多要素認証 (MFA) を使用することをお勧めします。詳細については、「 AWS IAM Identity Center ユーザーガイド」の「多要素認証」および「 ユーザーガイド」の「 での多要素認証 (MFA) AWS IAM の使用」を参照してください。

AWS アカウント ルートユーザー

を作成するときは AWS アカウント、アカウント内のすべての およびリソースへの AWS サービス 完全なアクセス権を持つ 1 つのサインインアイデンティティから始めます。この ID は AWS アカウント ルートユーザーと呼ばれ、アカウントの作成に使用した E メールアドレスとパスワードでサインインすることでアクセスできます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザーの認証情報は保護し、ルートユーザーでしか実行できないタスクを実行するときに使用します。ルートユーザーとしてサインインする必要があるタスクの完全なリストについては、「 IAMユーザーガイド」の「ルートユーザーの認証情報を必要とするタスク」を参照してください。

フェデレーティッドアイデンティティ

ベストプラクティスとして、管理者アクセスを必要とするユーザーを含む人間のユーザーに、一時的な認証情報を使用して にアクセスするための ID プロバイダーとのフェデレーションの使用を要求 AWS サービス します。

フェデレーティッド ID は、エンタープライズユーザーディレクトリ、ウェブ ID プロバイダー、AWS Directory Service、アイデンティティセンターディレクトリのユーザー、または ID ソースを通じて提供された認証情報 AWS サービス を使用して にアクセスするユーザーです。フェデレーティッド ID が にアクセスすると AWS アカウント、ロールが引き受けられ、ロールは一時的な認証情報を提供します。

アクセスを一元管理する場合は、 AWS IAM Identity Centerを使用することをお勧めします。Identity Center でユーザーとグループを作成することも、独自の IAM ID ソース内のユーザーとグループのセットに接続して同期して、すべての AWS アカウント とアプリケーションで使用することもできます。IAM Identity Center の詳細については、「 ユーザーガイド」のIAM「Identity Center とはAWS IAM Identity Center 」を参照してください。

IAM ユーザーとグループ

IAM ユーザーは、単一のユーザーまたはアプリケーションに対して特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。可能であれば、パスワードやアクセスキーなどの長期的な認証情報を持つIAMユーザーを作成する代わりに、一時的な認証情報を使用することをお勧めします。ただし、IAMユーザーとの長期的な認証情報を必要とする特定のユースケースがある場合は、アクセスキーをローテーションすることをお勧めします。詳細については、「 ユーザーガイド」の「長期的な認証情報を必要とするユースケースでアクセスキーを定期的にローテーションするIAM」を参照してください。

IAM グループは、IAMユーザーのコレクションを指定するアイデンティティです。グループとしてサインインすることはできません。グループを使用して、複数のユーザーに対して一度に権限を指定できます。多数のユーザーグループがある場合、グループを使用することで権限の管理が容易になります。例えば、 という名前のグループを作成しIAMAdmins、そのグループにIAMリソースを管理するアクセス許可を付与できます。

ユーザーは、ロールとは異なります。ユーザーは 1 人の人または 1 つのアプリケーションに一意に関連付けられますが、ロールはそれを必要とする任意の人が引き受けるようになっています。ユーザーには永続的な長期の認証情報がありますが、ロールでは一時認証情報が提供されます。詳細については、「 ユーザーガイド」のIAM「 (ロールの代わりに) ユーザーを作成する場合IAM」を参照してください。

IAM ロール

IAM ロールは、特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。ユーザーと似ていますがIAM、特定のユーザーに関連付けられていません。IAM ロール を切り替える AWS Management Console ことで、 で ロールを一時的に引き受けることができます。ロールを引き受けるには、 または AWS API オペレーションを AWS CLI 呼び出すか、カスタム を使用しますURL。ロールの使用方法の詳細については、「 ユーザーガイド」のIAM「ロールの使用IAM」を参照してください。

IAM 一時的な認証情報を持つ ロールは、以下の状況で役立ちます。

- フェデレーションユーザーアクセス – フェデレーティッド ID に許可を割り当てるには、ロールを作成してそのロールの許可を定義します。フェデレーティッド ID が認証されると、その ID はロールに関連付けられ、ロールで定義されている許可が付与されます。フェデレーションのロールの詳細については、「 ユーザーガイド」の「サードパーティー ID プロバイダーのロールの作成IAM」を参照してください。IAM Identity Center を使用する場合は、アクセス許可セットを設定します。ID が認証後にアクセスできる内容を制御するために、IAMIdentity Center はアクセス許可セットを のロールに関連付けますIAM。アクセス許可セットの詳細については、「AWS IAM Identity Center ユーザーガイド」の「アクセス許可セット」を参照してください。
- 一時的なIAMユーザーアクセス許可 – IAM ユーザーまたはロールは、 IAMロールを引き受けて、特定のタスクに対して異なるアクセス許可を一時的に引き受けることができます。
- クロスアカウントアクセス – IAMロールを使用して、別のアカウントのユーザー (信頼されたプリンシパル) が自分のアカウントのリソースにアクセスすることを許可できます。クロスアカウントアクセスを許可する主な方法は、ロールを使用することです。ただし、一部の では AWS サービス、 (ロールをプロキシとして使用する代わりに) ポリシーをリソースに直接アタッチできます。クロスアカウントアクセスのロールとリソースベースのポリシーの違いについては、「 ユーザーガイド」の「 でのクロスアカウントリソースアクセスIAMIAM」を参照してください。
- クロスサービスアクセス — 一部の は、他の の機能 AWS サービス を使用します AWS サービス。例えば、サービスで呼び出しを行うと、そのサービスが Amazon でアプリケーションを実行 EC2したり、Amazon S3 にオブジェクトを保存したりするのが一般的です。サービスでは、呼び出し元プリンシパルの許可、サービスロール、またはサービスリンクロールを使用してこれを行う場合があります。
  - 転送アクセスセッション (FAS) – IAM ユーザーまたはロールを使用して でアクションを実行すると AWS、プリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を AWS サービス、ダウンストリームサービス AWS サービス へのリクエストのリクエストと組み合わせて使用します。FAS リクエストは、サービスが他

の AWS サービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するための権限が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。

- サービスロール – サービスロールは、ユーザーに代わってアクションを実行するためにサービスが引き受ける IAMロールです。IAM 管理者は、 内からサービスロールを作成、変更、削除できますIAM。詳細については、「 ユーザーガイド」の「 にアクセス許可を委任するロールの作成 AWS サービスIAM」を参照してください。
- サービスにリンクされたロール – サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS サービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスにリンクされたロールのアクセス許可を表示できますが、編集することはできません。

- Amazon で実行されているアプリケーション EC2 – IAMロールを使用して、EC2インスタンスで実行され、 AWS CLI または AWS API リクエストを行うアプリケーションの一時的な認証情報を管理できます。これは、EC2インスタンス内にアクセスキーを保存するよりも望ましいです。 AWS ロールをEC2インスタンスに割り当て、そのすべてのアプリケーションで使用できるようにするには、インスタンスにアタッチされたインスタンスプロファイルを作成します。インスタンスプロファイルには ロールが含まれており、EC2インスタンスで実行されているプログラムが一時的な認証情報を取得できるようにします。詳細については、「 ユーザーガイド」の「 IAMロールを使用して Amazon EC2インスタンスで実行されているアプリケーションにアクセス許可を付与するIAM」を参照してください。

IAM ロールとIAMユーザーのどちらを使用するかについては、「 ユーザーガイド」の「 (ユーザーではなく) IAMロールを作成する場合IAM」を参照してください。

ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、 AWS ID またはリソースにアタッチします。ポリシーは AWS 、アイデンティティまたはリソースに関連付けられているときにアクセス許可を定義する のオブジェクトです。 は、プリンシパル (ユーザー、ルートユーザー、またはロールセッション) がリクエストを行うときに、これらのポリシー AWS を評価します。ポリシーでの権限により、リクエストが許可されるか拒否されるかが決まります。ほとんどのポリシーはJSONドキュメント AWS として に保存されます。JSON ポリシードキュメントの構造と内容の詳細については、「 ユーザーガイド」のJSON「ポリシーの概要IAM」を参照してください。

管理者はポリシーを使用して AWS JSON、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

デフォルトでは、ユーザーやロールに権限はありません。必要なリソースに対してアクションを実行するアクセス許可をユーザーに付与するために、IAM管理者はIAMポリシーを作成できます。その後、管理者はIAMポリシーをロールに追加し、ユーザーはロールを引き受けることができます。

IAM ポリシーは、オペレーションの実行に使用するメソッドに関係なく、アクションのアクセス許可を定義します。例えば、`iam:GetRole`アクションを許可するポリシーがあるとします。そのポリシーを持つユーザーは、 AWS Management Console、、 AWS CLIまたは AWS からロール情報を取得できますAPI。

アイデンティティベースのポリシー

ID ベースのポリシーは、IAMユーザー、ユーザーのグループ、ロールなどの ID にアタッチできるJSONアクセス許可ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースのポリシーを作成する方法については、「 ユーザーガイド」のIAM「ポリシーの作成IAM」を参照してください。

アイデンティティベースのポリシーは、さらにインラインポリシーまたはマネージドポリシーに分類できます。インラインポリシーは、単一のユーザー、グループ、またはロールに直接埋め込まれています。管理ポリシーは、 内の複数のユーザー、グループ、ロールにアタッチできるスタンドアロンポリシーです AWS アカウント。管理ポリシーには、 AWS 管理ポリシーとカスタマー管理ポリシーが含まれます。管理ポリシーとインラインポリシーのどちらかを選択する方法については、「 IAM ユーザーガイド」の「管理ポリシーとインラインポリシーの選択」を参照してください。

リソースベースのポリシー

リソースベースのポリシーは、リソースにアタッチするJSONポリシードキュメントです。リソースベースのポリシーの例としては、IAMロールの信頼ポリシー や Amazon S3 バケットポリシー などがあります。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスをコントロールできます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS サービス。

リソースベースのポリシーは、そのサービス内にあるインラインポリシーです。リソースベースのポリシーIAMでは、 の AWS 管理ポリシーを使用できません。

アクセスコントロールリスト (ACLs)

