aws

ユーザーガイド

Amazon Managed Service for Prometheus

Amazon Managed Service for Prometheus: ユーザーガイド

Table of Contents

Amazon Managed Service for Prometheus とは

Amazon Managed Service for Prometheus は、コンテナのメトリクスをモニタリングする Prometheus 互換のサーバーレスサービスです。これにより、大規模なコンテナ環境を安全にモニタリングすることが容易になります。Amazon Managed Service for Prometheus では、現在使用されているものと同じオープンソースの Prometheus データモデルとクエリ言語を使用して、コンテナ化されたワークロードのパフォーマンスをモニタリングできます。また、基盤のインフラストラクチャを管理する必要なく、スケーラビリティ、可用性、セキュリティを強化できます。

Amazon Managed Service for Prometheus は、ワークロードのスケールアップとスケールダウンに応じて自動的に運用メトリクスの取り込み、保存、クエリをスケールします。これは AWS セキュリティサービスと統合され、データへの高速で安全なアクセスを可能にします。

Amazon Managed Service for Prometheus は、複数のアベイラビリティーゾーン (マルチ AZ) 配置を使用して高い可用性を実現するように設計されています。ワークスペースに取り込まれたデータは、同じリージョンの 3 つのアベイラビリティーゾーンにレプリケートされます。

Amazon Managed Service for Prometheus は、Amazon Elastic Kubernetes Service 環境および自己管理型 Kubernetes 環境で実行されるコンテナクラスターと連携します。

Amazon Managed Service for Prometheus では、Prometheus と同じオープンソースの Prometheus データモデルおよび PromQL クエリ言語が使用されます。エンジニアリングチームは、PromQL を使用してメトリクスのフィルタリングや集計を行ったり、メトリクスにアラームを設定したりして、コードを変更することなくすばやくパフォーマンスを可視化できます。Amazon Managed Service for Prometheus により、運用コストや複雑さを伴わずに柔軟なクエリ機能が提供されます。

ワークスペースに取り込まれたメトリクスは、デフォルトで 150 日間保存され、その後自動的に削除されます。最大 1095 日 (3 年) までワークスペースを設定して、保持期間を調整できます。詳細については、「ワークスペースの設定」を参照してください。

サポート対象のリージョン

Amazon Managed Service for Prometheus では現在、次のリージョンがサポートされています。

リージョン名	リージョン	エンドポイント	プロトコル	
米国東部 (オハイオ)	us-east-2	aps.us-east-2.amazonaws.com	HTTPS	
		aps-workspaces.us-east-2.amazonaws.com	HTTPS	
		aps-workspaces-fips.us-east-2.amazonaws.com	HTTPS	
		aps-workspaces-fips.us-east-2.api.aws	HTTPS	
		aps-workspaces.us-east-2.api.aws	HTTPS	
		aps-fips.us-east-2.amazonaws.com	HTTPS	
		aps.us-east-2.api.aws	HTTPS	
		aps-fips.us-east-2.api.aws	HTTPS	
米国東部 (バージニア北部)	us-east-1	aps.us-east-1.amazonaws.com	HTTPS	
		aps-workspaces.us-east-1.amazonaws.com	HTTPS	
		aps-workspaces-fips.us-east-1.amazonaws.com	HTTPS	
		aps-workspaces-fips.us-east-1.api.aws	HTTPS	
		aps-workspaces.us-east-1.api.aws	HTTPS	
		aps-fips.us-east-1.amazonaws.com	HTTPS	
		aps.us-east-1.api.aws	HTTPS	
		aps-fips.us-east-1.api.aws	HTTPS	
米国西部 (北カリフォルニア)	us-west-1	aps.us-west-1.amazonaws.com	HTTPS	
		aps-workspaces.us-west-1.amazonaws.com	HTTPS	
			HTTPS	

リージョン名	リージョン	エンドポイント	プロトコル	
		aps-workspaces-fips.us-west-1.amazonaws.com	HTTPS	
		aps-workspaces-fips.us-west-1.api.aws	HTTPS	
		aps-workspaces.us-west-1.api.aws	HTTPS	
		aps-fips.us-west-1.amazonaws.com	HTTPS	
		aps.us-west-1.api.aws	HTTPS	
		aps-fips.us-west-1.api.aws	HTTPS	
米国西部 (オレゴン)	us-west-2	aps.us-west-2.amazonaws.com	HTTPS	
		aps-workspaces.us-west-2.amazonaws.com	HTTPS	
		aps-workspaces-fips.us-west-2.amazonaws.com	HTTPS	
		aps-workspaces-fips.us-west-2.api.aws	HTTPS	
		aps-workspaces.us-west-2.api.aws	HTTPS	
		aps-fips.us-west-2.amazonaws.com	HTTPS	
		aps.us-west-2.api.aws	HTTPS	
		aps-fips.us-west-2.api.aws	HTTPS	
アフリカ (ケープタウン)	af-south-1	aps.af-south-1.amazonaws.com	HTTPS	
		aps-workspaces.af-south-1.amazonaws.com	HTTPS	
		aps-workspaces.af-south-1.api.aws	HTTPS	
		aps.af-south-1.api.aws	HTTPS	

リージョン名	リージョン	エンドポイント	プロトコル
アジアパシフィック (香港)	ap-east-1	aps.ap-east-1.amazonaws.com	HTTPS
		aps-workspaces.ap-east-1.amazonaws.com	HTTPS
		aps-workspaces.ap-east-1.api.aws	HTTPS
		aps.ap-east-1.api.aws	HTTPS
アジアパシフィック (ハイデラバード)	ap-south-2	aps.ap-south-2.amazonaws.com	HTTPS
		aps-workspaces.ap-south-2.amazonaws.com	HTTPS
		aps-workspaces.ap-south-2.api.aws	HTTPS
		aps.ap-south-2.api.aws	HTTPS
アジアパシフィック (ジャカルタ)	ap-southeast-3	aps.ap-southeast-3.amazonaws.com	HTTPS
		aps-workspaces.ap-southeast-3.amazonaws.com	HTTPS
		aps-workspaces.ap-southeast-3.api.aws	HTTPS
		aps.ap-southeast-3.api.aws	HTTPS
アジアパシフィック (マレーシア)	ap-southeast-5	aps.ap-southeast-5.amazonaws.com	HTTPS
		aps-workspaces.ap-southeast-5.amazonaws.com	HTTPS
		aps-workspaces.ap-southeast-5.api.aws	HTTPS
		aps.ap-southeast-5.api.aws	HTTPS

リージョン名	リージョン	エンドポイント	プロトコル	
アジアパシフィック (メルボルン)	ap-southeast-4	aps.ap-southeast-4.amazonaws.com aps-workspaces.ap-southeast-4.amazonaws.com aps-workspaces.ap-southeast-4.api.aws aps.ap-southeast-4.api.aws	HTTPS HTTPS HTTPS HTTPS	
アジアパシフィック (ムンバイ)	ap-south-1	aps.ap-south-1.amazonaws.com aps-workspaces.ap-south-1.amazonaws.com aps-workspaces.ap-south-1.api.aws aps.ap-south-1.api.aws	HTTPS HTTPS HTTPS HTTPS	
アジアパシフィック (大阪)	ap-northeast-3	aps.ap-northeast-3.amazonaws.com aps-workspaces.ap-northeast-3.amazonaws.com aps-workspaces.ap-northeast-3.api.aws aps.ap-northeast-3.api.aws	HTTPS HTTPS HTTPS HTTPS	
アジアパシフィック (ソウル)	ap-northeast-2	aps.ap-northeast-2.amazonaws.com aps-workspaces.ap-northeast-2.amazonaws.com aps-workspaces.ap-northeast-2.api.aws aps.ap-northeast-2.api.aws	HTTPS HTTPS HTTPS HTTPS	

リージョン名	リージョン	エンドポイント	プロトコル
アジアパシフィック (シンガポール)	ap-southeast-1	aps.ap-southeast-1.amazonaws.com aps-workspaces.ap-southeast-1.amazonaws.com aps-workspaces.ap-southeast-1.api.aws aps.ap-southeast-1.api.aws	HTTPS HTTPS HTTPS HTTPS
アジアパシフィック (シドニー)	ap-southeast-2	aps.ap-southeast-2.amazonaws.com aps-workspaces.ap-southeast-2.amazonaws.com aps-workspaces.ap-southeast-2.api.aws aps.ap-southeast-2.api.aws	HTTPS HTTPS HTTPS HTTPS
アジアパシフィック (台北)	ap-east-2	aps.ap-east-2.amazonaws.com aps-workspaces.ap-east-2.amazonaws.com aps-workspaces.ap-east-2.api.aws aps.ap-east-2.api.aws	HTTPS HTTPS HTTPS HTTPS
アジアパシフィック (タイ)	ap-southeast-7	aps.ap-southeast-7.amazonaws.com aps-workspaces.ap-southeast-7.amazonaws.com aps-workspaces.ap-southeast-7.api.aws aps.ap-southeast-7.api.aws	HTTPS HTTPS HTTPS HTTPS

リージョン名	リージョン	エンドポイント	プロトコル
アジアパシフィック (東京)	ap-northeast-1	aps.ap-northeast-1.amazonaws.com aps-workspaces.ap-northeast-1.amazonaws.com aps-workspaces.ap-northeast-1.api.aws aps.ap-northeast-1.api.aws	HTTPS HTTPS HTTPS HTTPS
カナダ (中部)	ca-central-1	aps.ca-central-1.amazonaws.com aps-workspaces.ca-central-1.amazonaws.com aps-workspaces-fips.ca-central-1.amazonaws.com aps-workspaces-fips.ca-central-1.api.aws aps-workspaces.ca-central-1.api.aws aps-fips.ca-central-1.amazonaws.com aps.ca-central-1.api.aws aps-fips.ca-central-1.api.aws	HTTPS HTTPS HTTPS HTTPS HTTPS HTTPS HTTPS HTTPS

リージョン名	リージョン	エンドポイント	プロトコル	
カナダ西部 (カルガリー)	ca-west-1	aps.ca-west-1.amazonaws.com	HTTPS	
		aps-workspaces.ca-west-1.amazonaws.com	HTTPS	
		aps-workspaces-fips.ca-west-1.amazonaws.com	HTTPS	
		aps-workspaces-fips.ca-west-1.api.aws	HTTPS	
		aps-workspaces.ca-west-1.api.aws	HTTPS	
		aps-fips.ca-west-1.amazonaws.com	HTTPS	
		aps.ca-west-1.api.aws	HTTPS	
		aps-fips.ca-west-1.api.aws	HTTPS	
欧州 (フランクフルト)	eu-central-1	aps.eu-central-1.amazonaws.com	HTTPS	
		aps-workspaces.eu-central-1.amazonaws.com	HTTPS	
		aps-workspaces.eu-central-1.api.aws	HTTPS	
		aps.eu-central-1.api.aws	HTTPS	
欧州 (アイルランド)	eu-west-1	aps.eu-west-1.amazonaws.com	HTTPS	
		aps-workspaces.eu-west-1.amazonaws.com	HTTPS	
		aps-workspaces.eu-west-1.api.aws	HTTPS	
		aps.eu-west-1.api.aws	HTTPS	

リージョン名	リージョン	エンドポイント	プロトコル	
欧州 (ロンドン)	eu-west-2	aps.eu-west-2.amazonaws.com	HTTPS	
		aps-workspaces.eu-west-2.amazonaws.com	HTTPS	
		aps-workspaces.eu-west-2.api.aws	HTTPS	
		aps.eu-west-2.api.aws	HTTPS	
ヨーロッパ (ミラノ)	eu-south-1	aps.eu-south-1.amazonaws.com	HTTPS	
		aps-workspaces.eu-south-1.amazonaws.com	HTTPS	
		aps-workspaces.eu-south-1.api.aws	HTTPS	
		aps.eu-south-1.api.aws	HTTPS	
欧州 (パリ)	eu-west-3	aps.eu-west-3.amazonaws.com	HTTPS	
		aps-workspaces.eu-west-3.amazonaws.com	HTTPS	
		aps-workspaces.eu-west-3.api.aws	HTTPS	
		aps.eu-west-3.api.aws	HTTPS	
欧州 (スペイン)	eu-south-2	aps.eu-south-2.amazonaws.com	HTTPS	
		aps-workspaces.eu-south-2.amazonaws.com	HTTPS	
		aps-workspaces.eu-south-2.api.aws	HTTPS	
		aps.eu-south-2.api.aws	HTTPS	
欧州 (ストックホルム)	eu-north-1	aps.eu-north-1.amazonaws.com	HTTPS	
		aps-workspaces.eu-north-1.amazonaws.com	HTTPS	
		aps-workspaces.eu-north-1.api.aws	HTTPS	
		aps.eu-north-1.api.aws	HTTPS	

リージョン名	リージョン	エンドポイント	プロトコル
欧州 (チューリッヒ)	eu-central-2	aps.eu-central-2.amazonaws.com	HTTPS
		aps-workspaces.eu-central-2.amazonaws.com	HTTPS
		aps-workspaces.eu-central-2.api.aws	HTTPS
		aps.eu-central-2.api.aws	HTTPS
イスラエル (テルアビブ)	il-central-1	aps.il-central-1.amazonaws.com	HTTPS
		aps-workspaces.il-central-1.amazonaws.com	HTTPS
		aps-workspaces.il-central-1.api.aws	HTTPS
		aps.il-central-1.api.aws	HTTPS
メキシコ (中部)	mx-central-1	aps.mx-central-1.amazonaws.com	HTTPS
		aps-workspaces.mx-central-1.amazonaws.com	HTTPS
		aps-workspaces.mx-central-1.api.aws	HTTPS
		aps.mx-central-1.api.aws	HTTPS
中東 (バーレーン)	me-south-1	aps.me-south-1.amazonaws.com	HTTPS
		aps-workspaces.me-south-1.amazonaws.com	HTTPS
		aps-workspaces.me-south-1.api.aws	HTTPS
		aps.me-south-1.api.aws	HTTPS
中東 (アラブ首長国連邦)	me-central-1	aps.me-central-1.amazonaws.com	HTTPS
		aps-workspaces.me-central-1.amazonaws.com	HTTPS
		aps-workspaces.me-central-1.api.aws	HTTPS
		aps.me-central-1.api.aws	HTTPS

リージョン名	リージョン	エンドポイント	プロトコル	
南米 (サンパウロ)	sa-east-1	aps.sa-east-1.amazonaws.com	HTTPS	
		aps-workspaces.sa-east-1.amazonaws.com	HTTPS	
		aps-workspaces.sa-east-1.api.aws	HTTPS	
		aps.sa-east-1.api.aws	HTTPS	
AWS GovCloud (米国東部)	us-gov-east-1	aps.us-gov-east-1.amazonaws.com	HTTPS	
		aps-workspaces.us-gov-east-1.amazonaws.com	HTTPS	
		aps-workspaces-fips.us-gov-east-1.amazonaws.com	HTTPS	
		aps-workspaces-fips.us-gov-east-1.api.aws	HTTPS	
		aps-workspaces.us-gov-east-1.api.aws	HTTPS	
		aps-fips.us-gov-east-1.amazonaws.com	HTTPS	
		aps.us-gov-east-1.api.aws	HTTPS	
		aps-fips.us-gov-east-1.api.aws		

リージョン名	リージョン	エンドポイント	プロトコル
AWS GovCloud (米国西部)	us-gov-west-1	aps.us-gov-west-1.amazonaws.com	HTTPS
		aps-workspaces.us-gov-west-1.amazonaws.com	HTTPS
		aps-workspaces-fips.us-gov-west-1.amazonaws.com	HTTPS
		aps-workspaces-fips.us-gov-west-1.api.aws	HTTPS
		aps-workspaces.us-gov-west-1.api.aws	HTTPS
		aps-fips.us-gov-west-1.amazonaws.com	HTTPS
		aps.us-gov-west-1.api.aws	HTTPS
		aps-fips.us-gov-west-1.api.aws	

Amazon Managed Service for Prometheus には、(ワークスペース管理タスクを実行する) コントロールプレーンエンドポイント、および (ワークスペースインスタンスで Prometheus 互換データを処理する) データプレーンエンドポイントが含まれています。コントロールプレーンエンドポイントは `aps.*` で始まり、データプレーンエンドポイントは `aps-workspaces.*` で始まります。`.amazonaws.com` で終わるエンドポイントは IPv4 をサポートし、`.api.aws` で終わるエンドポイントは IPv4 と IPv6 の両方をサポートします。

料金

メトリクスの取り込みと保存には料金がかかります。ストレージ料金は、メトリクスサンプルとメタデータの圧縮サイズに基づきます。詳細については、「Amazon Managed Service for Prometheus の料金」を参照してください。

AWS Cost Explorer および AWS コストと使用状況レポートを使用して、料金をモニタリングできます。詳細については、Cost Explorer を使用したデータの探索」およびAWS 「コストと使用状況レポートとは」を参照してください。

プレミアムサポート

任意のレベルの AWS プレミアムサポートプランにサブスクライブする場合、プレミアムサポートは Amazon Managed Service for Prometheus に適用されます。

Amazon Managed Service for Prometheus の使用を開始する

Amazon Managed Service for Prometheus は、コンテナのメトリクスをモニタリングする Prometheus 互換のサーバーレスサービスです。これにより、大規模なコンテナ環境を安全にモニタリングすることが容易になります。このセクションでは、Amazon Managed Service for Prometheus の使用に関連した以下の 3 つの主要な領域について説明します。

- ワークスペースの作成 - メトリクスを保存およびモニタリングする Amazon Managed Service for Prometheus ワークスペースを作成します。
- メトリクスの取り込み - ワークスペースにメトリクスを取り込むまで、ワークスペースは空です。Amazon Managed Service for Prometheus にメトリクスを送信したり、Amazon Managed Service for Prometheus でメトリクスを自動的にスクレイピングしたりできます。
- メトリクスのクエリ - ワークスペースにメトリクスをデータとして取り込むと、そのデータをクエリしてメトリクスを探索またはモニタリングする準備が整います。

を初めて使用する場合、 AWSこのセクションには の設定に関する詳細 AWS アカウントも含まれています。

トピック

- セットアップ AWS
- Amazon Managed Service for Prometheus ワークスペースの作成
- ワークスペースへの Prometheus メトリクスの取り込み
- Prometheus メトリクスに対するクエリの実行

セットアップ AWS

このセクションのタスクを完了して、 AWS を初めてセットアップします。アカウントがすでにある場合は AWS 、「」に進みますAmazon Managed Service for Prometheus ワークスペースの作成。

にサインアップすると AWS、 AWS アカウントは Amazon Managed Service for Prometheus を含む AWSのすべてのサービスに自動的にアクセスできます。ただし、料金が発生するのは実際に使用したサービスの分だけです。

トピック

- にサインアップする AWS アカウント

にサインアップする AWS アカウント

の使用を開始するには AWS、 が必要です AWS アカウント。の作成の詳細については AWS アカウント、 AWS アカウント管理 リファレンスガイドの「 の開始方法 AWS アカウント」を参照してください。

Amazon Managed Service for Prometheus ワークスペースの作成

ワークスペースは、Prometheus メトリクスの保存とクエリに使用される専用の論理スペースです。ワークスペースでは、更新、一覧表示、記述、削除、メトリクスの取り込みとクエリなど、管理操作を認可するためのきめ細かいアクセスコントロールがサポートされます。アカウント内のリージョンごとに 1 つ以上のワークスペースを持つことができます。

ワークスペースをセットアップするには、次の手順に従います。

> **Note**
>
> ワークスペースの作成の詳細と利用可能なオプションについては、「Amazon Managed Service for Prometheus ワークスペースを作成する」を参照してください。

Amazon Managed Service for Prometheus ワークスペースを作成するには

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. [WorkSpace エイリアス] に、新しいワークスペースのエイリアスを入力します。

   ワークスペースエイリアスは、ワークスペースの識別に役立つわかりやすい名前です。これは一意でなくても構いません。2 つのワークスペースに同じエイリアスを付けることもできます。ただし、すべてのワークスペースには Amazon Managed Service for Prometheus によって生成された一意のワークスペース ID が割り当てられます。
3. (オプション) 名前空間にタグを追加するには、[新しいタグを追加] を選択します。

   [キー] にタグの名前を入力します。[値] では、任意でタグに値を追加できます。

   別のタグを追加するには、[新しいタグを追加] を再度選択します。

4. [ワークスペースを作成する] を選択します。

   ワークスペースの詳細ページが表示されます。ここには、このワークスペースのステータス、ARN、ワークスペース ID、リモート書き込み用とクエリ用のエンドポイント URL などの情報が表示されます。

   最初はステータスが [作成中] になる可能性があります。ステータスが [アクティブ] になるまで待ってから、メトリクスの取り込みの設定に進んでください。

   [エンドポイント - リモート書き込み URL] と [エンドポイント - クエリ URL] に表示された URL を書き留めます。これらの URL は、このワークスペースにメトリクスをリモートで書き込むように Prometheus サーバーを構成するときと、それらのメトリクスにクエリを実行するときに必要になります。

ワークスペースへの Prometheus メトリクスの取り込み

メトリクスを取り込む方法の 1 つは、スタンドアロンの Prometheus エージェント (エージェントモードで実行されている Prometheus インスタンス) を使用してクラスターからメトリクスを取得し、Amazon Managed Service for Prometheus に転送してストレージとモニタリングを行うことです。このセクションでは、Helm を使用して Prometheus エージェントの新しいインスタンスをセットアップすることにより、Amazon EKS から Amazon Managed Service for Prometheus ワークスペースへのメトリクスの取り込みを設定する方法について説明します。

Kubernetes やノードレベルのメトリクスなど、Amazon EKS でメトリクスを生成するには、Amazon EKS コミュニティアドオンを使用できます。詳細については、「Amazon EKS ユーザーガイド」の「利用可能なコミュニティアドオン」を参照してください。

メトリクスを保護する方法や可用性の高いメトリクスを作成する方法など、Amazon Managed Service for Prometheus にデータを取り込む他の方法については、「Amazon Managed Service for Prometheus ワークスペースにメトリクスを取り込む」を参照してください。

> **Note**
>
> ワークスペースに取り込まれたメトリクスは、デフォルトで 150 日間保存され、その後自動的に削除されます。最大 1095 日 (3 年) までワークスペースを設定して、保持期間を調整できます。詳細については、「ワークスペースの設定」を参照してください。

このセクションの手順に従うと、Amazon Managed Service for Prometheus を迅速に設定して稼働させることができます。既にワークスペースを作成していることを前提としています。このセクションでは、Amazon EKS クラスターに新しい Prometheus サーバーをセットアップします。新しいサーバーは、デフォルト設定を使用して Amazon Managed Service for Prometheus にメトリクスを送信するエージェントとして動作します。この方法には次の前提条件があります。

- 新しい Prometheus サーバーがメトリクスを収集する元の Amazon EKS クラスターが必要です。
- Amazon EKS クラスターには Amazon EBS CSI ドライバーをインストールしている必要があります (Helm で必要)。
- Helm CLI 3.0 以降を使用する必要があります。
- 以下のセクションの手順を実行するには、Linux または MacOS コンピュータを使用する必要があります。

ステップ 1: 新しい Helm チャートリポジトリを追加する

次のコマンドを入力して、新しい Helm チャートリポジトリを追加します。これらのコマンドの詳細については、「Helm Repo」を参照してください。

```
helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
helm repo add kube-state-metrics https://kubernetes.github.io/kube-state-metrics
helm repo update
```

ステップ 2: Prometheus 名前空間を作成する

次のコマンドを入力して、Prometheus サーバーとその他のモニタリングコンポーネント用の Prometheus 名前空間を作成します。*prometheus-agent-namespace* は、この名前空間に付ける名前に置き換えます。

```
kubectl create namespace prometheus-agent-namespace
```

ステップ 3: サービスアカウントの IAM ロールを設定する

この取り込み方法では、Prometheus エージェントが稼働している Amazon EKS クラスターのサービスアカウントの IAM ロールを使用する必要があります。

サービスアカウントの IAM ロールを使用すると、IAM ロールを Kubernetes サービスアカウントに関連付けることができます。このサービスアカウントは、そのサービスアカウントを使用する任意の

ポッドのコンテナにアクセス AWS 許可を付与できます。詳細については、「サービスアカウントの IAM ロール」を参照してください。

これらのロールをまだ設定していない場合は、「Amazon EKS クラスターからメトリクスを取り込むためのサービスロールの設定」の手順に従ってロールを設定します。そのセクションの手順では、`eksctl` を使用する必要があります。詳細については、「Amazon Elastic Kubernetes Service の開始方法 - `eksctl`」を参照してください。

> **Note**
>
> EKS または ではなく AWS 、アクセスキーとシークレットキーのみを使用して Amazon Managed Service for Prometheus にアクセスする場合、 `EKS-IAM-ROLE` ベースの SigV4 を使用することはできません。

ステップ 4: 新しいサーバーをセットアップしてメトリクスの取り込みを開始する

新しい Prometheus エージェントをインストールし、Amazon Managed Service for Prometheus ワークスペースにメトリクスを送信するには、以下の手順に従います。

新しい Prometheus エージェントをインストールし、Amazon Managed Service for Prometheus ワークスペースにメトリクスを送信するには

1. テキストエディタを使用して、`my_prometheus_values_yaml` という名前のファイルを作成し、次の内容を記述します。

   - *`IAM_PROXY_PROMETHEUS_ROLE_ARN`* は、「Amazon EKS クラスターからメトリクスを取り込むためのサービスロールの設定」で作成した amp-iamproxy-ingest-role の ARN に置き換えます。
   - *`WORKSPACE_ID`* は、Amazon Managed Service for Prometheus ワークスペースの ID に置き換えます。
   - *`REGION`* は、Amazon Managed Service for Prometheus のリージョンに置き換えます。

```
## The following is a set of default values for prometheus server helm chart which
 enable remoteWrite to AMP
## For the rest of prometheus helm chart values see: https://github.com/prometheus-
community/helm-charts/blob/main/charts/prometheus/values.yaml
```

```
##
serviceAccounts:
  server:
    name: amp-iamproxy-ingest-service-account
    annotations:
      eks.amazonaws.com/role-arn: ${IAM_PROXY_PROMETHEUS_ROLE_ARN}
server:
  remoteWrite:
    - url: https://aps-workspaces.${REGION}.amazonaws.com/workspaces/
${WORKSPACE_ID}/api/v1/remote_write
      sigv4:
        region: ${REGION}
      queue_config:
        max_samples_per_send: 1000
        max_shards: 200
        capacity: 2500
```

2. 次のコマンドを入力して、Prometheus サーバーを作成します。

   - *prometheus-chart-name* は、Prometheus リリース名に置き換えます。
   - *prometheus-agent-namespace* は、Prometheus 名前空間の名前に置き換えます。

```
helm install prometheus-chart-name prometheus-community/prometheus -n prometheus-
agent-namespace \
-f my_prometheus_values_yaml
```

Prometheus メトリクスに対するクエリの実行

ワークスペースにメトリクスが取り込まれるようになったら、それらのメトリクスに対してクエリを実行できます。メトリクスをクエリする一般的な方法は、Grafana などのサービスを使用してメトリクスをクエリすることです。このセクションでは、Amazon Managed Grafana を使用して Amazon Managed Service for Prometheus のメトリクスをクエリする方法を学習します。

> **Note**
>
> Amazon Managed Service for Prometheus のメトリクスをクエリする他の方法や、Amazon Managed Service for Prometheus API を使用する方法については、「Prometheus メトリクスに対するクエリの実行」を参照してください。

このセクションでは、既にワークスペースを作成していて、そのワークスペースにメトリクスを取り込んでいることを前提としています。

クエリの実行には、Prometheus の標準クエリ言語である PromQL を使用します。PromQL とその構文の詳細については、Prometheus ドキュメントの「Querying Prometheus」を参照してください。

Amazon Managed Grafana は、オープンソースの Grafana 向けのフルマネージドサービスで、オープンソースのサードパーティー ISV、およびデータソースを大規模に視覚化および分析するための AWS サービスへの接続を簡素化します。

Amazon Managed Service for Prometheus では、Amazon Managed Grafana を使用してワークスペース内のメトリクスにクエリを実行することがサポートされています。Amazon Managed Grafana コンソールで、既存の Amazon Managed Service for Prometheus アカウントを検出して、Amazon Managed Service for Prometheus ワークスペースをデータソースとして追加できます。Amazon Managed Grafana は、Amazon Managed Service for Prometheus にアクセスするために必要な認証情報の設定を管理します。Amazon Managed Grafana から Amazon Managed Service for Prometheus への接続を作成する方法の詳細については、「Amazon Managed Grafana User Guide」の手順を参照してください。

Amazon Managed Service for Prometheus のアラートを Amazon Managed Grafana で表示することもできます。アラートとの統合を設定する手順については、「アラートを Amazon Managed Grafana またはオープンソースの Grafana と統合する」を参照してください。

> **Note**
>
> Amazon Managed Grafana ワークスペースがプライベート VPC を使用するように設定している場合は、Amazon Managed Service for Prometheus のワークスペースを同じ VPC に接続する必要があります。詳細については、「プライベート VPC での Amazon Managed Grafana への接続」を参照してください。

Amazon Managed Service for Prometheus ワークスペースを管理する

ワークスペースは、Prometheus メトリクスの保存とクエリに使用される専用の論理スペースです。ワークスペースでは、更新、一覧表示、記述、削除、メトリクスの取り込みとクエリなど、管理操作を認可するためのきめ細かいアクセスコントロールがサポートされます。アカウント内のリージョンごとに 1 つ以上のワークスペースを持つことができます。

Amazon Managed Service for Prometheus ワークスペースを作成して管理するには、このセクションの手順を使用します。

トピック

Amazon Managed Service for Prometheus ワークスペースを作成する

Amazon Managed Service for Prometheus ワークスペースを作成するには、以下の手順に従います。 AWS CLI と Amazon Managed Service for Prometheus コンソールのどちらを使用するかを選択できます。

> **Note**
>
> Amazon EKS クラスターを実行している場合は、AWS Controllers for Kubernetes を使用して新しいワークスペースを作成することもできます。

を使用してワークスペースを作成するには AWS CLI

1. 以下のコマンドを入力して、ワークスペースを作成します。この例では `my-first-workspace` という名前のワークスペースを作成しますが、必要に応じて別のエイリアスを使

用することもできます。ワークスペースエイリアスは、ワークスペースの識別に役立つわかりやすい名前です。これは一意でなくても構いません。2 つのワークスペースに同じエイリアスを付けることもできます。ただし、すべてのワークスペースには Amazon Managed Service for Prometheus によって生成された一意のワークスペース ID が割り当てられます。

(オプション) 独自の KMS キーを使用してワークスペースに保存されているデータを暗号化するには、使用する AWS KMS キーに kmsKeyArnパラメータを含めることができます。Amazon Managed Service for Prometheus はカスタマーマネージドキーの使用に対して課金しませんが、キーに関連するコストが発生する場合があります AWS Key Management Service。Amazon Managed Service for Prometheus によるワークスペース内のデータの暗号化、または独自のカスタマーマネージドキーの作成、管理、使用方法の詳細については、「保管中の暗号化」を参照してください。

括弧 ([]) 内のパラメータはオプションであり、コマンドには括弧を含めないでください。

```
aws amp create-workspace [--alias my-first-workspace] [--kmsKeyArn arn:aws:aps:us-
west-2:111122223333:workspace/ws-sample-1234-abcd-56ef-7890abcd12ef]  [--
tags Status=Secret,Team=My-Team]
```

このコマンドは次のデータを返します。

- workspaceId は、このワークスペースの一意の ID です。この ID を書き留めてください。
- arn は、このワークスペースの ARN です。
- status は、ワークスペースの現在のステータスです。ワークスペースの作成直後は CREATING になる可能性があります。
- kmsKeyArn は、ワークスペースデータの暗号化に使用されるカスタマーマネージドキーです (指定されている場合)。

> **Note**
>
> カスタマーマネージドキーで作成されたワークスペースは、取り込み用にAWS マネージドコレクターを使用することはできません。
> カスタマーマネージドキーと AWS 所有キーのどちらを慎重に使用するかを選択します。カスタマーマネージドキーで作成されたワークスペースは、後で (またはその逆で) AWS 所有キーを使用するように変換することはできません。

- tags は、ワークスペースのタグ (ある場合) のリストを示します。

2. `create-workspace` コマンドが `CREATING` ステータスを返した場合は、後で次のコマンドを入力することで、ワークスペースの準備が整ったかどうかを確認できます。*my-workspace-id* は、`create-workspace` コマンドから返された `workspaceId` の値に置き換えます。

```
aws amp describe-workspace --workspace-id my-workspace-id
```

`describe-workspace` コマンドから `status` として `ACTIVE` が返されたら、ワークスペースを使用する準備ができています。

Amazon Managed Service for Prometheus コンソールを使用してワークスペースを作成するには

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. [作成] を選択します。
3. [WorkSpace エイリアス] に、新しいワークスペースのエイリアスを入力します。

   ワークスペースエイリアスは、ワークスペースの識別に役立つわかりやすい名前です。これは一意でなくても構いません。2 つのワークスペースに同じエイリアスを付けることもできます。ただし、すべてのワークスペースには Amazon Managed Service for Prometheus によって生成された一意のワークスペース ID が割り当てられます。
4. (オプション) 独自の KMS キーを使用してワークスペースに保存されているデータを暗号化するには、暗号化設定をカスタマイズを選択し、使用する AWS KMS キーを選択します (または新しいキーを作成します)。ドロップダウンリストからアカウントのキーを選択するか、アクセスできる任意のキーの ARN を入力できます。Amazon Managed Service for Prometheus はカスタマーマネージドキーの使用に対して課金しませんが、キーに関連するコストが発生する可能性があります AWS Key Management Service。

   Amazon Managed Service for Prometheus によるワークスペース内のデータの暗号化や、お客様独自のカスタマーマネージドキーの作成、管理、使用方法の詳細については、「保管中の暗号化」を参照してください。

> **Note**
>
> カスタマーマネージドキーで作成されたワークスペースは、取り込み用にAWS マネージドコレクターを使用することはできません。

カスタマーマネージドキーと AWS 所有キーのどちらを慎重に使用するかを選択します。カスタマーマネージドキーで作成されたワークスペースは、後で (またはその逆で) AWS 所有キーを使用するように変換することはできません。

5. (オプション) ワークスペースに 1 つ以上のタグを追加するには、[新しいタグを追加] を選択します。[キー] にタグの名前を入力します。[値] では、任意でタグに値を追加できます。

   別のタグを追加するには、[新しいタグを追加] を再度選択します。

6. [ワークスペースを作成する] を選択します。

   ワークスペースの詳細ページが表示されます。ここには、このワークスペースのステータス、ARN、ワークスペース ID、リモート書き込み用とクエリ用のエンドポイント URL などの情報が表示されます。

   ワークスペースの準備が整うまで、ステータスは [作成中] に戻ります。ステータスが [アクティブ] になるまで待ってから、メトリクスの取り込みの設定に進んでください。

   [エンドポイント - リモート書き込み URL] と [エンドポイント - クエリ URL] に表示された URL を書き留めます。これらの URL は、このワークスペースにメトリクスをリモートで書き込むように Prometheus サーバーを構成するときと、それらのメトリクスにクエリを実行するときに必要になります。