アクセスコントロールリスト (ACLs) は、リソースへのアクセス許可を持つプリンシパル (アカウントメンバー、ユーザー、またはロール) を制御します。ACLs はリソースベースのポリシーに似ていますが、JSONポリシードキュメント形式を使用しません。

Amazon S3、 AWS WAF、および Amazon VPCは、 をサポートするサービスの例ですACLs。の詳細についてはACLs、「Amazon Simple Storage Service デベロッパーガイド」の「アクセスコントロールリスト (ACL) の概要」を参照してください。

その他のポリシータイプ

AWS は、一般的ではない追加のポリシータイプをサポートします。これらのポリシータイプでは、より一般的なポリシータイプで付与された最大の権限を設定できます。

- アクセス許可の境界 – アクセス許可の境界は、アイデンティティベースのポリシーがIAMエンティティ (IAMユーザーまたはロール) に付与できるアクセス許可の上限を設定する高度な機能です。エンティティにアクセス許可の境界を設定できます。結果として得られる権限は、エンティティのアイデンティティベースポリシーとそのアクセス許可の境界の共通部分になります。`Principal` フィールドでユーザーまたはロールを指定するリソースベースのポリシーでは、アクセス許可の境界は制限されません。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。アクセス許可の境界の詳細については、「 IAMユーザーガイド」の「 IAMエンティティのアクセス許可の境界」を参照してください。
- サービスコントロールポリシー (SCPs) – SCPsは、 の組織または組織単位 (OU) に対する最大アクセス許可を指定するJSONポリシーです AWS Organizations。 AWS Organizations は、 AWS アカウント ビジネスが所有する複数の をグループ化して一元管理するためのサービスです。組織内のすべての機能を有効にすると、サービスコントロールポリシー (SCPs) をアカウントの一部またはすべてに適用できます。は、各 を含むメンバーアカウントのエンティティのアクセス許可SCP を制限します AWS アカウントのルートユーザー。Organizations と の詳細についてはSCPs、「 AWS Organizations ユーザーガイド」の「サービスコントロールポリシー」を参照してください。
- セッションポリシー - セッションポリシーは、ロールまたはフェデレーションユーザーの一時的なセッションをプログラムで作成する際にパラメータとして渡す高度なポリシーです。結果としてセッションの権限は、ユーザーまたはロールのアイデンティティベースポリシーとセッションポリシーの共通部分になります。また、リソースベースのポリシーから権限が派生する場合もあります。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。詳細については、「 ユーザーガイド」の「セッションポリシーIAM」を参照してください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成される権限を理解するのがさらに難しくなります。複数のポリシータイプが関与する場合にリクエストを許可するかどうか AWS を決定する方法については、 ユーザーガイドの「ポリシー評価ロジックIAM」を参照してください。

ス EventBridge ケジューラと の連携方法 IAM

IAM を使用してス EventBridge ケジューラへのアクセスを管理する前に、ス EventBridge ケジューラで使用できるIAM機能を確認してください。

IAM Amazon EventBridge Scheduler で使用できる の機能

IAM 機能	EventBridge スケジューラのサポート
アイデンティティベースのポリシー	あり
リソースベースのポリシー	なし
ポリシーアクション	あり
ポリシーリソース	はい
ポリシー条件キー (サービス固有)	あり
ACLs	なし
ABAC (ポリシー内のタグ)	部分的
一時的な認証情報	あり
プリンシパル権限	あり
サービスロール	あり
サービスリンクロール	なし

ス EventBridge ケジューラおよびその他の AWS のサービスがほとんどのIAM機能と連携する方法の概要を把握するには、IAM「 ユーザーガイド」のAWS 「 と連携する のサービスIAM」を参照してください。

ス EventBridge ケジューラのアイデンティティベースのポリシー

アイデンティティベースのポリシーのサポート: あり

ID ベースのポリシーは、IAMユーザー、ユーザーのグループ、ロールなどの ID にアタッチできるJSONアクセス許可ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースのポリシーを作成する方法については、「 ユーザーガイド」のIAM「ポリシーの作成IAM」を参照してください。

IAM アイデンティティベースのポリシーでは、許可または拒否されたアクションとリソース、およびアクションが許可または拒否される条件を指定できます。プリンシパルは、それが添付されているユーザーまたはロールに適用されるため、アイデンティティベースのポリシーでは指定できません。JSON ポリシーで使用できるすべての要素については、「 ユーザーガイド」の「 IAMJSONポリシー要素のリファレンスIAM」を参照してください。

ス EventBridge ケジューラのアイデンティティベースのポリシーの例

ス EventBridge ケジューラのアイデンティティベースのポリシーの例を表示するには、「」を参照してくださいス EventBridge ケジューラでのアイデンティティベースのポリシーの使用。

ス EventBridge ケジューラ内のリソースベースのポリシー

リソースベースのポリシーのサポート: なし

リソースベースのポリシーは、リソースにアタッチするJSONポリシードキュメントです。リソースベースのポリシーの例としては、IAMロールの信頼ポリシー や Amazon S3 バケットポリシー などがあります。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスをコントロールできます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS サービス。

クロスアカウントアクセスを有効にするには、リソースベースのポリシーで、アカウント全体または別のアカウントのIAMエンティティをプリンシパルとして指定できます。リソースベースのポリシーにクロスアカウントのプリンシパルを追加しても、信頼関係は半分しか確立されない点に注意してください。プリンシパルとリソースが異なる にある場合 AWS アカウント、信頼されたアカウントのIAM管理者は、プリンシパルエンティティ (ユーザーまたはロール) にリソースへのアクセス許可も付与する必要があります。IAM 管理者は、アイデンティティベースのポリシーをエンティティにアタッチすることで権限を付与します。ただし、リソースベースのポリシーで、同じアカウントのプリ

ンシパルへのアクセス権が付与されている場合は、アイデンティティベースのポリシーをさらに付与する必要はありません。詳細については、「 ユーザーガイド」の「 でのクロスアカウントリソースアクセスIAMIAM」を参照してください。

ス EventBridge ケジューラのポリシーアクション

ポリシーアクションのサポート: あり

管理者はポリシーを使用して AWS JSON、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

JSON ポリシーの `Action`要素は、ポリシーでアクセスを許可または拒否するために使用できるアクションを記述します。ポリシーアクションの名前は通常、関連する AWS APIオペレーションと同じです。一致するAPIオペレーションがないアクセス許可のみのアクションなど、いくつかの例外があります。また、ポリシーに複数のアクションが必要なオペレーションもあります。これらの追加アクションは、依存アクションと呼ばれます。

このアクションは、関連付けられたオペレーションを実行するための権限を付与するポリシーで使用されます。

ス EventBridge ケジューラアクションのリストを確認するには、「サービス認証リファレンス」の「Amazon EventBridge Scheduler で定義されるアクション」を参照してください。

ス EventBridge ケジューラのポリシーアクションは、アクションの前に次のプレフィックスを使用します。

```
scheduler
```

単一のステートメントで複数のアクションを指定するには、アクションをカンマで区切ります。

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

ワイルドカード (*) を使用して複数アクションを指定できます。例えば、`List` という単語で始まるすべてのアクションを指定するには、次のアクションを含めます。

```
"Action": [
```

```
    "scheduler:List*"
]
```

ス EventBridge ケジューラのポリシーリソース

ポリシーリソースのサポート: あり

管理者はポリシーを使用して AWS JSON、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

Policy ResourceJSON要素は、アクションが適用されるオブジェクトを指定します。ステートメントには、Resource または NotResource 要素を含める必要があります。ベストプラクティスとして、Amazon リソースネーム (ARN) を使用してリソースを指定します。これは、リソースレベルの許可と呼ばれる特定のリソースタイプをサポートするアクションに対して実行できます。

オペレーションのリスト化など、リソースレベルの権限をサポートしないアクションの場合は、ステートメントがすべてのリソースに適用されることを示すために、ワイルドカード (*) を使用します。

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

ス EventBridge ケジューラのリソースタイプとその のリストを確認するにはARNs、「サービス認証リファレンス」の「Amazon EventBridge Scheduler で定義されるリソース」を参照してください。各リソースARNの を指定できるアクションについては、「Amazon EventBridge Scheduler で定義されるアクション」を参照してください。

ス EventBridge ケジューラのアイデンティティベースのポリシーの例を表示するには、「」を参照してくださいス EventBridge ケジューラでのアイデンティティベースのポリシーの使用。

ス EventBridge ケジューラのポリシー条件キー

サービス固有のポリシー条件キーのサポート: あり

管理者はポリシーを使用して AWS JSON、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルが、どのリソースに対してどのような条件下でアクションを実行できるかということです。

Condition 要素 (または Condition ブロック) を使用すると、ステートメントが有効な条件を指定できます。Condition 要素はオプションです。イコールや未満などの 条件演算子 を使用して条件式を作成することで、ポリシーの条件とリクエスト内の値を一致させることができます。

1 つのステートメントに複数の Condition 要素を指定する場合、または 1 つの Condition 要素に複数のキーを指定する場合、 AWS では AND 論理演算子を使用してそれらを評価します。1 つの条件キーに複数の値を指定すると、 は論理ORオペレーションを使用して条件 AWS を評価します。ステートメントの権限が付与される前にすべての条件が満たされる必要があります。

条件を指定する際にプレースホルダー変数も使用できます。例えば、リソースにIAMユーザー名でタグ付けされている場合にのみ、リソースへのアクセス許可をIAMユーザーに付与できます。詳細については、「 ユーザーガイド」のIAM「ポリシー要素: 変数とタグIAM」を参照してください。

AWS は、グローバル条件キーとサービス固有の条件キーをサポートします。すべての AWS グローバル条件キーを確認するには、「 ユーザーガイド」のAWS 「 グローバル条件コンテキストキーIAM」を参照してください。

ス EventBridge ケジューラの条件キーのリストを確認するには、「サービス認証リファレンス」の「Amazon EventBridge Scheduler の条件キー」を参照してください。条件キーを使用できるアクションとリソースについては、「Amazon EventBridge Scheduler で定義されるアクション」を参照してください。

ス EventBridge ケジューラのアイデンティティベースのポリシーの例を表示するには、「」を参照してくださいス EventBridge ケジューラでのアイデンティティベースのポリシーの使用。

ACLs ス EventBridge ケジューラの

をサポートACLs： いいえ

アクセスコントロールリスト (ACLs) は、リソースへのアクセス許可を持つプリンシパル (アカウントメンバー、ユーザー、またはロール) を制御します。ACLs はリソースベースのポリシーに似ていますが、JSONポリシードキュメント形式を使用しません。

ABAC ス EventBridge ケジューラを使用する

サポート ABAC (ポリシー内のタグ): 部分的

属性ベースのアクセスコントロール (ABAC) は、属性に基づいてアクセス許可を定義する認可戦略です。では AWS、これらの属性はタグ と呼ばれます。タグは、IAMエンティティ (ユーザーまたはロール) および多くの AWS リソースにアタッチできます。エンティティとリソースのタグ付けは、の最初のステップですABAC。次に、プリンシパルのタグがアクセスしようとしているリソースのタグと一致する場合に、オペレーションを許可するABACポリシーを設計します。

ABAC は、急速に成長している環境や、ポリシー管理が煩雑になる状況に役立ちます。

タグに基づいてアクセスを管理するには、aws:ResourceTag/*key-name*、aws:RequestTag/*key-name*、または aws:TagKeys の条件キーを使用して、ポリシーの条件要素でタグ情報を提供します。

サービスがすべてのリソースタイプに対して 3 つの条件キーすべてをサポートする場合、そのサービスの値はありです。サービスが一部のリソースタイプに対してのみ 3 つの条件キーのすべてをサポートする場合、値は「部分的」になります。

の詳細についてはABAC、「 IAMユーザーガイド」の「 とはABAC」を参照してください。のセットアップ手順を含むチュートリアルを表示するにはABAC、「 ユーザーガイド」の「属性ベースのアクセスコントロール (ABAC) を使用するIAM」を参照してください。

ス EventBridge ケジューラでの一時的な認証情報の使用

一時的な認証情報のサポート: あり

一部の は、一時的な認証情報を使用してサインインすると機能 AWS サービス しません。一時的な認証情報 AWS サービス を使用する などの詳細については、 ユーザーガイドのAWS サービス 「 と連携する IAM IAM 」を参照してください。

ユーザー名とパスワード以外の AWS Management Console 方法で にサインインする場合、一時的な認証情報を使用します。例えば、会社のシングルサインオン (SSO) リンク AWS を使用して にアクセスすると、そのプロセスによって一時的な認証情報が自動的に作成されます。また、ユーザーとしてコンソールにサインインしてからロールを切り替える場合も、一時的な認証情報が自動的に作成されます。ロールの切り替えの詳細については、「 IAMユーザーガイド」の「ロールへの切り替え (コンソール) 」を参照してください。

一時的な認証情報は、 AWS CLI または を使用して手動で作成できます AWS API。その後、これらの一時的な認証情報を使用して . AWS recommends にアクセスできます AWS。これは、長期的なアクセスキーを使用する代わりに、一時的な認証情報を動的に生成することを推奨しています。詳細については、「」の「一時的なセキュリティ認証情報IAM」を参照してください。

ス EventBridge ケジューラのクロスサービスプリンシパル許可

転送アクセスセッションをサポート (FAS): はい

IAM ユーザーまたはロールを使用して でアクションを実行すると AWS、プリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を AWS サービス、ダウンストリームサービス AWS サービス へのリクエストのリクエストと組み合わせて使用

します。FAS リクエストは、サービスが他の AWS サービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するための権限が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。

ス EventBridge ケジューラのサービスロール

サービスロールのサポート: あり

サービスロールは、ユーザーに代わってアクションを実行するためにサービスが引き受ける IAM ロールです。IAM 管理者は、 内からサービスロールを作成、変更、削除できますIAM。詳細については、「 ユーザーガイド」の「 にアクセス許可を委任するロールの作成 AWS サービスIAM」を参照してください。

> **Warning**
>
> サービスロールのアクセス許可を変更すると、ス EventBridge ケジューラの機能が破損する可能性があります。ス EventBridge ケジューラが指示する場合以外は、サービスロールを編集しないでください。

ス EventBridge ケジューラのサービスにリンクされたロール

サービスにリンクされたロールのサポート: なし

サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS サービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスにリンクされたロールのアクセス許可を表示できますが、編集することはできません。

サービスにリンクされたロールの作成または管理の詳細については、「 AWS と連携する のサービスIAM」を参照してください。表の中から、[Service-linked role] (サービスにリンクされたロール) 列に Yes と記載されたサービスを見つけます。サービスリンクロールに関するドキュメントをサービスで表示するには、はい リンクを選択します。

ス EventBridge ケジューラでのアイデンティティベースのポリシーの使用

デフォルトでは、ユーザーとロールにはス EventBridge ケジューラリソースを作成または変更するアクセス許可はありません。また、、 AWS Command Line Interface （ AWS CLI ） AWS Management

Console、または を使用してタスクを実行することはできません AWS API。必要なリソースに対してアクションを実行するアクセス許可をユーザーに付与するために、IAM管理者はIAMポリシーを作成できます。その後、管理者はIAMポリシーをロールに追加し、ユーザーはロールを引き受けることができます。

これらのポリシードキュメント例を使用してIAMアイデンティティベースのJSONポリシーを作成する方法については、「 ユーザーガイド」のIAM「ポリシーの作成IAM」を参照してください。

各リソースタイプの の形式など、スケ EventBridge ジューラで定義されるアクションとリソースタイプの詳細については、「サービス認証リファレンスARNs」の「Amazon EventBridge Scheduler のアクション、リソース、および条件キー」を参照してください。

トピック

- ポリシーのベストプラクティス
- EventBridge スケジューラのアクセス許可
- AWS ス EventBridge ケジューラの マネージドポリシー
- ス EventBridge ケジューラのカスタマー管理ポリシー
- AWS マネージドポリシーの更新

ポリシーのベストプラクティス

ID ベースのポリシーは、ユーザーのアカウントで誰かがス EventBridge ケジューラリソースを作成、アクセス、または削除できるどうかを決定します。これらのアクションを実行すると、 AWS アカウントに料金が発生する可能性があります。アイデンティティベースポリシーを作成したり編集したりする際には、以下のガイドラインと推奨事項に従ってください:

- AWS 管理ポリシーを開始し、最小特権のアクセス許可に移行する – ユーザーとワークロードにアクセス許可を付与するには、多くの一般的なユースケースにアクセス許可を付与する AWS 管理ポリシーを使用します。これらは で使用できます AWS アカウント。ユースケースに固有の AWS カスタマー管理ポリシーを定義して、アクセス許可をさらに減らすことをお勧めします。詳細については、「 ユーザーガイド」の「 AWS 管理ポリシーAWS 」または「 ジョブ機能の 管理ポリシーIAM」を参照してください。
- 最小特権のアクセス許可を適用する – IAMポリシーでアクセス許可を設定する場合は、タスクの実行に必要なアクセス許可のみを付与します。これを行うには、特定の条件下で特定のリソースに対して実行できるアクションを定義します。これは、最小特権アクセス許可とも呼ばれています。IAM を使用してアクセス許可を適用する方法の詳細については、「 ユーザーガイド」の「 のポリシーとアクセス許可IAMIAM」を参照してください。

- IAM ポリシーの条件を使用してアクセスをさらに制限する – ポリシーに条件を追加して、アクションとリソースへのアクセスを制限できます。例えば、ポリシー条件を記述して、すべてのリクエストを を使用して送信する必要があることを指定できますSSL。条件を使用して、 などの特定の を介してサービスアクションが使用される場合に AWS サービス、サービスアクションへのアクセスを許可することもできます AWS CloudFormation。詳細については、「 ユーザーガイド」のIAMJSON「ポリシー要素: 条件IAM」を参照してください。
- IAM Access Analyzer を使用してIAMポリシーを検証し、安全で機能的なアクセス許可を確保する – IAM Access Analyzer は、ポリシーがポリシー言語 (JSON) とIAMベストプラクティスに準拠するように、新規および既存のIAMポリシーを検証します。IAM Access Analyzer には、安全で機能的なポリシーの作成に役立つ 100 を超えるポリシーチェックと実用的な推奨事項が用意されています。詳細については、「 ユーザーガイド」のIAM「Access Analyzer ポリシーの検証IAM」を参照してください。
- 多要素認証を要求する (MFA) – でIAMユーザーまたはルートユーザーを必要とするシナリオがある場合は AWS アカウント、セキュリティを強化MFAするために をオンにします。API オペレーションが呼び出されるMFAタイミングを要求するには、ポリシーにMFA条件を追加します。詳細については、「 IAMユーザーガイド」のMFA「 で保護されたAPIアクセスの設定」を参照してください。

のベストプラクティスの詳細についてはIAM、「 ユーザーガイド」の「 のセキュリティのベストプラクティスIAMIAM」を参照してください。

EventBridge スケジューラのアクセス許可

IAM プリンシパル (ユーザー、グループ、またはロール) がス EventBridge ケジューラでスケジュールを作成し、コンソールまたは を介してスケ EventBridge ジューラリソースにアクセスするには API、プリンシパルに一連のアクセス許可がアクセス許可ポリシーに追加されている必要があります。これらの権限は、プリンシパルの職務に応じて設定できます。例えば、ス EventBridge ケジューラコンソールのみを使用して既存のスケジュールのリストを表示するユーザーまたはロールには、`CreateSchedule` API オペレーションを呼び出すために必要なアクセス許可は必要ありません。アイデンティティベースの権限を調整して、最も権限の低いアクセス権のみを提供することをおすすめします。

次のリストは、ス EventBridge ケジューラのリソースと、それに対応するサポートされているアクションを示しています。

- スケジュール
  - `scheduler:ListSchedules`

  - scheduler:GetSchedule
  - scheduler:CreateSchedule
  - scheduler:UpdateSchedule
  - scheduler:DeleteSchedule
- スケジュールグループ
  - scheduler:ListScheduleGroups
  - scheduler:GetScheduleGroup
  - scheduler:CreateScheduleGroup
  - scheduler:DeleteScheduleGroup
  - scheduler:ListTagsForResource
  - scheduler:TagResource
  - scheduler:UntagResource

ス EventBridge ケジューラのアクセス許可を使用して、ス EventBridge ケジューラで使用する独自のカスタマー管理ポリシーを作成できます。次のセクションで説明する AWS マネージドポリシーを使用して、独自のポリシーを管理することなく、一般的なユースケースに必要なアクセス許可を付与することもできます。