ワークスペースにメトリクスを取り込む方法については、「ワークスペースへの Prometheus メトリクスの取り込み」を参照してください。

ワークスペースの設定

ワークスペースは以下に対して設定できます。

- ラベルセットを定義し、定義したラベルセットに一致するアクティブ時系列の制限を定義します。ラベルセットは、時系列メトリクスにコンテキストを与えるのに役立つ名前と値のペアである 1 つ以上のラベルのセットです。

  ラベルセットを定義し、アクティブ時系列制限を設定することにより、1 つのテナントまたはソースのスパイクを制限して、そのテナントまたはソースのみに影響を与えることができます。例えば、ラベルセット `team=A env=prod` にアクティブ時系列制限を 1,000,000 に設定した場合、そのラベルセットに一致する取り込まれた時系列の数が制限を超えると、ラベルセットに一致する時

系列のみがスロットリングされます。これにより、他のテナントやメトリクスソースは影響を受けません。

Prometheus のラベルの詳細については、「データモデル」を参照してください。

- ワークスペースに保持するデータの日数を定義する保持期間を設定します。

ワークスペースを設定するには

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ページの左上隅にあるメニューアイコンを選択し、[すべての WorkSpaces] を選択します。
3. ワークスペースのワークスペース ID を選択します。
4. [ワークスペース構成] タブを選択します。
5. ワークスペースの保持期間を設定するには、[保持期間] セクションで [編集] を選択します。次に、新しい保持期間を日数で指定します。最大は 1095 日 (3 年) です。
6. ラベルセットとそのアクティブシリーズ制限を追加または変更するには、[ラベルセット] セクションの [編集] を選択します。次に、以下の操作を実行します。

   a. (オプション) [デフォルトのバケット制限] に値を入力し、定義されたラベルセットと一致しない時系列のみをカウントして、ワークスペースに取り込むことができるアクティブな時系列の最大数に制限を設定します。

   b. ラベルセットを定義するには、[アクティブシリーズ上限] の下に新しいラベルセットの非アクティブ時系列の上限を入力します。

      次に、ラベルセットで使用される 1 つのラベルにラベルと値を入力し、[ラベルを追加] を選択します。

   c. (オプション) 別のラベルセットを定義するには、[別のラベルセットを追加] を選択し、前のステップを繰り返します。
7. 完了したら、[変更を保存] を選択します。

ワークスペースエイリアスを編集する

ワークスペースを編集してエイリアスを変更できます。 AWS CLIを使用してワークスペースエイリアスを変更するには、次のコマンドを入力します。

```
aws amp update-workspace-alias --workspace-id my-workspace-id --alias "new-alias"
```

Amazon Managed Service for Prometheus コンソールを使用してワークスペースを編集するには

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ページの左上隅にあるメニューアイコンを選択し、[すべての WorkSpaces] を選択します。
3. 編集するワークスペースのワークスペース ID を選択し、[編集] を選択します。
4. ワークスペースの新しいエイリアスを入力し、[保存] を選択します。

Amazon Managed Service for Prometheus ワークスペースの詳細 (ARN を含む) を確認する

Amazon Managed Service for Prometheus ワークスペースの詳細は、 AWS コンソールまたは AWS CLIを使用して確認できます。

Console

Amazon Managed Service for Prometheus コンソールを使用してワークスペースの詳細を確認するには

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ページの左上隅にあるメニューアイコンを選択し、[すべての WorkSpaces] を選択します。
3. ワークスペースのワークスペース ID を選択します。これにより、以下のような、ワークスペースの詳細が表示されます。

   - 現在のステータス – ワークスペースのステータス ([アクティブ] など) が、[ステータス] の下に表示されます。
   - ARN - ワークスペースの ARN が [ARN] の下に表示されます。
   - ID – ワークスペースの ID が [ワークスペース ID] の下に表示されます。
   - URL – ワークスペースに対する書き込みやデータのクエリのための URL など、ワークスペースの複数の URL がコンソールに表示されます。

> **Note**
>
> デフォルトの場合、返される URL は IPv4 URL です。デュアルスタック (IPv4 および IPv6 をサポート) URLを使用することもできます。これらは同じですが、ドメインはデフォルトの amazonaws.com ではなく、api.aws になります。例えば、以下 (IPv4 URL) が表示された場合:
>
> ```
> https://aps-workspaces.us-east-1.amazonaws.com/workspaces/ws-abcd1234-ef56-7890-ab12-example/api/v1/remote_write
> ```
>
> 次のようにデュアルスタック (IPv6 のサポートを含む) URL を作成できます。
>
> ```
> https://aps-workspaces.us-east-1.api.aws/workspaces/ws-abcd1234-ef56-7890-ab12-example/api/v1/remote_write
> ```

このセクションの下には、ルール、アラートマネージャー、ログ、設定、タグに関する情報を含むタブがあります。

AWS CLI

を使用してワークスペースの詳細を検索するには AWS CLI

次のコマンドは、ワークスペースの詳細を返します。*my-workspace-id* は、詳細を確認する対象のワークスペースのワークスペース ID に置き換える必要があります。

```
aws amp describe-workspace --workspace-id my-workspace-id
```

これにより、以下のような、ワークスペースの詳細が返されます。

- 現在のステータス – ワークスペースのステータス (ACTIVE など) が statusCode プロパティで返されます。
- ARN – ワークスペース ARN が arn プロパティで返されます。
- URLs は、 prometheusEndpointプロパティ内のワークスペースの基本 URL AWS CLI を返します。

> **Note**
>
> デフォルトの場合、返される URL は IPv4 URL です。デュアルスタック (IPv4 および IPv6 をサポート) URL を、デフォルトの amazonaws.com ではなく、ドメイン api.aws で使用することもできます。例えば、以下 (IPv4 URL) が表示された場合:
>
> ```
> https://aps-workspaces.us-east-1.amazonaws.com/workspaces/ws-abcd1234-ef56-7890-ab12-example/
> ```
>
> 次のようにデュアルスタック (IPv6 のサポートを含む) URL を作成できます。
>
> ```
> https://aps-workspaces.us-east-1.api.aws/workspaces/ws-abcd1234-ef56-7890-ab12-example/
> ```
>
> ワークスペースのリモート書き込み URL およびクエリ URL を作成することもできます。この場合、/api/v1/remote_write または /api/v1/query をそれぞれ追加します。

Amazon Managed Service for Prometheus ワークスペースを削除する

ワークスペースを削除すると、ワークスペースに取り込んだデータも削除されます。

> **Note**
>
> Amazon Managed Service for Prometheus ワークスペースを削除しても、メトリクスをスクレイピングしてワークスペースに送信している AWS マネージドコレクターは自動的に削除されません。詳細については、「スクレイパーの検出と削除」を参照してください。

を使用してワークスペースを削除するには AWS CLI

以下のコマンドを使用します。

```
aws amp delete-workspace --workspace-id my-workspace-id
```

Amazon Managed Service for Prometheus コンソールを使用してワークスペースを削除するには

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ページの左上隅にあるメニューアイコンを選択し、[すべての WorkSpaces] を選択します。
3. 削除するワークスペースのワークスペース ID を選択し、[削除] を選択します。
4. 確認ボックスに **`delete`** と入力し、[削除] を選択します。

Amazon Managed Service for Prometheus ワークスペースにメトリクスを取り込む

メトリクスにクエリを実行したり、アラートを設定したりするには、事前にメトリクスを Amazon Managed Service for Prometheus ワークスペースに取り込む必要があります。このセクションでは、ワークスペースへのメトリクスの取り込みを設定する方法について説明します。

> **Note**
>
> ワークスペースに取り込まれたメトリクスは、デフォルトで 150 日間保存され、その後自動的に削除されます。最大 1095 日 (3 年) までワークスペースを設定して、保持期間を調整できます。詳細については、「ワークスペースの設定」を参照してください。

Amazon Managed Service for Prometheus ワークスペースにメトリクスを取り込むには、2 つの方法があります。

- AWS マネージドコレクターの使用 – Amazon Managed Service for Prometheus は、Amazon Elastic Kubernetes Service (Amazon EKS) クラスターからメトリクスを自動的にスクレイプする、フルマネージド型のエージェントレススクレイパーを提供します。スクレイピングは、Prometheus 互換エンドポイントからメトリクスを自動的にプルします。
- カスタマーマネージドコレクターの使用 — 独自のコレクターを管理するためのオプションは多数あります。使用する最も一般的なコレクターの 2 つは、Prometheus の独自のインスタンスのインストール、エージェントモードでの実行、または AWS Distro for OpenTelemetry の使用です。これらの 2 つについては、次のセクションで詳しく説明します。

  コレクターは Amazon Managed Service for Prometheus に Prometheus のリモート書き込み機能を使用してメトリクスを送信します。独自のアプリケーション内の Prometheus リモート書き込みを使用して、メトリクスを Amazon Managed Service for Prometheus に直接送信できます。リモート書き込みの直接使用とリモート書き込み設定の詳細については、「Prometheus ドキュメント」の「remote_write」を参照してください。

トピック

- AWS マネージドコレクターによるメトリクスの取り込み
- カスタマーマネージドコレクター

AWS マネージドコレクターによるメトリクスの取り込み

Amazon Managed Service for Prometheus の一般的なユースケースは、Amazon Elastic Kubernetes Service (Amazon EKS) によって管理される Kubernetes クラスターを監視することです。Kubernetes クラスターや Amazon EKS 内で実行される多くのアプリケーションは、Prometheus 互換のスクレイパーがアクセスできるようにメトリクスを自動的にエクスポートします。

> **Note**
>
> Amazon EKS は、クラスター内の API サーバーのメトリクス、`kube-controller-manager` メトリクス、`kube-scheduler` メトリクスを公開します。Kubernetes 環境で実行されるその他多くのテクノロジーやアプリケーションは、Prometheus 互換のメトリクスを提供しています。十分に文書化されたエクスポーターのリストについては、「Prometheus ドキュメント」の「Exporters and integrations」を参照してください。

Amazon Managed Service for Prometheus は、完全マネージド型のエージェントレススクレイパー (コレクター) を提供し、Prometheus 互換のメトリクスを自動的に検出して取得します。エージェントやスクレイパーを管理、インストール、パッチ適用、または保守する必要はありません。Amazon Managed Service for Prometheus コレクターは Amazon EKS クラスター用に、信頼性が高く、安定性があり、可用性が高く、自動的にスケーリングされるメトリクスのコレクションを提供します。Amazon Managed Service for Prometheus マネージドコレクターは、EC2 や Fargate などの Amazon EKS クラスターと連携します。

Amazon Managed Service for Prometheus コレクターは、スクレイパーの作成時に指定されたサブネットごとに Elastic Network Interface (ENI) を作成します。コレクターはこれらの ENI を介してメトリクスをスクレイピングし、`remote_write` を使って、VPC エンドポイントを使用して Amazon Managed Service for Prometheus ワークスペースにデータをプッシュします。スクレイピングされたデータが、パブリックインターネット上を移動することはありません。

以下のトピックでは、Amazon EKS クラスターで Amazon Managed Service for Prometheus コレクターを使用する方法と、収集されたメトリクスについて詳しく説明します。

トピック

- Amazon EKS のマネージドコレクターを設定する
- Amazon MSK 用のマネージド Prometheus コレクターを設定する

- Prometheus と互換性のあるメトリクスとはどのようなものですか。
- 提供されたログを使用してコレクターをモニタリングする

Amazon EKS のマネージドコレクターを設定する

Amazon Managed Service for Prometheus コレクターを使用するには、Amazon EKS クラスター内のメトリクスを検出して取得するスクレイパーを作成します。Amazon Managed Streaming for Apache Kafka と統合するスクレイパーを作成することもできます。詳細については、「Amazon MSK の統合」を参照してください。

- Amazon EKS クラスターを作成するときに、スクレイパーを作成できます。スクレイパーの作成を含め、Amazon EKS クラスターの作成に関する詳細については、「Amazon EKS ユーザーガイド」の「Amazon EKS クラスターの作成」を参照してください。
- AWS API を使用してプログラムで、または を使用して、独自のスクレイパーを作成できます AWS CLI。

Amazon Managed Service for Prometheus コレクターは、Prometheus と互換性のあるメトリクスをスクレイピングします。Prometheus 互換メトリクスの詳細については、「Prometheus と互換性のあるメトリクスとはどのようなものですか。」を参照してください。Amazon EKS クラスターは、API サーバーのメトリクスを公開します。Kubernete バージョン `1.28` 以降の Amazon EKS クラスターは、`kube-scheduler` と `kube-controller-manager` のメトリクスも公開します。詳細については、「Amazon EKS ユーザーガイド」の「コントロールプレーンの未加工メトリクスを Prometheus 形式で取得する」を参照してください。

> **Note**
>
> クラスターからメトリクスをスクレイピングすると、ネットワークの使用に対して料金が発生する場合があります。これらのコストを最適化する 1 つの方法は、提供されたメトリクスを圧縮 (gzip などを使用) するように `/metrics` エンドポイントを設定し、ネットワーク全体で移動する必要があるデータを減らすことです。これを行う方法は、メトリクスを提供するアプリケーションまたはライブラリによって異なります。一部のライブラリは、デフォルトで gzip を使用します。

以下のトピックでは、スクレイパーを作成、管理、および設定する方法について説明します。

トピック

- スクレイパーの作成
- Amazon EKS クラスターの設定
- スクレイパーの検出と削除
- スクレイパー設定
- スクレイパー設定のトラブルシューティング
- スクレイパーの制限事項

スクレイパーの作成

Amazon Managed Service for Prometheus コレクターは、Amazon EKS クラスターからメトリクスを検出して収集するスクレイパーで構成されています。Amazon Managed Service for Prometheus ではお客様に代わってスクレイパーが管理されます。インスタンス、エージェント、スクレイパーをご自身で管理しなくても、必要なスケーラビリティ、セキュリティ、信頼性を実現できます。

スクレイパーを作成する方法は 3 つあります。

- Amazon EKS コンソールから Amazon EKS クラスターを作成して、Prometheus メトリクスをオンにすると、スクレイパーが自動的に作成されます。
- スクレイパーは、既存のクラスターの Amazon EKS コンソールから作成できます。Amazon EKS コンソールでクラスターを開き、[オブザーバビリティ] タブで [スクレイパーを追加] を選択します。

  使用可能な設定の詳細については、「Amazon EKS ユーザーガイド」の「Prometheus メトリクスを有効にする」を参照してください。
- AWS API または を使用してスクレイパーを作成できます AWS CLI。

  これらのオプションについて、次の手順で説明します。

独自のスクレイパーを作成するには、いくつかの前提条件があります。

- Amazon EKS クラスターが作成済みである必要があります。
- Amazon EKS クラスターは、クラスターエンドポイントアクセスコントロールがプライベートアクセスを含むように設定されている必要があります。プライベートとパブリックを含めることができますが、プライベートを含める必要があります。
- Amazon EKS クラスターが存在する Amazon VPC では、DNS が有効になっている必要があります。

> **Note**
>
> クラスターは、Amazon リソースネーム (ARN) によってスクレイパーと関連付けられます。クラスターを削除し、同じ名前で新しいクラスターを作成すると、ARN は新しいクラスターで再利用されます。このため、スクレイパーは新しいクラスターのメトリクスを収集しようとします。スクレイパーの削除は、クラスターの削除とは別個に行います。

AWS API

AWS API を使用してスクレイパーを作成するには

`CreateScraper` API オペレーションを使用して、 AWS API でスクレイパーを作成します。次の例では、`us-west-2` リージョンでスクレイパーを作成します。 AWS アカウント、ワークスペース、セキュリティ、Amazon EKS クラスター情報を独自の ID に置き換え、スクレイパーに使用する設定を指定する必要があります。

> **Note**
>
> セキュリティグループとサブネットは、接続先のクラスターのセキュリティグループとサブネットに設定する必要があります。
> 少なくとも 2 つ以上のアベイラビリティーゾーンにある 2 つ以上のサブネットを含める必要があります。

`scrapeConfiguration` は、base64 でエンコードされた Prometheus 設定 YAML ファイルです。`GetDefaultScraperConfiguration` API オペレーションで汎用設定をダウンロードできます。`scrapeConfiguration` の形式の詳細については、「スクレイパー設定」を参照してください。

```
POST /scrapers HTTP/1.1
Content-Length: 415
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: aws-cli/1.18.147 Python/2.7.18 Linux/5.4.58-37.125.amzn2int.x86_64
 botocore/1.18.6

{
    "alias": "myScraper",
```

```
    "destination":  {
        "ampConfiguration": {
            "workspaceArn": "arn:aws:aps:us-west-2:account-id:workspace/
ws-workspace-id"
        }
    },
    "source": {
        "eksConfiguration": {
            "clusterArn": "arn:aws:eks:us-west-2:account-id:cluster/cluster-name",
            "securityGroupIds": ["sg-security-group-id"],
            "subnetIds": ["subnet-subnet-id-1", "subnet-subnet-id-2"]
        }
    },
    "scrapeConfiguration": {
        "configurationBlob": <base64-encoded-blob>
    }
}
```

AWS CLI

AWS CLIを使用してスクレイパーを作成するには

`create-scraper` コマンドを使用して AWS CLIでスクレイパーを作成します。次の例では、`us-west-2` リージョンでスクレイパーを作成します。 AWS アカウント、ワークスペース、セキュリティ、Amazon EKS クラスター情報を独自の ID に置き換え、スクレイパーに使用する設定を指定する必要があります。

> **Note**
>
> セキュリティグループとサブネットは、接続先のクラスターのセキュリティグループとサブネットに設定する必要があります。
> 少なくとも 2 つ以上のアベイラビリティーゾーンにある 2 つ以上のサブネットを含める必要があります。

`scrape-configuration` は、base64 でエンコードされた Prometheus 設定 YAML ファイルです。汎用設定は、`get-default-scraper-configuration` コマンドを使用してダウンロードできます。`scrape-configuration` の形式の詳細については、「スクレイパー設定」を参照してください。

```
aws amp create-scraper \
```

```
  --source eksConfiguration="{clusterArn='arn:aws:eks:us-west-2:account-
id:cluster/cluster-name', securityGroupIds=['sg-security-group-
id'],subnetIds=['subnet-subnet-id-1', 'subnet-subnet-id-2']}" \
  --scrape-configuration configurationBlob=<base64-encoded-blob> \
  --destination ampConfiguration="{workspaceArn='arn:aws:aps:us-west-2:account-
id:workspace/ws-workspace-id'}"
```

以下は、 AWS API で使用できるスクレイパーオペレーションの完全なリストです。

- CreateScraper API オペレーションを使用してスクレイパーを作成します。
- ListScrapers API オペレーションを使用して既存のスクレイパーを一覧表示します。
- UpdateScraper API オペレーションを使用して、スクレイパーのエイリアス、設定、または送信先を更新します。
- DeleteScraper API オペレーションを使用してスクレイパーを削除します。
- DescribeScraper API オペレーションを使用してスクレイパーの詳細を取得します。
- GetDefaultScraperConfiguration API オペレーションを使用してスクレイパーの汎用設定を取得します。

> **Note**
>
> スクレイピングする Amazon EKS クラスターは、Amazon Managed Service for Prometheus がメトリクスにアクセスできるように設定されている必要があります。次のトピックでは、クラスターの設定方法について説明します。

クロスアカウントの設定

Amazon EKS クラスターと Amazon Managed Service for Prometheus ワークスペースが異なるアカウントにあるときにクロスアカウントスクレイパーを作成するには、次の手順を使用します。例えば、Amazon EKS クラスターを含むソースアカウント `account_id_source` と、Amazon Managed Service for Prometheus ワークスペースを含むターゲットアカウント `account_id_target` があるとします。

クロスアカウント設定でスクレイパーを作成するには

1. ソースアカウントで、ロール `arn:aws:iam::account_id_source:role/Source` を作成し、次の信頼ポリシーを追加します。

```
{
    "Effect": "Allow",
    "Principal": {
    "Service": [
        "scraper.aps.amazonaws.com"
     ]
    },
    "Action": "sts:AssumeRole",
    "Condition": {
        "ArnEquals": {
            "aws:SourceArn": "scraper_ARN"
        },
        "StringEquals": {
            "AWS:SourceAccount": "account_id"
        }
    }
}
```

2. ソース (Amazon EKS クラスター) とターゲット (Amazon Managed Service for Prometheus ワークスペース) のすべての組み合わせで、ロール `arn:aws:iam::account_id_target:role/Target` を作成し、AmazonPrometheusRemoteWriteAccess のアクセス許可を持つ次の信頼ポリシーを追加する必要があります。

```
{
  "Effect": "Allow",
  "Principal": {
     "AWS": "arn:aws:iam::account_id_source:role/Source"
  },
  "Action": "sts:AssumeRole",
  "Condition": {
     "StringEquals": {
        "sts:ExternalId": "scraper_ARN"
      }
  }
}
```

3. `--role-configuration` オプションを使用してスクレイパーを作成します。

```
aws amp create-scraper \
  --source eksConfiguration="{clusterArn='arn:aws:eks:us-west-2:account-
id_source:cluster/xarw,subnetIds=[subnet-subnet-id]}" \
  --scrape-configuration configurationBlob=<base64-encoded-blob> \
  --destination ampConfiguration="{workspaceArn='arn:aws:aps:us-west-2:account-
id_target:workspace/ws-workspace-id'}"\
  --role-configuration '{"sourceRoleArn":"arn:aws:iam::account-id_source:role/
Source", "targetRoleArn":"arn:aws:iam::account-id_target:role/Target"}'
```

4. スクレイパーの作成を検証します。

```
aws amp list-scrapers
{
    "scrapers": [
        {
            "scraperId": "scraper-id",
            "arn": "arn:aws:aps:us-west-2:account_id_source:scraper/scraper-id",
            "roleArn": "arn:aws:iam::account_id_source:role/aws-service-role/
scraper.aps.amazonaws.com/
AWSServiceRoleForAmazonPrometheusScraperInternal_cc319052-41a3-4",
            "status": {
                "statusCode": "ACTIVE"
            },
            "createdAt": "2024-10-29T16:37:58.789000+00:00",
            "lastModifiedAt": "2024-10-29T16:55:17.085000+00:00",
            "tags": {},
            "source": {
                "eksConfiguration": {
                    "clusterArn": "arn:aws:eks:us-west-2:account_id_source:cluster/
xarw",
                    "securityGroupIds": [
                        "sg-security-group-id",
                        "sg-security-group-id"
                    ],
                    "subnetIds": [
                        "subnet-subnet_id"
                    ]
                }
            },
            "destination": {
                "ampConfiguration": {
```

```
                    "workspaceArn": "arn:aws:aps:us-
west-2:account_id_target:workspace/ws-workspace-id"
                }
            }
        }
    ]
}
```

RoleConfiguration とサービスにリンクされたロールの間の変更

RoleConfiguration ではなくサービスにリンクされたロールに切り替えて Amazon Managed Service for Prometheus ワークスペースに書き込む場合は、UpdateScraper を更新し、RoleConfiguration なしでスクレイパーと同じアカウントのワークスペースを提供する必要があります。RoleConfiguration はスクレイパーから削除され、サービスにリンクされたロールが使用されます。

スクレイパーと同じアカウントのワークスペースを変更し、引き続き RoleConfiguration を使用する場合は、UpdateScraper で RoleConfiguration を再度指定する必要があります。

カスタマーマネージドキーで有効になっているワークスペース用のスクレイパーの作成

カスタマーマネージドキーを使用して Amazon Managed Service for Prometheus ワークスペースにメトリクスを取り込むスクレイパーを作成するには、ソースとターゲットの両方を同じアカウントに設定した --role-configuration を使用します。

```
aws amp create-scraper \
  --source eksConfiguration="{clusterArn='arn:aws:eks:us-west-2:account-id:cluster/
xarw,subnetIds=[subnet-subnet_id]}" \
  --scrape-configuration configurationBlob=<base64-encoded-blob> \
  --destination ampConfiguration="{workspaceArn='arn:aws:aps:us-west-2:account-
id:workspace/ws-workspace-id'}"\
  --role-configuration '{"sourceRoleArn":"arn:aws:iam::account_id:role/Source",
 "targetRoleArn":"arn:aws:iam::account_id:role/Target"}'
```

スクレイパー作成時の一般的なエラー

以下は、新しいスクレイパーの作成時に発生する場合がある最も一般的な問題です。

- 必要な AWS リソースが存在しません。指定したセキュリティグループ、サブネット、Amazon EKS クラスターが存在している必要があります。
- IP アドレス領域が不足しています。`CreateScraper` API に渡すサブネットごとに、少なくとも 1 つの IP アドレスが必要です。

Amazon EKS クラスターの設定

Amazon EKS クラスターは、スクレイパーがメトリクスにアクセスできるように設定する必要があります。この設定には 2 つのオプションがあります。

- Amazon EKS のアクセスエントリを使用して、Amazon Managed Service for Prometheus コレクターにクラスターへのアクセス権を自動的に付与します。
- Amazon EKS クラスターをマネージドメトリクススクレイピング用に手動で設定します。

以下のトピックで、これらの各手順について詳しく説明します。

アクセスエントリを使用してスクレイパーアクセス用に Amazon EKS を設定する

Amazon EKS のアクセスエントリを使用することは、クラスターからメトリクスをスクレイピングするためのアクセス権を Amazon Managed Service for Prometheus に付与する最も簡単な方法です。

スクレイピングする Amazon EKS クラスターは、API 認証を許可するように設定する必要があります。クラスター認証モードは、`API` または `API_AND_CONFIG_MAP` に設定する必要があります。これは、Amazon EKS コンソールのクラスター詳細の [アクセス設定] タブで確認できます。詳細については、「Amazon EKS ユーザーガイド」の「Amazon EKS クラスターで Kubernetes オブジェクトへのアクセスを IAM ロールまたはユーザーに許可する」を参照してください。

クラスターの作成時または作成後にスクレイパーを作成できます。

- クラスターの作成時 - Amazon EKS コンソールを使用して Amazon EKS クラスターを作成するときに (クラスターの一部としてスクレイパーを作成する手順に従って)、このアクセスを設定できます。アクセスエントリポリシーが自動的に作成され、クラスターメトリクスへのアクセスが Amazon Managed Service for Prometheus に許可されます。
- クラスターの作成後に追加 - Amazon EKS クラスターが既に存在する場合は、認証モードを `API` または `API_AND_CONFIG_MAP` に設定します。Amazon Managed Service for Prometheus API または CLI を使用して、または Amazon EKS コンソールを使用して作成したスクレイパーには、自

動的に正しいアクセスエントリポリシーが作成され、スクレイパーからクラスターにアクセスできるようになります。

アクセスエントリポリシーの作成

スクレイパーを作成し、Amazon Managed Service for Prometheus でアクセスエントリポリシーを自動的に生成できるようにすると、次のポリシーが生成されます。アクセスエントリの詳細については、「Amazon EKS ユーザーガイド」の「IAM ロールまたはユーザーに Kubernetes へのアクセスを許可する」を参照してください。

```
{
    "rules": [
        {
            "effect": "allow",
            "apiGroups": [
                ""
            ],
            "resources": [
                "nodes",
                "nodes/proxy",
                "nodes/metrics",
                "services",
                "endpoints",
                "pods",
                "ingresses",
                "configmaps"
            ],
            "verbs": [
                "get",
                "list",
                "watch"
            ]
        },
        {
            "effect": "allow",
            "apiGroups": [
                "extensions",
                "networking.k8s.io"
            ],
            "resources": [
                "ingresses/status",
                "ingresses"
```

```
            ],
            "verbs": [
                "get",
                "list",
                "watch"
            ]
        },
        {
            "effect": "allow",
            "apiGroups": [
                "metrics.eks.amazonaws.com"
            ],
            "resources": [
                "kcm/metrics",
                "ksh/metrics"
            ],
            "verbs": [
                "get"
            ]
        },
        {
            "effect": "allow",
            "nonResourceURLs": [
                "/metrics"
            ],
            "verbs": [
                "get"
            ]
        }
    ]
}
```

スクレイパーアクセス用に Amazon EKS を手動で設定する

`aws-auth ConfigMap` を使用して kubernetes クラスターへのアクセスを制御する場合でも、Amazon Managed Service for Prometheus スクレイパーにメトリクスへのアクセスを許可できます。次の手順では、Amazon Managed Service for Prometheus に対して、Amazon EKS クラスターのメトリクスをスクレイピングするためのアクセス権を付与します。

> **Note**
>
> ConfigMap およびアクセスエントリの詳細については、「Amazon EKS ユーザーガイド」の「IAM ロールまたはユーザーに Kubernetes へのアクセスを許可する」を参照してください。

この手順では、 kubectlと CLI AWS を使用します。kubectl のインストールの詳細については、「Amazon EKS ユーザーガイド」の「kubectl のインストール」を参照してください。

Amazon EKS クラスターをマネージドメトリクスクレイピング用に手動で設定するには

1. clusterrole-binding.yml という名前のファイルを作成し、次のテキストを記述します。

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: aps-collector-role
rules:
  - apiGroups: [""]
    resources: ["nodes", "nodes/proxy", "nodes/metrics", "services", "endpoints",
 "pods", "ingresses", "configmaps"]
    verbs: ["describe", "get", "list", "watch"]
  - apiGroups: ["extensions", "networking.k8s.io"]
    resources: ["ingresses/status", "ingresses"]
    verbs: ["describe", "get", "list", "watch"]
  - nonResourceURLs: ["/metrics"]
    verbs: ["get"]
  - apiGroups: ["metrics.eks.amazonaws.com"]
    resources: ["kcm/metrics", "ksh/metrics"]
    verbs: ["get"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: aps-collector-user-role-binding
subjects:
- kind: User
  name: aps-collector-user
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
```

```
  name: aps-collector-role
  apiGroup: rbac.authorization.k8s.io
```

2. クラスターで次のコマンドを実行します。

```
kubectl apply -f clusterrole-binding.yml
```

これにより、クラスターのロールバインディングとルールが作成されます。この例では、aps-collector-role をロール名、aps-collector-user をユーザー名として使用しています。

3. 次のコマンドは、*scraper-id* という ID のスクレイパーに関する情報を提供します。これは、前のセクションのコマンドを使用して作成したスクレイパーです。

```
aws amp describe-scraper --scraper-id scraper-id
```

4. describe-scraper の結果から roleArn を探します。この形式は次のようになります。

```
arn:aws:iam::account-id:role/aws-service-role/scraper.aps.amazonaws.com/
AWSServiceRoleForAmazonPrometheusScraper_unique-id
```

Amazon EKS では、この ARN に別の形式が必要です。次のステップで使用するために、返される ARN の形式を調整する必要があります。この形式に合わせて編集してください。

```
arn:aws:iam::account-id:role/AWSServiceRoleForAmazonPrometheusScraper_unique-id
```

例えば、この ARN の場合、

```
arn:aws:iam::111122223333:role/aws-service-role/scraper.aps.amazonaws.com/
AWSServiceRoleForAmazonPrometheusScraper_1234abcd-56ef-7
```

以下のように記述する必要があります。

```
arn:aws:iam::111122223333:role/
AWSServiceRoleForAmazonPrometheusScraper_1234abcd-56ef-7
```

5. 前のステップで変更した roleArn と、クラスター名およびリージョンを使用して、クラスター内で以下のコマンドを実行します。

```
eksctl create iamidentitymapping --cluster cluster-name --region region-id --
arn roleArn --username aps-collector-user
```

これにより、スクレイパーは clusterrole-binding.yml ファイルに作成したロールとユーザーを使用してクラスターにアクセスできます。

スクレイパーの検出と削除

AWS API または を使用して AWS CLI 、アカウントのスクレイパーを一覧表示したり、削除したりできます。

> **Note**
>
> 最新バージョンの AWS CLI または SDK を使用していることを確認します。最新バージョンには、最新の特長と機能に加え、セキュリティアップデートも含まれています。または、常に最新のコマンドラインエクスペリエンスを提供する AWS CloudShell を自動的に使用します。

アカウント内のすべてのスクレイパーを一覧表示するには、ListScrapers API オペレーションを使用します。

または、 を使用して AWS CLIを呼び出します。

```
aws amp list-scrapers --region aws-region
```

ListScrapers は、アカウント内のすべてのスクレイパーを返します。例:

```
{
    "scrapers": [
        {
            "scraperId": "s-1234abcd-56ef-7890-abcd-1234ef567890",
            "arn": "arn:aws:aps:us-west-2:123456789012:scraper/s-1234abcd-56ef-7890-
abcd-1234ef567890",
            "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
AWSServiceRoleForAmazonPrometheusScraper_1234abcd-2931",
            "status": {
                "statusCode": "DELETING"
            },
```

```
            "createdAt": "2023-10-12T15:22:19.014000-07:00",
            "lastModifiedAt": "2023-10-12T15:55:43.487000-07:00",
            "tags": {},
            "source": {
                "eksConfiguration": {
                    "clusterArn": "arn:aws:eks:us-west-2:123456789012:cluster/my-
cluster",
                    "securityGroupIds": [
                        "sg-1234abcd5678ef90"
                    ],
                    "subnetIds": [
                        "subnet-abcd1234ef567890",
                        "subnet-1234abcd5678ab90"
                    ]
                }
            },
            "destination": {
                "ampConfiguration": {
                    "workspaceArn": "arn:aws:aps:us-west-2:123456789012:workspace/
ws-1234abcd-5678-ef90-ab12-cdef3456a78"
                }
            }
        }
    ]
}
```

スクレイパーを削除するには、ListScrapers オペレーションを使用して削除するスクレイパーの scraperId を見つけ、DeleteScraper オペレーションを使用して削除します。

または、 を使用して AWS CLIを呼び出します。

```
aws amp delete-scraper --scraper-id scraperId
```

スクレイパー設定

Prometheus 互換のスクレイパー設定を使用して、スクレイパーがメトリクスを検出して収集する方法を制御できます。例えば、メトリクスをワークスペースに送信する間隔を変更できます。再ラベル付けを使用して、メトリクスのラベルを動的に書き換えることもできます。スクレイパー設定は、スクレイパーの定義の一部である YAML ファイルです。

新しいスクレイパーを作成したら、API コールで base64 でエンコードされた YAML ファイルを提供して設定を指定します。Amazon Managed Service for Prometheus API の

GetDefaultScraperConfiguration オペレーションを含む汎用設定ファイルをダウンロードできます。

スクレイパーの設定を変更するために、UpdateScraper オペレーションを使用できます。メトリクスのソース (別の Amazon EKS クラスターなど) を更新する必要がある場合は、スクレイパーを削除し、新しいソースで再作成する必要があります。

サポートされている設定

スクレイパーの設定形式に関する情報 (可能な値の詳細な内訳を含む) については、Prometheus ドキュメントの「Configuration」を参照してください。グローバル設定オプションと <scrape_config> オプションには、最も一般的に必要なオプションが記載されています。

サポートされているサービスは Amazon EKS のみであるため、サポートされるサービス検出設定 (<*_sd_config>) は <kubernetes_sd_config> のみです。

許可される設定セクションの完全なリスト:

- <global>
- <scrape_config>
- <static_config>
- <relabel_config>
- <metric_relabel_configs>
- <kubernetes_sd_config>

これらのセクション内の制限は、サンプル設定ファイルの後に一覧表示されます。

設定ファイルの例

以下は、30 秒のスクレイプ間隔の YAML 設定ファイルのサンプルです。このサンプルには、kube API サーバーメトリクスと、kube-controller-manager および kube-scheduler メトリクスのサポートが含まれています。詳細については、「Amazon EKS ユーザーガイド」の「コントロールプレーンの未加工メトリクスを Prometheus 形式で取得する」を参照してください。

```
global:
   scrape_interval: 30s
   external_labels:
     clusterArn: apiserver-test-2
scrape_configs:
  - job_name: pod_exporter
```

```
    kubernetes_sd_configs:
      - role: pod
  - job_name: cadvisor
    scheme: https
    authorization:
      type: Bearer
      credentials_file: /var/run/secrets/kubernetes.io/serviceaccount/token
    kubernetes_sd_configs:
      - role: node
    relabel_configs:
      - action: labelmap
        regex: __meta_kubernetes_node_label_(.+)
      - replacement: kubernetes.default.svc:443
        target_label: __address__
      - source_labels: [__meta_kubernetes_node_name]
        regex: (.+)
        target_label: __metrics_path__
        replacement: /api/v1/nodes/$1/proxy/metrics/cadvisor
  # apiserver metrics
  - scheme: https
    authorization:
      type: Bearer
      credentials_file: /var/run/secrets/kubernetes.io/serviceaccount/token
    job_name: kubernetes-apiservers
    kubernetes_sd_configs:
    - role: endpoints
    relabel_configs:
    - action: keep
      regex: default;kubernetes;https
      source_labels:
      - __meta_kubernetes_namespace
      - __meta_kubernetes_service_name
      - __meta_kubernetes_endpoint_port_name
  # kube proxy metrics
  - job_name: kube-proxy
    honor_labels: true
    kubernetes_sd_configs:
    - role: pod
    relabel_configs:
    - action: keep
      source_labels:
      - __meta_kubernetes_namespace
      - __meta_kubernetes_pod_name
      separator: '/'
```

```
      regex: 'kube-system/kube-proxy.+'
    - source_labels:
      - __address__
      action: replace
      target_label: __address__
      regex: (.+?)(\\:\\d+)?
      replacement: $1:10249
  # Scheduler metrics
  - job_name: 'ksh-metrics'
    kubernetes_sd_configs:
    - role: endpoints
    metrics_path: /apis/metrics.eks.amazonaws.com/v1/ksh/container/metrics
    scheme: https
    bearer_token_file: /var/run/secrets/kubernetes.io/serviceaccount/token
    relabel_configs:
    - source_labels:
      - __meta_kubernetes_namespace
      - __meta_kubernetes_service_name
      - __meta_kubernetes_endpoint_port_name
      action: keep
      regex: default;kubernetes;https
  # Controller Manager metrics
  - job_name: 'kcm-metrics'
    kubernetes_sd_configs:
    - role: endpoints
    metrics_path: /apis/metrics.eks.amazonaws.com/v1/kcm/container/metrics
    scheme: https
    bearer_token_file: /var/run/secrets/kubernetes.io/serviceaccount/token
    relabel_configs:
    - source_labels:
      - __meta_kubernetes_namespace
      - __meta_kubernetes_service_name
      - __meta_kubernetes_endpoint_port_name
      action: keep
      regex: default;kubernetes;https
```

以下は、 AWS マネージドコレクターに固有の制限です。

- スクレイプ間隔 — スクレイパー設定では、30 秒未満のスクレイプ間隔を指定できません。
- ターゲット — `static_config` 内のターゲットは IP アドレスとして指定する必要があります。

- DNS 解決 — ターゲット名に関連して、この設定で認識されるサーバー名は Kubernetes api サーバー `kubernetes.default.svc` のみです。他のすべてのマシン名は IP アドレスで指定する必要があります。
- 認可 - 認可が必要ない場合は省略します。必要な場合、認可は `Bearer` でなければならず、ファイル `/var/run/secrets/kubernetes.io/serviceaccount/token` を指す必要があります。つまり、使用する場合、認可セクションは次のようになる必要があります。

```
authorization:
  type: Bearer
  credentials_file: /var/run/secrets/kubernetes.io/serviceaccount/token
```

> **Note**
>
> `type: Bearer` はデフォルトであるため、省略できます。

スクレイパー設定のトラブルシューティング

Amazon Managed Service for Prometheus コレクターは、メトリクスの検出と収集を自動的に行います。しかし、予想したメトリクスが Amazon Managed Service for Prometheus ワークスペースに表示されない場合、どのようにトラブルシューティングできるでしょうか。

> **Important**
>
> Amazon EKS クラスターのプライベートアクセスが有効になっていることを確認します。詳細については、「Amazon EKS ユーザーガイド」の「クラスタープライベートエンドポイント」を参照してください。

up メトリクスは便利なツールです。Amazon Managed Service for Prometheus コレクターが検出した各エンドポイントについて、このメトリクスは自動的に送信されます。このメトリクスには 3 つの状態があり、コレクター内で発生している問題のトラブルシューティングに役立ちます。

- up が存在しない — エンドポイントの up メトリクスが存在しない場合、コレクターがエンドポイントを検出できなかったことを意味します。

  エンドポイントが存在することが確実な場合、コレクターがエンドポイントを検出できない理由はいくつかあります。

  - スクレイプ設定を調整する必要がある場合があります。検出 `relabel_config` を調整する必要がある場合があります。
  - 検出に使用される `role` に問題がある可能性があります。
  - Amazon EKS クラスターで使用される Amazon VPC で DNS が有効になっていない場合があり、コレクターがエンドポイントを検出できない可能性があります。
- up は存在するものの、常に 0 — up が存在するが 0 の場合、コレクターはエンドポイントを検出できますが、Prometheus 互換のメトリクスを検出できません。

  この場合は、`curl` エンドポイントに対して直接コマンドを実行してみるといいかもしれません。プロトコル (`http` または `https`)、エンドポイント、使用しているポートなど、詳細が正しいことを検証できます。また、エンドポイントの応答が有効な `200` レスポンスであり、Prometheus 形式に従っていることを確認することもできます。最後に、レスポンスの本文を最大許容サイズより大きくすることはできません (AWS マネージドコレクターの制限については、次のセクションを参照してください) 。
- up が存在し、0 より大きい — up が存在し、かつ 0 より大きい場合、メトリクスは Amazon Managed Service for Prometheus に送信されています。

  Amazon Managed Service for Prometheus (または Amazon Managed Grafana などの代替ダッシュボード) で正しいメトリクスを検出していることを確認します。curl をもう一度使用して、`/metrics` エンドポイントに予想したデータがあるかどうかを確認できます。また、スクレイパーあたりのエンドポイント数など、他の制限を超えていないことも確認してください。スクレイピングされるメトリクスエンドポイントの数を調べるには、`count(up)` を使用して up メトリクスの数を確認します。

スクレイパーの制限事項

Amazon Managed Service for Prometheus が提供するフルマネージド型スクレイパーには、いくつかの制限があります。

- リージョン — EKS クラスター、マネージドスクレイパー、Amazon Managed Service for Prometheus ワークスペースはすべて同じ AWS リージョンにある必要があります。
- コレクター — 1 リージョンの 1 アカウントあたり、最大 10 個の Amazon Managed Service for Prometheus スクレイパーを設定できます。

> **Note**
>
> クォータの引き上げをリクエストすることで、この上限を引き上げることができます。

- メトリクスレスポンス — 任意の 1 つの `/metrics` エンドポイントリクエストからのレスポンスの本文は 50 メガバイト (MB) を超えることはできません。
- スクレイパーあたりのエンドポイント — スクレイパーは最大 30,000 の `/metrics` エンドポイントをスクレイピングできます。
- スクレイプ間隔 — スクレイパー設定では、30 秒未満のスクレイプ間隔を指定できません。

Amazon MSK 用のマネージド Prometheus コレクターを設定する

Amazon Managed Service for Prometheus コレクターを使用するには、Amazon Managed Streaming for Apache Kafka クラスター内のメトリクスを検出して取得するスクレイパーを作成します。Amazon Elastic Kubernetes Service と統合するスクレイパーを作成することもできます。詳細については、「Amazon EKS の統合」を参照してください。

スクレイパーの作成

Amazon Managed Service for Prometheus コレクターは、Amazon MSK クラスターからメトリクスを検出して収集するスクレイパーで構成されています。Amazon Managed Service for Prometheus ではお客様に代わってスクレイパーが管理されます。インスタンス、エージェント、スクレイパーをご自身で管理しなくても、必要なスケーラビリティ、セキュリティ、信頼性を実現できます。

スクレイパーは、次の手順で説明 AWS CLI するように、 AWS API または を使用して作成できます。

独自のスクレイパーを作成するには、いくつかの前提条件があります。

- Amazon MSK クラスターが作成済みである必要があります。
- スクレイパーは Prometheus メトリクスを収集するためにこれらの DNS レコードにアクセスする必要があるため、Amazon VPC 内のポート 11001 (JMX Exporter) および 11002 (Node Exporter) でのインバウンドトラフィックを許可するように Amazon MSK クラスターのセキュリティグループを設定します。
- Amazon MSK クラスターが存在する Amazon VPC では、DNS が有効になっている必要があります。

> **Note**
>
> クラスターは、Amazon リソースネーム (ARN) によってスクレイパーと関連付けられます。クラスターを削除し、同じ名前で新しいクラスターを作成すると、ARN は新しいクラスターで再利用されます。このため、スクレイパーは新しいクラスターのメトリクスを収集しようとします。スクレイパーの削除は、クラスターの削除とは別個に行います。

To create a scraper using the AWS API

`CreateScraper` API オペレーションを使用して、 AWS API でスクレイパーを作成します。次の例のコマンドは、米国東部 (バージニア北部) リージョンにスクレイパーを作成します。`###` `#`コンテンツを Amazon MSK クラスター情報に置き換え、スクレイパー設定を指定します。

> **Note**
>
> ターゲットクラスターと一致するようにセキュリティグループとサブネットを設定します。2 つのアベイラビリティーゾーンにまたがって少なくとも 2 つのサブネットが含まれます。

```
               POST /scrapers HTTP/1.1
Content-Length: 415
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: aws-cli/1.18.147 Python/2.7.18 Linux/5.4.58-37.125.amzn2int.x86_64
 botocore/1.18.6

{
    "alias": "myScraper",
    "destination":  {
        "ampConfiguration": {
            "workspaceArn": "arn:aws:aps:us-east-1:123456789012:workspace/ws-
workspace-id"
        }
    },
    "source": {
        "vpcConfiguration": {
            "securityGroupIds": ["sg-security-group-id"],
```

```
            "subnetIds": ["subnet-subnet-id-1", "subnet-subnet-id-2"]
        }
    },
    "scrapeConfiguration": {
        "configurationBlob": base64-encoded-blob
    }
}
```

この例では、scrapeConfiguration パラメータには、MSK クラスターの DNS レコードを指定する base64 でエンコードされた Prometheus 設定 YAML ファイルが必要です。

各 DNS レコードは、特定のアベイラビリティーゾーンのブローカーエンドポイントを表し、クライアントは選択した AZ 全体に分散されたブローカーに接続して高可用性を実現できます。

MSK クラスタープロパティの DNS レコードの数は、クラスター設定のブローカーノードとアベイラビリティーゾーンの数に対応します。

- デフォルト設定 – 3 つの AZ にまたがる 3 つのブローカーノード = 3 つの DNS レコード
- カスタム設定 – 2 つの AZ にまたがる 2 つのブローカーノード = 2 つの DNS レコード

MSK クラスターの DNS レコードを取得するには、https://console.aws.amazon.com/msk/home?region=us-east-1#/home/ で、MSK コンソールを開きます。MSK クラスターに移動します。[プロパティ]、[ブローカー]、[エンドポイント] を選択します。

MSK クラスターからメトリクスをスクレイプするように Prometheus を設定するには、次の 2 つのオプションがあります。

1. クラスターレベルの DNS 解決 (推奨) – クラスターのベース DNS 名を使用して、すべてのブローカーを自動的に検出します。ブローカーエンドポイントが b-1.clusterName.xxx.xxx.xxx の場合、DNS レコードとして clusterName.xxx.xxx.xxx を使用します。これにより、Prometheus はクラスター内のすべてのブローカーを自動的にスクレイプできます。

   個々のブローカーエンドポイント – きめ細かな制御のために各ブローカーエンドポイントを個別に指定します。設定で完全なブローカー識別子 (b-1、b-2) を使用します。例えば、次のようになります。

   ```
   dns_sd_configs:
     - names:
   ```

```
    - b-1.clusterName.xxx.xxx.xxx
    - b-2.clusterName.xxx.xxx.xxx
    - b-3.clusterName.xxx.xxx.xxx
```

> **Note**
>
> を AWS コンソールの実際の MSK クラスターエンドポイントclusterName.xxx.xxx.xxxに置き換えます。

詳細については、Prometheus ドキュメントの「<dns_sd_config>」を参照してください。

以下はスクレイパー設定ファイルの例です。

```
global:
  scrape_interval: 30s
  external_labels:
    clusterArn: msk-test-1

scrape_configs:
  - job_name: msk-jmx
    scheme: http
    metrics_path: /metrics
    scrape_timeout: 10s
    dns_sd_configs:
      - names:
          - dns-record-1
          - dns-record-2
          - dns-record-3
        type: A
        port: 11001
    relabel_configs:
      - source_labels: [__meta_dns_name]
        target_label: broker_dns
      - source_labels: [__address__]
        target_label: instance
        regex: '(.*)'
        replacement: '${1}'

  - job_name: msk-node
    scheme: http
    metrics_path: /metrics
```

```
    scrape_timeout: 10s
    dns_sd_configs:
      - names:
          - dns-record-1
          - dns-record-2
          - dns-record-3
        type: A
        port: 11002
    relabel_configs:
      - source_labels: [__meta_dns_name]
        target_label: broker_dns
      - source_labels: [__address__]
        target_label: instance
        regex: '(.*)'
        replacement: '${1}'
```

次のいずれかのコマンドを実行して、YAML ファイルを base64 に変換します。任意のオンライン base64 コンバーターを使用してファイルを変換することもできます。

Example Linux/macOS

```
echo -n scraper config updated with dns records | base64
```

Example Windows PowerShell

```
[Convert]::ToBase64String([System.Text.Encoding]::UTF8.GetBytes(scraper config
 updated with dns records))
```

To create a scraper using the AWS CLI

`create-scraper` コマンドを使用して AWS Command Line Interfaceでスクレイパーを作成します。次の例のコマンドは、米国東部 (バージニア北部) リージョンにスクレイパーを作成します。*####*コンテンツを Amazon MSK クラスター情報に置き換え、スクレイパー設定を指定します。

> **Note**
>
> ターゲットクラスターと一致するようにセキュリティグループとサブネットを設定します。2 つのアベイラビリティーゾーンにまたがって少なくとも 2 つのサブネットが含まれます。

```
aws amp create-scraper \
 --source vpcConfiguration="{securityGroupIds=['sg-security-group-
id'],subnetIds=['subnet-subnet-id-1', 'subnet-subnet-id-2']}" \
--scrape-configuration configurationBlob=base64-encoded-blob \
 --destination ampConfiguration="{workspaceArn='arn:aws:aps:us-
west-2:123456789012:workspace/ws-workspace-id'}"
```

- AWS API で使用できるスクレイパーオペレーションの完全なリストを次に示します。

  CreateScraper API オペレーションを使用してスクレイパーを作成します。
- ListScrapers API オペレーションを使用して既存のスクレイパーを一覧表示します。
- UpdateScraper API オペレーションを使用して、スクレイパーのエイリアス、設定、または送信先を更新します。
- DeleteScraper API オペレーションを使用してスクレイパーを削除します。
- DescribeScraper API オペレーションを使用してスクレイパーの詳細を取得します。

クロスアカウントの設定

メトリクスを収集する Amazon MSK クラスターが Amazon Managed Service for Prometheus コレクターとは異なるアカウントにある場合に、クロスアカウント設定でスクレイパーを作成するには、次の手順を使用します。

例えば、Amazon MSK がある最初のソースアカウント `account_id_source`、および Amazon Managed Service for Prometheus ワークスペースがある 2 番目のターゲットアカウント `account_id_target` の 2 つのアカウントがある場合です。

クロスアカウント設定でスクレイパーを作成するには

1. ソースアカウントで、ロール `arn:aws:iam::111122223333:role/Source` を作成し、次の信頼ポリシーを追加します。

```
{
    "Effect": "Allow",
    "Principal": {
    "Service": [
        "scraper.aps.amazonaws.com"
     ]
    },
```

```
    "Action": "sts:AssumeRole",
    "Condition": {
        "ArnEquals": {
            "aws:SourceArn": "arn:aws:aps:aws-region:111122223333:scraper/scraper-
id"
        },
        "StringEquals": {
            "AWS:SourceAccount": "111122223333"
        }
    }
}
```

2. ソース (Amazon MSK クラスター) とターゲット (Amazon Managed Service for Prometheus ワークスペース) のすべての組み合わせで、ロール `arn:aws:iam::444455556666:role/Target` を作成し、AmazonPrometheusRemoteWriteAccess のアクセス許可を持つ次の信頼ポリシーを追加する必要があります。

```
{
  "Effect": "Allow",
  "Principal": {
     "AWS": "arn:aws:iam::111122223333:role/Source"
  },
  "Action": "sts:AssumeRole",
  "Condition": {
     "StringEquals": {
        "sts:ExternalId": "arn:aws:aps:aws-region:111122223333:scraper/scraper-id"
      }
  }
}
```

3. `--role-configuration` オプションを使用してスクレイパーを作成します。

```
aws amp create-scraper \ --source vpcConfiguration="{subnetIds=[subnet-
subnet-id], "securityGroupIds": ["sg-security-group-id"]}" \ --
scrape-configuration configurationBlob=<base64-encoded-blob> \
 --destination ampConfiguration="{workspaceArn='arn:aws:aps:aws-
region:444455556666:workspace/ws-workspace-id'}"\ --role-configuration
```

```
 '{"sourceRoleArn":"arn:aws:iam::111122223333:role/Source",
 "targetRoleArn":"arn:aws:iam::444455556666:role/Target"}'
```

4. スクレイパーの作成を検証します。

```
aws amp list-scrapers
{
    "scrapers": [
        {
            "scraperId": "s-example123456789abcdef0",
            "arn": "arn:aws:aps:aws-region:111122223333:scraper/s-
example123456789abcdef0": "arn:aws:iam::111122223333:role/Source",
            "status": "ACTIVE",
            "creationTime": "2025-10-27T18:45:00.000Z",
            "lastModificationTime": "2025-10-27T18:50:00.000Z",
            "tags": {},
            "statusReason": "Scraper is running successfully",
            "source": {
                "vpcConfiguration": {
                    "subnetIds": ["subnet-subnet-id"],
                    "securityGroupIds": ["sg-security-group-id"]
                }
            },
            "destination": {
                "ampConfiguration": {
                    "workspaceArn": "arn:aws:aps:aws-region:444455556666:workspace/
ws-workspace-id'"
                }
            },
            "scrapeConfiguration": {
                "configurationBlob": "<base64-encoded-blob>"
            }
        }
    ]
}
```

RoleConfiguration とサービスにリンクされたロールの間の変更

RoleConfiguration ではなくサービスにリンクされたロールに切り替えて Amazon Managed Service for Prometheus ワークスペースに書き込む場合は、UpdateScraper を更新し、RoleConfiguration なしでスクレイパーと同じアカウントのワークスペースを提供する必要があります。RoleConfiguration はスクレイパーから削除され、サービスにリンクされたロールが使用されます。

スクレイパーと同じアカウントのワークスペースを変更し、引き続き RoleConfiguration を使用する場合は、UpdateScraper で RoleConfiguration を再度指定する必要があります。

スクレイパーの検出と削除

AWS API または を使用して AWS CLI 、アカウントのスクレイパーを一覧表示したり、削除したりできます。

> **Note**
>
> 最新バージョンの AWS CLI または SDK を使用していることを確認します。最新バージョンには、最新の特長と機能に加え、セキュリティアップデートも含まれています。または、常に最新のコマンドラインエクスペリエンスを提供する AWS CloudShell を自動的に使用します。

アカウント内のすべてのスクレイパーを一覧表示するには、ListScrapers API オペレーションを使用します。

または、 を使用して AWS CLIを呼び出します。

```
aws amp list-scrapers
```

ListScrapers は、アカウント内のすべてのスクレイパーを返します。例:

```
{
    "scrapers": [
        {
            "scraperId": "s-1234abcd-56ef-7890-abcd-1234ef567890",
            "arn": "arn:aws:aps:aws-region:123456789012:scraper/s-1234abcd-56ef-7890-
abcd-1234ef567890",
```

```
            "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
AWSServiceRoleForAmazonPrometheusScraper_1234abcd-2931",
            "status": {
                "statusCode": "DELETING"
            },
            "createdAt": "2023-10-12T15:22:19.014000-07:00",
            "lastModifiedAt": "2023-10-12T15:55:43.487000-07:00",
            "tags": {},
            "source": {
                "vpcConfiguration": {
                   "securityGroupIds": [
                        "sg-1234abcd5678ef90"
                    ],
                    "subnetIds": [
                        "subnet-abcd1234ef567890",
                        "subnet-1234abcd5678ab90"
                    ]
                }
            },
            "destination": {
                "ampConfiguration": {
                    "workspaceArn": "arn:aws:aps:aws-region:123456789012:workspace/
ws-1234abcd-5678-ef90-ab12-cdef3456a78"
                }
            }
        }
    ]
}
```

スクレイパーを削除するには、ListScrapers オペレーションを使用して削除するスクレイパーの scraperId を見つけ、DeleteScraper オペレーションを使用して削除します。

または、 を使用して AWS CLIを呼び出します。

```
aws amp delete-scraper --scraper-id scraperId
```

Amazon MSK から収集されたメトリクス

Amazon MSK と統合すると、Amazon Managed Service for Prometheus コレクターは次のメトリクスを自動的にスクレイピングします。

メトリクス: jmx_exporter ジョブと pod_exporter ジョブ

メトリクス	説明/目的
jmx_config_reload_failure_total	JMX エクスポーターが設定ファイルの再ロードに失敗した合計回数。
jmx_scrape_duration_seconds	現在の収集サイクルで JMX メトリクスをスクレイプするのにかかった秒単位の時間。
jmx_scrape_error	JMX メトリクススクレイピング中にエラーが発生したかどうかを示します (1 = エラー、0 = 成功)。
java_lang_Memory_HeapMemoryUsage_used	現在、JVM に使用されているヒープメモリの量 (バイト単位)。
java_lang_Memory_HeapMemoryUsage_max	メモリ管理に使用できるヒープメモリの最大数 (バイト単位)。
java_lang_Memory_NonHeapMemoryUsage_used	現在、JVM に使用されているヒープ以外のメモリの量 (バイト単位)。
kafka_cluster_Partition_Value	パーティション ID とトピック別に分類される、Kafka クラスターパーティションに関連する現在の状態または値。
kafka_consumer_consumer_coordinator_metrics_assigned_partitions	このコンシューマーに現在割り当てられているパーティションの数。
kafka_consumer_consumer_coordinator_metrics_commit_latency_avg	オフセットをコミットするのにかかったミリ秒単位の平均時間。
kafka_consumer_consumer_coordinator_metrics_commit_rate	1 秒あたりのオフセットコミットの数。
kafka_consumer_consumer_coordinator_metrics_failed_rebalance_total	失敗したコンシューマーグループの再調整の合計数。

メトリクス	説明/目的
kafka_consumer_consumer_coordinator_metrics_last_heartbeat_seconds_ago	最後のハートビートがコーディネーターに送信されてからの秒数。
kafka_consumer_consumer_coordinator_metrics_rebalance_latency_avg	コンシューマーグループの再調整にかかるミリ秒単位の平均時間。
kafka_consumer_consumer_coordinator_metrics_rebalance_total	コンシューマーグループの再調整の合計数。
kafka_consumer_consumer_fetch_manager_metrics_bytes_consumed_rate	コンシューマーごとの 1 秒あたりの平均消費バイト数。
kafka_consumer_consumer_fetch_manager_metrics_fetch_latency_avg	フェッチリクエストにかかるミリ秒単位の平均時間。
kafka_consumer_consumer_fetch_manager_metrics_fetch_rate	1 秒あたりのフェッチリクエストの数。
kafka_consumer_consumer_fetch_manager_metrics_records_consumed_rate	1 秒あたりに消費されるレコードの平均数。
kafka_consumer_consumer_fetch_manager_metrics_records_lag_max	このコンシューマー内の任意のパーティションのレコード数に関する最大ラグ。
kafka_consumer_consumer_metrics_connection_count	アクティブな接続の現在の数。
kafka_consumer_consumer_metrics_incoming_byte_rate	すべてのサーバーから 1 秒あたりに受信した平均バイト数。
kafka_consumer_consumer_metrics_last_poll_seconds_ago	前回のコンシューマー poll() 呼び出しからの秒数。
kafka_consumer_consumer_metrics_request_rate	1 秒あたりの送信リクエストの数。
kafka_consumer_consumer_metrics_response_rate	1 秒あたりの受信レスポンスの数。

メトリクス	説明/目的
kafka_consumer_group_ConsumerLagMetrics_Value	コンシューマーがどの程度遅れているかを示す、コンシューマーグループの現在のコンシューマーラグ値。
kafka_controller_KafkaController_Value	Kafka コントローラーの現在の状態または値 (1 = アクティブなコントローラー、0 = アクティブではない)。
kafka_controller_ControllerEventManager_Count	処理されたコントローラーイベントの合計数。
kafka_controller_ControllerEventManager_Mean	コントローラーイベントの処理にかかる平均時間。
kafka_controller_ControllerStats_MeanRate	1 秒あたりのコントローラー統計オペレーションの平均レート。
kafka_coordinator_group_GroupMetadataManager_Value	コンシューマーグループのグループメタデータマネージャーの現在の状態または値。
kafka_log_LogFlushStats_Count	ログフラッシュ操作の合計数。
kafka_log_LogFlushStats_Mean	ログフラッシュ操作にかかる平均時間。
kafka_log_LogFlushStats_MeanRate	1 秒あたりのログフラッシュ操作の平均レート。
kafka_network_RequestMetrics_Count	処理されたネットワークリクエストの合計数。
kafka_network_RequestMetrics_Mean	ネットワークリクエストの処理にかかる平均時間。
kafka_network_RequestMetrics_MeanRate	1 秒あたりのネットワークリクエストの平均レート。
kafka_network_Acceptor_MeanRate	1 秒あたりに受け入れられる接続の平均レート。

メトリクス	説明/目的
kafka_server_Fetch_queue_size	フェッチリクエストキューの現在のサイズ。
kafka_server_Produce_queue_size	作成リクエストキューの現在のサイズ。
kafka_server_Request_queue_size	一般的なリクエストキューの現在のサイズ。
kafka_server_BrokerTopicMetrics_Count	ブローカートピックオペレーション (メッセージイン/アウト、バイトイン/アウト) の合計数。
kafka_server_BrokerTopicMetrics_MeanRate	ブローカートピックオペレーションの 1 秒あたりの平均レート。
kafka_server_BrokerTopicMetrics_OneMinuteRate	ブローカートピックオペレーションの 1 分間の移動平均レート。
kafka_server_DelayedOperationPurgatory_Value	処理待ち状態での遅延オペレーションの現在の数 (完了待ち)。
kafka_server_DelayedFetchMetrics_MeanRate	1 秒あたりの遅延フェッチオペレーションの平均レート。
kafka_server_FetcherLagMetrics_Value	レプリカフェッチャースレッドの現在のラグ値 (リーダーからどの程度遅延しているか)。
kafka_server_FetcherStats_MeanRate	1 秒あたりのフェッチャーオペレーションの平均レート。
kafka_server_ReplicaManager_Value	レプリカマネージャーの現在の状態または値。
kafka_server_ReplicaManager_MeanRate	1 秒あたりのレプリカマネージャーオペレーションの平均レート。
kafka_server_LeaderReplication_byte_rate	このブローカーがリーダーであるパーティションの 1 秒あたりのレプリケートされたバイト数。
kafka_server_group_coordinator_metrics_group_completed_rebalance_count	完了したコンシューマーグループの再調整の合計数。

メトリクス	説明/目的
kafka_server_group_coordinator_metrics_offset_commit_count	オフセットコミット操作の合計数。
kafka_server_group_coordinator_metrics_offset_commit_rate	1 秒あたりのオフセットコミット操作のレート。
kafka_server_socket_server_metrics_connection_count	アクティブな接続の現在の数。
kafka_server_socket_server_metrics_connection_creation_rate	1 秒あたりの新しい接続作成のレート。
kafka_server_socket_server_metrics_connection_close_rate	1 秒あたりの接続閉鎖のレート。
kafka_server_socket_server_metrics_failed_authentication_total	失敗した認証試行の合計数。
kafka_server_socket_server_metrics_incoming_byte_rate	1 秒あたりの受信バイトのレート。
kafka_server_socket_server_metrics_outgoing_byte_rate	1 秒あたりの送信バイトのレート。
kafka_server_socket_server_metrics_request_rate	1 秒あたりのリクエストのレート。
kafka_server_socket_server_metrics_response_rate	1 秒あたりのレスポンスのレート。
kafka_server_socket_server_metrics_network_io_rate	1 秒あたりのネットワーク I/O オペレーションのレート。
kafka_server_socket_server_metrics_io_ratio	I/O オペレーションに費やされた時間の割合。
kafka_server_controller_channel_metrics_connection_count	コントローラーチャネルのアクティブな接続の現在の数。

メトリクス	説明/目的
kafka_server_controller_channel_metrics_incoming_byte_rate	コントローラーチャネルの 1 秒あたりの受信バイトのレート。
kafka_server_controller_channel_metrics_outgoing_byte_rate	コントローラーチャネルの 1 秒あたりの送信バイトのレート。
kafka_server_controller_channel_metrics_request_rate	コントローラーチャネルの 1 秒あたりのリクエストのレート。
kafka_server_replica_fetcher_metrics_connection_count	レプリカフェッチャーのアクティブな接続の現在の数。
kafka_server_replica_fetcher_metrics_incoming_byte_rate	レプリカフェッチャーの 1 秒あたりの受信バイトのレート。
kafka_server_replica_fetcher_metrics_request_rate	レプリカフェッチャーの 1 秒あたりのリクエストのレート。
kafka_server_replica_fetcher_metrics_failed_authentication_total	レプリカフェッチャーの失敗した認証試行の合計数。
kafka_server_ZooKeeperClientMetrics_Count	ZooKeeper クライアントオペレーションの合計数。
kafka_server_ZooKeeperClientMetrics_Mean	ZooKeeper クライアントオペレーションの平均レイテンシー。
kafka_server_KafkaServer_Value	Kafka サーバーの現在の状態または値 (通常はサーバーが実行中であることを示します)。
node_cpu_seconds_total	CPU が各モード (ユーザー、システム、アイドル、など) で費やした合計秒数 (CPU とモード別に分類)。
node_disk_read_bytes_total	ディスクから正常に読み取られた合計バイト数 (デバイスごとに分類)。

メトリクス	説明/目的
node_disk_reads_completed_total	ディスクに対して正常に完了した読み取りの合計数 (デバイスごとに分類)。
node_disk_writes_completed_total	ディスクに対して正常に完了した書き込みの合計数 (デバイスごとに分類)。
node_disk_written_bytes_total	ディスクに正常に書き込まれた合計バイト数 (デバイスごとに分類)。
node_filesystem_avail_bytes	非ルートユーザー用のバイト単位の使用可能なファイルシステムスペース (デバイスとマウントポイント別に分類)。
node_filesystem_size_bytes	バイト単位のファイルシステムの合計サイズ (デバイスとマウントポイント別に分類)。
node_filesystem_free_bytes	バイト単位の空きファイルシステムスペース (デバイスとマウントポイント別に分類)。
node_filesystem_files	ファイルシステム上のファイルノード (inode) の合計数 (デバイスとマウントポイント別に分類)。
node_filesystem_files_free	ファイルシステム上の空きファイルノード (inode) の数 (デバイスとマウントポイント別に分類)。
node_filesystem_readonly	ファイルシステムが読み取り専用でマウントされているかどうかを示します (1 = 読み取り専用、0 = 読み取り/書き込み)。
node_filesystem_device_error	ファイルシステム統計の取得中にエラーが発生したかどうかを示します (1 = エラー、0 = 成功)。

制限事項

現在の Amazon MSK と Amazon Managed Service for Prometheus の統合には、次の制限があります。

- Amazon MSK プロビジョンドクラスターでのみサポート (Amazon MSK Serverless では利用できません)
- KRaft メタデータモードと組み合わせてパブリックアクセスが有効になっている Amazon MSK クラスターではサポートされていません
- Amazon MSK Express ブローカーではサポートされていません
- 現在、Amazon MSK クラスターと Amazon Managed Service for Prometheus コレクター/ワークスペース間の 1:1 マッピングをサポートしています。

Prometheus と互換性のあるメトリクスとはどのようなものですか。

Prometheus メトリクスをアプリケーションやインフラストラクチャからスクレイピングして Amazon Managed Service for Prometheus で使用するには、Prometheus 互換の `/metrics` エンドポイントから Prometheus 互換のメトリクスをインストルメントして公開する必要があります。独自のメトリクスを実装することができますが、必須ではありません。Kubernetes (Amazon EKS を含む) や他の多くのライブラリやサービスは、これらのメトリクスを直接実装しています。

Amazon EKS のメトリクスを Prometheus 互換のエンドポイントにエクスポートすると、それらのメトリクスを Amazon Managed Service for Prometheus コレクターで自動的にスクレイピングすることができます。

詳細については、以下の各トピックを参照してください。

- メトリクスを Prometheus メトリクスとしてエクスポートする既存のライブラリとサービスの詳細については、「Prometheus ドキュメント」の「Exporters and integrations」を参照してください。
- Prometheus 互換メトリクスを独自のコードからエクスポートする方法の詳細については、「Prometheus ドキュメント」の「Writing exporters」を参照してください。
- Amazon Managed Service for Prometheus コレクターを設定して Amazon EKS クラスターからメトリクスを自動的にスクレイピングする方法の詳細については、「Amazon EKS のマネージドコレクターを設定する」を参照してください。

提供されたログを使用してコレクターをモニタリングする

Amazon Managed Service for Prometheus コレクターは、メトリクス収集プロセスのモニタリングとトラブルシューティングに役立つ公開ログを提供します。これらのログは自動的に Amazon CloudWatch Logs に送信され、サービス検出、メトリクス収集、データエクスポートオペレーションを可視化します。コレクターは、メトリクス収集パイプラインの 3 つの主要コンポーネントのログを提供します。

トピック

サービス検出ログ

サービス検出ログは、以下を含むターゲット検出プロセスに関する情報を提供します。

- Kubernetes API リソースへのアクセス時の認証またはアクセス許可の問題。
- サービス検出設定の設定エラー。

次の例は、サービス検出中に発生する可能性のある一般的な認証エラーとアクセス許可エラーを示しています。

存在しない Amazon EKS クラスター

指定された Amazon EKS クラスターが存在しない場合、次のエラーが表示されます。