AWS ス EventBridge ケジューラの マネージドポリシー

AWS は、 が AWS 作成および管理するスタンドアロンIAMポリシーを提供することで、多くの一般的なユースケースに対処します。管理ポリシー、つまり事前定義ポリシーは、一般的ユースケースに必要なアクセス許可を付与するため、どの許可が必要なのかをユーザーが調査する必要はありません。詳細については、「 ユーザーガイド」の「 AWS 管理ポリシーIAM」を参照してください。アカウントのユーザーにアタッチできる以下の AWS 管理ポリシーは、スケ EventBridge ジューラに固有です。

- the section called “AmazonEventBridgeSchedulerFullAccess” – コンソールと を使用してス EventBridge ケジューラへのフルアクセスを許可しますAPI。
- the section called “AmazonEventBridgeSchedulerReadOnlyAccess” – ス EventBridge ケジューラへの読み取り専用アクセスを許可します。

AmazonEventBridgeSchedulerFullAccess

AmazonEventBridgeSchedulerFullAccess 管理ポリシーは、スケジュールおよびスケジュールグループに対してすべてのス EventBridge ケジューラアクションを使用するアクセス許可を付与します。

```
{
    "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 マネージドポリシーは、スケジュールとスケジュールグループに関する詳細を表示する読み取り専用の権限を付与します。

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

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

ス EventBridge ケジューラのカスタマー管理ポリシー

次の例を使用して、スケ EventBridge ジューラ用の独自のカスタマー管理ポリシーを作成します。カスタマー管理ポリシーを使用すると、プリンシパルの職務に応じて、チーム内のアプリケーションとユーザーに必要なアクションとリソースのみにアクセス許可を付与できます。

トピック

- 例: CreateSchedule
- 例: GetSchedule
- 例: UpdateSchedule
- 例: DeleteScheduleGroup

例: **CreateSchedule**

新しいスケジュールを作成するときは、 またはカスタマーマネージドキー を使用してAWS 所有のキース EventBridge ケジューラでデータを暗号化するかどうかを選択します。

次のポリシーでは、プリンシパルはスケジュールを作成し、 AWS 所有のキーを使用して暗号化を適用できます。では AWS 所有のキー、 が AWS Key Management Service （ AWS KMS) のリソース AWS を管理するため、 を操作するための追加のアクセス許可は必要ありません 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"
                }
            }
        }
    ]
}
```

次のポリシーを使用して、プリンシパルがスケジュールを作成し、 AWS KMS カスタマーマネージドキーを暗号化に使用できるようにします。カスタマーマネージドキーを使用するには、プリンシパルにアカウントの AWS KMS リソースへのアクセス許可が必要です。このポリシーは、スケ EventBridge ジューラ上のデータの暗号化に使用する 1 つの指定されたKMSキーへのアクセスを許可します。または、ワイルドカード (*) 文字を使用して、アカウント内のすべてのキー、または特定の名前パターンに一致するサブセットへのアクセスを許可することもできます。

```
{
    "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"
                }
            }
        }
    ]
}
```

例: **GetSchedule**

以下のポリシーを使用して、プリンシパルがスケジュールに関する情報を取得できるようにします。

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

```
            "Resource":
            [
                "arn:aws:scheduler:us-west-2:123456789012:schedule/my-group/my-
schedule-name"
            ]
        }
    ]
}
```

例: **UpdateSchedule**

以下のポリシーを使用して、プリンシパルが scheduler:UpdateSchedule アクションを呼び出してスケジュールを更新できるようにします。と同様にCreateSchedule、ポリシーは、スケジュールが暗号化に AWS KMS AWS 所有のキー を使用するか、カスタマーマネージドキーを使用するかによって異なります。で設定されたスケジュールの場合は AWS 所有のキー、次のポリシーを使用します。

```
{
    "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"
                }
            }
        }
```

```
    ]
}
```

カスタマーマネージドキーを使用して設定されたスケジュールの場合は、以下のポリシーを使用します。このポリシーには、プリンシパルがアカウントの AWS KMS リソースにアクセスできるようにする追加のアクセス許可が含まれています。

```
{
    "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"
                }
            }
        }
    ]
}
```

例: **DeleteScheduleGroup**

以下のポリシーを使用して、プリンシパルがスケジュールグループを削除できるようにします。グループを削除すると、そのグループに関連付けられているスケジュールも削除されます。グループを削除するプリンシパルには、そのグループに関連付けられているスケジュールも削除する権限が必要です。このポリシーは、指定されたスケジュールグループとグループ内のすべてのスケジュールに対して scheduler:DeleteScheduleGroup アクションを呼び出す権限をプリンシパルに付与します。

> **Note**
>
> EventBridge スケジューラは、個々のスケジュールのリソースレベルのアクセス許可の指定をサポートしていません。例えば、次の記述は無効であり、ポリシーには含めないでください。
> "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 マネージドポリシーの更新

変更	説明	日付
the section called “AmazonEventBridgeSchedulerFullAccess” – 新しいマネージドポリシー	EventBridge スケジューラは、スケジュールやスケジュールグループを含むすべてのリソースへのフルアクセスをユーザーに付与する新しいマネージドポリシーのサポートを追加します。	2022 年 11 月 10 日
the section called “AmazonEventBridgeSchedulerReadOnlyAccess” – 新しいマネージドポリシー	EventBridge スケジューラは、スケジュールやスケジュールグループを含むすべてのリソースへの読み取り専用アクセスをユーザーに許可する新しい マネージドポリシーのサポートを追加します。	2022 年 11 月 10 日
EventBridge スケジューラが変更の追跡を開始しました	EventBridge スケジューラが AWS マネージドポリシーの変更の追跡を開始しました。	2022 年 11 月 10 日

ス EventBridge ケジューラでの混乱した代理の防止

混乱した代理問題は、アクションを実行するためのアクセス許可を持たないエンティティが、より特権のあるエンティティにアクションの実行を強制できてしまう場合に生じる、セキュリティ上の問題です。では AWS、サービス間のなりすましにより、混乱した代理問題が発生する可能性があります。サービス間でのなりすましは、1 つのサービス (呼び出し元サービス) が、別のサービス (呼び出し対象サービス) を呼び出すときに発生する可能性があります。呼び出し元サービスは、本来ならアクセスすることが許可されるべきではない方法でその許可を使用して、別のお客様のリソースに対する処理を実行するように操作される場合があります。これを防ぐため、 AWS では、アカウント内のリソースへのアクセス権が付与されたサービスプリンシパルですべてのサービスのデータを保護するために役立つツールを提供しています。

スケジュール実行ロールで aws:SourceArnおよび aws:SourceAccount グローバル条件コンテキストキーを使用して、ス EventBridge ケジューラが別のサービスに付与するリソースへのアクセス許可を制限することをお勧めします。クロスサービスアクセスにリソースを 1 つだけ関連付けたい場合は、aws:SourceArn を使用します。そのアカウント内のリソースをクロスサービスの使用に関連付けることを許可する場合は、aws:SourceAccount を使用します。

混乱した代理問題から保護する最も効果的な方法は、リソースARNがいっぱいになった aws:SourceArn グローバル条件コンテキストキーを使用することです。以下の条件は個々のスケジュールグループに限定されます: arn:aws:scheduler:*:*123456789012*:schedule-group/*your-schedule-group*

リソースARNの全体がわからない場合や、複数のリソースを指定する場合は、 の不明な部分にワイルドカード文字 (*) を含むaws:SourceArnグローバルコンテキスト条件キーを使用しますARN。例: arn:aws:scheduler:*:*123456789012*:schedule-group/*。

の値は、ARNこの条件の範囲を設定するス EventBridge ケジューラのスケジュールグループaws:SourceArnである必要があります。

> **Important**
>
> aws:SourceArn ステートメントの範囲を特定のスケジュールやスケジュール名のプレフィックスに限定しないでください。ARN 指定する はスケジュールグループである必要があります。

次の例では、aws:SourceArn および aws:SourceAccount グローバル条件コンテキストキーを実行ロールの信頼ポリシーで使用して、混乱した代理問題を回避する方法を示します。

```
{
    "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 Scheduler のアイデンティティとアクセスのトラブルシューティング

次の情報は、スケ EventBridge ジューラと の使用時に発生する可能性がある一般的な問題の診断と修正に役立ちますIAM。

トピック

- ス EventBridge ケジューラでアクションを実行する権限がない
- iam を実行する権限がありません。PassRole
- 自分の 以外のユーザーにス EventBridge ケジューラリソース AWS アカウント へのアクセスを許可したい

ス EventBridge ケジューラでアクションを実行する権限がない

「I am not authorized to perform an action in Amazon Bedrock」というエラーが表示された場合、そのアクションを実行できるようにポリシーを更新する必要があります。

次の例のエラーは、mateojacksonIAMユーザーが コンソールを使用して架空の*my-example-widget*リソースの詳細を表示しようとしているが、架空のscheduler:*GetWidget*アクセス許可がない場合に発生します。

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

この場合、Mateo のポリシーでは、*my-example-widget* アクションを使用して scheduler:*GetWidget* リソースへのアクセスを許可するように更新する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン認証情報を提供した担当者が管理者です。

iam を実行する権限がありません。PassRole

iam:PassRole アクションを実行する権限がないというエラーが表示された場合は、ポリシーを更新してスケ EventBridge ジューラにロールを渡すことができるようにする必要があります。

一部の AWS サービス では、新しいサービスロールまたはサービスにリンクされたロールを作成する代わりに、そのサービスに既存のロールを渡すことができます。そのためには、サービスにロールを渡す権限が必要です。

次の例のエラーは、 というIAMユーザーがコンソールを使用してmarymajorス EventBridge ケジューラでアクションを実行しようする場合に発生します。ただし、このアクションをサービスが実行するには、サービスロールから付与された権限が必要です。メアリーには、ロールをサービスに渡す許可がありません。

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

この場合、Mary のポリシーを更新してメアリーに iam:PassRole アクションの実行を許可する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン認証情報を提供した担当者が管理者です。

自分の 以外のユーザーにス EventBridge ケジューラリソース AWS アカウント へのアクセスを許可したい

他のアカウントのユーザーや組織外の人が、リソースにアクセスするために使用できるロールを作成できます。ロールの引き受けを委託するユーザーを指定できます。リソースベースのポリシーまたは

アクセスコントロールリスト (ACLs) をサポートするサービスでは、これらのポリシーを使用して、ユーザーにリソースへのアクセスを許可できます。

詳細については、以下を参照してください。

- ス EventBridge ケジューラがこれらの機能をサポートしているかどうかを確認するには、「」を参照してくださいス EventBridge ケジューラと の連携方法 IAM。
- 所有している のリソースへのアクセスを提供する方法については、 AWS アカウント 「 ユーザーガイド」の「所有 AWS アカウント している別の のIAMユーザーへのアクセスを提供するIAM」を参照してください。
- リソースへのアクセスをサードパーティー に提供する方法については AWS アカウント、「 ユーザーガイド」の「サードパーティー AWS アカウント が所有する へのアクセスを提供するIAM」を参照してください。
- ID フェデレーションを通じてアクセスを提供する方法については、 IAMユーザーガイドの「外部認証されたユーザーへのアクセスの提供 (ID フェデレーション) 」を参照してください。
- クロスアカウントアクセスでのロールとリソースベースのポリシーの使用の違いについては、ユーザーガイドの「 でのクロスアカウントリソースアクセスIAMIAM」を参照してください。

Amazon EventBridge Scheduler でのデータ保護

責任 AWS 共有モデル、Amazon EventBridge Scheduler でのデータ保護に適用されます。このモデルで説明されているように、 AWS はすべての を実行するグローバルインフラストラクチャを保護する責任があります AWS クラウド。お客様は、このインフラストラクチャでホストされているコンテンツに対する管理を維持する責任があります。また、使用する AWS サービス のセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーの詳細については、「データプライバシーFAQ」を参照してください。欧州におけるデータ保護の詳細については、AWS 「 セキュリティブログ」のAWS 「責任共有モデル」とGDPRブログ記事を参照してください。

データ保護の目的で、認証情報を保護し AWS アカウント 、 AWS IAM Identity Center または AWS Identity and Access Management () を使用して個々のユーザーを設定することをお勧めしますIAM。この方法により、それぞれのジョブを遂行するために必要な権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。1TLS.2 が必要で、1.3 TLS をお勧めします。
- を使用して APIとユーザーアクティビティのログ記録を設定します AWS CloudTrail。

- AWS 暗号化ソリューションと、 内のすべてのデフォルトのセキュリティコントロールを使用します AWS サービス。
- Amazon Macie などの高度なマネージドセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。
- コマンドラインインターフェイスまたは AWS を介して にアクセスするときに FIPS 140-3 検証済みの暗号化モジュールが必要な場合はAPI、FIPSエンドポイントを使用します。利用可能なFIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-3」を参照してください。

お客様の E メールアドレスなどの極秘または機密情報は、タグ、または名前フィールドなどの自由形式のテキストフィールドに配置しないことを強くお勧めします。これは、コンソール、、またはを使用してス EventBridge ケジューラまたは他の AWS サービス を使用する場合API AWS CLIも同様です AWS SDKs。名前に使用する自由記述のテキストフィールドやタグに入力したデータは、課金や診断ログに使用される場合があります。URL を外部サーバーに提供する場合は、そのサーバーへのリクエストを検証URLするために認証情報を に含めないことを強くお勧めします。

ス EventBridge ケジューラでの保管時の暗号化

このセクションでは、Amazon EventBridge Scheduler が保管中のデータを暗号化および復号する方法について説明します。保管中のデータは、ス EventBridge ケジューラとサービスの基盤となるコンポーネントに保存されるデータです。 EventBridge スケジューラは AWS Key Management Service （ AWS KMS) と統合して、 を使用してデータを暗号化および復号化しますAWS KMS key。EventBridge スケジューラは、、AWS 所有のキーカスタマーマネージドKMSキーの 2 種類のキーをサポートしています。 https://docs.aws.amazon.com/customer managed KMS key

> **Note**
>
> EventBridge スケジューラは、対称暗号化KMSキーの使用のみをサポートします。

AWS 所有のキー は、 AWS サービスが複数の AWS アカウントで使用するために所有および管理するKMSキーです。ス AWS 所有のキー EventBridge ケジューラが使用する は AWS アカウントに保存されませんが、ス EventBridge ケジューラはそれらを使用してデータとリソースを保護します。デフォルトでは、ス EventBridge ケジューラは AWS 所有キーを使用してすべてのデータを暗号化および復号します。自分の AWS 所有のキー またはアクセスポリシーを管理する必要はありません。ス EventBridge ケジューラが AWS 所有のキー を使用してデータを保護する場合、料金は発生しません。また、その使用量はアカウントの AWS KMS クォータの一部としてカウントされません。

カスタマーマネージドキーは、ユーザーが作成、所有、管理する AWS アカウントに保存されているKMSキーです。特定のユースケースで、スケ EventBridge ジューラ上のデータを保護する暗号化キーを制御および監査する必要がある場合は、カスタマーマネージドキーを使用できます。カスタマーマネージドキーを選択した場合、キーポリシーを管理する必要があります。カスタマーマネージドキーの使用には、月額料金と、無料利用枠を超えた使用に対する料金がかかります。カスタマーマネージドキーの使用も AWS KMS クォータの一部としてカウントされます。料金の詳細については、「AWS Key Management Service の料金」を参照してください。

トピック

- 暗号化アーティファクト
- KMS キーの管理
- CloudTrail イベントの例

暗号化アーティファクト

次の表は、ス EventBridge ケジューラが保管時に暗号化するさまざまなタイプのデータと、各カテゴリでサポートされるKMSキーのタイプを示しています。

データ型	説明	AWS 所有のキー	カスタマーマネージドキー
ペイロード (最大 256 KB)	スケジュールをターゲットに配信するように設定するときに、スケジュールの `TargetInput` パラメータで指定するデータ。	サポート	サポート
識別子と状態	スケジュールの固有の名前と状態 (有効、無効)。	サポート	サポートされていません
スケジューリング設定	繰り返しのスケジュールの場合は rate 式や cron 式などのスケジューリング式、1	サポート	サポートされていません

データ型	説明	AWS 所有のキー	カスタマーマネージドキー
	回限りの呼び出しの場合はタイムスタンプ、スケジュールの開始日、終了日、タイムゾーン。		
ターゲット設定	ターゲットの Amazon リソースネーム (ARN) 、およびその他のターゲット関連の設定の詳細。	サポート	サポートされていません
呼び出しと障害動作の設定	柔軟な時間枠設定、スケジュールの再試行ポリシー、配信失敗時に使用するデッドレターキューの詳細。	サポート	サポートされていません

EventBridge スケジューラは、前の表で説明したように、ターゲットペイロードを暗号化および復号化するときにのみカスタマーマネージドキーを使用します。カスタマーマネージドキーを使用することを選択した場合、ス EventBridge ケジューラはペイロードを 2 回暗号化および復号します。1 回はデフォルトの を使用し AWS 所有のキー、もう 1 回は指定したカスタマーマネージドキーを使用します。他のすべてのデータ型では、ス EventBridge ケジューラは保管中のデータ AWS 所有のキーを保護するためにデフォルトのみを使用します。

次のthe section called “KMS キーの管理”セクションでは、ス EventBridge ケジューラでカスタマーマネージドキーを使用するには、IAMリソースとキーポリシーを管理する方法を説明します。

KMS キーの管理

オプションで、スケジュールがターゲットに配信するペイロードを暗号化および復号化するためのカスタマーマネージドキーを指定できます。ス EventBridge ケジューラは、最大 256KB のデータをペイロードを暗号化および復号化します。カスタマーマネージドキーの使用には、月額料金と、無料利用枠を超えた使用に対する料金がかかります。カスタマーマネージドキーの使用は AWS KMS

クォータの一部としてカウントされます。料金の詳細については、「AWS Key Management Service の料金」を参照してください。

EventBridge スケジューラは、スケジュールを作成するプリンシパルに関連付けられたIAMアクセス許可を使用してデータを暗号化します。つまり、ス EventBridge ケジューラ を呼び出すユーザーまたはロールに必要な AWS KMS 関連アクセス許可をアタッチする必要がありますAPI。さらに、EventBridge Scheduler はリソースベースのポリシーを使用してデータを復号します。つまり、スケジュールに関連付けられた実行ロールには、データを復号するときに を呼び AWS KMS API出すために必要な AWS KMS 関連アクセス許可も必要です。

> **Note**
>
> EventBridge スケジューラは、一時的なアクセス許可の許可の使用をサポートしていません。

次のセクションでは、 AWS KMS キーポリシーを管理する方法と、スケ EventBridge ジューラでカスタマーマネージドキーを使用するために必要なIAMアクセス許可について説明します。

トピック

- アクセスIAM許可を追加する
- キーポリシーの管理

アクセスIAM許可を追加する

カスタマーマネージドキーを使用するには、スケジュールを作成するアイデンティティベースのIAMプリンシパルと、スケジュールに関連付ける実行ロールに次のアクセス許可を追加する必要があります。

カスタマーマネージドキーに対するアイデンティティベースのアクセス許可

スケジュールの作成API時にス EventBridge ケジューラを呼び出すプリンシパル (ユーザー、グループ、またはロール) に関連付けられたアクセス許可ポリシーに、次の AWS KMS アクションを追加する必要があります。

```
{
    "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** – 指定したキーが対称暗号化KMSキーであることを検証するために必要です。
- **kms:GenerateDataKey** – スケ EventBridge ジューラがクライアント側の暗号化を実行するために使用するデータキーを生成するために必要です。
- **kms:Decrypt** – ス EventBridge ケジューラが暗号化されたデータとともに保存する暗号化されたデータキーを復号化する必要があります。

カスタマーマネージドキーの実行ロール権限

スケジュールの実行ロールのアクセス許可ポリシーに次のアクションを追加して、データを復号するときに を呼び AWS KMS API出すス EventBridge ケジューラへのアクセスを許可する必要があります。

```
{
    "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`** – ス EventBridge ケジューラが暗号化されたデータとともに保存する暗号化されたデータキーを復号化する必要があります。

新しいスケジュールを作成するときにス EventBridge ケジューラコンソールを使用して新しい実行ロールを作成する場合、ス EventBridge ケジューラは必要なアクセス許可を実行ロールに自動的にアタッチします。ただし、既存の実行ロールを選択した場合、カスタマーマネージドキーを使用できるようにするには、必要な権限をロールに追加する必要があります。

キーポリシーの管理

を使用してカスタマーマネージドキーを作成する場合 AWS KMS、デフォルトでは、キーにはスケジュールの実行ロールへのアクセスを提供する次のキーポリシーがあります。