```
{
  "component": "SERVICE_DISCOVERY",
  "timestamp": "2025-04-30T17:25:41.946Z",
  "message": {
    "log": "Failed to watch Service - Verify your scraper source exists."
  },
  "scrapeConfigId": "s-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

サービスの無効なアクセス許可

コレクターにサービスを監視するための適切なロールベースのアクセスコントロール (RBAC) アクセス許可がない場合、次のエラーが表示されます。

```
{
  "component": "SERVICE_DISCOVERY",
  "timestamp": "2025-04-30T17:25:41.946Z",
  "message": {
    "log": "Failed to watch Service - Verify your scraper source permissions are
 valid."
  },
  "scrapeConfigId": "s-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

エンドポイントの無効なアクセス許可

コレクターにエンドポイントを監視するための適切なロールベースのアクセスコントロール (RBAC) アクセス許可がない場合、次のエラーが表示されます。

```
{
  "component": "SERVICE_DISCOVERY",
  "timestamp": "2025-04-30T17:25:41.946Z",
  "message": {
    "log": "Failed to watch Endpoints - Verify your scraper source permissions are
 valid."
  },
  "scrapeConfigId": "s-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

コレクターログ

コレクターログは、以下を含むメトリクススクレイピングプロセスに関する情報を提供します。

- エンドポイントが利用できないためのスクレイピングの失敗。
- ターゲットをスクレイピングしようとするときの接続の問題。
- スクレイピングオペレーション中のタイムアウト。
- スクレイピングターゲットによって返される HTTP ステータスエラー。

次の例は、メトリクススクレイピングプロセス中に発生する可能性がある一般的なコレクターエラーを示しています。

欠落しているメトリクスエンドポイント

ターゲットインスタンスで /metrics エンドポイントが使用できない場合、次のエラーが表示されます。

```
{
    "component": "COLLECTOR",
    "message": {
        "log": "Failed to scrape Prometheus endpoint - verify /metrics endpoint is
 available",
        "job": "pod_exporter",
        "targetLabels": "{__name__=\"up\", instance=\10.24.34.0\", job=
\"pod_exporter\"}"
    },
    "timestamp": "1752787969551",
    "scraperId": "s-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

接続が拒否されました

コレクターがターゲットエンドポイントへの接続を確立できない場合、次のエラーが表示されます。

```
{
  "scrapeConfigId": "s-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "timestamp": "2025-04-30T17:25:41.946Z",
  "message": {
    "message": "Scrape failed",
    "scrape_pool": "pod_exporter",
    "target": "http://10.24.34.0:80/metrics",
    "error": "Get \"http://10.24.34.0:80/metrics\": dial tcp 10.24.34.0:80: connect:
 connection refused"
  },
  "component": "COLLECTOR"
}
```

エクスポーターログ

エクスポーターログは、収集したメトリクスを Amazon Managed Service for Prometheus ワークスペースに送信するプロセスに関する、以下を含む情報を提供します。

- 処理されたメトリクスとデータポイントの数。
- ワークスペースの問題によるエクスポートの失敗。
- メトリクスの書き込み試行時のアクセス許可エラー。
- エクスポートパイプラインの依存関係の失敗。

次の例は、メトリクスのエクスポートプロセス中に発生する可能性がある一般的なエクスポーターエラーを示しています。

名前空間が見つからない

メトリクスエクスポートのターゲットワークスペースが見つからない場合、次のエラーが表示されます。

```
{
    "component": "EXPORTER",
    "message": {
        "log": "Failed to export to the target workspace - Verify your scraper
 destination.",
        "samplesDropped": 5
    },
    "timestamp": "1752787969664",
    "scraperId": "s-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
}
```

コレクター公開ログの理解と使用

ログ構造

すべてのコレクター公開ログは、以下のフィールドと一貫した構造に従います。

scrapeConfigId

ログを生成したスクレイピング設定の一意の識別子。

timestamp

ログエントリが生成された時刻。

メッセージ

ログメッセージのコンテンツ。追加の構造化フィールドが含まれる場合があります。

コンポーネント

ログを生成したコンポーネント (SERVICE_DISCOVERY、COLLECTOR、または EXPORTER)

トラブルシューティングに公開ログを使用する

コレクター公開ログは、メトリクス収集に関する一般的な問題のトラブルシューティングに役立ちます。

1. サービス検出の問題
   - SERVICE_DISCOVERY ログで認証エラーまたはアクセス許可エラーを確認します。
   - コレクターが Kubernetes リソースにアクセスするために必要なアクセス許可を持っていることを確認します。
2. メトリクススクレイピングの問題
   - COLLECTOR ログでスクレイピングの失敗を確認します。
   - ターゲットエンドポイントがアクセス可能であり、メトリクスを返すことを確認します。
   - ファイアウォールルールで、コレクターのターゲットエンドポイントへの接続が許可されていることを確認します。
3. メトリクスエクスポートの問題
   - EXPORTER ログでエクスポートの失敗を確認します。
   - ワークスペースが存在し、正しく設定されていることを確認します。
   - コレクターにワークスペースへの書き込みに必要なアクセス許可があることを確認します。

コレクター公開ログへのアクセス

コレクター公開ログは、Amazon CloudWatch Logs に自動的に送信されます。これらのログにアクセスするには:

1. CloudWatch コンソールの https://console.aws.amazon.com/cloudwatch/ を開いてください。
2. ナビゲーションペインで、[Log groups] (ロググループ) を選択します。

3. コレクター /aws/prometheus/workspace_id/collector/collector_id のロググループを見つけて選択します。
4. ログイベントを参照または検索して、関連情報を特定します。

CloudWatch Logs Insights を使用してコレクターログをクエリおよび分析することもできます。例えば、すべてのサービス検出エラーを特定するには:

```
fields @timestamp, message.message
| filter component = "SERVICE_DISCOVERY" and message.message like /Failed/
| sort @timestamp desc
```

コレクターをモニタリングするためのベストプラクティス

Amazon Managed Service for Prometheus コレクターを効果的にモニタリングするには:

1. 永続的なスクレイピングエラーやエクスポートエラーなど、重要なコレクターの問題に対して CloudWatch アラームを設定します。詳細については、「Amazon CloudWatch ユーザーガイド」の「アラーム」を参照してください。
2. CloudWatch ダッシュボードを作成して、公開されたログデータと共にコレクターのパフォーマンスメトリクスを視覚化します。詳細については、「Amazon CloudWatch ユーザーガイド」の「ダッシュボード」を参照してください。
3. サービス検出ログを定期的に確認し、ターゲットが正しく検出されていることを確認します。
4. 削除されたターゲットの数をモニタリングして、設定の潜在的な問題を特定します。
5. エクスポートの失敗を追跡して、メトリクスがワークスペースに正常に送信されていることを確認します。

カスタマーマネージドコレクター

このセクションには、Prometheus リモート書き込みを使用して Amazon Managed Service for Prometheus にメトリクスを送信する独自のコレクターを設定してデータを取り込む方法に関する情報が含まれています。

独自のコレクターを使用して Amazon Managed Service for Prometheus にメトリクスを送信する場合、メトリクスを保護し、取り込みプロセスが可用性のニーズを満たしていることを確認する責任はお客様にあります。

ほとんどのカスタマーマネージドコレクターは、以下のツールのいずれかを使用します。

- AWS Distro for OpenTelemetry (ADOT) – ADOT は、OpenTelemetry の完全なサポートを受け、安全で、本番環境に対応したオープンソースディストリビューションであり、エージェントがメトリクスを収集できるようにします。ADOT を使用してメトリクスを収集し、それらを Amazon Managed Service for Prometheus ワークスペースに送信できます。ADOT コレクターの詳細については、「AWS Distro for OpenTelemetry」を参照してください。
- Prometheus エージェント — オープンソースの Prometheus サーバーの独自のインスタンスをセットアップし、エージェントとして実行することでメトリクスを収集し Amazon Managed Service for Prometheus ワークスペースに転送することができます。

以下のトピックは、これら両方のツールの使用方法について説明し、独自のコレクターの設定に関する一般的な情報も含んでいます。

トピック

メトリクスの取り込みの保護

Amazon Managed Service for Prometheus には、メトリクスの取り込みを保護するための手段が用意されています。

Amazon Managed Service for Prometheus AWS PrivateLink での の使用

Amazon Managed Service for Prometheus にメトリクスを取り込むネットワークトラフィックは、パブリックインターネットエンドポイントを介して、または VPC エンドポイントを介して行うことができます AWS PrivateLink。 AWS PrivateLink を使用すると、VPC からのネットワークトラフィックはパブリックインターネットを経由せず、 AWS のネットワーク内で保護されます。Amazon Managed Service for Prometheus の AWS PrivateLink VPC エンドポイントを作成するには、「」を参照してくださいインターフェイス VPC エンドポイントでの Amazon Managed Service for Prometheus の使用。

認証と認可

AWS Identity and Access Management (IAM) は、 AWS リソースへのアクセスを安全に制御するのに役立つウェブサービスです。IAM を使用して、誰を認証 (サインイン) し、誰にリソースの使用

を認可する (アクセス許可を付与する) かを制御します。Amazon Managed Service for Prometheus は IAM と統合されているため、データを安全に保つことができます。Amazon Managed Service for Prometheus をセットアップするときは、Prometheus サーバーからメトリクスを取り込めるようにする IAM ロールと、Amazon Managed Service for Prometheus ワークスペースに保存されたメトリクスに対して Grafana サーバーからクエリを実行できるようにする IAM ロールを作成する必要があります。IAM の詳細については、「IAM とは」を参照してください。

Amazon Managed Service for Prometheus のセットアップに役立つもう 1 つの AWS セキュリティ機能は、 AWS 署名バージョン 4 の署名プロセス (AWS SigV4) です。署名バージョン 4 は、HTTP によって送信された AWS リクエストに認証情報を追加するプロセスです。セキュリティのため、へのほとんどのリクエストは、アクセスキー ID とシークレットアクセスキーで構成されるアクセスキーで署名 AWS する必要があります。これらの 2 つのキーは、一般的にセキュリティ認証情報と呼ばれます。SigV4 の詳細については、「Signature Version 4 の署名プロセス」を参照してください。

AWS Distro for OpenTelemetry をコレクターとして使用する

このセクションでは、 AWS Distro for OpenTelemetry (ADOT) Collector を設定して Prometheus 計測アプリケーションからスクレイプし、メトリクスを Amazon Managed Service for Prometheus に送信する方法について説明します。ADOT コレクターの詳細については、「AWS Distro for OpenTelemetry」を参照してください。

以下のトピックでは、Amazon EKS、Amazon ECS、Amazon EC2 インスタンスのいずれのメトリクスであるかに基づいて、メトリクスのコレクターとして ADOT を設定する 3 つの異なる方法について説明します。

トピック

- Amazon Elastic Kubernetes Service クラスターで AWS Distro for OpenTelemetry を使用してメトリクスの取り込みを設定する
- AWS Distro for Open Telemetry を使用して Amazon ECS からのメトリクス取り込みを設定する
- リモート書き込みを使用した Amazon EC2 インスタンスからのメトリクスの取り込みの設定

Amazon Elastic Kubernetes Service クラスターで AWS Distro for OpenTelemetry を使用してメトリクスの取り込みを設定する

AWS Distro for OpenTelemetry (ADOT) コレクターを使用して、Prometheus で計測されたアプリケーションからメトリクスをスクレイプし、そのメトリクスを Amazon Managed Service for Prometheus に送信できます。

> **Note**
>
> ADOT コレクターの詳細については、「AWS Distro for OpenTelemetry」を参照してください。
> Prometheus でインストルメント化したアプリケーションの詳細については、「Prometheus と互換性のあるメトリクスとはどのようなものですか。」を参照してください。

ADOT による Prometheus メトリクスの収集には、Prometheus Receiver、Prometheus Remote Write Exporter、Sigv4 Authentication Extension という 3 つの OpenTelemetry コンポーネントが使用されます。

既存の Prometheus の設定を使用して Prometheus Receiver を構成して、サービス検出とメトリクスのスクレイピングを実行できます。Prometheus Receiver は、メトリクスを Prometheus 公開形式でスクレイピングします。スクレイピング対象のアプリケーションやエンドポイントは、Prometheus クライアントライブラリで構成する必要があります。Prometheus Receiver は、Prometheus ドキュメントの「Configuration」で説明されている Prometheus のスクレイピングと再ラベル付けの設定をすべてサポートしています。これらの設定を直接 ADOT コレクターの設定に貼り付けることができます。

Prometheus Remote Write Exporter は、`remote_write` エンドポイントを使用して、スクレイピングされたメトリクスを管理ポータルワークスペースに送信します。データをエクスポートする HTTP リクエストは、安全な認証の AWS プロトコルである AWS SigV4 と Sigv4 Authentication Extension で署名されます。詳細については、「Signature Version 4 の署名プロセス」を参照してください。

コレクターは、Amazon EKS 上の Prometheus メトリクスエンドポイントを自動的に検出し、<kubernetes_sd_config> にある設定を使用します。

以下のデモは、Amazon Elastic Kubernetes Service または自己管理型 Kubernetes を実行しているクラスターでのこの設定の例を示しています。これらのステップを実行するには、デフォルトの認証情報チェーンの潜在的なオプションのいずれかからの AWS 認証情報が必要です AWS 。詳細については、AWS 「 SDK for Go の設定」を参照してください。このデモでは、プロセスの統合テストに使用されるサンプルアプリを使用します。このサンプルアプリは、Prometheus クライアントライブラリのように、`/metrics` エンドポイントでメトリクスを公開します。

前提条件

以下の取り込み設定手順を開始する前に、サービスアカウントの IAM ロールと信頼ポリシーを設定する必要があります。

サービスアカウントの IAM ロールと信頼ポリシーを設定するには

1. 「Amazon EKS クラスターからメトリクスを取り込むためのサービスロールの設定」の手順に従って、サービスアカウントの IAM ロールを作成します。

   ADOT コレクターは、メトリクスをスクレイピングしてエクスポートするときにこのロールを使用します。

2. 次に、信頼ポリシーを編集します。IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。

3. 左側のナビゲーションペインで [ロール] を選択し、ステップ 1 で作成した amp-iamproxy-ingest-role を探します。

4. [信頼関係] タブを選択し、[信頼関係の編集] を選択します。

5. 信頼関係ポリシーの JSON で、`aws-amp` を `adot-col` に置き換えて [信頼ポリシーの更新] を選択します。最終的なポリシーは次のようになります。

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.us-east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.eks.us-east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:adot-col:amp-iamproxy-ingest-service-account",
          "oidc.eks.us-east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
        }
      }
    }
  ]
}
```

6. [アクセス許可] タブを選択し、次のアクセス許可ポリシーがロールにアタッチされていることを確認します。

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "aps:RemoteWrite",
                "aps:GetSeries",
                "aps:GetLabels",
                "aps:GetMetricMetadata"
            ],
            "Resource": "*"
        }
    ]
}
```

Prometheus メトリクスの収集の有効化

> **Note**
>
> Amazon EKS で名前空間を作成すると、`alertmanager` とノードエクスポーターはデフォルトで無効になっています。

Amazon EKS または Kubernetes クラスターで Prometheus の収集を有効にするには

1. aws-otel-community のリポジトリから、サンプルアプリをフォークしてクローンします。

   次に、以下のコマンドを実行します。

```
cd ./sample-apps/prometheus-sample-app
docker build . -t prometheus-sample-app:latest
```

2. このイメージを Amazon ECR や DockerHub などのレジストリにプッシュします。

3. 次のように Kubernetes 設定をコピーして適用し、サンプルアプリをクラスターにデプロイします。prometheus-sample-app.yaml ファイル内の {{PUBLIC_SAMPLE_APP_IMAGE}} は、先ほどプッシュしたイメージに置き換えます。

```
curl https://raw.githubusercontent.com/aws-observability/aws-otel-collector/
main/examples/eks/aws-prometheus/prometheus-sample-app.yaml -o prometheus-sample-
app.yaml
kubectl apply -f prometheus-sample-app.yaml
```

4. 次のコマンドを入力して、サンプルアプリが起動したことを確認します。コマンドの出力で、NAME 列に prometheus-sample-app が表示されます。

```
kubectl get all -n aoc-prometheus-pipeline-demo
```

5. ADOT コレクターのデフォルトのインスタンスを起動します。そのためには、まず次のコマンドを入力して、ADOT コレクターの Kubernetes 設定を取得します。

```
curl https://raw.githubusercontent.com/aws-observability/aws-otel-collector/main/
examples/eks/aws-prometheus/prometheus-daemonset.yaml -o prometheus-daemonset.yaml
```

次に、テンプレートファイルを編集して、YOUR_ENDPOINT を Amazon Managed Service for Prometheus ワークスペースの remote_write エンドポイントに、YOUR_REGION を使用中のリージョンに置き換えます。ワークスペースの詳細を確認したときに Amazon Managed Service for Prometheus コンソールに表示される remote_write エンドポイントを使用してください。

また、Kubernetes 設定のサービスアカウントセクションYOUR_ACCOUNT_IDの を AWS アカウント ID に変更する必要があります。

この例では、ADOT コレクターの設定で注釈 (scrape=true) を使用して、スクレイピングするターゲットエンドポイントを指定しています。ADOT コレクターは、これによってサンプルアプリのエンドポイントをクラスター内の kube-system エンドポイントから区別できます。別のサンプルアプリをスクレイピングする場合は、これを再ラベル付けの設定から削除できます。

6. 次のコマンドを入力して、ADOT コレクターをデプロイします。

```
kubectl apply -f prometheus-daemonset.yaml
```

7. 次のコマンドを入力して、ADOT コレクターが起動したことを確認します。NAMESPACE 列で adot-col を探してください。

```
kubectl get pods -n adot-col
```

8. ログエクスポーターを使用して、パイプラインが機能することを確認します。サンプルテンプレートは既にログエクスポーターと統合されています。次のコマンドを入力します。

```
kubectl get pods -A
kubectl logs -n adot-col name_of_your_adot_collector_pod
```

サンプルアプリからスクレイピングされたメトリクスの一部は、次の例のようになります。

```
Resource labels:
     -> service.name: STRING(kubernetes-service-endpoints)
     -> host.name: STRING(192.168.16.238)
     -> port: STRING(8080)
     -> scheme: STRING(http)
InstrumentationLibraryMetrics #0
Metric #0
Descriptor:
     -> Name: test_gauge0
     -> Description: This is my gauge
     -> Unit:
     -> DataType: DoubleGauge
DoubleDataPoints #0
StartTime: 0
Timestamp: 1606511460471000000
Value: 0.000000
```

9. Amazon Managed Service for Prometheus がメトリクスを受け取ったかどうかをテストするには、awscurl を使用します。このツールを使用すると、 AWS Sigv4 認証を使用してコマンドラインを介して HTTP リクエストを送信できるため、Amazon Managed Service for Prometheus からクエリを実行するための正しいアクセス許可を持つ AWS 認証情報をローカルに設定する必要があります。 のインストール手順についてはawscurl、「awscurl」を参照してください。

次のコマンドの AMP_REGION と AMP_ENDPOINT は、Amazon Managed Service for Prometheus ワークスペースの情報に置き換えます。

```
awscurl --service="aps" --region="AMP_REGION" "https://AMP_ENDPOINT/api/v1/query?
query=adot_test_gauge0"
{"status":"success","data":{"resultType":"vector","result":[{"metric":
{"__name__":"adot_test_gauge0"},"value":[1606512592.493,"16.87214000011479"]}]}}
```

レスポンスとしてメトリクスを受け取れば、パイプラインの設定が成功し、サンプルアプリから Amazon Managed Service for Prometheus にメトリクスが正常に伝搬されたことを意味します。

クリーンアップ

このデモをクリーンアップするには、次のコマンドを入力します。

```
kubectl delete namespace aoc-prometheus-pipeline-demo
kubectl delete namespace adot-col
```

詳細設定

Prometheus Receiver は、Prometheus ドキュメントの「Configuration」で説明されている Prometheus のスクレイピングと再ラベル付けの設定をすべてサポートしています。これらの設定を直接 ADOT コレクターの設定に貼り付けることができます。

Prometheus Receiver の設定には、サービス検出、スクレイピング設定、再ラベル設定が含まれます。レシーバーの設定は次のようになります。

```
receivers:
  prometheus:
    config:
      [Your Prometheus configuration]
```

設定ファイルの例を以下に示します。

```
receivers:
  prometheus:
    config:
      global:
        scrape_interval: 1m
        scrape_timeout: 10s

      scrape_configs:
      - job_name: kubernetes-service-endpoints
        sample_limit: 10000
        kubernetes_sd_configs:
        - role: endpoints
```

```
        tls_config:
          ca_file: /var/run/secrets/kubernetes.io/serviceaccount/ca.crt
          insecure_skip_verify: true
        bearer_token_file: /var/run/secrets/kubernetes.io/serviceaccount/token
```

既存の Prometheus 設定がある場合は、値が環境変数で置き換えられないように、$ 文字を $$ に置き換える必要があります。*これは、relabel_configurations の replacement の値で特に重要です。例えば、次のような relabel_configurations があるとします。

```
relabel_configs:
- source_labels:
 [__meta_kubernetes_ingress_scheme,__address__,__meta_kubernetes_ingress_path]
  regex: (.+);(.+);(.+)
  replacement: ${1}://${2}${3}
  target_label: __param_target
```

これは次のように変更します。

```
relabel_configs:
- source_labels:
 [__meta_kubernetes_ingress_scheme,__address__,__meta_kubernetes_ingress_path]
  regex: (.+);(.+);(.+)
  replacement: $${1}://${2}${3}
  target_label: __param_target
```

Prometheus Remote Write Exporter と Sigv4 Authentication Extension

Prometheus Remote Write Exporter と Sigv4 Authentication Extension の設定は、Prometheus Receiver よりも簡単です。パイプラインのこの段階では、既にメトリクスが取り込まれていて、このデータを Amazon Managed Service for Prometheus にエクスポートする準備ができています。次の例は、Amazon Managed Service for Prometheus と通信するための適切な設定の最小要件を示しています。

```
extensions:
  sigv4auth:
    service: "aps"
    region: "user-region"
exporters:
  prometheusremotewrite:
    endpoint: "https://aws-managed-prometheus-endpoint/api/v1/remote_write"
    auth:
```

```
      authenticator: "sigv4auth"
```

この設定は、デフォルトの AWS 認証情報チェーンの AWS 認証情報を使用して、 AWS SigV4 によって署名された HTTPS リクエストを送信します。詳細については、「Configuring the AWS SDK for Go」を参照してください。サービスには aps を指定する必要があります。

デプロイの方法にかかわらず、ADOT コレクターはデフォルトの AWS 認証情報チェーンにリストされているオプションのいずれかにアクセスできる必要があります。Sigv4 Authentication Extension は に依存し AWS SDK for Go 、それを使用して認証情報を取得して認証します。これらの認証情報に、Amazon Managed Service for Prometheus のリモート書き込みアクセス許可があることを確認する必要があります。

AWS Distro for Open Telemetry を使用して Amazon ECS からのメトリクス取り込みを設定する

このセクションでは、Amazon Elastic Container Service (Amazon ECS) からメトリクスを収集し、AWS Distro for Open Telemetry (ADOT) を使用して Amazon Managed Service for Prometheus に取り込む方法について説明します。また、Amazon Managed Grafana でメトリクスを視覚化する方法についても説明します。

前提条件

> **Important**
>
> 開始する前に、 AWS Fargate クラスター上のデフォルト設定の Amazon ECS 環境と、Amazon Managed Service for Prometheus ワークスペースおよび Amazon Managed Grafana ワークスペースが必要です。ユーザーがコンテナのワークロード、Amazon Managed Service for Prometheus、Amazon Managed Grafana に精通していることを前提としています。

詳細については、以下のリンクを参照してください。

- Fargate クラスターにデフォルト設定で Amazon ECS 環境を作成する方法については、「Amazon ECS デベロッパーガイド」の「クラスターの作成」を参照してください。
- Amazon Managed Service for Prometheus ワークスペースを作成する方法については、「Amazon Managed Service for Prometheus ユーザーガイド」の「ワークスペースの作成」を参照してください。

- Amazon Managed Grafana ワークスペースを作成する方法については、「Amazon Managed Grafana User Guide」の「Creating a workspace」を参照してください。

ステップ 1: カスタム ADOT コレクターコンテナイメージを定義する

以下の設定ファイルをテンプレートとして使用して、独自の ADOT コレクターコンテナイメージを定義します。*my-remote-URL* と *my-region* は、使用中の `endpoint` と `region` の値に置き換えます。設定を adot-config.yaml というファイルに保存します。

> **Note**
>
> この設定では、`sigv4auth` 拡張機能を使用して Amazon Managed Service for Prometheus への呼び出しを認証します。`sigv4auth` の構成方法の詳細については、GitHub の「Authenticator - Sigv4」を参照してください。

```
receivers:
  prometheus:
    config:
      global:
        scrape_interval: 15s
        scrape_timeout: 10s
      scrape_configs:
      - job_name: "prometheus"
        static_configs:
        - targets: [ 0.0.0.0:9090 ]
  awsecscontainermetrics:
    collection_interval: 10s
processors:
  filter:
    metrics:
      include:
        match_type: strict
        metric_names:
          - ecs.task.memory.utilized
          - ecs.task.memory.reserved
          - ecs.task.cpu.utilized
          - ecs.task.cpu.reserved
          - ecs.task.network.rate.rx
          - ecs.task.network.rate.tx
          - ecs.task.storage.read_bytes
```

```
          - ecs.task.storage.write_bytes
exporters:
  prometheusremotewrite:
    endpoint: my-remote-URL
    auth:
      authenticator: sigv4auth
  logging:
    loglevel: info
extensions:
  health_check:
  pprof:
    endpoint: :1888
  zpages:
    endpoint: :55679
  sigv4auth:
    region: my-region
    service: aps
service:
  extensions: [pprof, zpages, health_check, sigv4auth]
  pipelines:
    metrics:
      receivers: [prometheus]
      exporters: [logging, prometheusremotewrite]
    metrics/ecs:
      receivers: [awsecscontainermetrics]
      processors: [filter]
      exporters: [logging, prometheusremotewrite]
```

ステップ 2: ADOT コレクターコンテナイメージを Amazon ECR リポジトリにプッシュする

Dockerfile を使用して、コンテナイメージを作成して Amazon Elastic Container Registry (ECR) リポジトリにプッシュします。

1. Dockerfile をビルドして、コンテナイメージをコピーして OTEL Docker イメージに追加します。

   ```
   FROM public.ecr.aws/aws-observability/aws-otel-collector:latest
   COPY adot-config.yaml /etc/ecs/otel-config.yaml
   CMD ["--config=/etc/ecs/otel-config.yaml"]
   ```

2. Amazon ECR リポジトリを作成します。

   ```
   # create repo:
   ```

```
COLLECTOR_REPOSITORY=$(aws ecr create-repository --repository aws-otel-collector \
                                --query repository.repositoryUri --output text)
```

3. コンテナイメージを作成します。

```
# build ADOT collector image:
docker build -t $COLLECTOR_REPOSITORY:ecs .
```

> **Note**
>
> コンテナのビルドは、そのコンテナが実行される環境と同じ環境で行うことを前提としています。そうでない場合、イメージのビルド時に `--platform` パラメータの使用が必要になることがあります。

4. Amazon ECR リポジトリにサインインします。*my-region* は、使用中の `region` の値に置き換えます。

```
# sign in to repo:
aws ecr get-login-password --region my-region | \
        docker login --username AWS --password-stdin $COLLECTOR_REPOSITORY
```

5. コンテナイメージをプッシュします。

```
# push ADOT collector image:
docker push $COLLECTOR_REPOSITORY:ecs
```

ステップ 3: Amazon ECS タスク定義を作成して Amazon Managed Service for Prometheus をスクレイピングする

Amazon Managed Service for Prometheus をスクレイピングする Amazon ECS タスク定義を作成します。タスク定義には、`adot-collector` という名前のコンテナと、`prometheus` という名前のコンテナを含める必要があります。`prometheus` はメトリクスを生成し、`adot-collector` は `prometheus` をスクレイピングします。

> **Note**
>
> Amazon Managed Service for Prometheus はサービスとして実行され、コンテナからメトリクスを収集します。この場合のコンテナは、Prometheus をエージェントモードでローカル

で実行し、ローカルのメトリクスを Amazon Managed Service for Prometheus に送信します。

例: タスク定義

以下の例は、タスク定義がどのようなものかを示しています。この例をテンプレートとして使用して、独自のタスク定義を作成できます。adot-collector の image の値は、リポジトリの URL とイメージタグ ($COLLECTOR_REPOSITORY:ecs) に置き換えます。adot-collector と prometheus の region の値は、使用中の region の値に置き換えます。

```
{
  "family": "adot-prom",
  "networkMode": "awsvpc",
  "containerDefinitions": [
    {
      "name": "adot-collector",
      "image": "account_id.dkr.ecr.region.amazonaws.com/image-tag",
      "essential": true,
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/ecs-adot-collector",
          "awslogs-region": "my-region",
          "awslogs-stream-prefix": "ecs",
          "awslogs-create-group": "True"
        }
      }
    },
    {
      "name": "prometheus",
      "image": "prom/prometheus:main",
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/ecs-prom",
          "awslogs-region": "my-region",
          "awslogs-stream-prefix": "ecs",
          "awslogs-create-group": "True"
        }
      }
    }
```

```
  ],
  "requiresCompatibilities": [
    "FARGATE"
  ],
  "cpu": "1024"
}
```

ステップ 4: Amazon Managed Service for Prometheus にアクセスする許可をタスクに付与する

スクレイピングされたメトリクスを Amazon Managed Service for Prometheus に送信するには、Amazon ECS タスクに AWS API オペレーションを呼び出すための正しいアクセス許可が必要です。タスク用の IAM ロールを作成し、そのロールに `AmazonPrometheusRemoteWriteAccess` ポリシーをアタッチする必要があります。このロールを作成してポリシーをアタッチする方法の詳細については、「タスク用の IAM ロールとポリシーの作成」を参照してください。

IAM ロールに `AmazonPrometheusRemoteWriteAccess` をアタッチし、そのロールをタスクに使用したら、スクレイピングされたメトリクスを Amazon ECS によって Amazon Managed Service for Prometheus に送信できます。

ステップ 5: Amazon Managed Grafana でメトリクスを視覚化する

> **Important**
>
> 開始する前に、Amazon ECS タスク定義に対して Fargate タスクを実行する必要があります。そうしないと、Amazon Managed Service for Prometheus でメトリクスを使用することができません。

1. Amazon Managed Grafana ワークスペースのナビゲーションペインで、 AWS アイコンの下にあるデータソースを選択します。
2. [データソース] タブの [サービス] で、[Amazon Managed Service for Prometheus] を選択し、[デフォルトのリージョン] を選択します。
3. [データソースの追加] を選択します。
4. `ecs` および `prometheus` プレフィックスを使用して、メトリクスのクエリと表示を行います。

リモート書き込みを使用した Amazon EC2 インスタンスからのメトリクスの取り込みの設定

このセクションでは、Amazon Elastic Compute Cloud (Amazon EC2) インスタンス上で、リモート書き込みを使用する Prometheus サーバーを実行する方法について説明します。Go で記述されたデモアプリケーションからメトリクスを収集し、それらを Amazon Managed Service for Prometheus ワークスペースに送信する方法について説明します。

前提条件

> **Important**
>
> 開始する前に、Prometheus v2.26 以降をインストールしておく必要があります。ユーザーが Prometheus、Amazon EC2、Amazon Managed Service for Prometheus に精通していることを前提としています。Prometheus のインストール方法については、Prometheus ウェブサイトの「Getting started」を参照してください。

Amazon EC2 または Amazon Managed Service for Prometheus に慣れていない場合は、まず以下のセクションを読むことをお勧めします。

- Amazon Elastic Compute Cloud とは
- Amazon Managed Service for Prometheus とは

Amazon EC2 用の IAM ロールの作成

メトリクスをストリーミングするには、まず AWS 管理ポリシー AmazonPrometheusRemoteWriteAccess を使用して IAM ロールを作成する必要があります。その後、そのロールを持つインスタンスを起動し、Amazon Managed Service for Prometheus ワークスペースにメトリクスをストリーミングできます。

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. ナビゲーションペインで [ロール] を選択し、[ロールを作成] を選択します。
3. 信頼されたエンティティの種類として [AWS のサービス] を選択します。ユースケースとして [EC2] を選択します。[次へ: アクセス許可] を選択します。
4. 検索バーに「AmazonPrometheusRemoteWriteAccess」と入力します。[ポリシー名] で [AmazonPrometheusRemoteWriteAccess] を選択し、[ポリシーをアタッチ] を選択します。[次へ: タグ] を選択します。

5. (オプション) IAM ロールに IAM タグを作成します。[次へ: 確認] を選択します。
6. ロールの名前を入力します。[ポリシーを作成] を選択します。

Amazon EC2 インスタンスの起動

Amazon EC2 インスタンスを起動するには、「Amazon Elastic Compute Cloud Linux インスタンス用ユーザーガイド」の「インスタンスの起動」の手順に従います。

デモアプリケーションの実行

IAM ロールを作成し、そのロールを使用して EC2 インスタンスを起動したら、デモアプリケーションを実行して動作を確認できます。

デモアプリケーションとテストメトリクスを実行するには

1. 以下のテンプレートを使用して、main.go という名前の Go ファイルを作成します。

```
package main

import (
    "github.com/prometheus/client_golang/prometheus/promhttp"
    "net/http"
)

func main() {
    http.Handle("/metrics", promhttp.Handler())

    http.ListenAndServe(":8000", nil)
}
```

2. 次のコマンドを実行して、適切な依存関係をインストールします。

```
sudo yum update -y
sudo yum install -y golang
go get github.com/prometheus/client_golang/prometheus/promhttp
```

3. デモアプリケーションを実行します。

```
go run main.go
```

デモアプリケーションはポート 8000 で実行され、公開されているすべての Prometheus メトリクスを表示します。これらのメトリクスの例を以下に示します。