```
{
    "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": "*"
        }
    ]
}
```

オプションで、キーポリシーの範囲を実行ロールへのアクセスのみに限定することもできます。これは、ス EventBridge ケジューラリソースでのみカスタマーマネージドキーを使用する場合に実行できます。次のキーポリシーの例を使用して、キーを使用できるス EventBridge ケジューラリソースを制限します。

```
{
    "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 イベントの例

AWS CloudTrail は、すべてのAPI呼び出しイベントをキャプチャします。これには、スEventBridge ケジューラがカスタマーマネージドキーを使用してデータを復号するたびにAPI呼び出されます。次の例は、カスタマーマネージドキーを使用した kms:Decryptアクションを使用したスEventBridge ケジューラを示す CloudTrail イベントエントリを示しています。

```
{
  "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"
  }
}
```

スケ EventBridge ジューラで転送中の暗号化

EventBridge スケジューラは、ネットワークを移動するときに転送中のデータを暗号化します。Transport Layer Security (TLS) は、ス EventBridge ケジューラAPIオペレーションを呼び出すとき、およびス EventBridge ケジューラがスケジュールを呼び出すときにターゲットを呼び出すAPIsときに、データを暗号化します。デフォルトでは、ス EventBridge ケジューラTLSは転送中のデータを暗号化するときに 1.2 を使用します。転送中の暗号化を設定する必要はなく、ス EventBridge ケジューラの使用時に別のTLSバージョンを選択することはできません。

ス EventBridge ケジューラの使用 API – などのAPIオペレーションを実行すると`CreateSchedule`、ス EventBridge ケジューラはHTTPリクエスト本文やヘッダーを含むリクエスト全体を暗号化します。 EventBridge また、スケジューラは、 から受信したレスポンスオブジェクト全体を暗号化しますAPIs。

ターゲットの使用 APIs – ス EventBridge ケジューラはスケジュールを呼び出すときに、スケジュールの作成時にAPI指定したターゲットを呼び出します。イベントをターゲットに配信すると、ス EventBridge ケジューラはリクエスト本文とすべてのヘッダーを含むリクエスト全体と、ターゲットから受信したレスポンスを暗号化します。

Amazon EventBridge Scheduler のコンプライアンス検証

AWS サービス が特定のコンプライアンスプログラムの範囲内にあるかどうかを確認するには、コンプライアンスプログラムAWS サービス による対象範囲内のコンプライアンスプログラムを参照し、関心のあるコンプライアンスプログラムを選択します。一般的な情報については、AWS「コンプライアンスプログラム」を参照してください。

を使用して、サードパーティーの監査レポートをダウンロードできます AWS Artifact。詳細については、「 でのレポートのダウンロード AWS Artifact」の」を参照してください。

を使用する際のお客様のコンプライアンス責任 AWS サービス は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 は、コンプライアンスに役立つ以下のリソース AWS を提供しています。

- セキュリティとコンプライアンスのクイックスタートガイド – これらのデプロイガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスに重点を置いたベースライン環境 AWS を にデプロイする手順について説明します。
- アマゾン ウェブ サービスHIPAAのセキュリティとコンプライアンスのためのアーキテクチャ – このホワイトペーパーでは、企業が AWS を使用して HIPAA対象アプリケーションを作成する方法について説明します。

> **Note**
>
> すべての AWS サービス がHIPAA対象となるわけではありません。詳細については、HIPAA「対象サービスリファレンス」を参照してください。

- AWS コンプライアンスリソース – このワークブックとガイドのコレクションは、お客様の業界や地域に適用される場合があります。
- AWS カスタマーコンプライアンスガイド — コンプライアンスの観点から責任共有モデルを理解します。このガイドでは、ガイダンスを保護し AWS サービス 、複数のフレームワーク (米国国立標準技術研究所 (NIST) 、Payment Card Industry Security Standards Council ()、PCI国際標準化機構 (ISO) など) のセキュリティコントロールにマッピングするためのベストプラクティスをまとめています。
- 「 デベロッパーガイド」の「 ルールによるリソースの評価」 – この AWS Config サービスは、リソース設定が社内プラクティス、業界ガイドライン、および規制にどの程度準拠しているかを評価します。 AWS Config
- AWS Security Hub – これにより AWS サービス 、 内のセキュリティ状態を包括的に確認できます AWS。Security Hub では、セキュリティコントロールを使用して AWS リソースを評価し、セキュリティ業界標準とベストプラクティスに対するコンプライアンスをチェックします。サポートされているサービスとコントロールのリストについては、Security Hub のコントロールリファレンスを参照してください。
- Amazon GuardDuty – これにより AWS アカウント、疑わしいアクティビティや悪意のあるアクティビティがないか環境を監視することで、、、ワークロード、コンテナ、データに対する潜在的な脅威 AWS サービス を検出します。 GuardDuty は、特定のコンプライアンスフレームワークで義務付けられている侵入検知要件を満たすことでDSS、 PCI などのさまざまなコンプライアンス要件に対応するのに役立ちます。

- AWS Audit Manager – これにより AWS サービス 、 AWS 使用状況を継続的に監査し、リスクの管理方法と規制や業界標準への準拠を簡素化できます。

Amazon EventBridge Scheduler の耐障害性

AWS グローバルインフラストラクチャは、 AWS リージョン およびアベイラビリティーゾーンを中心に構築されています。物理的に分離および分離された複数のアベイラビリティーゾーン AWS リージョン を提供し、低レイテンシー、高スループット、高冗長ネットワークで接続されます。アベイラビリティーゾーンでは、ゾーン間で中断することなく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。アベイラビリティーゾーンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性が高く、フォールトトレラントで、スケーラブルです。

AWS リージョン およびアベイラビリティーゾーンの詳細については、AWS 「 グローバルインフラストラクチャ」を参照してください。

ス EventBridge ケジューラには、 AWS グローバルインフラストラクチャに加えて、データの耐障害性とバックアップのニーズをサポートするのに役立ついくつかの機能が用意されています。

Amazon EventBridge Scheduler のインフラストラクチャセキュリティ

マネージドサービスである Amazon EventBridge Scheduler は AWS グローバルネットワークセキュリティで保護されています。 AWS セキュリティサービスと がインフラストラクチャ AWS を保護する方法については、AWS 「 クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティのベストプラクティスを使用して AWS 環境を設計するには、「セキュリティの柱 AWS Well-Architected Framework」の「Infrastructure Protection」を参照してください。

が AWS 公開したAPI呼び出しを使用して、ネットワーク経由でス EventBridge ケジューラにアクセスします。クライアントは以下をサポートする必要があります:

- Transport Layer Security (TLS) 。1TLS.2 が必要で、1.3 TLS をお勧めします。
- (Ephemeral Diffie-HellmanPFS) や DHE (Elliptic Curve Ephemeral Diffie-Hellman) などの完全前方秘匿性 ECDHE () を備えた暗号スイート。これらのモードは、Java 7 以降など、ほとんどの最新システムでサポートされています。

さらに、リクエストは、 IAMプリンシパルに関連付けられたアクセスキー ID とシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token Service (AWS STS) を使用して、一時セキュリティ認証情報を生成し、リクエストに署名することもできます。

Amazon EventBridge Scheduler のモニタリングとメトリクス

モニタリングは、Amazon EventBridge Scheduler およびその他の AWS ソリューションの信頼性、可用性、パフォーマンスを維持する上で重要な部分です。 AWS には、ス EventBridge ケジューラを監視し、問題が発生した場合に報告し、必要に応じて自動アクションを実行するための以下のモニタリングツールが用意されています。

- Amazon CloudWatch は、 AWS リソースと、 で実行しているアプリケーションを AWS リアルタイムでモニタリングします。メトリクスの収集と追跡、カスタマイズしたダッシュボードの作成、および指定したメトリクスが指定したしきい値に達したときに通知またはアクションを実行するアラームの設定を行うことができます。詳細については、「Amazon ユーザーガイド CloudWatch 」を参照してください。
- AWS CloudTrail は、アカウントによって、または AWS アカウントに代わって行われたAPI呼び出しおよび関連イベントをキャプチャし、指定した Amazon S3 バケットにログファイルを配信します。を呼び出したユーザーとアカウント AWS、呼び出し元のソース IP アドレス、呼び出しが発生した日時を特定できます。詳細については、AWS CloudTrail ユーザーガイドをご参照ください。

トピック

- Amazon による Amazon ス EventBridge ケジューラのモニタリング CloudWatch
- を使用した Amazon EventBridge Scheduler API呼び出しの口グ記録 AWS CloudTrail

Amazon による Amazon ス EventBridge ケジューラのモニタリング CloudWatch

を使用して Amazon EventBridge Scheduler をモニタリングできます。これにより CloudWatch、raw データを収集し、読み取り可能なほぼリアルタイムのメトリクスに加工できます。 EventBridge Scheduler は、すべてのスケジュールのメトリクスのセットと、デッドレターキュー () が関連付けられているスケジュールのメトリクスの追加セットを発行しますDLQ。スケジュールに を設定するとDLQ、スケジュールが再試行ポリシーを使い果たすと、ス EventBridge ケジューラは追加のメトリクスを発行します。

これらの統計は 15 か月間保持されるため、履歴情報にアクセスして、スケジュールが失敗する理由をより的確に把握し、根本的な問題のトラブルシューティングを行うことができます。また、特定のしきい値をモニタリングするアラームを設定し、しきい値に達したときに通知を送信したりアクションを実行したりできます。詳細については、「Amazon ユーザーガイド CloudWatch 」を参照してください。

トピック

用語

名前空間

名前空間は、 AWS サービスの CloudWatch メトリクスのコンテナです。ス EventBridge ケジューラの場合、名前空間は ですAWS/Scheduler。

CloudWatch メトリクス

CloudWatch メトリクスは、 に固有の時系列のデータポイントのセットを表します CloudWatch。

ディメンション

ディメンションは、メトリクスのアイデンティティの一部である名前と値のペアです。

単位

統計には、測定単位があります。ス EventBridge ケジューラの場合、単位にはカウント が含まれます。

ディメンション

このセクションでは、 のス EventBridge ケジューラメトリクスの CloudWatch ディメンショングループについて説明します CloudWatch。

ディメンション	説明
ScheduleGroup	を使用してメトリクスを表示するスケジュールのグループ CloudWatch。グループをまだ作成していない場合、 EventBridge Scheduler はスケジュールを`default`グループに関連付けます。

メトリクスへのアクセス

このセクションでは、特定のス EventBridge ケジューラスケジュール CloudWatch の でパフォーマンスメトリクスにアクセスする方法について説明します。

ディメンションのパフォーマンスメトリクスを表示する方法

1. CloudWatch コンソールでメトリクスページを開きます。
2. AWS リージョンセレクタを使用して、スケジュールのリージョンを選択します。
3. [スケジューラ] 名前空間を選択します。
4. [すべてのメトリクス]タブで、[スケジュールグループメトリクス] などのディメンションを選択します。選択したリージョンで作成したすべてのスケジュールのメトリクスを表示するには、[アカウントメトリクス] を選択します。
5. ディメンションの CloudWatch メトリクスを選択します。例えば、 InvocationAttemptCountまたは の場合はInvocationDroppedCount、グラフ検索 を選択します。
6. グラフ化されたメトリクスタブを選択すると、ス EventBridge ケジューラメトリクスのパフォーマンス統計が表示されます。

メトリクスの一覧

次の表に、すべてのス EventBridge ケジューラスケジュールのメトリクスと、 を設定したスケジュールの追加メトリクスを示しますDLQ。

すべてのスケジュールのメトリクス

名前空間	メトリクス	単位	説明
AWS/Scheduler	InvocationAttemptCount	カウント	呼び出しを試みるたびに発生します。このメトリクスを使用して、ス EventBridge ケジューラがスケジュールを呼び出そうとしていることを確認し、呼び出しがアカウントクォータに近づくタイミングを確認します。
AWS/Scheduler	TargetErrorCount	カウント	EventBridge スケジューラがターゲット を呼び出した後にターゲットが例外を返したときに発行されますAPI。これを使用して、ターゲットへの配信がいつ失敗したかを確認します。
AWS/Scheduler	TargetErrorThrottledCount	カウント	ターゲットによるAPIスロットリングが原因でターゲットの呼び出しが失敗した場合に発生します。これを使用して、ス EventBridge ケジューラによって行われたターゲットAPIスロットリング呼び出しが

名前空間	メトリクス	単位	説明
			根本的な理由である場合に配信の失敗を診断します。
AWS/Scheduler	InvocationThrottleCount	カウント	ス EventBridge ケジューラによって設定されたサービスクォータを超えているため、ス EventBridge ケジューラがターゲット呼び出しをスロットリングするときに発生します。これを使用して、呼び出しスロットリング制限のクォータをいつ超過したかを判断します。サービスクォータの詳細については、「クォータ」を参照してください。

名前空間	メトリクス	単位	説明
AWS/Scheduler	InvocationDroppedCount	カウント	スケジュールの再試行ポリシーを使い果たした後、スEventBridge ケジューラがターゲットの呼び出しを停止したときに発行されます。再試行ポリシーの詳細については、スEventBridge ケジューラAPIリファレンスRetryPolicyの「」を参照してください。

を使用したスケジュールのメトリクス DLQ

名前空間	メトリクス	単位	説明
AWS/Scheduler	InvocationsSentToDeadLetterCount	カウント	スケジュールの への正常な配信ごとに発行されますDLQ。これを使用して、イベントが に送信されるタイミングを判断しDLQ、スケジュールのに配信されたイベントDLQをチェックして、障害の原

名前空間	メトリクス	単位	説明
			因を特定するのに役立つ追加の詳細を確認します。

名前空間	メトリクス	単位	説明
AWS/Scheduler	InvocationsFailedToBeSentToDeadLetterCount	カウント	ス EventBridge ケジューラが にイベントを配信できない場合に発行されます DLQ。これら 2 つのメトリクスを使用して、ス EventBridge ケジューラがにイベントを送信できない理由を特定し DLQ、問題を解決するためにDLQ設定を変更します。 として指定した Amazon SQSキュー DLQが存在しない場合の InvocationsFailedToBeSentToDeadLetterCount_<error_code> メトリクスの例を次に示します。Invocatio
AWS/Scheduler	InvocationsFailedToBeSentToDeadLetterCount_<error_code>	カウント	

名前空間	メトリクス	単位	説明
			nsFailedToBeSentToDeadLetterCount_ **AWS.SimpleQueueService.NonExistentQueue**
AWS/Scheduler	InvocationsSentToDeadLetterCount_Truncated_MessageSizeExceeded	カウント	に送信されたイベントのペイロードがAmazon で許可されている最大サイズDLQを超えると発行されSQS、スEventBridgeケジューラはスケジュールの Input 属性で指定したペイロードを切り捨てます。

EventBridge スケジューラ使用状況メトリクス

CloudWatch は、一部の AWS リソースの使用状況を追跡するメトリクスを収集します。これらのメトリクスは AWS Service Quotas に対応しています。これらのメトリクスを追跡することで、クォータを積極的に管理できます。次のメトリクスを使用して、ス EventBridge ケジューラのクォータをいつ超過したかを判断します。サービスクォータの詳細については、「クォータ」を参照してください。

これらのメトリクスは、 ではなく AWS/Usage名前空間に含まれAWS/Scheduler、1 分ごとに収集されます。

現在、この名前空間で CloudWatch 公開されるメトリクス名は のみですCallCount。このメトリクスは、Resource、Service、および Type のディメンションで発行されます。Resource ディメンションは、追跡されるAPIオペレーションの名前を指定します。

例えば、次のディメンションを持つ CallCountメトリクスは、アカウントで EventBridge スケジューラ CreateScheduleAPIオペレーションが呼び出された回数を示します。

- 「サービス」：「スケジューラ」
- 「タイプ」：API「」
- 「リソース」：CreateSchedule「」

CallCount メトリクスには指定された単位がありません。メトリクスの最も有用な統計は SUM です。これは、1 分間の合計オペレーション数を表します。

メトリクス

メトリクス	説明		
CallCount	アカウントで実行された指定されたオペレーションの数。		

ディメンション

ディメンション	説明		
Service	リソースを含む AWS サービスの名前。 EventBridge スケジューラ 使用状況メトリクスの場合、このディメンションの値は ですScheduler 。		
Class	追跡されているリソースのクラス。		

ディメンション	説明		
	EventBridge スケジューラ API 使用状況メトリクスは、このディメンションを の値で使用しますNone。		
Type	追跡されるリソースのタイプ。 現在、Service ディメンションが Scheduler である場合、Type の有効な値は API のみです。		
Resource	API オペレーションの名前。有効な値には次のようなものがあります。 • CreateSchedule • CreateScheduleGroup • DeleteSchedule • DeleteScheduleGroup • GetSchedule • GetScheduleGroup • ListScheduleGroups • ListSchedules • ListTagsForResource • TagResource • UntagResource • UpdateSchedule		

を使用した Amazon EventBridge Scheduler API呼び出しの口グ記録 AWS CloudTrail

Amazon EventBridge Scheduler は、ユーザー AWS CloudTrail、ロール、または EventBridge Scheduler の サービスによって実行されたアクションを記録する AWS サービスである と統合されています。 は、ス EventBridge ケジューラのすべてのAPI呼び出しをイベントとして CloudTrail キャ

プチャします。キャプチャされた呼び出しには、ス EventBridge ケジューラコンソールからの呼び出しと、スケ EventBridge ジューラAPIオペレーションへのコード呼び出しが含まれます。証跡を作成する場合は、ス EventBridge ケジューラの CloudTrail イベントなど、Amazon S3 バケットへのイベントの継続的な配信を有効にすることができます。証跡を設定しない場合でも、 CloudTrail コンソールのイベント履歴 で最新のイベントを表示できます。によって収集された情報を使用して CloudTrail、スケ EventBridge ジューラに対するリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

の詳細については CloudTrail、「 AWS CloudTrail ユーザーガイド」を参照してください。

EventBridge のスケジューラ情報 CloudTrail

CloudTrail アカウントを作成する AWS アカウント と、 で が有効になります。ス EventBridge ケジューラでアクティビティが発生すると、そのアクティビティは CloudTrail イベント履歴 の他の AWS サービスイベントとともにイベントに記録されます。で最近のイベントを表示、検索、ダウンロードできます AWS アカウント。詳細については、「イベント履歴 で CloudTrail イベントを表示する」を参照してください。

ス EventBridge ケジューラのイベントなど AWS アカウント、 のイベントの継続的な記録については、証跡を作成します。証跡により CloudTrail 、 はログファイルを Amazon S3 バケットに配信できます。デフォルトでは、コンソールで証跡を作成するときに、証跡がすべての AWS リージョンに適用されます。証跡は、 AWS パーティション内のすべてのリージョンからのイベントをログに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、 CloudTrail ログで収集されたイベントデータをより詳細に分析し、それに基づいて行動するように他の AWS サービスを設定できます。詳細については、次を参照してください:

- 追跡を作成するための概要
- CloudTrail がサポートするサービスと統合
- の Amazon SNS通知の設定 CloudTrail
- 複数のリージョンからの CloudTrail ログファイルの受信と複数のアカウントからの CloudTrail ログファイルの受信

すべてのス EventBridge ケジューラAPIアクションは によってログに記録 CloudTrail され、Amazon EventBridge Scheduler APIリファレンス に記載されています。例えば、、、 `DeleteSchedule`アクションを呼び出す`UpdateSchedule`と`CreateSchedule`、 CloudTrail ログファイルにエントリが生成されます。

各イベントまたはログエントリには、誰がリクエストを生成したかという情報が含まれます。アイデンティティ情報は、以下を判別するのに役立ちます:

- リクエストが root または AWS Identity and Access Management （ IAM) ユーザー認証情報を使用して行われたかどうか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが別の AWS サービスによって行われたかどうか。