```
curl -s http://localhost:8000/metrics
...
process_max_fds 4096# HELP process_open_fds Number of open file descriptors.# TYPE
 process_open_fds gauge
process_open_fds 10# HELP process_resident_memory_bytes Resident memory size in
 bytes.# TYPE process_resident_memory_bytes gauge
process_resident_memory_bytes 1.0657792e+07# HELP process_start_time_seconds Start
 time of the process since unix epoch in seconds.# TYPE process_start_time_seconds
 gauge
process_start_time_seconds 1.61131955899e+09# HELP process_virtual_memory_bytes
 Virtual memory size in bytes.# TYPE process_virtual_memory_bytes gauge
process_virtual_memory_bytes 7.77281536e+08# HELP process_virtual_memory_max_bytes
 Maximum amount of virtual memory available in bytes.# TYPE
 process_virtual_memory_max_bytes gauge
process_virtual_memory_max_bytes -1# HELP
 promhttp_metric_handler_requests_in_flight Current number of scrapes being
 served.# TYPE promhttp_metric_handler_requests_in_flight gauge
promhttp_metric_handler_requests_in_flight 1# HELP
 promhttp_metric_handler_requests_total Total number of scrapes by HTTP status
 code.# TYPE promhttp_metric_handler_requests_total counter
promhttp_metric_handler_requests_total{code="200"} 1
promhttp_metric_handler_requests_total{code="500"} 0
promhttp_metric_handler_requests_total{code="503"} 0
```

Amazon Managed Service for Prometheus ワークスペースの作成

Amazon Managed Service for Prometheus ワークスペースを作成するには、「Create a workspace」の手順に従います。

Prometheus サーバーの実行

1. 以下の YAML ファイルの例をテンプレートとして使用して、`prometheus.yaml` という名前の新しいファイルを作成します。`url` については、*`my-region`* を使用中のリージョンの値に、*`my-workspace-id`* を Amazon Managed Service for Prometheus で生成されたワークスペース ID に置き換えます。`region` については、*`my-region`* を使用中のリージョンの値に置き換えます。

例: YAML ファイル

```
global:
  scrape_interval: 15s
  external_labels:
    monitor: 'prometheus'

scrape_configs:
  - job_name: 'prometheus'
    static_configs:
      - targets: ['localhost:8000']

remote_write:
  -
    url: https://aps-workspaces.my-region.amazonaws.com/workspaces/my-workspace-id/
api/v1/remote_write
    queue_config:
        max_samples_per_send: 1000
        max_shards: 200
        capacity: 2500
    sigv4:
         region: my-region
```

2. Prometheus サーバーを実行して、デモアプリケーションのメトリクスを Amazon Managed Service for Prometheus ワークスペースに送信します。

```
prometheus --config.file=prometheus.yaml
```

これで、Prometheus サーバーによってデモアプリケーションのメトリクスが Amazon Managed Service for Prometheus ワークスペースに送信されます。

Prometheus インスタンスをコレクターとして使用する

エージェントモード (Prometheus エージェント と呼ばれる) で実行している Prometheus インスタンスを使用し、メトリクスをスクレイピングして Amazon Managed Service for Prometheus ワークスペースに送信できます。

以下のトピックでは、エージェントモードで実行されている Prometheus インスタンスをメトリクスのコレクターとして設定するさまざまな方法について説明します。

> **Warning**
>
> Prometheus エージェントを作成する場合、その設定とメンテナンスはユーザーが担当します。セキュリティ機能を有効にして、Prometheus のスクレイピングエンドポイントがパブリックインターネットに公開されないようにします。

同じメトリクスセットをモニタリングする複数の Prometheus インスタンスをセットアップし、それらを 1 つの Amazon Managed Service for Prometheus ワークスペースに送信して高可用性を実現する場合は、重複排除を設定する必要があります。重複排除を設定する手順に従わない場合、Amazon Managed Service for Prometheus に送信されるすべてのデータサンプルが、重複サンプルも含めて課金対象になります。重複排除の設定方法については、「Amazon Managed Service for Prometheus に送信される高可用性メトリクスの重複排除」を参照してください。

トピック

- Helm を使用した新しい Prometheus サーバーからの取り込みの設定
- EC2 上の Kubernetes 内にある既存の Prometheus サーバーからの取り込みの設定
- Fargate 上の Kubernetes にある既存の Prometheus サーバーからの取り込みの設定

Helm を使用した新しい Prometheus サーバーからの取り込みの設定

このセクションの手順に従うと、Amazon Managed Service for Prometheus を迅速に設定して稼働させることができます。ここでは、Amazon EKS クラスターに新しい Prometheus サーバーをセットアップします。新しいサーバーは、デフォルト設定を使用して Amazon Managed Service for Prometheus にメトリクスを送信します。この方法には次の前提条件があります。

- 新しい Prometheus サーバーがメトリクスを収集する元の Amazon EKS クラスターが必要です。
- Amazon EKS クラスターには Amazon EBS CSI ドライバーをインストールしている必要があります (Helm で必要)。
- Helm CLI 3.0 以降を使用する必要があります。
- 以下のセクションの手順を実行するには、Linux または macOS コンピュータを使用する必要があります。

ステップ 1: 新しい Helm チャートリポジトリを追加する

次のコマンドを入力して、新しい Helm チャートリポジトリを追加します。これらのコマンドの詳細については、「Helm Repo」を参照してください。

```
helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
helm repo add kube-state-metrics https://kubernetes.github.io/kube-state-metrics
helm repo update
```

ステップ 2: Prometheus 名前空間を作成する

次のコマンドを入力して、Prometheus サーバーとその他のモニタリングコンポーネント用の Prometheus 名前空間を作成します。*prometheus-namespace* は、この名前空間に付ける名前に置き換えます。

```
kubectl create namespace prometheus-namespace
```

ステップ 3: サービスアカウントの IAM ロールを設定する

ここで説明するオンボーディング方法では、Prometheus サーバーが実行されている Amazon EKS クラスターでサービスアカウントの IAM ロールを使用する必要があります。

サービスアカウントの IAM ロールを使用すると、IAM ロールを Kubernetes サービスアカウントに関連付けることができます。このサービスアカウントは、そのサービスアカウントを使用するポッド内のコンテナに AWS アクセス許可を提供できます。詳細については、「サービスアカウントの IAM ロール」を参照してください。

これらのロールをまだ設定していない場合は、「Amazon EKS クラスターからメトリクスを取り込むためのサービスロールの設定」の手順に従ってロールを設定します。そのセクションの手順では、`eksctl` を使用する必要があります。詳細については、「Amazon Elastic Kubernetes Service の開始方法 - `eksctl`」を参照してください。

> **Note**
>
> EKS または ではなく AWS 、アクセスキーとシークレットキーのみを使用して Amazon Managed Service for Prometheus にアクセスする場合、 `EKS-IAM-ROLE` ベースの SigV4 を使用することはできません。

ステップ 4: 新しいサーバーをセットアップしてメトリクスの取り込みを開始する

Amazon Managed Service for Prometheus ワークスペースにメトリクスを送信する新しい Prometheus サーバーをインストールするには、以下の手順に従います。

Amazon Managed Service for Prometheus ワークスペースにメトリクスを送信する新しい Prometheus サーバーをインストールするには

1. テキストエディタを使用して、my_prometheus_values_yaml という名前のファイルを作成し、次の内容を記述します。

   - *IAM_PROXY_PROMETHEUS_ROLE_ARN* は、「Amazon EKS クラスターからメトリクスを取り込むためのサービスロールの設定」で作成した amp-iamproxy-ingest-role の ARN に置き換えます。
   - *WORKSPACE_ID* は、Amazon Managed Service for Prometheus ワークスペースの ID に置き換えます。
   - *REGION* は、Amazon Managed Service for Prometheus のリージョンに置き換えます。

```
## The following is a set of default values for prometheus server helm chart which
 enable remoteWrite to AMP
## For the rest of prometheus helm chart values see: https://github.com/prometheus-
community/helm-charts/blob/main/charts/prometheus/values.yaml
##
serviceAccounts:
  server:
    name: amp-iamproxy-ingest-service-account
    annotations:
      eks.amazonaws.com/role-arn: ${IAM_PROXY_PROMETHEUS_ROLE_ARN}
server:
  remoteWrite:
    - url: https://aps-workspaces.${REGION}.amazonaws.com/workspaces/
${WORKSPACE_ID}/api/v1/remote_write
      sigv4:
        region: ${REGION}
      queue_config:
        max_samples_per_send: 1000
        max_shards: 200
        capacity: 2500
```

2. 次のコマンドを入力して、Prometheus サーバーを作成します。

- *prometheus-chart-name* は、Prometheus リリース名に置き換えます。
- *prometheus-namespace* は、Prometheus 名前空間の名前に置き換えます。

```
helm install prometheus-chart-name prometheus-community/prometheus -n prometheus-
namespace \
-f my_prometheus_values_yaml
```

> **Note**
>
> `helm install` コマンドはさまざまな方法でカスタマイズできます。詳細については、「Helm ドキュメント」の「Helm install」を参照してください。

EC2 上の Kubernetes 内にある既存の Prometheus サーバーからの取り込みの設定

Amazon Managed Service for Prometheus は、Amazon EKS を実行しているクラスターおよび Amazon EC2 上で動作する自己管理型 Kubernetes クラスター内の Prometheus サーバーからのメトリクスの取り込みをサポートしています。このセクションの詳細な手順は、Amazon EKS クラスター内の Prometheus サーバーを対象としています。Amazon EC2 上の自己管理型 Kubernetes クラスターの場合も手順は同じですが、Kubernetes クラスターでサービスアカウントの OIDC プロバイダーと IAM ロールを手動で設定する必要がある点が異なります。

このセクションの手順では、Kubernetes パッケージマネージャーとして Helm を使用します。

トピック

- ステップ 1: サービスアカウントの IAM ロールを設定する
- ステップ 2: Helm を使用して既存の Prometheus サーバーをアップグレードする

ステップ 1: サービスアカウントの IAM ロールを設定する

ここで説明するオンボーディング方法では、Prometheus サーバーが実行されている Amazon EKS クラスターでサービスアカウントの IAM ロールを使用する必要があります。これらのロールはサービスロールとも呼ばれます。

サービスロールを使用すると、IAM ロールを Kubernetes サービスアカウントに関連付けることができます。このサービスアカウントは、そのサービスアカウントを使用する任意のポッドのコンテナに

アクセス AWS 許可を付与できます。詳細については、「サービスアカウントの IAM ロール」を参照してください。

これらのロールをまだ設定していない場合は、「Amazon EKS クラスターからメトリクスを取り込むためのサービスロールの設定」の手順に従ってロールを設定します。

ステップ 2: Helm を使用して既存の Prometheus サーバーをアップグレードする

このセクションの手順には、リモート書き込みと sigv4 を設定して、Amazon Managed Service for Prometheus ワークスペースへのリモート書き込みを行えるように Prometheus サーバーを認証および認可する方法が含まれます。

Prometheus バージョン 2.26.0 以降を使用している場合

バージョン 2.26.0 以降の Prometheus サーバーイメージで Helm チャートを使用している場合は、以下の手順に従います。

Helm チャートを使用して Prometheus サーバーからのリモート書き込みを設定するには

1. Helm 設定ファイルに新しいリモート書き込みセクションを作成します。

   - `${IAM_PROXY_PROMETHEUS_ROLE_ARN}` は、「ステップ 1: サービスアカウントの IAM ロールを設定する」で作成した amp-iamproxy-ingest-role の ARN に置き換えます。ロールの ARN は `arn:aws:iam::`*`your account ID`*`:role/amp-iamproxy-ingest-role` という形式になります。
   - `${WORKSPACE_ID}` は、Amazon Managed Service for Prometheus のワークスペース ID に置き換えます。
   - `${REGION}` は、Amazon Managed Service for Prometheus ワークスペースのリージョン (`us-west-2` など) に置き換えます。

```
## The following is a set of default values for prometheus server helm chart which
 enable remoteWrite to AMP
    ## For the rest of prometheus helm chart values see: https://github.com/
prometheus-community/helm-charts/blob/main/charts/prometheus/values.yaml
    ##
    serviceAccounts:
      server:
        name: amp-iamproxy-ingest-service-account
        annotations:
          eks.amazonaws.com/role-arn: ${IAM_PROXY_PROMETHEUS_ROLE_ARN}
```

```
    server:
      remoteWrite:
        - url: https://aps-workspaces.${REGION}.amazonaws.com/workspaces/
${WORKSPACE_ID}/api/v1/remote_write
          sigv4:
            region: ${REGION}
          queue_config:
            max_samples_per_send: 1000
            max_shards: 200
            capacity: 2500
```

2. Helm を使用して既存の Prometheus サーバーの構成を更新します。

   - `prometheus-chart-name` は、Prometheus リリース名に置き換えます。
   - `prometheus-namespace` は、Prometheus サーバーがインストールされている Kubernetes 名前空間に置き換えます。
   - `my_prometheus_values_yaml` は、Helm 設定ファイルのパスに置き換えます。
   - `current_helm_chart_version` は、Prometheus サーバーの Helm チャートの現在のバージョンに置き換えます。現在のチャートのバージョンは、helm list コマンドを使用して確認できます。

```
helm upgrade prometheus-chart-name prometheus-community/prometheus \
       -n prometheus-namespace \
       -f my_prometheus_values_yaml \
       --version current_helm_chart_version
```

以前のバージョンの Prometheus 使う

2.26.0 より前のバージョンの Prometheus を使用している場合は、以下の手順に従います。これらのステップではサイドカーアプローチを使用します。これは、以前のバージョンの Prometheus では AWS 署名バージョン 4 の署名プロセス (AWS SigV4) がネイティブにサポートされていないためです。

これらの手順では、Prometheus のデプロイに Helm を使用しているものと想定します。

Prometheus サーバーからのリモート書き込みを設定するには

1. Prometheus サーバーで、新しいリモート書き込み設定を作成します。まず、新しい更新ファイルを作成します。このファイルの名前を `amp_ingest_override_values.yaml`. とします。

この YAML ファイルに次の値を追加します。

```
serviceAccounts:
        server:
            name: "amp-iamproxy-ingest-service-account"
            annotations:
                eks.amazonaws.com/role-arn:
 "${SERVICE_ACCOUNT_IAM_INGEST_ROLE_ARN}"
    server:
        sidecarContainers:
            - name: aws-sigv4-proxy-sidecar
              image: public.ecr.aws/aws-observability/aws-sigv4-proxy:1.0
              args:
              - --name
              - aps
              - --region
              - ${REGION}
              - --host
              - aps-workspaces.${REGION}.amazonaws.com
              - --port
              - :8005
              ports:
              - name: aws-sigv4-proxy
                containerPort: 8005
        statefulSet:
            enabled: "true"
        remoteWrite:
            - url: http://localhost:8005/workspaces/${WORKSPACE_ID}/api/v1/
remote_write
```

${REGION} は、Amazon Managed Service for Prometheus ワークスペースのリージョンに置き換えます。

${SERVICE_ACCOUNT_IAM_INGEST_ROLE_ARN} は、「ステップ 1: サービスアカウントの IAM ロールを設定する」で作成した amp-iamproxy-ingest-role の ARN に置き換えます。ロールの ARN は arn:aws:iam::*your account ID*:role/amp-iamproxy-ingest-role という形式になります。

${WORKSPACE_ID} は、ワークスペース ID に置き換えます。

2. Prometheus Helm チャートをアップグレードします。まず、以下のコマンドを入力して Helm チャート名を確認します。このコマンドの出力で、名前に prometheus という文字列を含むチャートを探します。

```
helm ls --all-namespaces
```

次に、以下のコマンドを入力します。

```
helm upgrade --install prometheus-helm-chart-name prometheus-community/prometheus -
n prometheus-namespace -f ./amp_ingest_override_values.yaml
```

prometheus-helm-chart-name は、前のコマンドで返された Prometheus Helm チャートの名前に置き換えます。*prometheus-namespace* は、名前空間の名前に置き換えます。

Helm チャートのダウンロード

Helm チャートをまだローカルにダウンロードしていない場合は、次のコマンドを使用してダウンロードできます。

```
helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
    helm pull prometheus-community/prometheus --untar
```

Fargate 上の Kubernetes にある既存の Prometheus サーバーからの取り込みの設定

Amazon Managed Service for Prometheus は、Fargate 上で動作する自己管理型 Kubernetes クラスター内の Prometheus サーバーからのメトリクスの取り込みをサポートしています。Fargate 上で動作する Amazon EKS クラスター内の Prometheus サーバーからメトリクスを取り込むには、amp_ingest_override_values.yaml という名前のファイルのデフォルト設定を次のようにオーバーライドします。

```
prometheus-node-exporter:
        enabled: false

    alertmanager:
        enabled: false

    serviceAccounts:
      server:
        name: amp-iamproxy-ingest-service-account
```

```
        annotations:
          eks.amazonaws.com/role-arn: ${IAM_PROXY_PROMETHEUS_ROLE_ARN}

    server:
      persistentVolume:
        enabled: false
      remoteWrite:
        - url: https://aps-workspaces.${REGION}.amazonaws.com/workspaces/
${WORKSPACE_ID}/api/v1/remote_write
          sigv4:
            region: ${REGION}
          queue_config:
            max_samples_per_send: 1000
            max_shards: 200
            capacity: 2500
```

次のコマンドを実行して、オーバーライドを使用して Prometheus をインストールします。

```
helm install prometheus-for-amp prometheus-community/prometheus \
                  -n prometheus \
                  -f amp_ingest_override_values.yaml
```

この Helm チャートの設定では、ノードエクスポーターとアラートマネージャーを無効にし、さらに Prometheus サーバーのデプロイの実行を無効にしています。

次のテストクエリの例を実行すると、インストールを確認できます。

```
$ awscurl --region region --service aps "https://aps-
workspaces.region_id.amazonaws.com/workspaces/workspace_id/api/v1/query?
query=prometheus_api_remote_read_queries"
    {"status":"success","data":{"resultType":"vector","result":[{"metric":
{"__name__":"prometheus_api_remote_read_queries","instance":"localhost:9090","job":"prometheus"
[1648461236.419,"0"]}]}}21
```

高可用性データ用に Amazon Managed Service for Prometheus をセットアップする

Amazon Managed Service for Prometheus にデータを送信すると、そのデータはリージョン内の AWS アベイラビリティーゾーン間で自動的にレプリケートされ、スケーラビリティ、可用性、セキュリティを提供するホストのクラスターから提供されます。特定の環境によっては、さらに可用性

を高めるフェイルセーフ機能を追加することが望ましい場合があります。環境に高可用性セーフティを追加する一般的な方法は 2 つあります。

- 同じデータを持つ複数のコンテナまたはインスタンスがある場合は、そのデータを Amazon Managed Service for Prometheus に送信し、データの重複排除を自動的に行わせることができます。これは、データを Amazon Managed Service for Prometheus ワークスペースに確実に送信するために役立ちます。

  高可用性データの重複排除の詳細については、「Amazon Managed Service for Prometheus に送信される高可用性メトリクスの重複排除」を参照してください。
- AWS リージョンが利用できない場合でもデータにアクセスできるようにする場合は、別のリージョンの 2 つ目のワークスペースにメトリクスを送信できます。

  メトリクスデータを複数のワークスペースに送信する方法の詳細については、「クロスリージョンワークスペースを使用して Amazon Managed Service for Prometheus に高可用性を追加する」を参照してください。

トピック

Amazon Managed Service for Prometheus に送信される高可用性メトリクスの重複排除

複数の Prometheus エージェント (エージェントモードで実行されている Prometheus インスタンス) から、Amazon Managed Service for Prometheus ワークスペースにデータを送信できます。

これらのインスタンスのいくつかが同じメトリクスを記録して送信している場合、データの高可用性が確保されます (いずれかのエージェントがデータの送信を停止しても、Amazon Managed Service for Prometheus ワークスペースは別のインスタンスから引き続きデータを受信します)。ただし、Amazon Managed Service for Prometheus ワークスペースでは、メトリクスの重複が自動的に排除されるようにすることが望まれます。これにより、メトリクスが複数回表示されるのを防ぎ、データインジェストとストレージに対して複数回課金が発生することを回避できます。

Amazon Managed Service for Prometheus で複数の Prometheus エージェントからのデータを自動的に重複排除するには、重複データを送信しているエージェントのセットに単一のクラスター名を割り当て、各インスタンスにレプリカ名を割り当てます。クラスター名により、これらのインスタンスが共有データを持つものとして識別されます。レプリカ名により、Amazon Managed Service for Prometheus で各メトリクスのソースを識別することが可能になります。最終的に保存されるメトリクスにはクラスターラベルが含まれますが、レプリカは含まれないため、メトリクスは単一のソースから取得されているように見えます。

> **Note**
>
> 特定のバージョンの Kubernetes (1.28 および 1.29) では、`cluster` ラベルが付いた独自のメトリクスが出力される場合があります。これにより、Amazon Managed Service for Prometheus の重複排除に問題が発生する可能性があります。詳細については、高可用性に関するよくある質問を参照してください。

以下のトピックでは、Amazon Managed Service for Prometheus でデータを自動的に重複排除するように、データを送信する際に `cluster` ラベルと `__replica__` ラベルを含める方法を示します。

> **Important**
>
> 重複排除を設定しない場合、Amazon Managed Service for Prometheus に送信されるすべてのデータサンプルが課金対象になります。これらのデータサンプルには、重複するサンプルが含まれます。

Prometheus による Amazon Managed Service for Prometheus への高可用性データの送信

Prometheus で高可用性設定をセットアップするには、高可用性グループのすべてのインスタンスに外部ラベルを適用して、Amazon Managed Service for Prometheus でそれらを識別できるよ

うにする必要があります。Prometheus インスタンスのエージェントを高可用性グループの一部として識別するには、cluster ラベルを使用します。グループ内の各レプリカを個別に識別するには、__replica__ ラベルを使用します。重複排除を機能させるには、__replica__ と cluster の両方のラベルを適用する必要があります。

> **Note**
>
> __replica__ ラベルは、replica という単語の前後に 2 つのアンダースコア記号が付いた形式です。

例: コードスニペット

次のコードスニペットでは、cluster ラベルは Prometheus インスタンスのエージェント prom-team1 を識別し、_replica_ ラベルはレプリカ replica1 と replica2 を識別します。

```
cluster: prom-team1
__replica__: replica1
```

```
cluster: prom-team1
__replica__: replica2
```

Amazon Managed Service for Prometheus は、これらのラベルを持つ高可用性レプリカからのデータサンプルを保存する場合、サンプルを受け入れるときに replica ラベルを取り除きます。つまり、レプリカごとにシリーズが保存されるのではなく、現在のシリーズに対して 1:1 のシリーズマッピングが作成されます。cluster ラベルは保持されます。

> **Note**
>
> 特定のバージョンの Kubernetes (1.28 および 1.29) では、cluster ラベルが付いた独自のメトリクスが出力される場合があります。これにより、Amazon Managed Service for Prometheus の重複排除に問題が発生する可能性があります。詳細については、高可用性に関するよくある質問を参照してください。

Prometheus Operator Helm チャートを使用して Amazon Managed Service for Prometheus への高可用性データをセットアップする

Helm の Prometheus Operator で高可用性設定をセットアップするには、高可用性グループのすべてのインスタンスに外部ラベルを適用し、これらのラベルを Amazon Managed Service for Prometheus で識別できるようにする必要があります。さらに、Prometheus Operator Helm チャートで `replicaExternalLabelName` および `externalLabels` 属性を設定する必要もあります。

例: YAML ヘッダー

次の YAML ヘッダーでは、`externalLabel` に `cluster` が追加され、Prometheus インスタンスのエージェントを高可用性グループの一部として識別します。また、`replicaExternalLabels` はグループ内の各レプリカを識別します。

```
replicaExternalLabelName: __replica__
externalLabels:
cluster: prom-dev
```

> **Note**
>
> 特定のバージョンの Kubernetes (1.28 および 1.29) では、`cluster` ラベルが付いた独自のメトリクスが出力される場合があります。これにより、Amazon Managed Service for Prometheus の重複排除に問題が発生する可能性があります。詳細については、高可用性に関するよくある質問を参照してください。

AWS Distro for OpenTelemetry を使用して Amazon Managed Service for Prometheus に高可用性データを送信する

AWS Distro for OpenTelemetry (ADOT) は、OpenTelemetry プロジェクトの安全で本番環境に対応したディストリビューションです。ADOT は、アプリケーションモニタリング用の分散トレースとメトリクスを取集できるように、ソース API、ライブラリ、エージェントを提供します。ADOT の詳細については、AWS 「Distro for Open Telemetry について」を参照してください。

高可用性設定で ADOT を設定するには、ADOT コレクターコンテナイメージを設定し、外部ラベルと `cluster__replica__` を AWS Prometheus リモート書き込みエクスポーターに適用する必要があります。このエクスポーターは、スクレイピングされたメトリクスを `remote_write` エンドポイント経由で Amazon Managed Service for Prometheus ワークスペースに送信します。これらのラ

ベルを Remote Write Exporter に設定すると、冗長レプリカの実行中に重複するメトリクスが保持されるのを防ぐことができます。 AWS Prometheus リモート書き込みエクスポーターの詳細については、「Amazon Managed Service for Prometheus の Prometheus リモート書き込みエクスポーターの開始方法」を参照してください。

> **Note**
>
> 特定のバージョンの Kubernetes (1.28 および 1.29) では、`cluster` ラベルが付いた独自のメトリクスが出力される場合があります。これにより、Amazon Managed Service for Prometheus の重複排除に問題が発生する可能性があります。詳細については、高可用性に関するよくある質問を参照してください。

Prometheus コミュニティ Helm チャートによる Amazon Managed Service for Prometheus への高可用性データの送信

Prometheus コミュニティ Helm チャートで高可用性設定をセットアップするには、高可用性グループのすべてのインスタンスに外部ラベルを適用して、Amazon Managed Service for Prometheus でそれらを識別できるようにする必要があります。以下は、Prometheus コミュニティ Helm チャートから Prometheus の 1 つのインスタンスに `external_labels` を追加する方法の例を示しています。

```
server:
global:
  external_labels:
      cluster: monitoring-cluster
      __replica__: replica-1
```

> **Note**
>
> 複数のレプリカが必要な場合は、異なるレプリカ値を使用してチャートを複数回デプロイする必要があります。Prometheus コミュニティ Helm チャートでは、コントローラーグループから直接レプリカの数を増やすとき、レプリカ値を動的に設定することができないためです。`replica` ラベルを自動設定するには、Prometheus Operator Helm チャートを使用します。

> **Note**
>
> 特定のバージョンの Kubernetes (1.28 および 1.29) では、`cluster` ラベルが付いた独自のメトリクスが出力される場合があります。これにより、Amazon Managed Service for Prometheus の重複排除に問題が発生する可能性があります。詳細については、高可用性に関するよくある質問を参照してください。

Amazon Managed Service for Prometheus の高可用性設定に関する一般的な質問への回答

サンプルポイントを追跡するために、別のラベルに値 __replica__ を含める必要がありますか?

高可用性設定では、Amazon Managed Service for Prometheus は Prometheus インスタンスのクラスターからリーダーを選出することで、データサンプルが重複しないようにします。リーダーレプリカからのデータサンプルの送信が 30 秒間停止した場合、Amazon Managed Service for Prometheus は自動的に別の Prometheus インスタンスをリーダーレプリカに設定し、欠落したデータを含めてデータを新しいリーダーから取り込みます。したがって、答えは「いいえ」であり、推奨もされません。 これを行った場合、次のような問題が発生する可能性があります。

- PromQL で `count` のクエリを実行すると、新しいリーダーの選出期間中に、想定よりも高い値が返されることがあります。
- 新しいリーダーの選出期間中にそのリーダーが `active series limits` になると、`active series` の数が増加します。詳細については、「AMP のクォータ」を参照してください。

Kubernetes には独自の cluster ラベルがあり、メトリクスを重複排除していないようです。どうすればこの問題を解決できますか。

新しいメトリクス `apiserver_storage_size_bytes` が Kubernetes 1.28 で導入され、`cluster` ラベルが追加されました。この `cluster` ラベルにより、Amazon Managed Service for Prometheus の重複排除の問題が発生する可能性があります。Kubernetes 1.3 では、ラベル名が `storage-cluster_id` に変更されています (1.28 と 1.29 の後のパッチでも名前が変更されています)。クラスターが、このメトリクスを `cluster` ラベル付きで出力している場合、Amazon Managed Service for Prometheus は関連付する時系列を重複排除できません。この問題を回避するには、Kubernetes クラスターを最新のパッチ適用バージョンにアップグレードすることをお勧めします。または、Amazon Managed Service for Prometheus に取り込む前

に、apiserver_storage_size_bytes メトリクスの cluster ラベルを再ラベル付けすることもできます。

> **Note**
>
> Kubernetes の変更の詳細については、Kubernetes GitHub プロジェクトの「Rename Label cluster to storage_cluster_id for apiserver_storage_size_bytes metric」を参照してください。

クロスリージョンワークスペースを使用して Amazon Managed Service for Prometheus に高可用性を追加する

クロスリージョン可用性をデータに追加するには、 AWS リージョン間で複数のワークスペースにメトリクスを送信できます。Prometheus では、複数のライターとクロスリージョンでの書き込みの両方がサポートされています。

以下の例は、Helm を使用してエージェントモードで動作する Prometheus サーバーをセットアップして、異なるリージョンの 2 つのワークスペースにメトリクスを送信する方法を示しています。

```
extensions:
      sigv4auth:
        service: "aps"

    receivers:
      prometheus:
        config:
          scrape_configs:
            - job_name: 'kubernetes-kubelet'
              scheme: https
              tls_config:
                ca_file: /var/run/secrets/kubernetes.io/serviceaccount/ca.crt
                insecure_skip_verify: true
              bearer_token_file: /var/run/secrets/kubernetes.io/serviceaccount/token
              kubernetes_sd_configs:
              - role: node
              relabel_configs:
              - action: labelmap
                regex: __meta_kubernetes_node_label_(.+)
              - target_label: __address__
                replacement: kubernetes.default.svc.cluster.local:443
              - source_labels: [__meta_kubernetes_node_name]
```

```
                regex: (.+)
                target_label: __metrics_path__
                replacement: /api/v1/nodes/$${1}/proxy/metrics

    exporters:
      prometheusremotewrite/one:
        endpoint: "https://aps-workspaces.workspace_1_region.amazonaws.com/workspaces/
ws-workspace_1_id/api/v1/remote_write"
        auth:
          authenticator: sigv4auth
      prometheusremotewrite/two:
        endpoint: "https://aps-workspaces.workspace_2_region.amazonaws.com/workspaces/
ws-workspace_2_id/api/v1/remote_write"
        auth:
          authenticator: sigv4auth

    service:
      extensions: [sigv4auth]
      pipelines:
        metrics/one:
          receivers: [prometheus]
          exporters: [prometheusremotewrite/one]
        metrics/two:
          receivers: [prometheus]
          exporters: [prometheusremotewrite/two]
```

Prometheus メトリクスに対するクエリの実行

ワークスペースにメトリクスが取り込まれるようになったら、それらのメトリクスに対してクエリを実行できます。

メトリクスを視覚的に表現するダッシュボードを作成するには、Amazon Managed Grafana などのサービスを使用できます。Amazon Managed Grafana (または Grafana のスタンドアロンインスタンス) は、さまざまなディスプレイプレゼンテーションスタイルでメトリクスを表示するグラフィカルインターフェイスを構築できます。Amazon Managed Grafana の詳細については、「Amazon Managed Grafana ユーザーガイド」を参照してください。

また、ダイレクトクエリを使用して、1 回限りのクエリの作成、データの探索、またはメトリクスを使用する独自のアプリケーションの作成を行うこともできます。ダイレクトクエリは、Amazon Managed Service for Prometheus API と標準の Prometheus クエリ言語 PromQL を使用して、Prometheus ワークスペースからデータを取得します。PromQL とその構文の詳細については、Prometheus ドキュメントの「Querying Prometheus」を参照してください。

トピック

PromQL チートシート

Amazon Managed Service for Prometheus ワークスペースでメトリクスをクエリするときは、この PromQL (Prometheus クエリ言語) チートシートをクイックリファレンスとして使用します。PromQL を使用すると、機能クエリ言語を使用して時系列データをリアルタイムで選択および集計できます。

PromQL の詳細については、PromLabs ウェブサイトの「PromQL チートシート」を参照してください。

基本セレクタ

メトリクス名とラベルマッチャーで時系列を選択します。

```
# Select all time series with the metric name http_requests_total
http_requests_total

# Select time series with specific label values
http_requests_total{job="prometheus", method="GET"}

# Use label matchers
http_requests_total{status_code!="200"}          # Not equal
http_requests_total{status_code=~"2.."}          # Regex match
http_requests_total{status_code!~"4.."}          # Negative regex match
```

範囲ベクトルセレクタ

時間の経過に伴うサンプルの範囲を選択します。

```
# Select 5 minutes of data
http_requests_total[5m]

# Time units: s (seconds), m (minutes), h (hours), d (days), w (weeks), y (years)
cpu_usage[1h]
memory_usage[30s]
```

集約演算子

複数の時系列にまたがるデータを集約します。

```
# Sum all values
sum(http_requests_total)

# Sum by specific labels
sum by (job) (http_requests_total)
sum without (instance) (http_requests_total)

# Other aggregation operators
avg(cpu_usage)                    # Average
min(response_time)                # Minimum
max(response_time)                # Maximum
count(up)                         # Count of series
stddev(cpu_usage)                 # Standard deviation
```

共通関数

関数を適用してデータを変換します。

```
# Rate of increase per second (for counters)
rate(http_requests_total[5m])

# Increase over time range
increase(http_requests_total[1h])

# Derivative (for gauges)
deriv(cpu_temperature[5m])

# Mathematical functions
abs(cpu_usage - 50)               # Absolute value
round(cpu_usage, 0.1)             # Round to nearest 0.1
sqrt(memory_usage)                # Square root

# Time functions
time()                            # Current Unix timestamp
```

```
hour()                          # Hour of day (0-23)
day_of_week()                   # Day of week (0-6, Sunday=0)
```

二項演算子

算術演算と論理演算を実行します。

```
# Arithmetic operators
cpu_usage + 10
memory_total - memory_available
disk_usage / disk_total * 100

# Comparison operators (return 0 or 1)
cpu_usage > 80
memory_usage < 1000
response_time >= 0.5

# Logical operators
(cpu_usage > 80) and (memory_usage > 1000)
(status_code == 200) or (status_code == 201)
```

実用的なクエリの例

Amazon Managed Service for Prometheus ワークスペースで使用できる一般的なモニタリングクエリ:

```
# CPU usage percentage
100 - (avg by (instance) (rate(node_cpu_seconds_total{mode="idle"}[5m])) * 100)

# Memory usage percentage
(1 - (node_memory_MemAvailable_bytes / node_memory_MemTotal_bytes)) * 100

# Request rate per second
sum(rate(http_requests_total[5m])) by (job)

# Error rate percentage
```

```
sum(rate(http_requests_total{status_code=~"5.."}[5m])) /
sum(rate(http_requests_total[5m])) * 100

# 95th percentile response time
histogram_quantile(0.95, sum(rate(http_request_duration_seconds_bucket[5m])) by (le))

# Top 5 instances by CPU usage
topk(5, avg by (instance) (cpu_usage))
```

メトリクスのクエリを保護する

Amazon Managed Service for Prometheus には、メトリクスのクエリの実行を保護するための手段が用意されています。

Amazon Managed Service for Prometheus AWS PrivateLink での の使用

Amazon Managed Service for Prometheus でメトリクスをクエリするためのネットワークトラフィックは、パブリックインターネットエンドポイントを介して、または VPC エンドポイントを介して実行できます AWS PrivateLink。を使用すると AWS PrivateLink、VPCs からのネットワークトラフィックは、パブリックインターネットを経由せずに AWS ネットワーク内で保護されます。Amazon Managed Service for Prometheus の AWS PrivateLink VPC エンドポイントを作成するには、「」を参照してくださいインターフェイス VPC エンドポイントでの Amazon Managed Service for Prometheus の使用。

認証と認可

AWS Identity and Access Management は、 リソースへのアクセス AWS を安全に制御するのに役立つウェブサービスです。IAM を使用して、誰を認証 (サインイン) し、誰にリソースの使用を認可する (アクセス許可を付与する) かを制御します。Amazon Managed Service for Prometheus は IAM と統合されているため、データを安全に保つことができます。Amazon Managed Service for Prometheus をセットアップするときは、Amazon Managed Service for Prometheus ワークスペースに保存されたメトリクスに対して Grafana サーバーからクエリを実行できるようにする IAM ロールを作成する必要があります。IAM の詳細については、「IAM とは」を参照してください。

Amazon Managed Service for Prometheus のセットアップに役立つもう 1 つの AWS セキュリティ機能は、 AWS 署名バージョン 4 の署名プロセス (AWS SigV4) です。署名バージョン 4 は、HTTP によって送信された AWS リクエストに認証情報を追加するプロセスです。セキュリティのため、

へのほとんどのリクエストは、アクセスキー ID とシークレットアクセスキーで構成されるアクセスキーで署名 AWS する必要があります。これらの 2 つのキーは、一般的にセキュリティ認証情報と呼ばれます。SigV4 の詳細については、「Signature Version 4 の署名プロセス」を参照してください。

Amazon Managed Service for Prometheus で使用するための Amazon Managed Grafana のセットアップ

Amazon Managed Grafana は、オープンソースの Grafana 向けのフルマネージドサービスで、オープンソースのサードパーティー ISV、およびデータソースを大規模に視覚化および分析するための AWS サービスへの接続を簡素化します。

Amazon Managed Service for Prometheus では、Amazon Managed Grafana を使用してワークスペース内のメトリクスにクエリを実行することがサポートされています。Amazon Managed Grafana コンソールで、既存の Amazon Managed Service for Prometheus アカウントを検出して、Amazon Managed Service for Prometheus ワークスペースをデータソースとして追加できます。Amazon Managed Grafana は、Amazon Managed Service for Prometheus にアクセスするために必要な認証情報の設定を管理します。Amazon Managed Grafana から Amazon Managed Service for Prometheus への接続を作成する方法の詳細については、「Amazon Managed Grafana User Guide」の手順を参照してください。

Amazon Managed Service for Prometheus のアラートを Amazon Managed Grafana で表示することもできます。アラートとの統合を設定する手順については、「アラートを Amazon Managed Grafana またはオープンソースの Grafana と統合する」を参照してください。

プライベート VPC での Amazon Managed Grafana への接続

Amazon Managed Service for Prometheus は、Amazon Managed Grafana がメトリクスやアラートのクエリを実行するときに接続するサービスエンドポイントを提供しています。

Amazon Managed Grafana は、プライベート VPC を使用するように構成できます (Grafana でプライベート VPC をセットアップする方法の詳細については、「Amazon Managed Grafana User Guide」の「Connecting to Amazon VPC」を参照してください)。設定によっては、この VPC から Amazon Managed Service for Prometheus のサービスエンドポイントにアクセスできない場合があります。

特定のプライベート VPC を使用するように構成されている Amazon Managed Grafana ワークスペースに、Amazon Managed Service for Prometheus をデータソースとして追加するには、まず VPC エンドポイントを作成して、Amazon Managed Service for Prometheus を同じ VPC に接続す

る必要があります。VPC エンドポイントの作成の詳細については、「Amazon Managed Service for Prometheus 用のインターフェイス VPC エンドポイントの作成」を参照してください。

Amazon Managed Service for Prometheus で使用する Grafana オープンソースまたは Grafana Enterprise のセットアップ

Grafana のインスタンスを使用して、Amazon Managed Service for Prometheus のメトリクスをクエリできます。このトピックでは、Grafana のスタンドアロンインスタンスを使用して Amazon Managed Service for Prometheus のメトリクスをクエリする方法について説明します。

前提条件

Grafana インスタンス – Amazon Managed Service for Prometheus で認証できる Grafana インスタンスが必要です。

Amazon Managed Service for Prometheus では、Grafana バージョン 7.3.5 以降を使用してワークスペース内のメトリクスにクエリを実行することがサポートされています。バージョン 7.3.5 以降には、 AWS 署名バージョン 4 (SigV4) 認証のサポートが含まれています。

Grafana のバージョンを確認するには、次のコマンドを入力します。*grafana_install_directory* は、Grafana インストールへのパスに置き換えます。

```
grafana_install_directory/bin/grafana-server -v
```

スタンドアロンの Grafana をまだ使用していないか、より新しいバージョンが必要な場合は、新しいインスタンスをインストールできます。スタンドアロンの Grafana をセットアップする手順については、Grafana ドキュメントの「Grafana のインストール」を参照してください。Grafana の開始方法については、Grafana ドキュメントの「Getting started with Grafana」を参照してください。

AWS アカウント – Amazon Managed Service for Prometheus のメトリクスにアクセスするには、正しいアクセス許可を持つ AWS アカウント が必要です。

Grafana を Amazon Managed Service for Prometheus と連携するように設定するには、AmazonPrometheusQueryAccess ポリシーまたは `aps:QueryMetrics`、`aps:GetMetricMetadata`、`aps:GetSeries`、`aps:GetLabels` のアクセス許可を持つアカウントにログオンする必要があります。詳細については、「IAM のアクセス許可とポリシー」を参照してください。

次のセクションでは、Grafana からの認証のセットアップについて詳しく説明します。

ステップ 1: AWS SigV4 をセットアップする

Amazon Managed Service for Prometheus は AWS Identity and Access Management (IAM) と連携して、Prometheus APIsで保護します。デフォルトでは、Grafana の Prometheus データソースは、Prometheus が認証を必要としないものと想定します。Grafana で Amazon Managed Service for Prometheus の認証および認可機能を利用できるようにするには、Grafana データソースで SigV4 認証サポートを有効にする必要があります。自己管理型の Grafana オープンソースサーバーまたは Grafana Enterprise サーバーを使用している場合は、このページの手順に従ってください。Amazon Managed Grafana を使用している場合、SigV4 認証は完全に自動化されます。Amazon Managed Grafana の詳細については、「Amazon Managed Grafana とは」を参照してください。

Grafana で SigV4 を有効にするには、`AWS_SDK_LOAD_CONFIG` および `GF_AUTH_SIGV4_AUTH_ENABLED` 環境変数を `true` に設定して Grafana を起動します。`GF_AUTH_SIGV4_AUTH_ENABLED` 環境変数は、Grafana のデフォルト設定をオーバーライドして SigV4 サポートを有効にします。詳細については、Grafana ドキュメントの「Configuration」を参照してください。

Linux

Linux 上のスタンドアロン Grafana サーバーで SigV4 を有効にするには、次のコマンドを入力します。

```
export AWS_SDK_LOAD_CONFIG=true
```

```
export GF_AUTH_SIGV4_AUTH_ENABLED=true
```

```
cd grafana_install_directory
```

```
./bin/grafana-server
```

Windows

Windows 上のスタンドアロン Grafana で SigV4 を有効にするには、Windows のコマンドプロンプトを使用して、次のコマンドを入力します。

```
set AWS_SDK_LOAD_CONFIG=true
```

```
set GF_AUTH_SIGV4_AUTH_ENABLED=true
```

```
cd grafana_install_directory
```

```
.\bin\grafana-server.exe
```

ステップ 2: Grafana で Prometheus データソースを追加する

以下の手順では、Grafana で Prometheus データソースを設定して、Amazon Managed Service for Prometheus メトリクスに対するクエリを実行する方法を説明します。

Grafana サーバーに Prometheus データソースを追加するには

1. Grafana コンソールを開きます。
2. [設定] で、[データソース] を選択します。
3. [データソースを追加] を選択します。
4. [Prometheus] を選択します。
5. HTTP URL として、Amazon Managed Service for Prometheus コンソールのワークスペースの詳細ページに表示される [エンドポイント - クエリ URL] を指定します。
6. 指定した HTTP URL から、URL に追加されている `/api/v1/query` という文字列を削除します。これは、Prometheus データソースによって自動的に追加されるためです。

   正しい URL は、https://aps-workspaces.us-west-2.amazonaws.com/workspaces/ws-1234a5b6-78cd-901e-2fgh-3i45j6k178l9 のようになります。
7. [認証] で、[SigV4 認証] のトグルを選択して有効にします。
8. SigV4 認可は、Grafana で長期認証情報を直接指定するか、デフォルトのプロバイダーチェーンを使用して構成できます。長期認証情報を直接指定する方がすぐに開始できるため、以下では最初にその手順を説明します。Amazon Managed Service for Prometheus で Grafana を使用することに慣れてきたら、デフォルトのプロバイダーチェーンを使用することをお勧めします。これにより、柔軟性とセキュリティが向上します。デフォルトのプロバイダーチェーンを設定する方法の詳細については、「Specifying Credentials」を参照してください。
   - 長期認証情報を直接使用するには、以下を実行します。
     a. [SigV4 認証の詳細] で、[認証プロバイダー] として [アクセスとシークレットキー] を選択します。
     b. [アクセスキー ID] に、 AWS アクセスキー ID を入力します。
     c. [シークレットアクセスキー] に、 AWS シークレットアクセスキーを入力します。

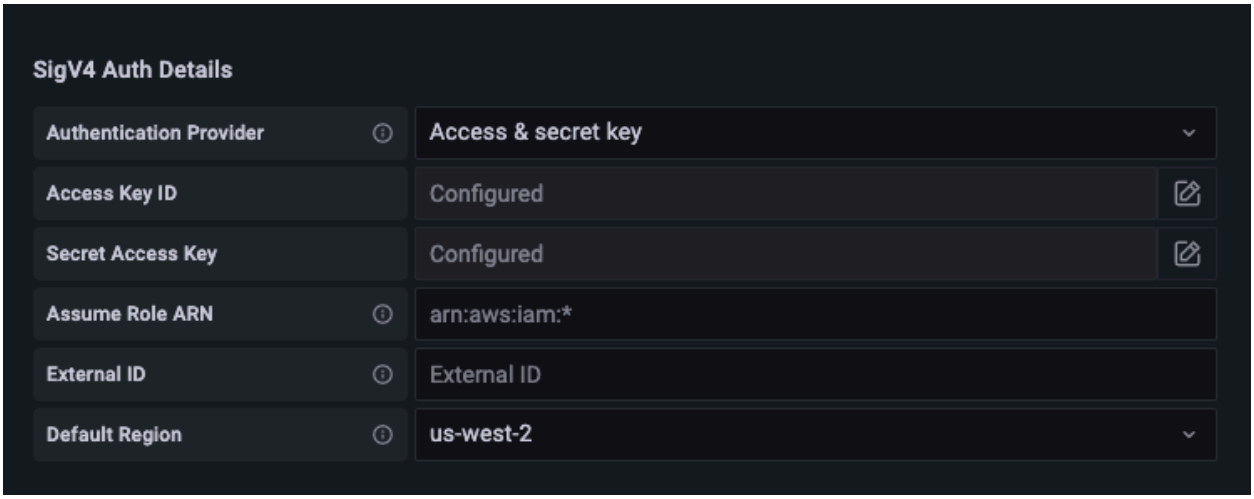
d. [引き受けロールの ARN] と [外部 ID] フィールドは空白のままにします。

e. [デフォルトのリージョン] で、Amazon Managed Service for Prometheus ワークスペースのリージョンを選択します。このリージョンは、ステップ 5 で指定した URL に含まれているリージョンと一致する必要があります。

f. [保存してテスト] を選択します。

「Data source is working」というメッセージが表示されます。

次のスクリーンショットは、アクセスキーとシークレットキーを含む SigV4 認証の詳細設定を示しています。



- 代わりにデフォルトのプロバイダーチェーンを使用するには (本番環境に推奨)、以下を実行します。

a. [SigV4 認証の詳細] で、[認証プロバイダー] として [AWS SDK のデフォルト] を選択します。

b. [引き受けロールの ARN] と [外部 ID] フィールドは空白のままにします。

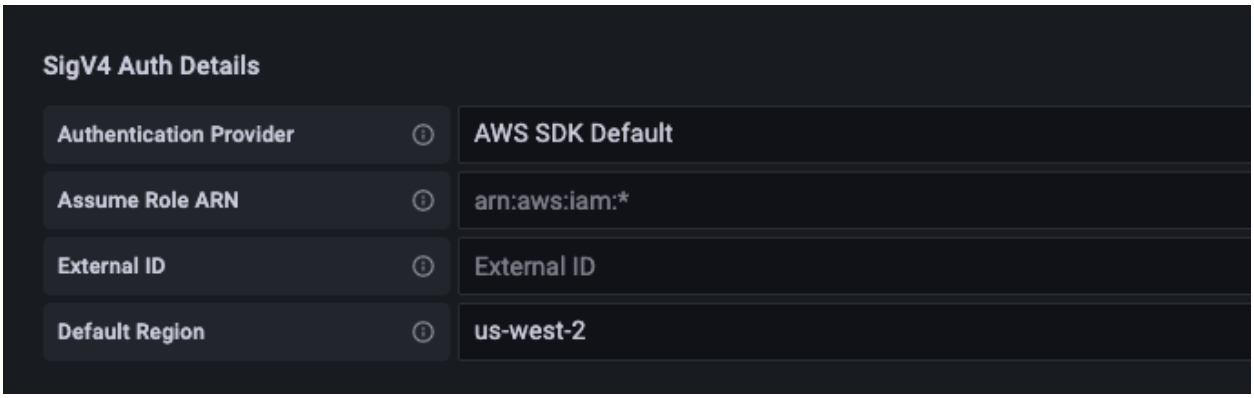
c. [デフォルトのリージョン] で、Amazon Managed Service for Prometheus ワークスペースのリージョンを選択します。このリージョンは、ステップ 5 で指定した URL に含まれているリージョンと一致する必要があります。

d. [保存してテスト] を選択します。

「Data source is working」というメッセージが表示されます。

このメッセージが表示されない場合は、次のセクションで、接続に関するトラブルシューティングのヒントを参照してください。

次のスクリーンショットは、SDK のデフォルトの SigV4 認証の詳細設定を示しています。



9. 新しいデータソースに対して PromQL クエリをテストします。

   a. [調査] を選択します。

   b. 次のようなサンプル PromQL クエリを実行します。

```
prometheus_tsdb_head_series
```

ステップ 3: (オプション) [保存してテスト] が機能しない場合のトラブルシューティング

前の手順で [保存してテスト] を選択したときにエラーが表示される場合は、以下を確認してください。

HTTP エラー: 見つかりません

URL に含まれているワークスペース ID が正しいことを確認してください。

HTTP エラー: 禁止されています

このエラーは、認証情報が無効であることを意味します。以下をチェックしてください:

- [デフォルトのリージョン] に指定したリージョンが正しいことを確認します。
- 認証情報に誤字がないことを確認します。
- 使用している認証情報に AmazonPrometheusQueryAccess ポリシーが適用されていることを確認します。詳細については、「IAM のアクセス許可とポリシー」を参照してください。

- 使用している認証情報に、この Amazon Managed Service for Prometheus ワークスペースへのアクセス権があることを確認します。

HTTP エラー: 不正なゲートウェイ

このエラーのトラブルシューティングを行うには、Grafana サーバーのログを確認します。詳細については、Grafana ドキュメントの「Troubleshooting」を参照してください。

が表示された場合**`Error http: proxy error: NoCredentialProviders: no valid providers in chain`**、デフォルトの認証情報プロバイダーチェーンは、使用する有効な AWS 認証情報を見つけることができませんでした。「Specifying Credentials」に従って認証情報が設定されていることを確認してください。共有設定を使用する場合は、`AWS_SDK_LOAD_CONFIG` 環境が `true` に設定されていることを確認してください。

Amazon EKS クラスターで動作する Grafana を使用したクエリ

Amazon Managed Service for Prometheus では、Grafana バージョン 7.3.5 以降を使用して Amazon Managed Service for Prometheus ワークスペース内のメトリクスにクエリを実行することがサポートされています。バージョン 7.3.5 以降には、 AWS 署名バージョン 4 (SigV4) 認証のサポートが含まれています。

Grafana を Amazon Managed Service for Prometheus と連携するように設定するには、AmazonPrometheusQueryAccess ポリシーまたは `aps:QueryMetrics`、`aps:GetMetricMetadata`、`aps:GetSeries`、`aps:GetLabels` のアクセス許可を持つアカウントにログオンする必要があります。詳細については、「IAM のアクセス許可とポリシー」を参照してください。

AWS SigV4 をセットアップする

Grafana は、 AWS 署名バージョン 4 (SigV4) 認証をサポートする新機能を追加しました。詳細については、「Signature Version 4 の署名プロセス」を参照してください。Grafana サーバーでは、この機能はデフォルトで有効になっていません。ここでは、Kubernetes クラスターへの Grafana のデプロイに Helm を使用しているものと想定して、この機能を有効にする手順を説明します。

Grafana 7.3.5 以降のサーバーで SigV4 を有効にするには

1. Grafana の設定をオーバーライドする新しい更新ファイルを作成し、`amp_query_override_values.yaml` という名前を付けます。

2. 以下の内容をファイルに入力し、ファイルを保存します。*account-id* を、Grafana サーバーが実行されている AWS アカウント ID に置き換えます。

```
serviceAccount:
    name: "amp-iamproxy-query-service-account"
    annotations:
        eks.amazonaws.com/role-arn: "arn:aws:iam::account-id:role/amp-iamproxy-
query-role"
grafana.ini:
  auth:
    sigv4_auth_enabled: true
```

この YAML ファイル内の amp-iamproxy-query-role は、次の「サービスアカウントの IAM ロールの設定」セクションで作成するロールの名前です。ワークスペースに対してクエリを実行するためのロールが既に作成されている場合は、ファイル内のロールを独自のロール名に置き換えることができます。

このファイルは、後の「Helm を使用した Grafana サーバーのアップグレード」で使用します。

サービスアカウントの IAM ロールの設定

Amazon EKS クラスターで Grafana サーバーを使用している場合は、アクセス制御にサービスアカウントの IAM ロール (サービスロールとも呼ばれます) を使用することをお勧めします。これを実行して IAM ロールを Kubernetes サービスアカウントに関連付けると、サービスアカウントはそのサービスアカウントを使用する任意のポッドのコンテナにアクセス AWS 許可を付与できます。詳細については、「サービスアカウントの IAM ロール」を参照してください。

これらのクエリ用のサービスロールをまだ設定していない場合は、「メトリクスのクエリを実行するためのサービスアカウントの IAM ロールの設定」の手順に従ってロールを設定します。

その後、信頼関係の条件に Grafana サービスアカウントを追加する必要があります。

信頼関係の条件に Grafana サービスアカウントを追加するには

1. ターミナルウィンドウから、Grafana サーバーの名前空間とサービスアカウント名を確認します。例えば、次のコマンドを使用できます。

```
kubectl get serviceaccounts -n grafana_namespace
```

2. Amazon EKS コンソールで、EKS クラスターに関連付けられているサービスアカウントの IAM ロールを開きます。
3. [信頼関係の編集] を選択します。
4. Condition を更新して、ステップ 1 のコマンド出力で確認した Grafana 名前空間と Grafana サービスアカウント名を含めます。以下に例を示します。

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.us-
east-1.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.eks.us-east-1.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": [
            "system:serviceaccount:aws-amp:amp-iamproxy-query-service-
account",
            "system:serviceaccount:grafana-namespace:grafana-service-account-
name"
          ],
          "oidc.eks.us-east-1.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
        }
      }
    }
  ]
}
```

5. [信頼ポリシーの更新] を選択します。

Helm を使用した Grafana サーバーのアップグレード

このステップでは、前のセクションで amp_query_override_values.yaml ファイルに追加したエントリを使用するように Grafana サーバーをアップグレードします。

以下のコマンドを実行します。Grafana 用の Helm チャートの詳細については、「Grafana Community Kubernetes Helm Charts」を参照してください。

```
helm repo add grafana https://grafana.github.io/helm-charts
```

```
helm upgrade --install grafana grafana/grafana -n grafana_namespace -f ./
amp_query_override_values.yaml
```

Grafana での Prometheus データソースの追加

以下の手順では、Grafana で Prometheus データソースを設定して、Amazon Managed Service for Prometheus メトリクスに対するクエリを実行する方法を説明します。

Grafana サーバーに Prometheus データソースを追加するには

1. Grafana コンソールを開きます。
2. [設定] で、[データソース] を選択します。
3. [データソースを追加] を選択します。
4. [Prometheus] を選択します。
5. HTTP URL として、Amazon Managed Service for Prometheus コンソールのワークスペースの詳細ページに表示される [エンドポイント - クエリ URL] を指定します。
6. 指定した HTTP URL から、URL に追加されている /api/v1/query という文字列を削除します。これは、Prometheus データソースによって自動的に追加されるためです。
7. [認証] で、[SigV4 認証] のトグルを選択して有効にします。

   [引き受けロールの ARN] と [外部 ID] フィールドは空白のままにします。次に、[デフォルトのリージョン] で、Amazon Managed Service for Prometheus ワークスペースのあるリージョンを選択します。
8. [保存してテスト] を選択します。

   「Data source is working」というメッセージが表示されます。

9. 新しいデータソースに対して PromQL クエリをテストします。

   a. [調査] を選択します。

   b. 次のようなサンプル PromQL クエリを実行します。

```
prometheus_tsdb_head_series
```

Prometheus 互換 API を使用したクエリ

メトリクスの表示とクエリを行うには Amazon Managed Grafana などのツールを使用する方法が最も簡単ですが、Amazon Managed Service for Prometheus では、メトリクスのクエリに使用できる Prometheus 互換 API もいくつかサポートされています。使用可能なすべての Prometheus 互換 API の詳細については、「Prometheus 互換 API」を参照してください。

Prometheus 互換 API は、Prometheus クエリ言語 PromQL を使用して、返すデータを指定します。PromQL とその構文の詳細については、Prometheus ドキュメントの「Querying Prometheus」を参照してください。

これらの API を使用してメトリクスにクエリを実行する場合、リクエストは AWS Signature Version 4 の署名プロセスで署名されている必要があります。AWS Signature Version 4 をセットアップすると、署名プロセスを簡略化できます。詳細については、「aws-sigv4-proxy」を参照してください。

AWS SigV4 プロキシを介した署名は、 を使用して実行できますawscurl。次のトピックでは、awscurl を使用して Prometheus 互換 APIs をクエリし、 を使用して AWS SigV4 を設定する方法について説明します。 awscurl

トピック

- awscurl を使用して Prometheus 互換 API でクエリを実行する

awscurl を使用して Prometheus 互換 API でクエリを実行する

Amazon Managed Service for Prometheus の API リクエストは、SigV4 で署名する必要があります。awscurl を使用すると、クエリのプロセスを簡略化できます。

awscurl をインストールするには、Python 3 と pip パッケージマネージャーがインストールされている必要があります。

Linux ベースのインスタンスでは、次のコマンドで awscurl をインストールします。

```
$ pip3 install awscurl
```

macOS マシンでは、次のコマンドで awscurl をインストールします。

```
$ brew install awscurl
```

次の例は、サンプルの awscurl クエリです。*Region*、*Workspace-id*、*QUERY* の各入力は、ユースケースに応じた値に置き換えます。

```
# Define the Prometheus query endpoint URL. This can be found in the Amazon Managed
 Service for Prometheus console page
# under the respective workspace.

$ export AMP_QUERY_ENDPOINT=https://aps-workspaces.Region.amazonaws.com/
workspaces/Workspace-id/api/v1/query

# credentials are infered from the default profile
$ awscurl -X POST --region Region \
                  --service aps "${AMP_QUERY_ENDPOINT}" -d 'query=QUERY'  --header
 'Content-Type: application/x-www-form-urlencoded'
```

> **Note**
>
> クエリ文字列は、URL エンコードする必要があります。

query=up などのクエリでは、次のような結果が得られます。

```
{
  "status": "success",
  "data": {
    "resultType": "vector",
    "result": [
      {
        "metric": {
          "__name__": "up",
          "instance": "localhost:9090",
          "job": "prometheus",
          "monitor": "monitor"
        },
        "value": [
```

```
          1652452637.636,
          "1"
        ]
      },
    ]
  }
}
```

指定したリクエストに awscurl で署名するには、有効な認証情報を以下のいずれかの方法で渡す必要があります。

- IAM ロールのアクセスキー ID とシークレットキーを指定する。ロールのアクセスキーとシークレットキーは、https://console.aws.amazon.com/iam/ で確認できます。

  例えば、次のようになります。

  ```
  $ export AMP_QUERY_ENDPOINT=https://aps-workspaces.Region.amazonaws.com/
  workspaces/Workspace_id/api/v1/query

  $ awscurl -X POST --region <Region> \
                    --access_key <ACCESS_KEY> \
                    --secret_key <SECRET_KEY> \
                    --service aps "$AMP_QUERY_ENDPOINT?query=<QUERY>"
  ```

- .aws/credentials および /aws/config ファイルに保存されている設定ファイルを参照する。使用するプロファイルの名前を指定することもできます。指定しない場合、 default ファイルが使用されます。例えば、次のようになります。

  ```
  $ export AMP_QUERY_ENDPOINT=https://aps-workspaces.<Region>.amazonaws.com/workspaces/
  <Workspace_ID>/api/v1/query
  $ awscurl -X POST --region <Region> \
                    --profile <PROFILE_NAME>
                    --service aps "$AMP_QUERY_ENDPOINT?query=<QUERY>"
  ```

- EC2 インスタンスに関連付けられているインスタンスプロファイルを使用する。

awscurl コンテナを使用したクエリリクエストの実行

別のバージョンの Python がインストールされていて、関連する依存関係を満たすことができない場合は、コンテナを使用して awscurl アプリケーションとその依存関係をパッケージ化できます。次

の例では Docker ランタイムを使用して awscurl をデプロイしますが、OCI 準拠の任意のランタイムとイメージを使用できます。

```
$ docker pull okigan/awscurl
$ export AMP_QUERY_ENDPOINT=https://aps-workspaces.Region.amazonaws.com/
workspaces/Workspace_id/api/v1/query
$ docker run --rm -it okigan/awscurl --access_key $AWS_ACCESS_KEY_ID  --secret_key
 $AWS_SECRET_ACCESS_KEY \ --region Region --service aps "$AMP_QUERY_ENDPOINT?
query=QUERY"
```

各クエリのクエリ使用状況に関する統計を取得する

クエリの料金は、実行されたクエリから 1 か月間に処理されたクエリサンプルの合計数に基づいて計算されます。行った各クエリに関する統計を取得して、処理されたサンプルを追跡できます。リクエストにクエリパラメータ stats=all を含めることで、query または queryRange API のクエリレスポンスに、処理されたクエリサンプルに関する統計データを含めることができます。samples オブジェクトが stats オブジェクト内に作成され、stats データがレスポンスで返されます。

samples オブジェクトは以下の属性で構成されます。

属性	説明
totalQueryableSamples	処理されたクエリサンプルの合計数。これが請求に使用される情報です。
totalQueryableSamplesPerStep	各ステップで処理されたクエリサンプルの数。これは、エポックタイムスタンプと、その特定のステップでロードされたサンプルの数から成る配列の配列として表されます。

サンプルのリクエストと、stats 情報を含むレスポンスの例を以下に示します。

query の例:

GET

```
endpoint/api/v1/query?query=up&time=1652382537&stats=all
```

[応答]

```
{
    "status": "success",
    "data": {
        "resultType": "vector",
        "result": [
            {
                "metric": {
                    "__name__": "up",
                    "instance": "localhost:9090",
                    "job": "prometheus"
                },
                "value": [
                    1652382537,
                    "1"
                ]
            }
        ],
        "stats": {
            "timings": {
                "evalTotalTime": 0.00453349,
                "resultSortTime": 0,
                "queryPreparationTime": 0.000019363,
                "innerEvalTime": 0.004508405,
                "execQueueTime": 0.000008786,
                "execTotalTime": 0.004554219
            },
            "samples": {
                "totalQueryableSamples": 1,
                "totalQueryableSamplesPerStep": [
                    [
                        1652382537,
                        1
                    ]
                ]
            }
        }
    }
}
```


queryRange の例:

GET

```
endpoint/api/v1/query_range?query=sum+%28rate+%28go_gc_duration_seconds_count%5B1m%5D
%29%29&start=1652382537&end=1652384705&step=1000&stats=all
```

[応答]

```
{
    "status": "success",
    "data": {
        "resultType": "matrix",
        "result": [
            {
                "metric": {},
                "values": [
                    [
                        1652383000,
                        "0"
                    ],
                    [
                        1652384000,
                        "0"
                    ]
                ]
            }
        ],
        "stats": {
            "samples": {
                "totalQueryableSamples": 8,
                "totalQueryableSamplesPerStep": [
                    [
                        1652382000,
                        0
                    ],
                    [
                        1652383000,
                        4
                    ],
                    [
                        1652384000,
                        4
                    ]
                ]
            }
        }
```

```
    }
}
```

異常検出

Amazon Managed Service for Prometheus は、機械学習アルゴリズムを使用してメトリクスデータの異常なパターンを自動的に識別する異常検出機能を提供します。この機能は、静的しきい値ではなく真に異常な動作に焦点を当てることにより、潜在的な問題を積極的に検出し、アラートの疲労を軽減し、モニタリングの有効性を向上させるのに役立ちます。

Amazon Managed Service for Prometheus の異常検出では、ランダムカットフォレスト (RCF) アルゴリズムを使用します。このアルゴリズムは、時系列データを分析して通常の動作パターンを確立し、それらのパターンからの偏差を特定します。このアルゴリズムは季節的な傾向に適応し、欠落したデータを適切に処理して、検出された異常の信頼度スコアを提供します。

異常検出が動作する仕組み

Amazon Managed Service for Prometheus の異常検出では、機械学習を使用して、手動しきい値設定なしでメトリクスデータの異常なパターンを特定します。システムは通常の動作パターンと季節的なバリエーションを学習し、誤検出を減らして、問題を早期に検出できるようにします。アプリケーションの変更に継続的に適応するため、動的なクラウド環境に適しています。

異常検出は、応答時間やエラー率などのアプリケーションパフォーマンスメトリクスをモニタリングし、CPU とメモリの使用状況を通じてインフラストラクチャの状態を追跡して、異常なユーザー動作を検出し、トラフィック分析を通じてキャパシティプランニングのニーズを特定して、予期しない変更がないかビジネスメトリクスをモニタリングします。予測可能なパターン、季節的な変動、または段階的な成長傾向に最適です。

ランダムカットフォレスト (RCF) アルゴリズムは、時系列データの分析に使用されます。RCF は、データスペースを分割し、正規分布から遠く離れた分離ポイントを識別する決定木を作成します。このアルゴリズムは、受信データから学習し、各メトリクスの通常の動作の動的モデルを構築します。

有効にすると、履歴データを分析してベースラインパターンと季節的傾向を確立し、予想される値の予測を生成して偏差を特定します。アルゴリズムは 4 つのキー出力を生成します。

- upper_band - 予想される正常値の上限
- lower_band - 予想される法線値の下限
- score - データポイントの異常を示す数値異常スコア
- value - 実際の観測メトリクス値

異常検出の開始方法

Prometheus メトリクスで異常検出の使用を開始するには、アルゴリズムが通常のパターンを学習するための十分な履歴データが必要です。最適な結果を得るためには、異常検出を有効にする前に、少なくとも 14 日間の一貫したメトリクスデータを用意することをお勧めします。

PreviewAnomalyDetector API を使用して、異常検出がメトリクスとどのように連携するかをプレビューできます。PreviewAnomalyDetector を使用して、履歴データに対してアルゴリズムをテストし、実稼働モニタリングに実装する前にその有効性を評価します。詳細については、「PreviewAnomalyDetector API」を参照してください。

異常検出を実装するときは、以下のベストプラクティスを考慮してください。

- 安定したメトリクスから始める – 一貫したパターンを持つメトリクスから始め、最初は揮発性が高いデータやスパースなデータを避けます。
- 集約データを使用する – パフォーマンスと精度を向上させるために、未加工の高カーディナリティデータではなく、集約されたメトリクス (平均や合計など) に異常検出を適用します。
- 感度の調整 — 特定のユースケースと誤検出と見逃した異常の許容度に基づいてアルゴリズムパラメータを調整します。
- アルゴリズムのパフォーマンスをモニタリングする – 検出された異常を定期的にレビューして、システムが進化するにつれてアルゴリズムが引き続き貴重なインサイトを提供していることを確認します。

PreviewAnomalyDetector API

PreviewAnomalyDetector オペレーションを使用して、指定した期間に異常検出アルゴリズムによってメトリクスデータがどのように分析されるかを示すエンドポイントを作成します。このエンドポイントは、実装前にディテクターのパフォーマンスを評価および検証するのに役立ちます。

有効な HTTP 動詞

GET, POST

サポートされているペイロードタイプ

URL エンコードされたパラメータ

POST 用の application/x-www-form-urlencoded

サポートされているパラメータ

query=<string> Prometheus 式のクエリ文字列。

start=<rfc3339 | unix_timestamp> query_range を使用して一定期間にわたってクエリを評価する場合、開始タイムスタンプ。

end=<rfc3339 | unix_timestamp> query_range を使用して一定期間にわたってクエリを評価する場合、終了タイムスタンプ。

step=<duration | float> クエリの解決ステップ幅 (duration 形式または float の秒数)。query_range を使用して一定期間にわたってクエリを評価するときに、そのクエリで必要とされる場合にのみ使用します。

クエリパラメータの形式

元の PromQL 式をクエリパラメータの RandomCutForest (RCF) 擬似関数でラップします。詳細については、「Amazon Managed Service for Prometheus API リファレンス」の「RandomCutForestConfiguration」を参照してください。

RCF 関数は次の形式を使用します。

```
RCF(<query>
[,shingle size
[,sample size
[,ignore near expected from above
[,ignore near expected from below
[,ignore near expected from above ratio
[,ignore near expected from below ratio]]]]])
```

クエリを除くすべてのパラメータはオプションであり、省略するとデフォルト値が使用されます。最小構文は次のとおりです。

```
RCF(<query>)
```

クエリを集計関数でラップする必要があります。他のパラメータを省略するときに特定のオプションパラメータを使用するには、関数に空の位置を残します。

```
RCF(<query>,,,,,1.0,1.0)
```

この例では、期待値と観測値の比率に基づいて、異常検出のスパイクとドロップを無視する比率パラメータのみを設定します。

API リクエストと応答

成功した呼び出しは、QueryMetrics API と同じ形式を返します。API は十分なサンプルが利用可能になると、元の時系列に加えて、これらの新しい時系列を返します。

- anomaly_detector_preview:lower_band — PromQL 式の結果の期待値の下限帯域
- anomaly_detector_preview:score – 0 ~ 1 の異常スコア。1 はそのデータポイントでの異常の信頼度が高いことを示します。
- anomaly_detector_preview:upper_band — PromQL 式の結果の期待値の上限帯域