詳細については、CloudTrail userIdentity「」要素を参照してください。

ス EventBridge ケジューラのログファイルエントリについて

証跡は、指定した Amazon S3 バケットにイベントをログファイルとして配信できるようにする設定です。 CloudTrail ログファイルには 1 つ以上のログエントリが含まれます。イベントは任意のソースからの単一のリクエストを表し、リクエストされたアクション、アクションの日時、リクエストパラメータなどに関する情報が含まれます。 CloudTrail ログファイルはパブリックAPIコールの順序付けられたスタックトレースではないため、特定の順序では表示されません。

Amazon EventBridge Scheduler のクォータ

AWS アカウントには、各 AWS サービスについて、以前は制限と呼ばれていたデフォルトのクォータがあります。特に明記されていない限り、クォータは地域固有です。ほとんどのクォータの引き上げをリクエストできますが、一部は引き上げることができません。

ス EventBridge ケジューラのクォータを表示するには、Service Quotas コンソール を開きます。ナビゲーションペインで、AWS サービス を選択し、スEventBridge ケジューラ を選択します。

クォータの引き上げをリクエストするには、「Service Quotas ユーザーガイド」の「Requesting a quota increase」(クォータ引き上げリクエスト) を参照してください。Service Quotas でクォータがまだ利用できない場合は、[上限引き上げ] フォームを使用してください。

AWS アカウントには、ス EventBridge ケジューラに関連する次のクォータがあります。

名前	デフォルト	引き上げ可能	説明
CreateSchedule リクエストレート	ca-central-1: 250 eu-central-1: 1,000 他のサポートされている各リージョン: 50	はい	1 秒あたりの最大 CreateSchedule リクエスト数。このクォータに達すると、ス EventBridge ケジューラは間隔の残りの間、このオペレーションのリクエストを拒否します。
CreateScheduleGroup リクエストレート	サポートされている各リージョン: 10	はい	1 秒あたりの最大 CreateScheduleGroup リクエスト数。このクォータに達すると、ス EventBridge ケジューラは間隔の残りの間、この

名前	デフォルト	引き上げ可能	説明
			オペレーションのリクエストを拒否します。
DeleteSchedule リクエストレート	ca-central-1: 250 eu-central-1: 1,000 他のサポートされている各リージョン: 50	はい	1 秒あたりの最大 DeleteSchedule リクエスト数。このクォータに達すると、ス EventBridge ケジューラは間隔の残りの間、このオペレーションのリクエストを拒否します。
DeleteScheduleGroup リクエストレート	サポートされている各リージョン: 10	はい	1 秒あたりの最大 DeleteScheduleGroup リクエスト数。このクォータに達すると、ス EventBridge ケジューラは間隔の残りの間、このオペレーションのリクエストを拒否します。
GetSchedule リクエストレート	ca-central-1: 250 eu-central-1: 1,000 他のサポートされている各リージョン: 50	はい	1 秒あたりの最大 GetSchedule リクエスト数。このクォータに達すると、ス EventBridge ケジューラは間隔の残りの間、このオペレーションのリクエストを拒否します。

名前	デフォルト	引き上げ可能	説明
GetScheduleGroup リクエストレート	サポートされている各リージョン: 10	はい	1 秒あたりの最大 GetScheduleGroup リクエスト数。このクォータに達すると、スEventBridge ケジューラは間隔の残りの間、このオペレーションのリクエストを拒否します。
Invocations スロットリング制限 (トランザクション/秒)	eu-central-1: 1,000 他のサポートされている各リージョン:500	はい	呼び出しは、定義されたターゲットに配信されるスケジュールペイロードです。制限に到達後、呼び出しが調整されます。つまり、引き続き呼び出しは行われますが、遅延が発生します。
ListScheduleGroups リクエストレート	サポートされている各リージョン: 10	はい	1 秒あたりの最大 ListScheduleGroups リクエスト数。このクォータに達すると、スEventBridge ケジューラは間隔の残りの間、このオペレーションのリクエストを拒否します。

名前	デフォルト	引き上げ可能	説明
ListSchedules リクエストレート	サポートされている各リージョン: 50	はい	1 秒あたりの最大 ListSchedules リクエスト数。このクォータに達すると、ス EventBridge ケジューラは間隔の残りの間、このオペレーションのリクエストを拒否します。
ListTagsForResource リクエストレート	サポートされている各リージョン: 10	はい	スケジューラリソースに関連付けられているすべてのタグを一覧表示します。
スケジュールグループの数	サポートされている各リージョン : 500	はい	リージョンあたりのスケジュールグループの最大数。
スケジュールの数	ca-central-1: 10,000,000 eu-central-1: 10,000,000 他のサポートされている各リージョン: 1,000,000	はい	リージョンあたりのスケジュールの最大数。このクォータには、実行が完了した 1 回限りのスケジュールが含まれます。ActionAfterCompletion 機能を使用して、完了後に自動的に削除するようにスケジュールを設定することをお勧めします。

名前	デフォルト	引き上げ可能	説明
TagResource リクエストレート	サポートされている各リージョン: 1	はい	指定されたスケジューラリソースに 1 つ以上のタグ (キーと値のペア) を割り当てます。
UntagResource リクエストレート	サポートされている各リージョン: 1	はい	指定したスケジューラリソースから 1 つまたは複数のタグを削除します。
UpdateSchedule リクエストレート	ca-central-1: 250 eu-central-1: 1,000 他のサポートされている各リージョン: 50	はい	1 秒あたりの最大 UpdateSchedule リクエスト数。このクォータに達すると、ス EventBridge ケジューラは間隔の残りの間、このオペレーションのリクエストを拒否します。

ス EventBridge ケジューラのクォータとサービスエンドポイントの詳細については、 AWS 全般のリファレンスガイドの「Amazon EventBridge Scheduler エンドポイントとクォータ」を参照してください。

ス EventBridge ケジューラのクォータのトラブルシューティング

次の情報は、ス EventBridge ケジューラのクォータに関して発生する可能性がある一般的な問題の診断と修正に役立ちます。

ServiceQuotaExceededException

デフォルトのレート制限を下回っていてもCreateSchedule、、、、GetScheduleまたは DeleteScheduleUpdateScheduleリクエストレートでスロットリングエラーが発生しています。

一般的な原因

2023 年 9 月 7 日、ス EventBridge ケジューラは実行ロールの ScheduleGroup ARN信頼ポリシー ARNでスケジュールの代わりに (Amazon リソースネーム) のサポートを開始しました。信頼ポリシーARNsでスケジュールの使用を引き続き許可リストに登録されているお客様はTPS、デフォルトの制限である 250 ~ 1000 TPS (リージョンによって異なります) ではなく、50 の制限がある場合があります。

解決方法

サポートに連絡して、上限の引き上げをリクエストしてください。

防止

次のいずれかの方法で、既存の信頼ポリシーを変更します。

- ロールからすべてのスコープを削除します。
- スケジュールARNまたは を使用して引き受けることができるようにロールをスコープします ScheduleGroup ARN。

  例えば、次の既存の信頼ポリシーがあるとします。

```
{
    "Effect": "Allow",
    "Principal": {
        "Service": "scheduler.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
        "StringEquals": {
            "aws:SourceArn":
 "arn:aws:scheduler:region:account:schedule/schedule_group/schedule"
        }
    }
}
```

信頼ポリシーを次のように更新できます。

```
{
    "Effect": "Allow",
    "Principal": {
        "Service": "scheduler.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
        "ForAnyValue:StringEquals": {
            "aws:SourceArn": [
                "arn:aws:scheduler:region:account:schedule/schedule_group/schedule",
                "arn:aws:scheduler:region:account:schedule-group/schedule_group"
            ]
        }
    }
}
```

ス EventBridge ケジューラユーザーガイドのドキュメント履歴

次の表に、ス EventBridge ケジューラのドキュメントリリースを示します。

変更	説明	日付
実行ロールと混乱した代理の防止の変更	今回の更新では、ロールの権限ポリシーで混乱した代理の防止を実装した場合に、スケジュールグループリソースに実行ロールが適用される方法の変更点について説明しています。 • the section called “混乱した代理の防止”	2023 年 9 月 7 日
完了後のスケジュールの自動削除	EventBridge スケジューラは自動削除をサポートしています。自動削除を設定すると、ス EventBridge ケジューラは最後に計画された呼び出し後にスケジュールを削除します。 • the section called “スケジュール完了後の削除”	2023 年 8 月 2 日
ユニバーサルターゲットの使用に関するトピックを更新	ス EventBridge ケジューラがターゲットにして統合できるサポートされているサービスのリストを更新しました。この更新には、サポートされていないGETAPIオペレーションのリスト、ユニバーサルターゲットの例の改善、およびガ	2023 年 3 月 17 日

	イド全体での のその他の軽微な改善も含まれています。 • the section called “ユニバーサルターゲットの使用”	
開始日が設定されていないレートベースのスケジュールに関する情報を更新	を指定しない場合にス EventBridge ケジューラがレートベースのスケジュールを処理する方法に関する情報を追加しました`StartDate` 。 • the section called “レートベースのスケジュール”	2023 年 3 月 17 日
スケジューラグループの管理に関する新しいトピック	スケジューラを使用してス EventBridge ケジューラグループを作成する方法に関する新しい章を追加しました。この章では、グループの作成、グループへのスケジュールの追加、ス EventBridge ケジューラリソースをより簡単に管理およびモニレートするためのタグの適用、最後にグループの削除を行う方法について説明します。 • スケジュールグループの管理	2023 年 3 月 17 日

夏時間とタイムゾーンに関する新しいトピック	ス EventBridge ケジューラが夏時間を処理する方法と、異なるタイムゾーンでスケジュールを作成する方法を説明する新しいセクションを追加しました。 • the section called “夏時間” • the section called “タイムゾーン”	2022 年 11 月 17 日
メトリクスに関する新しいトピック	ス EventBridge ケジューラがに発行するメトリクスを説明する新しいトピックを追加しました CloudWatch。これらのメトリクスを使用して、呼び出しの失敗を監視し、スケジュールの問題を解決する方法を把握できます。 • the section called “によるモニタリング CloudWatch”	2022 年 11 月 15 日
初回リリース	ス EventBridge ケジューラユーザーガイドの初回リリース。	2022 年 11 月 10 日

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。