リクエスト例

```
POST /workspaces/workspace-id/anomalydetectors/preview
Content-Type: application/x-www-form-urlencoded

query=RCF%28avg%28vector%28time%28%29%29%29%2C%208%2C
%20256%29&start=1735689600&end=1735695000&step=1m
```

レスポンス例

```
200 OK
...

{
  "status": "success",
  "data": {
    "result": [
      {
        "metric": {},
        "values": [
          [
            1735689600,
            "1735689600"
          ],
          [
            1735689660,
```

```
            "1735689660"
          ],
          .........
        ]
      },
      {
        "metric": {
          "anomaly_detector_preview": "upper_band"
        },
        "values": [
          [
            1735693500,
            "1.7356943E9"
          ],
          [
            1735693560,
            "1.7356945E9"
          ]
          ],
          .........
        ]
      },
      {
        "metric": {
          "anomaly_detector_preview": "lower_band"
        },
        "values": [
          [
            1735693500,
            "1.7356928E9"
          ],
          [
            1735693560,
            "1.7356929E9"
          ],
          .........
        ]
      },
      {
        "metric": {
          "anomaly_detector_preview": "score"
        },
        "values": [
          [
```

```
            1735693500,
            "0.0"
          ],
          [
            1735695000,
            "0.0"
          ],
          .........
        ]
      }
    ],
    "resultType": "matrix"
  }
}
```

ルールを使用して、受信したメトリクスを変更またはモニタリングする

Amazon Managed Service for Prometheus で受信したメトリクスに対してアクションを実行するルールを設定できます。これらのルールは、メトリクスをモニタリングしたり、受信したメトリクスに基づいて新しい計算されたメトリクスを作成したりできます。

Amazon Managed Service for Prometheus は、定期的に評価される 2 種類のルールをサポートしています。

- 記録ルールでは、頻繁に必要になる式や計算負荷の高い式を事前に計算し、その結果を新しい時系列セットとして保存できます。多くの場合、事前に計算された結果に対してクエリを実行する方が、元の式を必要時に毎回実行するよりもはるかに高速です。
- アラートルールでは、PromQL としきい値に基づいてアラート条件を定義できます。ルールによってしきい値がトリガーされると、アラートマネージャーに通知が送信されます。アラートマネージャーでルールを管理するように設定したり、ルールを通知ダウンストリームのレシーバー (Amazon Simple Notification Service など) に転送するように設定したりできます。

Amazon Managed Service for Prometheus でルールを使用するには、ルールを定義する 1 つ以上の YAML ルールファイルを作成します。Amazon Managed Service for Prometheus のルールファイルの形式は、スタンドアロンの Prometheus のルールファイルと同じです。詳細については、Prometheus ドキュメントの「Defining Recording rules」と「Alerting rules」を参照してください。

ワークスペースには複数のルールファイルを含めることができます。それぞれのルールファイルは、別々の名前空間に格納されます。ルールファイルを複数にすれば、既存の Prometheus ルールファイルを変更したり結合したりする必要なく、そのままワークスペースにインポートできます。また、異なるルールグループ名前空間には、異なるタグを付けることができます。

ルールの順序

ルールファイル内では、ルールはルールグループに格納されます。ルールファイルの 1 つのルールグループ内のルールは、常に上から下に順番に評価されます。したがって、記録ルールでは、ある記録ルールの結果を、同じルールグループに含まれている後の記録ルールの計算やアラートルールで使用できます。ただし、個々のルールファイルの実行順序は指定できないため、ある記録ルールの結果を使用して別のルールグループまたは別のルールファイル内のルールを計算することはできません。

トピック

ルールの使用に必要な IAM アクセス許可を理解する

ユーザーに、Amazon Managed Service for Prometheus でルールを使用するためのアクセス許可を付与する必要があります。次のアクセス許可を持つ AWS Identity and Access Management (IAM) ポリシーを作成し、そのポリシーをユーザー、グループ、またはロールに割り当てます。

> **Note**
>
> IAM の詳細については、「Amazon Managed Service for Prometheus の Identity and Access Management」を参照してください。

ルールを使用するためのアクセス権を付与するポリシー

次のポリシーは、ルールを使用するためのアクセス権を、アカウント内のすべてのリソースに付与します。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "aps:CreateRuleGroupsNamespace",
                "aps:ListRuleGroupsNamespaces",
                "aps:DescribeRuleGroupsNamespace",
```

```
                "aps:PutRuleGroupsNamespace",
                "aps:DeleteRuleGroupsNamespace"
            ],
            "Resource": "*"
        }
    ]
}
```

1 つの名前空間にのみアクセス権を付与するポリシー

特定のポリシーにのみアクセス権を付与するポリシーを作成することもできます。次のサンプルポリシーは、指定された RuleGroupNamespace にのみアクセス権を付与します。このポリシーを使用するには、*<account>*、*<region>*、*<workspace-id>*、*<namespace-name>* を、アカウントに応じた適切な値に置き換えます。

ルールファイルを作成する

Amazon Managed Service for Prometheus でルールを使用するには、ルールを定義するルールファイルを作成します。Amazon Managed Service for Prometheus のルールファイルは、スタンドアロンの Prometheus のルールファイルと同じ形式の YAML テキストファイルです。詳細については、Prometheus ドキュメントの「Defining Recording rules」と「Alerting rules」を参照してください。

基本的なルールファイルの例を以下に示します。

```
groups:
  - name: cpu_metrics
    interval: 60s
    rules:
     - record: avg_cpu_usage
       expr: avg(rate(node_cpu_seconds_total[5m])) by (instance)
     - alert: HighAverageCPU
       expr: avg_cpu_usage > 0.8
       for: 10m
       keep_firing_for: 20m
       labels:
         severity: critical
       annotations:
         summary: "Average CPU usage across cluster is too high"
```

この例では、60 秒ごとに評価されるルールグループ cpu_metrics を作成します。このルールグループは、avg_cpu_usage という名前の記録ルールを使用して新しいメトリクスを作成し、それをアラートで使用します。使用されるプロパティの一部について以下に説明します。含めることができるアラートルールやその他のプロパティの詳細については、Prometheus ドキュメントの「Alerting rules」を参照してください。

- `record: avg_cpu_usage` – この記録ルールは、avg_cpu_usage という新しいメトリクスを作成します。
- `interval` プロパティが指定されていない場合、ルールグループのデフォルトの評価間隔は 60 秒です。
- `expr: avg(rate(node_cpu_seconds_total[5m])) by (instance)` – この記録ルールの式は、各ノードの過去 5 分間の CPU 平均使用率を計算し、instance ラベル別にグループ化します。
- `alert: HighAverageCPU` – このアラートルールは、HighAverageCPU という新しいアラートを作成します。
- `expr: avg_cpu_usage > 0.8` – この式は、CPU 平均使用率が 80% を超えるサンプルを検索するようにアラートに指示します。
- `for: 10m` – アラートは、CPU の平均使用率が少なくとも 10 分間 80% を超えた場合にのみ発生します。

  この場合、メトリクスは過去 5 分間の平均として計算されます。したがって、アラートは、平均 CPU 使用率が 80% を超える 5 分間のサンプル (合計 10 分) が連続して 2 つ以上ある場合にのみ発生します。
- `keep_firing_for: 20m` – このアラートは、サンプルが少なくとも 20 分間しきい値を下回るまで引き続き発生します。これは、アラートが連続して上昇と下降を繰り返すのを防ぐのに役立ちます。

> **Note**
>
> ルール定義ファイルをローカルで作成して Amazon Managed Service for Prometheus にアップロードするか、Amazon Managed Service for Prometheus コンソール内で直接、定義を作成、編集、アップロードできます。どちらの場合でも、同じフォーマットルールが適用されます。ファイルのアップロードと編集の詳細については、「Amazon Managed Service for Prometheus にルール設定ファイルをアップロードする」を参照してください。

Amazon Managed Service for Prometheus にルール設定ファイルをアップロードする

ルール設定ファイルに必要なルールを確認したら、コンソール内でファイルを作成して編集できます。または、コンソールや AWS CLIを使用してファイルをアップロードできます。

> **Note**
>
> Amazon EKS クラスターを実行している場合は、AWS Controllers for Kubernetes を使用してルール設定ファイルをアップロードすることもできます。

Amazon Managed Service for Prometheus コンソールを使用してルール設定の編集または置換、および名前空間の作成を行うには

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ページの左上隅にあるメニューアイコンを選択し、[すべての WorkSpaces] を選択します。
3. ワークスペースのワークスペース ID を選択し、[ルール管理] タブを選択します。
4. [名前空間を追加] を選択します。
5. [ファイルを選択] を選択し、ルール定義ファイルを選択します。

   または、[構成を定義] を選択して、Amazon Managed Service for Prometheus コンソール内で直接、ルール定義ファイルを作成および編集することもできます。これにより、サンプルデフォルト定義ファイルが作成されます。これを編集してアップロードします。
6. (オプション) 名前空間にタグを追加するには、[新しいタグを追加] を選択します。

   [キー] にタグの名前を入力します。[値] では、任意でタグに値を追加できます。

   別のタグを追加するには、[新しいタグを追加] を選択します。
7. [続行] を選択します。Amazon Managed Service for Prometheus は、選択したルールファイルと同じ名前で新しい名前空間を作成します。

を使用して AWS CLI アラートマネージャー設定を新しい名前空間のワークスペースにアップロードするには

1. アラートマネージャーファイルの内容を base64 でエンコードします。Linux では、次のコマンドを使用できます。

   ```
   base64 input-file output-file
   ```

   macOS では、次のコマンドを使用できます。

   ```
   openssl base64 input-file output-file
   ```

2. 以下のいずれかのコマンドを入力して、名前空間の作成とファイルのアップロードを行います。

   AWS CLI バージョン 2 で、次のように入力します。

   ```
   aws amp create-rule-groups-namespace --data file://path_to_base_64_output_file --
   name namespace-name  --workspace-id my-workspace-id --region region
   ```

   AWS CLI バージョン 1 で、次のように入力します。

   ```
   aws amp create-rule-groups-namespace --data fileb://path_to_base_64_output_file --
   name namespace-name  --workspace-id my-workspace-id --region region
   ```

3. アラートマネージャーの設定が有効になるまで数秒かかります。ステータスを確認するには、次のコマンドを入力します。

   ```
   aws amp describe-rule-groups-namespace --workspace-id workspace_id --
   name namespace-name --region region
   ```

   `status` が `ACTIVE` であれば、ルールファイルが有効になっています。

ルール設定ファイルを編集または置換する

Amazon Managed Service for Prometheus に既にアップロードしたルールファイルのルールを変更する場合は、新しいルールファイルをアップロードして既存の設定を置き換えるか、コンソールで現在の設定を直接編集できます。必要に応じて、現在のファイルをダウンロードし、テキストエディタで編集して、新しいバージョンをアップロードできます。

Amazon Managed Service for Prometheus コンソールを使用してルール設定を編集するには

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ページの左上隅にあるメニューアイコンを選択し、[すべての WorkSpaces] を選択します。
3. ワークスペースのワークスペース ID を選択し、[ルール管理] タブを選択します。
4. 編集するルール設定ファイルの名前を選択します。
5. (オプション) 現在のルール設定ファイルをダウンロードする場合は、[ダウンロード] または [コピー] を選択します。
6. [変更] を選択して、コンソール内で設定を直接編集します。完了したら、[保存] を選択します。

   または、[構成を置き換える] を選択して、新しい設定ファイルをアップロードすることもできます。その場合は、新しいルール定義ファイルを選択し、[続行] を選択してアップロードします。

を使用してルール設定ファイル AWS CLI を編集するには

1. ルールファイルの内容を base64 でエンコードします。Linux では、次のコマンドを使用できます。

   ```
   base64 input-file output-file
   ```

   macOS では、次のコマンドを使用できます。

   ```
   openssl base64 input-file output-file
   ```

2. 以下のいずれかのコマンドを入力して、新しいファイルをアップロードします。

   AWS CLI バージョン 2 で、次のように入力します。

   ```
   aws amp put-rule-groups-namespace --data file://path_to_base_64_output_file --
   name namespace-name  --workspace-id my-workspace-id --region region
   ```

   AWS CLI バージョン 1 で、次のように入力します。

   ```
   aws amp put-rule-groups-namespace --data fileb://path_to_base_64_output_file --
   name namespace-name  --workspace-id my-workspace-id --region region
   ```

3. ルールファイルが有効になるまで数秒かかります。ステータスを確認するには、次のコマンドを入力します。

```
aws amp describe-rule-groups-namespace --workspace-id workspace_id --
name namespace-name --region region
```

`status` が `ACTIVE` であれば、ルールファイルが有効になっています。それまでは、このルールファイルの以前のバージョンが有効なままになります。

ルール評価のトラブルシューティング

このガイドでは、Amazon Managed Service for Prometheus (AMP) のルール評価に関する一般的な問題のトラブルシューティング手順をステップバイステップで説明します。アラートルールと記録ルールの問題を診断して解決するには、次の手順に従います。

トピック

- アラート実行ステータスを検証する
- 欠落しているアラート通知を解決する
- ルールのヘルスステータスを確認する
- クエリでオフセットを使用して取り込みの遅延を処理する
- 一般的な の問題と解決策
- ルール評価のベストプラクティス

アラート実行ステータスを検証する

ルール評価の問題をトラブルシューティングするときは、まず合成時系列 `ALERTS` をクエリしてアラートが実行されたかどうかを確認します。`ALERTS` 時系列には次のラベルが含まれます。

- alertname – アラートの名前。
- alertstate – [保留中] または [実行中]。
  - pending – アラートは `for` 句で指定された期間待機しています。
  - firing — アラートが指定された期間、条件を満たしています。追加のラベルはアラートルールで定義されます。

> **Note**
>
> アラートが [実行中] または [保留中] の場合、サンプル値は [1] です。アラートがアイドル状態の場合、サンプルは生成されません。

欠落しているアラート通知を解決する

アラートが実行中でも通知が届かない場合は、次のアラートマネージャー設定を確認します。

1. アラートマネージャーの設定を確認する – ルートレシーバーと設定が正しく設定されていることを確認します。アラートの実行に影響を与える可能性のある待機時間、時間間隔、必要なラベルなどのルートブロック設定を確認します。アラートルールと対応するルートとレシーバーを比較して、適切な一致を確認します。`time_interval` を持つルートの場合、タイムスタンプが指定された間隔内にあることを確認します。
2. アラートレシーバーのアクセス許可を確認する – Amazon SNS トピックを使用する場合は、AMP が通知を発行するために必要なアクセス許可を持っていることを確認します。詳細については、「Amazon SNS トピックにアラートメッセージを送信するためのアクセス許可を Amazon Managed Service for Prometheus に付与する」を参照してください。
3. レシーバーペイロードの互換性を検証する – アラートレシーバーがアラートマネージャーのペイロード形式を受け入れていることを確認します。Amazon SNS の要件については、「Amazon SNS メッセージ検証ルールを理解する」を参照してください。
4. アラートマネージャーログの確認 – AMP はアラートマネージャーから公開ログを提供し、通知問題のデバッグに役立ちます。詳細については、「CloudWatch Logs で Amazon Managed Service for Prometheus イベントをモニタリングする」を参照してください。

アラートマネージャーの詳細については、「アラートマネージャーを使用して Amazon Managed Service for Prometheus でアラートを管理および転送する」を参照してください。

ルールのヘルスステータスを確認する

ルールの形式が正しくないと、評価が失敗する可能性があります。ルールが評価に失敗した理由を特定するには、次の方法を使用します。

Example

ListRules API を使用する

ListRules API は、ルールのヘルスに関する情報を提供します。health および lastError フィールドをチェックして、問題を診断します。

レスポンスの例:

```
{
  "status": "success",
  "data": {
    "groups": [
      {
        "name": "my_rule_group",
        "file": "my_namespace",
        "rules": [
          {
            "state": "firing",
            "name": "broken_alerting_rule",
            "query": "...",
            "duration": 0,
            "keepFiringFor": 0,
            "labels": {},
            "annotations": {},
            "alerts": [],
            "health": "err",
            "lastError": "vector contains metrics with the same labelset after applying
 alert labels",
            "type": "alerting",
            "lastEvaluation": "1970-01-01T00:00:00.00000000Z",
            "evaluationTime": 0.08
          }
        ]
      }
    ]
  }
}
```

Example

公開ログを使用する

ListRules API には、最新の情報のみが表示されます。より詳細な履歴については、ワークスペースで公開ログを有効にして以下にアクセスします。

- 評価失敗のタイムスタンプ

- 詳しいエラーメッセージ
- 履歴評価データ

公開ログメッセージの例:

```
{
  "workspaceId": "ws-a2c55905-e0b4-4065-a310-d83ce597a391",
  "message": {
    "log": "Evaluating rule failed, name=broken_alerting_rule, group=my_rule_group,
 namespace=my_namespace, err=vector contains metrics with the same labelset after
 applying alert labels",
    "level": "ERROR",
    "name": "broken_alerting_rule",
    "group": "my_rule_group",
    "namespace": "my_namespace"
  },
  "component": "ruler"
}
```

ルーラーまたはアラートマネージャーからのログのその他の例については、「ルーラーのトラブルシューティング」および「アラートマネージャーを使用して Amazon Managed Service for Prometheus でアラートを管理および転送する」を参照してください。

クエリでオフセットを使用して取り込みの遅延を処理する

デフォルトでは、式は評価時に値を使用してオフセットなし (インスタントクエリ) で評価されます。メトリクスの取り込みが遅れた場合、記録ルールは、すべてのメトリクスが取り込まれた後に式を手動で評価する場合と同じ値を表さない場合があります。

> **Tip**
>
> オフセット修飾子を使用すると、取り込みの遅延による問題を減らすことができます。詳細については、Prometheus ドキュメントの「オフセット修飾子」を参照してください。

例: 遅延メトリクスの処理

ルールが 12:00 に評価して、取り込みの遅延によりメトリクスの最新のサンプルが 11:45 からのものである場合、ルールでは 12:00 タイムスタンプでサンプルは見つかりません。これを軽減するには、**`my_metric_name offset 15m`** などのオフセットを追加します。

例: 複数のソースからのメトリクスを処理する

メトリクスが、2 つのサーバーなど異なるソースから生成される場合、メトリクスは異なる時間に取り込まれる可能性があります。これを軽減するには、**`metric_from_server_A / metric_from_server_B`** のような式を形成します。

ルールがサーバー A とサーバー B の取り込み時間の間で評価されると、予期しない結果が得られます。オフセットを使用すると、評価時間を調整できます。

一般的な の問題と解決策

記録ルールデータにおけるギャップ

手動評価と比較した記録ルールデータのギャップに気付いた場合 (クエリ API または UI を使用して記録ルールの元の PromQL 式を直接実行した場合)、次のいずれかが原因である可能性があります。

1. 長い評価時間 – ルールグループは複数の同時評価を持つことはできません。評価時間が設定された間隔を超えると、後続の評価が失われる可能性があります。設定された間隔を超える複数の連続した評価の欠落により、記録ルールが古くなる可能性があります。詳細については、Prometheus ドキュメントの「古さ」を参照してください。CloudWatch メトリクス `RuleGroupLastEvaluationDuration` を使用して評価期間をモニタリングし、評価に時間がかかりすぎているルールグループを特定できます。
2. 欠落した評価のモニタリング – AMP は、評価がスキップされたタイミングを追跡するための `RuleGroupIterationsMissed` CloudWatch メトリクスを提供します。ListRules API には、各ルール/グループの評価時刻と最終評価時刻が表示されます。これにより、欠落した評価のパターンを特定できます。詳細については、「ListRules」を参照してください。

推奨事項: ルールを別々のグループに分割する

評価期間を短縮するために、ルールを別々のルールグループに分割します。グループ内のルールは順番に実行されますが、ルールグループは並行して実行できます。相互に依存する関連ルールを同じグループに保持します。一般的に、ルールグループを小さくすると、評価の一貫性が向上し、ギャップが少なくなります。

ルール評価のベストプラクティス

1. ルールグループサイズの最適化 – ルールグループを小さくしておき、一貫した評価を確保します。関連するルールをグループ化しますが、大きなルールグループは避けます。

2. 適切な評価間隔を設定する — タイムリーなアラートとシステム負荷のバランスをとります。モニタリング対象のメトリクスの安定性パターンを確認して、通常の変動範囲を理解します。
3. 遅延メトリクスにオフセット修飾子を使用する – 取り込みの遅延を補うオフセットを追加します。観測された取り込みパターンに基づいてオフセット期間を調整します。
4. 評価パフォーマンスのモニタリング – `RuleGroupIterationsMissed` メトリクスを追跡します。ListRules API で評価時間を確認します。
5. ルール式の検証 – ルール定義と手動クエリの間で式が完全に一致することを確認します。さまざまな時間範囲で式をテストして動作を理解します。
6. ルールのヘルスを定期的に確認する – ルール評価でエラーがないか確認します。定期的な問題がないか、公開ログをモニタリングします。

これらのトラブルシューティング手順とベストプラクティスに従うことにより、Amazon Managed Service for Prometheus のルール評価に関する一般的な問題を特定して解決できます。

ルーラーのトラブルシューティング

CloudWatch Logs で Amazon Managed Service for Prometheus イベントをモニタリングする を使用すると、アラートマネージャーとルーラー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。このセクションには、ルーラー関連のトラブルシューティングトピックが含まれています。

ログに次のルーラー失敗エラーが含まれている場合

```
{
    "workspaceId": "ws-12345c67-89c0-4d12-345b-f14db70f7a99",
    "message": {
        "log": "Evaluating rule failed, name=failure,
 group=canary_long_running_vl_namespace, namespace=canary_long_running_vl_namespace,
 err=found duplicate series for the match group {dimension1=\\\"1\\\"} on the right
 hand-side of the operation: [{__name__=\\\"fake_metric2\\\", dimension1=\\\"1\\
\", dimension2=\\\"b\\\"}, {__name__=\\\"fake_metric2\\\", dimension1=\\\"1\\\",
 dimension2=\\\"a\\\"}];many-to-many matching not allowed: matching labels must be
 unique on one side",
        "level": "ERROR",
        "name": "failure",
        "group": "canary_long_running_vl_namespace",
        "namespace": "canary_long_running_vl_namespace"
    },
    "component": "ruler"
```

```
}
```

これは、ルールの実行中に何らかのエラーが発生したことを示します。

実行するアクション

エラーメッセージを使用して、ルールの実行のトラブルシューティングを行います。

アラートマネージャーを使用して Amazon Managed Service for Prometheus でアラートを管理および転送する

Amazon Managed Service for Prometheus が実行するアラートルールが発動すると、送信されたアラートはアラートマネージャーによって処理されます。アラートの重複排除、グループ化、ダウンストリームのレシーバーへのルーティングを行います。Amazon Managed Service for Prometheus は、レシーバーとして Amazon Simple Notification Service のみをサポートし、同じアカウントの Amazon SNS トピックにメッセージをルーティングできます。アラートマネージャーを使用して、アラートを無音にしたり禁止したりすることもできます。

アラートマネージャーは、Prometheus の Alertmanager と同様の機能を提供します。

アラートマネージャーの設定ファイルを使用すると、次の機能を構成できます。

- グループ化 - 類似するアラートを 1 つの通知にまとめます。これは特に、多数のシステムに同時に障害が発生し、大量のアラートが同時に発生する可能性のある大規模なシステム停止時に役立ちます。例えば、ネットワーク障害により、多くのノードに同時に障害が発生したとします。これらのタイプのアラートがグループ化されていると、アラートマネージャーが送信する通知は 1 つになります。

  アラートのグループ化とグループ化された通知のタイミングは、アラートマネージャー設定ファイルのルーティングツリーによって構成されます。詳細については、「<route>」を参照してください。
- 禁止 - 他の特定のアラートが既に発動している場合に、特定のアラートの通知を抑制します。例えば、クラスターに到達できないというアラートが発動している場合に、そのクラスターに関する他のすべてのアラートをミュートするようにアラートマネージャーを構成できます。これにより、実際の問題とは関係のないアラートが大量に発生するのを防ぐことができます。禁止ルールの記述方法の詳細については、「<inhibit_rule>」を参照してください。
- サイレンス - メンテナンスの時間帯など、指定した時間だけアラートをミュートします。受信したアラートに対して、アクティブなサイレンスのすべての等価式または正規表現マッチャーと一致するかどうかのチェックが行われます。一致した場合、そのアラートに関する通知は送信されません。

  サイレンスを作成するには、`PutAlertManagerSilences` API を使用します。詳細については、「PutAlertManagerSilences」を参照してください。

Prometheus テンプレート

スタンドアロンの Prometheus では、個別のテンプレートファイルを使用したテンプレートをサポートしています。テンプレートでは、条件の使用、データのフォーマットなど、さまざまな処理を行うことができます。

Amazon Managed Service for Prometheus では、アラートマネージャー設定と同じアラートマネージャー設定ファイルにテンプレートを格納します。

トピック

- アラートマネージャーの操作に必要な IAM アクセス許可を理解する
- Amazon Managed Service for Prometheus でアラートを管理およびルーティングするためのアラートマネージャー設定を作成します。
- Amazon Managed Service for Prometheus でアラートマネージャーを使用してアラートレシーバーにアラートを転送する
- Amazon Managed Service for Prometheus にアラートマネージャー設定ファイルをアップロードする
- アラートを Amazon Managed Grafana またはオープンソースの Grafana と統合する
- CloudWatch Logs を使用してアラートマネージャーのトラブルシューティングを行う

アラートマネージャーの操作に必要な IAM アクセス許可を理解する

Amazon Managed Service for Prometheus でアラートマネージャーを使用するためのアクセス許可をユーザーに付与する必要があります。以下のアクセス許可を含む AWS Identity and Access Management (IAM) ポリシーを作成し、そのポリシーをユーザー、グループ、ロールに割り当てます。

Amazon Managed Service for Prometheus でアラートを管理およびルーティングするためのアラートマネージャー設定を作成します。

Amazon Managed Service for Prometheus でアラートマネージャーとテンプレートを使用するには、アラートマネージャーの設定 YAML ファイルを作成します。Amazon Managed Service for Prometheus アラートマネージャーファイルには、次の 2 つの主要なセクションがあります。

- `template_files:` には、レシーバーから送信されたメッセージに使用されるテンプレートが含まれます。詳細については、Prometheus ドキュメントの「Template Reference」と「Template Examples」を参照してください。
- `alertmanager_config:` には、アラートマネージャーの設定が含まれます。これには、スタンドアロンの Prometheus のアラートマネージャー設定ファイルと同じ構造が使用されます。詳細については、Alertmanager ドキュメントの「Configuration」を参照してください。

> **Note**
>
> Amazon Managed Service for Prometheus では、上記の Prometheus ドキュメントで説明されている `repeat_interval` 設定に追加の制限があります。許容される最大値は 5 日間です。5 日より大きい値に設定しても 5 日間として扱われ、5 日間の期間が経過すると通知が再送信されます。

> **Note**
>
> 設定ファイルは Amazon Managed Service for Prometheus コンソールで直接編集することもできますが、ここで指定する形式に従う必要があります。設定ファイルのアップロードまたは編集の詳細については、「Amazon Managed Service for Prometheus にアラートマネージャー設定ファイルをアップロードする」を参照してください。

Amazon Managed Service for Prometheus のアラートマネージャー設定ファイルでは、YAML ファイルのルートにある `alertmanager_config` キーの中に、アラートマネージャーの設定内容をすべて含める必要があります。

以下は、アラートマネージャー設定ファイルの基本的な例です。

```
alertmanager_config: |
  route:
    receiver: 'default'
  receivers:
    - name: 'default'
      sns_configs:
      - topic_arn: arn:aws:sns:us-east-2:123456789012:My-Topic
        sigv4:
          region: us-east-2
        attributes:
```

```
          key: key1
          value: value1
```

現在サポートされているレシーバーは、Amazon Simple Notification Service (Amazon SNS) だけです。設定に他の種類のレシーバーが指定されている場合、その設定は拒否されます。

以下は別のアラートマネージャー設定ファイルの例で、template_files ブロックと alertmanager_config ブロックの両方を使用しています。

```
template_files:
  default_template: |
    {{ define "sns.default.subject" }}[{{ .Status | toUpper }}{{ if eq .Status
 "firing" }}:{{ .Alerts.Firing | len }}{{ end }}]{{ end }}
    {{ define "__alertmanager" }}AlertManager{{ end }}
    {{ define "__alertmanagerURL" }}{{ .ExternalURL }}/#/alerts?receiver={{ .Receiver |
 urlquery }}{{ end }}
alertmanager_config: |
  global:
  templates:
    - 'default_template'
  route:
    receiver: default
  receivers:
    - name: 'default'
      sns_configs:
      - topic_arn: arn:aws:sns:us-east-2:accountid:My-Topic
        sigv4:
          region: us-east-2
        attributes:
          key: severity
          value: SEV2
```

デフォルトの Amazon SNS テンプレートブロック

デフォルトの Amazon SNS の設定では、明示的にオーバーライドしない限り、以下のテンプレートが使用されます。

```
{{ define "sns.default.message" }}{{ .CommonAnnotations.SortedPairs.Values | join "
 " }}
  {{ if gt (len .Alerts.Firing) 0 -}}
  Alerts Firing:
    {{ template "__text_alert_list" .Alerts.Firing }}
```

```
  {{- end }}
  {{ if gt (len .Alerts.Resolved) 0 -}}
  Alerts Resolved:
    {{ template "__text_alert_list" .Alerts.Resolved }}
  {{- end }}
{{- end }}
```

Amazon Managed Service for Prometheus でアラートマネージャーを使用してアラートレシーバーにアラートを転送する

アラートは、アラートルールによって生成されると、アラートマネージャーに送信されます。アラートマネージャーは、アラートの重複排除、メンテナンス中のアラートの禁止、必要に応じたグループ化などの機能を実行します。次に、アラートをメッセージとしてアラートレシーバーに転送します。オペレーターへの通知、自動応答、または他の方法でのアラートへの応答ができるアラートレシーバーを設定できます。

Amazon Simple Notification Service (Amazon SNS) と PagerDuty を Amazon Managed Service for Prometheus でアラートレシーバーとして設定できます。以下のトピックでは、アラートレシーバーを作成して設定する方法について説明します。

トピック

- Amazon SNS をアラートレシーバーとして使用する
- PagerDuty をアラートレシーバーとして使用する

Amazon SNS をアラートレシーバーとして使用する

既存の Amazon SNS トピックを Amazon Managed Service for Prometheus のアラートレシーバーとして使用することも、新しいトピックを作成することもできます。[標準] タイプのトピックを使用することをお勧めします。このタイプでは、トピックから E メール、SMS、HTTP にアラートを転送できます。

アラートマネージャーのレシーバーとして使用する新しい Amazon SNS トピックを作成するには、「ステップ 1: トピックを作成する」の手順に従います。トピックのタイプとして、必ず [標準] を選択してください。

その Amazon SNS トピックにメッセージが送信されるたびに E メールを受信するには、「ステップ 2: トピックに対するサブスクリプションを作成する」の手順に従います。

新規または既存の Amazon SNS トピックを使用するかどうかにかかわらず、以下のタスクを完了するには、Amazon SNS トピックの Amazon リソースネーム (ARN) が必要です。

トピック

- Amazon SNS トピックにアラートメッセージを送信するためのアクセス許可を Amazon Managed Service for Prometheus に付与する
- Amazon SNS トピックにメッセージを送信するようにアラートマネージャーを設定する
- Amazon SNS にメッセージを JSON として送信するようにアラートマネージャーを設定する
- アラートのメッセージを他の送信先に送信するように Amazon SNS を設定する
- Amazon SNS メッセージ検証ルールを理解する

Amazon SNS トピックにアラートメッセージを送信するためのアクセス許可を Amazon Managed Service for Prometheus に付与する

Amazon SNS トピックにメッセージを送信するには、Amazon Managed Service for Prometheus にアクセス許可を付与する必要があります。次のポリシーステートメントは、そのアクセス許可を付与します。これには、混乱した代理問題と呼ばれるセキュリティ問題の防止に役立つ `Condition` ステートメントが含まれています。`Condition` ステートメントは、Amazon SNS トピックへのアクセスを制限し、この特定のアカウントと Amazon Managed Service for Prometheus ワークスペースからのオペレーションのみを許可します。混乱した代理に関する問題の詳細については、「サービス間での不分別な代理処理の防止」を参照してください。

Amazon SNS トピックにメッセージを送信するためのアクセス許可を Amazon Managed Service for Prometheus に付与するには

1. Amazon SNS コンソールのhttps://console.aws.amazon.com/sns/v3/homeを開いてください。
2. ナビゲーションペインで、[トピック] を選択します。
3. Amazon Managed Service for Prometheus で使用しているトピックの名前を選択します。
4. [編集] を選択します。
5. [アクセスポリシー] を選択し、次のポリシーステートメントを既存のポリシーに追加します。

```
{
    "Sid": "Allow_Publish_Alarms",
    "Effect": "Allow",
    "Principal": {
        "Service": "aps.amazonaws.com"
```

```
    },
    "Action": [
        "sns:Publish",
        "sns:GetTopicAttributes"
    ],
    "Condition": {
        "ArnEquals": {
            "aws:SourceArn": "workspace_ARN"
        },
        "StringEquals": {
            "AWS:SourceAccount": "account_id"
        }
    },
    "Resource": "arn:aws:sns:region:account_id:topic_name"
}
```

[オプション] Amazon SNS トピックでサービス側の暗号化 (SSE) が有効になっている場合は、トピックの暗号化に使用されるキーの AWS KMS キーポリシーに `kms:GenerateDataKey*`および アクセス`kms:Decrypt`許可を追加して、Amazon Managed Service for Prometheus がこの暗号化されたトピックにメッセージを送信できるようにする必要があります。

例えば、以下をポリシーに追加できます。

```
{
  "Statement": [{
    "Effect": "Allow",
    "Principal": {
      "Service": "aps.amazonaws.com"
    },
    "Action": [
      "kms:GenerateDataKey*",
      "kms:Decrypt"
    ],
    "Resource": "*"
  }]
}
```

詳細については、「SNS トピックに対するAWS KMS アクセス許可」を参照してください。

6. [変更を保存] を選択します。

> **Note**
>
> デフォルトでは、Amazon SNS は `AWS:SourceOwner` に条件を設定したアクセスポリシーを作成します。詳細については、「SNS アクセスポリシー」を参照してください。

> **Note**
>
> IAM は、最も制限の厳しいポリシーを優先するルールに従います。SNS トピックに、ドキュメント化された Amazon SNS ポリシーブロックよりも制限の厳しいポリシーブロックがある場合、トピックポリシーのアクセス許可は付与されません。ポリシーを評価して何が許可されているかを確認するには、「ポリシーの評価論理」を参照してください。

オプトインリージョンの SNS トピック設定

`aps.amazonaws.com` を使用して、Amazon Managed Service for Prometheus ワークスペースと同じ で Amazon SNS トピックを設定できます。 AWS リージョン 非オプトインリージョン (us-east-1 など) の SNS トピックをオプトインリージョン (af-south-1 など) で使用するには、リージョンサービスプリンシパル形式を使用する必要があります。リージョンサービスの原則では、*`us-east-1`* を、使用する非オプトインリージョン (**`aps.`*`us-east-1`*`.amazonaws.com`**) に置き換えます。

次の表に、オプトインリージョンとそれに対応するリージョンサービスプリンシパルを示します。

オプトインリージョンとそのリージョンサービスプリンシパル

リージョン名	リージョン	リージョンサービスプリンシパル
アフリカ (ケープタウン)	af-south-1	af-south-1.aps.amazonaws.com
アジアパシフィック (香港)	ap-east-1	ap-east-1.aps.amazonaws.com
アジアパシフィック (タイ)	ap-southeast-7	ap-southeast-7.aps.amazonaws.com

リージョン名	リージョン	リージョンサービスプリンシパル
欧州 (ミラノ)	eu-south-1	eu-south-1.aps.amazonaws.com
欧州 (チューリッヒ)	eu-central-2	eu-central-2.aps.amazonaws.com
中東 (UAE)	me-central-1	me-central-1.aps.amazonaws.com
アジアパシフィック (マレーシア)	ap-southeast-5	ap-southeast-5.aps.amazonaws.com

オプトインリージョンを有効にする方法については、 Amazon Web Services 全般のリファレンスの「IAM ユーザーガイド」の「AWS リージョンの管理」を参照してください。

これらのオプトインリージョンに Amazon SNS トピックを設定するときは、正しいリージョンサービスプリンシパルを使用して、リージョン間のアラート配信を有効にしてください。

サービス間での不分別な代理処理の防止

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。これを防ぐために、 AWS には、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が用意されています。

リソースポリシーで `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、Amazon Managed Service for Prometheus が Amazon SNS 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。両方のグローバル条件コンテキストキーを同じポリシーステートメントで使用する場合は、`aws:SourceAccount` 値と、`aws:SourceArn` 値に含まれるアカウントが、同じアカウント ID を示している必要があります。

aws:SourceArn の値は、Amazon Managed Service for Prometheus ワークスペースの ARN でなければなりません。

混乱した代理問題から保護するための最も効果的な方法は、リソースの完全な ARN を指定して 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。リソースの完全な ARN が不明な場合や、複数のリソースを指定する場合は、aws:SourceArn グローバルコンテキスト条件キーを使用して、ARN の未知部分をワイルドカード (*) で表します。例えば、arn:aws:*servicename*::*123456789012*:* です。

「Amazon SNS トピックにアラートメッセージを送信するためのアクセス許可を Amazon Managed Service for Prometheus に付与する」に記載されているポリシーは、Amazon Managed Service for Prometheus 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、混乱した代理問題を防止する方法を示しています。

Amazon SNS トピックにメッセージを送信するようにアラートマネージャーを設定する

標準タイプの (新規または既存の) Amazon SNS トピックを作成したら、これをアラートレシーバーとしてアラートマネージャー設定に追加できます。アラートマネージャーは、設定したアラートレシーバーにアラートを転送できます。そのためには、Amazon SNS トピックの Amazon リソースネーム (ARN) を知っている必要があります。

Amazon SNS のレシーバー設定の詳細については、Prometheus の構成に関するドキュメントの「<sns_configs>」を参照してください。

サポートされないプロパティ

Amazon Managed Service for Prometheus では、アラートのレシーバーとして Amazon SNS がサポートされています。ただし、サービスの制約により、Amazon SNS レシーバーのすべてのプロパティがサポートされるわけではありません。以下のプロパティは、Amazon Managed Service for Prometheus のアラートマネージャー設定ファイルでは使用できません。

- api_url: - api_url は Amazon Managed Service for Prometheus によって自動的に設定されるため、このプロパティは使用できません。
- Http_config - このプロパティは、外部プロキシを設定できるようにするものです。Amazon Managed Service for Prometheus では、この機能は現在サポートされていません。

また、SigV4 設定にはリージョンプロパティを含める必要があります。リージョンプロパティを指定しない場合、認可リクエストを行うための十分な情報が Amazon Managed Service for Prometheus に提供されません。

Amazon SNS トピックをレシーバーとしてアラートマネージャーを構成するには

1. 既存のアラートマネージャー設定ファイルを使用している場合は、テキストエディタでそのファイルを開きます。
2. `receivers` ブロックに Amazon SNS 以外の現在のレシーバーがある場合は、それらを削除します。複数の Amazon SNS トピックをレシーバーとして構成するには、各トピックを `receivers` ブロック内の個別の `sns_config` ブロックに追加します。
3. `receivers` セクションに次の YAML ブロックを追加します。

```
- name: name_of_receiver
  sns_configs:
    - sigv4:
        region: AWS #####
      topic_arn: ARN_of_SNS_topic
      subject: yoursubject
      attributes:
        key: yourkey
        value: yourvalue
```

`subject` を指定しない場合、デフォルトでは、ラベル名と値を使用するデフォルトテンプレートから件名が生成されますが、この値は SNS には長すぎる可能性があります。件名に適用されるテンプレートを変更するには、このガイドの「Amazon SNS にメッセージを JSON として送信するようにアラートマネージャーを設定する」を参照してください。

この後は、アラートマネージャー設定ファイルを Amazon Managed Service for Prometheus にアップロードする必要があります。詳細については、「Amazon Managed Service for Prometheus にアラートマネージャー設定ファイルをアップロードする」を参照してください。

Amazon SNS にメッセージを JSON として送信するようにアラートマネージャーを設定する

Amazon Managed Service for Prometheus アラートマネージャーは、デフォルトでは、メッセージをプレーンテキストリスト形式で出力します。これは、他のサービスで解析しにくい場合があります。代わりに、アラートを JSON 形式で送信するようにアラートマネージャーを設定できま

す。JSON を使用すると、 AWS Lambda またはウェブフック受信エンドポイントで Amazon SNS のダウンストリームのメッセージを簡単に処理できます。デフォルトのテンプレートを使用する代わりに、メッセージの内容を JSON で出力するカスタムテンプレートを定義すると、ダウンストリーム関数での解析が容易になります。

アラートマネージャーから Amazon SNS に JSON 形式でメッセージを出力するには、アラートマネージャーの設定を更新して、`template_files` ルートセクション内に次のコードを含めます。

```
default_template: |
   {{ define "sns.default.message" }}{{ "{" }}"receiver": "{{ .Receiver }}","status":
 "{{ .Status }}","alerts": [{{ range $alertIndex, $alerts := .Alerts }}{{ if
 $alertIndex }}, {{ end }}{{ "{" }}"status": "{{ $alerts.Status }}"{{ if
 gt (len $alerts.Labels.SortedPairs) 0 -}},"labels": {{ "{" }}{{ range
 $index, $label := $alerts.Labels.SortedPairs }}{{ if $index }},
 {{ end }}"{{ $label.Name }}": "{{ $label.Value }}"{{ end }}
{{ "}" }}{{- end }}{{ if gt (len $alerts.Annotations.SortedPairs )
 0 -}},"annotations": {{ "{" }}{{ range $index, $annotations :=
 $alerts.Annotations.SortedPairs }}{{ if $index }}, {{ end }}"{{ $annotations.Name }}":
 "{{ $annotations.Value }}"{{ end }}{{ "}" }}{{- end }},"startsAt":
 "{{ $alerts.StartsAt }}","endsAt": "{{ $alerts.EndsAt }}","generatorURL":
 "{{ $alerts.GeneratorURL }}","fingerprint": "{{ $alerts.Fingerprint }}"{{ "}" }}
{{ end }}]{{ if gt (len .GroupLabels) 0 -}},"groupLabels": {{ "{" }}{{ range
 $index, $groupLabels := .GroupLabels.SortedPairs }}{{ if $index }},
 {{ end }}"{{ $groupLabels.Name }}": "{{ $groupLabels.Value }}"{{ end }}
{{ "}" }}{{- end }}{{ if gt (len .CommonLabels) 0 -}},"commonLabels": {{ "{" }}
{{ range $index, $commonLabels := .CommonLabels.SortedPairs }}{{ if $index }},
 {{ end }}"{{ $commonLabels.Name }}": "{{ $commonLabels.Value }}"{{ end }}{{ "}" }}{{-
 end }}{{ if gt (len .CommonAnnotations) 0 -}},"commonAnnotations": {{ "{" }}{{ range
 $index, $commonAnnotations := .CommonAnnotations.SortedPairs }}{{ if $index }},
 {{ end }}"{{ $commonAnnotations.Name }}": "{{ $commonAnnotations.Value }}"{{ end }}
{{ "}" }}{{- end }}{{ "}" }}{{ end }}
   {{ define "sns.default.subject" }}[{{ .Status | toUpper }}{{ if eq .Status
 "firing" }}:{{ .Alerts.Firing | len }}{{ end }}]{{ end }}
```

> **Note**
>
> このテンプレートは、英数字データから JSON を作成します。データに特殊文字が含まれている場合は、このテンプレートを使用する前にそれらをエンコードしてください。

このテンプレートが送信通知で使用されるようにするには、次のように alertmanager_config ブロックでテンプレートを参照します。

```
alertmanager_config: |
  global:
  templates:
    - 'default_template'
```

> **Note**
>
> このテンプレートは、メッセージ本文全体を JSON として出力するものです。このテンプレートにより、メッセージ本文全体が上書きされます。この特定のテンプレートを使用する場合、メッセージ本文をオーバーライドすることはできません。手動で行ったオーバーライドは、テンプレートよりも優先されます。

詳細については、以下を参照してください。

- アラートマネージャー設定ファイルについては、「Amazon Managed Service for Prometheus でアラートを管理およびルーティングするためのアラートマネージャー設定を作成します。」を参照してください。
- 設定ファイルのアップロードについては、「Amazon Managed Service for Prometheus にアラートマネージャー設定ファイルをアップロードする」を参照してください。

アラートのメッセージを他の送信先に送信するように Amazon SNS を設定する

Amazon Managed Service for Prometheus は、Amazon Simple Notification Service (Amazon SNS) にのみアラートメッセージを送信できます。E メール、ウェブフック、Slack、OpsGenie などの他の送信先にメッセージを送信するには、これらのエンドポイントにメッセージを転送するように Amazon SNS を設定する必要があります。

以下のセクションでは、他の送信先にアラートを転送するように Amazon SNS を設定する方法について説明します。

トピック

- E メール
- ウェブフック

- Slack
- OpsGenie

E メール

E メールにメッセージを出力するように Amazon SNS トピックを構成するには、サブスクリプションを作成します。Amazon SNS コンソールで、[サブスクリプション] タブを選択して、[サブスクリプション] リストページを開きます。[サブスクリプションの作成] を選択し、[E メール] を選択します。Amazon SNS から、指定した E メールアドレスに確認メールが送信されます。確認を受け入れると、サブスクライブしたトピックから Amazon SNS 通知を E メールとして受信できます。詳細については、「Amazon SNS トピックへサブスクライブする」を参照してください。

ウェブフック

ウェブフックエンドポイントにメッセージを出力するように Amazon SNS トピックを構成するには、サブスクリプションを作成します。Amazon SNS コンソールで、[サブスクリプション] タブを選択して、[サブスクリプション] リストページを開きます。[サブスクリプションの作成] を選択し、[HTTP/HTTPS] を選択します。サブスクリプションを作成したら、確認手順に従ってサブスクリプションをアクティブにする必要があります。アクティブになると、HTTP エンドポイントで Amazon SNS 通知が受信されます。詳細については、「Amazon SNS トピックへサブスクライブする」を参照してください。Slack ウェブフックを使用してさまざまな宛先にメッセージを発行する方法の詳細については、「ウェブフックを使用して Amazon SNS メッセージを Amazon Chime、Slack、または Microsoft Teams に発行するにはどうすればよいですか?」を参照してください。

Slack

Slack にメッセージを出力するように Amazon SNS トピックを構成するには、2 つの方法があります。Slack による E メールからチャンネルへの統合を使用すると、Slack で E メールメッセージを受け取って Slack チャンネルに転送できます。または、Lambda 関数を使用して、Amazon SNS 通知を Slack に書き換えることができます。E メールを slack チャネルに転送する方法の詳細については、「Slack Webhook AWS の SNS トピックサブスクリプションの確認」を参照してください。Lambda 関数を作成して Amazon SNS メッセージを Slack に変換する方法の詳細については、「How to integrate Amazon Managed Service for Prometheus with Slack」を参照してください。

OpsGenie

OpsGenie にメッセージを出力するように Amazon SNS トピックを構成する方法については、「Integrate Opsgenie with Incoming Amazon SNS」を参照してください。

Amazon SNS メッセージ検証ルールを理解する

Amazon Simple Notification Service (Amazon SNS) では、メッセージが特定の基準を満たしている必要があります。これらの標準を満たしていないメッセージは、受信時に変更されます。アラートメッセージは、以下のルールに基づいて、Amazon SNS レシーバーにより必要に応じて検証、切り捨て、または変更されます。

- メッセージに UTF 以外の文字が含まれている場合。
  - メッセージは「Error - not a valid UTF-8 encoded string」に置き換えられます。
  - キーが [truncated] で値が [true] のメッセージ属性が 1 つ追加されます。
  - キーが [modified] で値が [Message: Error - not a valid UTF-8 encoded string] のメッセージ属性が 1 つ追加されます。
- メッセージが空の場合。
  - メッセージは [Error - Message should not be empty] に置き換えられます。
  - キーが [modified] で値が [Message: Error - Message should not be empty] のメッセージ属性が 1 つ追加されます。
- メッセージが切り捨てられた場合。
  - メッセージは切り捨てられたコンテンツになります。
  - キーが [truncated] で値が [true] のメッセージ属性が 1 つ追加されます。
  - キーが [modified] で値が [Message: Error - Message has been truncated from ***X*** KB, because it exceeds the 256 KB size limit] のメッセージ属性が 1 つ追加されます。
- 件名にコントロール文字または非 ASCII 文字が含まれています。
  - サブジェクトにコントロール文字または非 ASCII 文字が含まれている場合、SNS はサブジェクトを [Error - contains control- or non-ASCII characters] に置き換えます。
  - SNS E メールの件名の場合は、改行 (\n) などのコントロール文字を削除します。
- 件名が ASCII でない場合。
  - 件名は [Error - contains non printable ASCII characters] に置き換えられます。
  - キーが [modified] で値が [Subject: Error - contains non-printable ASCII characters] のメッセージ属性が 1 つ追加されます。
- 件名が切り捨てられた場合。
  - 件名は切り捨てられたコンテンツになります。
  - キーが [modified] で値が [Subject: Error - Subject has been truncated from ***X*** characters, because it exceeds the 100 character size limit] のメッセージ属性が 1 つ追加されます。

- メッセージ属性のキー/値が無効な場合。
  - 無効なメッセージ属性は削除されます。
  - キーが [modified] で値が [MessageAttribute: Error - ***X*** of the message attributes have been removed because of invalid MessageAttributeKey or MessageAttributeValue] のメッセージ属性が 1 つ追加されます。
- メッセージ属性が切り捨てられた場合。
  - 余分なメッセージ属性は削除されます。
  - キーが [modified] で値が [MessageAttribute: Error - ***X*** of the message attributes have been removed, because it exceeds the 256KB size limit] のメッセージ属性が 1 つ追加されます。

PagerDuty をアラートレシーバーとして使用する

PagerDuty に直接アラートを送信するように Amazon Managed Service for Prometheus を設定できます。この統合では、PagerDuty 統合キーを に保存 AWS Secrets Manager し、Amazon Managed Service for Prometheus にシークレットを読み取るアクセス許可を付与する必要があります。

PagerDuty の統合により、自動化されたインシデント対応ワークフローが可能になり、重要なアラートが適切なチームメンバーに適切なタイミングで届くようになります。PagerDuty をアラートレシーバーとして使用すると、PagerDuty のエスカレーションポリシー、オンコールスケジューリング、インシデント管理機能を活用して、アラートを迅速に確認して解決できます。この統合は、サービスの可用性を維持し、SLA 要件を満たすためにシステムの問題への迅速な対応が不可欠な本番環境に特に役立ちます。詳細については、PagerDuty ウェブサイトの「PagerDuty ナレッジベース」を参照してください。

PagerDuty 設定オプション

オプション	説明	必須
`routing_key`	サービス上の統合用 PagerDuty ルーティングキー。これを Secrets Manager ARN として指定する必要があります	はい
`service_key`	サービス上の統合用 PagerDuty サービスキー。こ	はい (イベント API v1 の場合)

オプション	説明	必須
	れを Secrets Manager ARN として指定する必要があります	
client	通知機能のクライアント ID	いいえ
client_url	通知の送信者へのバックリンク	いいえ
description	インテントの説明	いいえ
details	インシデントに関する詳細を提供する任意のキーと値のペアのセット	いいえ
severity	インシデントの重要度	いいえ
class	イベントのクラスまたはタイプ	いいえ
component	イベントを担当するソースマシンのコンポーネント	いいえ
group	コンポーネントの論理グループ	いいえ
source	影響を受けるシステムの一意の場所	いいえ

> **Note**
>
> url、service_key_file、routing_key_file、http_config オプションはサポートされていません。

以下のトピックでは、Amazon Managed Service for Prometheus で PagerDuty をアラートレシーバーとして設定する方法について説明します。

トピック

- AWS Secrets Manager および アクセス許可を設定する
- PagerDuty にアラートを送信するようにアラートマネージャーを設定する

AWS Secrets Manager および アクセス許可を設定する

PagerDuty にアラートを送信する前に、PagerDuty 統合キーを安全に保存し、必要なアクセス許可を設定する必要があります。このプロセスでは AWS Secrets Manager、 でシークレットを作成し、カスタマーマネージド AWS Key Management Service (AWS KMS) キーを使用して暗号化し、シークレットとその暗号化キーの両方にアクセスするために必要なアクセス許可を Amazon Managed Service for Prometheus に付与します。次の手順では、この設定プロセスの各ステップについて説明します。

Secrets Manager for PagerDuty でシークレットを作成するには

PagerDuty をアラートレシーバーとして使用するには、PagerDuty 統合キーを Secrets Manager に保存する必要があります。以下の手順に従ってください。

1. Secrets Manager コンソールを開きます。
2. 新しいシークレットを保存 を選択します。
3. [Secret type] (シークレットタイプ) で、[Other type of secret] (他の種類のシークレット) を選択します。
4. [キーと値のペア] には、PagerDuty 統合キーをシークレット値として入力します。これは、PagerDuty 統合のルーティングキーまたはサービスキーです。
5. [次へ] を選択します。
6. シークレットの名前と説明を入力して [次へ] を選択します。
7. 必要に応じてローテーション設定を行い、[次へ] を選択します。
8. 設定を確認し、[保存] を選択します。
9. シークレットを作成したら、その ARN を書き留めます。アラートマネージャーを設定するときに必要になります。

シークレットをカスタマーマネージド AWS KMS キーで暗号化するには

シークレットとその暗号化キーにアクセスするためのアクセス許可を Amazon Managed Service for Prometheus に付与する必要があります。

1. シークレットリソースポリシー: Secrets Manager コンソールでシークレットを開きます。

   a. [リソースのアクセス許可] を選択します。

   b. [アクセス許可の編集] を選択します。

   c. 次のポリシーステートメントを追加します。ステートメントで、######## を特定の値に置き換えます。

```
{
  "Effect": "Allow",
  "Principal": {
    "Service": "aps.amazonaws.com"
  },
  "Action": "secretsmanager:GetSecretValue",
  "Resource": "*",
  "Condition": {
    "ArnEquals": {
      "aws:SourceArn": "arn:aws:aps:aws-
region:123456789012:workspace/WORKSPACE_ID"
    },
    "StringEquals": {
      "aws:SourceAccount": "123456789012"
    }
  }
}
```

   d. [保存] を選択します。

2. KMS キーポリシー: AWS KMS コンソールで AWS KMS キーを開きます。

   a. [キーポリシー] を選択します。

   b. [編集] を選択します。

   c. 次のポリシーステートメントを追加します。ステートメントで、######## を特定の値に置き換えます。

```
{
  "Effect": "Allow",
  "Principal": {
    "Service": "aps.amazonaws.com"
  },
  "Action": "kms:Decrypt",
  "Resource": "*",
  "Condition": {
```

```
    "ArnEquals": {
      "aws:SourceArn": "arn:aws:aps:aws-
region:123456789012:workspace/WORKSPACE_ID"
    },
    "StringEquals": {
      "aws:SourceAccount": "123456789012"
    }
  }
}
```

d. [保存] を選択します。

次のステップ – 次のトピック、PagerDuty にアラートを送信するようにアラートマネージャーを設定する に進みます。

PagerDuty にアラートを送信するようにアラートマネージャーを設定する

PagerDuty にアラートを送信するようにアラートマネージャーを設定するには、アラートマネージャーの定義を更新する必要があります。これは、 AWS マネジメントコンソール、 AWS CLI、または AWS SDKsを使用して実行できます。

Exampleアラートマネージャーの設定

以下は、PagerDuty にアラートを送信するアラートマネージャー設定の例です。この例では、*########*を特定の値に置き換えます。

```
alertmanager_config: |
  route:
    receiver: 'pagerduty-receiver'
    group_by: ['alertname']
    group_wait: 30s
    group_interval: 5m
    repeat_interval: 1h
  receivers:
    - name: 'pagerduty-receiver'
      pagerduty_configs:
      - routing_key:
          aws_secrets_manager:
            secret_arn: 'arn:aws:secretsmanager:aws-
region:123456789012:secret:YOUR_SECRET_NAME'
            secret_key: 'YOUR_SECRET_KEY'
            refresh_interval: 5m
```

```
        description: '{{ .CommonLabels.alertname }}'
        severity: 'critical'
        details:
          firing: '{{ .Alerts.Firing | len }}'
          status: '{{ .Status }}'
          instance: '{{ .CommonLabels.instance }}'
```

Example AWS CLI

以下は、アラートマネージャーの定義を更新するために使用される AWS CLI コマンドです。この例では、*########*を特定の値に置き換えます。

```
aws amp put-alert-manager-definition \
  --workspace-id WORKSPACE_ID \
  --data file://alertmanager-config.yaml
```

PagerDuty 統合のトラブルシューティング

アラートが PagerDuty に送信されていない場合は、次の項目を確認してください。

- シークレットが存在し、正しい PagerDuty 統合キーが含まれていることを確認します。
- シークレットがカスタマー管理の KMS キーで暗号化されていることを確認します。
- シークレットと KMS キーの両方のリソースポリシーが、Amazon Managed Service for Prometheus に必要なアクセス許可を付与していることを確認します。
- アラートマネージャー設定の ARN がシークレットを正しく参照していることを確認します。
- PagerDuty アカウントで PagerDuty 統合キーが有効でアクティブであることを確認します。

Amazon Managed Service for Prometheus は、トラブルシューティングに役立つ Amazon CloudWatch Logs と以下の CloudWatch メトリクスをサポートします。詳細については、「CloudWatch Logs で Amazon Managed Service for Prometheus イベントをモニタリングする」および「CloudWatch メトリクスを使用して Amazon Managed Service for Prometheus のリソースモニタリングする」を参照してください。

- `SecretFetchFailure`
- `AlertManagerNotificationsThrottledByIntegration`
- `AlertManagerNotificationsFailedByIntegration`

Amazon Managed Service for Prometheus にアラートマネージャー設定ファイルをアップロードする

アラートマネージャー設定ファイルに必要な事項を確認したら、コンソール内で作成して編集できます。または、Amazon Managed Service for Prometheus コンソールまたは AWS CLIを使用して既存のファイルをアップロードできます。

> **Note**
>
> Amazon EKS クラスターを実行している場合は、AWS Controllers for Kubernetes を使用してアラートマネージャー設定ファイルをアップロードすることもできます。

Amazon Managed Service for Prometheus コンソールを使用してアラートマネージャー設定を編集または置換するには

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ページの左上隅にあるメニューアイコンを選択し、[すべての WorkSpaces] を選択します。
3. ワークスペースのワークスペース ID を選択し、[アラートマネージャー] タブを選択します。
4. ワークスペースにまだアラートマネージャーの定義がない場合は、[定義を追加] を選択します。

   > **Note**
   >
   > 置換するアラートマネージャー定義がワークスペースに既にある場合は、代わりに [変更] を選択します。

5. [ファイルを選択] を選択し、アラートマネージャー定義ファイルを選択して、[続行] を選択します。

   > **Note**
   >
   > または、[定義を作成] オプションを選択して、新しいファイルを作成してコンソールで直接編集することもできます。これにより、サンプルのデフォルト設定が作成されます。この設定を編集してからアップロードします。

を使用してアラートマネージャー設定をワークスペースに初めて AWS CLI アップロードするには

1. アラートマネージャーファイルの内容を base64 でエンコードします。Linux では、次のコマンドを使用できます。

   ```
   base64 input-file output-file
   ```

   macOS では、次のコマンドを使用できます。

   ```
   openssl base64 input-file output-file
   ```

2. 以下のいずれかのコマンドを入力して、ファイルをアップロードします。

   AWS CLI バージョン 2 で、次のように入力します。

   ```
   aws amp create-alert-manager-definition --data file://path_to_base_64_output_file
    --workspace-id my-workspace-id --region region
   ```

   AWS CLI バージョン 1 で、次のように入力します。

   ```
   aws amp create-alert-manager-definition --data fileb://path_to_base_64_output_file
    --workspace-id my-workspace-id --region region
   ```

3. アラートマネージャーの設定が有効になるまで数秒かかります。ステータスを確認するには、次のコマンドを入力します。

   ```
   aws amp describe-alert-manager-definition --workspace-id workspace_id --
   region region
   ```

   `status` が `ACTIVE` であれば、新しいアラートマネージャーの定義が有効になっています。

を使用してワークスペースのアラートマネージャー設定を新しい設定 AWS CLI に置き換えるには

1. アラートマネージャーファイルの内容を base64 でエンコードします。Linux では、次のコマンドを使用できます。

   ```
   base64 input-file output-file
   ```

   macOS では、次のコマンドを使用できます。

```
openssl base64 input-file output-file
```

2. 以下のいずれかのコマンドを入力して、ファイルをアップロードします。

   AWS CLI バージョン 2 で、次のように入力します。

   ```
   aws amp put-alert-manager-definition --data file://path_to_base_64_output_file --
   workspace-id my-workspace-id --region region
   ```

   AWS CLI バージョン 1 で、次のように入力します。

   ```
   aws amp put-alert-manager-definition --data file://path_to_base_64_output_file --
   workspace-id my-workspace-id --region region
   ```

3. 新しいアラートマネージャーの設定が有効になるまで数秒かかります。ステータスを確認するには、次のコマンドを入力します。

   ```
   aws amp describe-alert-manager-definition --workspace-id workspace_id --
   region region
   ```

   `status` が `ACTIVE` であれば、新しいアラートマネージャーの定義が有効になっています。それまでは、以前のアラートマネージャーの設定が有効なままになります。

アラートを Amazon Managed Grafana またはオープンソースの Grafana と統合する

Amazon Managed Service for Prometheus 内の Alertmanager で作成したアラートルールは、Amazon Managed Grafana や Grafana に転送して表示することができます。これにより、アラートルールとアラートを単一の環境に統合できます。Amazon Managed Grafana 内で、アラートルールと生成されたアラートを表示できます。

前提条件

Amazon Managed Service for Prometheus を Amazon Managed Grafana に統合する前に、以下の前提条件が満たされている必要があります。

- Amazon Managed Service for Prometheus AWS アカウント および IAM ロールをプログラムで作成するには、既存の および IAM 認証情報が必要です。

  AWS アカウント および IAM 認証情報の作成の詳細については、「」を参照してくださいセットアップ AWS。

- Amazon Managed Service for Prometheus ワークスペースがあり、そこにデータが取り込まれている必要があります。新しいワークスペースをセットアップするには、「Amazon Managed Service for Prometheus ワークスペースの作成」を参照してください。また、Alertmanager やルーラーなどの Prometheus の概念を理解しておく必要もあります。これらのトピックの詳細については、Prometheus のドキュメントを参照してください。

- Amazon Managed Service for Prometheus で、Alertmanager の設定とルールファイルが既に構成されている必要があります。Amazon Managed Service for Prometheus での Alertmanager の詳細については、「アラートマネージャーを使用して Amazon Managed Service for Prometheus でアラートを管理および転送する」を参照してください。ルールの詳細については、「ルールを使用して、受信したメトリクスを変更またはモニタリングする」を参照してください。

- Amazon Managed Grafana がセットアップされているか、オープンソースバージョンの Grafana が実行されている必要があります。

  - Amazon Managed Grafana を使用する場合は、Grafana アラートを使用している必要があります。詳細については、「Migrating legacy dashboard alerts to Grafana alerting」を参照してください。

  - オープンソースバージョンの Grafana を使用する場合は、バージョン 9.1 以降を実行している必要があります。

    > **Note**
    >
    > 以前のバージョンの Grafana を使用することもできますが、統合アラート (Grafana アラート) 機能を有効にする必要があります。また、Grafana から Amazon Managed Service for Prometheus を呼び出すには sigv4 プロキシのセットアップが必要になる場合があります。詳細については、「Amazon Managed Service for Prometheus で使用する Grafana オープンソースまたは Grafana Enterprise のセットアップ」を参照してください。

- Amazon Managed Grafana には、Prometheus リソースに対する次のアクセス許可が必要です。これらのアクセス許可は、https://docs.aws.amazon.com/grafana/latest/userguide/AMG-manage-permissions.html で説明されているサービス管理ポリシーとカスタマー管理ポリシーのいずれかに追加する必要があります。

- aps:ListRules
- aps:ListAlertManagerSilences
- aps:ListAlertManagerAlerts
- aps:GetAlertManagerStatus
- aps:ListAlertManagerAlertGroups
- aps:PutAlertManagerSilences
- aps:DeleteAlertManagerSilence

Amazon Managed Grafana のセットアップ

既に Amazon Managed Service for Prometheus インスタンスでルールとアラートが設定されている場合、Amazon Managed Grafana をそれらのアラートのダッシュボードとして使用するための構成は、すべて Amazon Managed Grafana 内で完結します。

Amazon Managed Grafana をアラートのダッシュボードとして構成するには

1. ワークスペースの Grafana コンソールを開きます。
2. [設定] で、[データソース] を選択します。
3. Prometheus データソースを作成するか開きます。まだ Prometheus データソースを設定していない場合、詳細については「ステップ 2: Grafana で Prometheus データソースを追加する」を参照してください。
4. Prometheus データソースで、[Alertmanager UI を使用してアラートを管理] を選択します。
5. [データソース] インターフェイスに戻ります。
6. 新しい Alertmanager データソースを作成します。
7. Alertmanager データソースの設定ページで、次の設定を追加します。
   - [実装] を Prometheus に設定します。
   - [URL] 設定には、Prometheus ワークスペースの URL を使用し、ワークスペース ID 以降の文字をすべて削除して、末尾に /alertmanager を追加します。次の例では、*variables* を自分の (アカウント固有の) 情報に置き換えます。

     ```
     https://aps-workspaces.US East (N. Virginia).amazonaws.com/workspaces/ws-example-1234-5678-abcd-xyz00000001/alertmanager.
     ```

- [認証] で、[SigV4Auth] をオンにします。これにより、リクエストに AWS 認証を使用するように Grafana に指示します。
- [SigV4Auth の詳細] で、[デフォルトのリージョン] に Prometheus インスタンスのリージョンを指定します。例えば、`us-east-1` を指定します。
- [デフォルト] オプションを `true` に設定します。

8. [保存してテスト] を選択します。
9. これで、Amazon Managed Service for Prometheus のアラートが Grafana インスタンスと連携するように構成されました。Amazon Managed Service for Prometheus インスタンスのアラートルール、アラートグループ (アクティブなアラートを含む)、サイレンスが、Grafana の [アラート] ページに表示されることを確認します。

CloudWatch Logs を使用してアラートマネージャーのトラブルシューティングを行う

CloudWatch Logs で Amazon Managed Service for Prometheus イベントをモニタリングする を使用すると、アラートマネージャーとルーラー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。このセクションには、アラートマネージャー関連のトラブルシューティングトピックが含まれています。

トピック

- アクティブなアラートの警告
- アラート集約グループサイズの警告
- アラートサイズが大きすぎる警告
- 空のコンテンツに関する警告
- 無効な key/value に関する警告
- メッセージの制限に関する警告
- リソースベースのポリシーがないことによるエラー
- 非 ASCII 文字に関する警告
- KMS を呼び出す権限がありません
- テンプレートエラー

アクティブなアラートの警告

ログに次の警告が含まれている場合

```
{
    "workspaceId": "ws-efdc5b42-b051-11ec-b123-4567ac120002",
    "message": {
        "log": "too many alerts, limit: 1000",
        "level": "WARN"
    },
    "component": "alertmanager"
}
```

つまり、アラートマネージャーの [アクティブなアラート] のクォータを超えています。

実行するアクション

クォータの引き上げをリクエストする。にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/servicequotas/ で Service Quotas コンソールを開きます。

アラート集約グループサイズの警告

ログに次の警告が含まれている場合

```
{
    "workspaceId": "ws-efdc5b42-b051-11ec-b123-4567ac120002",
    "message": {
        "log": "Too many aggregation groups, cannot create new group for alert,
 groups=1000, limit=1000, alert=sample-alert",
        "level": "WARN"
    },
    "component": "alertmanager"
}
```

つまり、アラートマネージャーのアラート集約グループのサイズクォータを超えました。

実行するアクション

group_by パラメータを使用して、アラート集約グループのサイズを小さくします。詳細については、Prometheus ドキュメントの「ルート関連設定」を参照してください。

また、クォータの引き上げをリクエストすることも可能です。にサインイン AWS マネジメントコンソール し、https://console.aws.amazon.com/servicequotas/ で Service Quotas コンソールを開きます。

アラートサイズが大きすぎる警告

ログに次の警告が含まれている場合

```
{
    "workspaceId": "ws-efdc5b42-b051-11ec-b123-4567ac120002",
    "message": {
        "log": "alerts too big, total size limit: 20000000 bytes",
        "level": "WARN"
    },
    "component": "alertmanager"
}
```

つまり、ワークスペースあたりのアラートマネージャーアラートのサイズクォータを超えました。

実行するアクション

不要な注釈とラベルを削除して、アラートサイズを減らします。

空のコンテンツに関する警告

ログに次の警告が含まれている場合

```
{
   "workspaceId": "ws-abcd1234-ef56-78ab-cd90-1234abcd0000",
   "message": {
       "log": "Message has been modified because the content was empty."
       "level": "WARN"
   },
   "component": "alertmanager"
}
```

これは、アラートマネージャーテンプレートにより、送信アラートが空のメッセージに解決されたことを意味します。

実行するアクション

アラートマネージャーのテンプレートを検証し、すべてのレシーバーのパスに有効なテンプレートがあることを確認します。

無効な key/value に関する警告

ログに次の警告が含まれている場合

```
{
   "workspaceId": "ws-abcd1234-ef56-78ab-cd90-1234abcd0000",
   "message": {
       "log": "MessageAttributes has been removed because of invalid key/value,
 numberOfRemovedAttributes=1"
       "level": "WARN"
   },
   "component": "alertmanager"
}
```

これは、キーや値が無効なため、メッセージ属性の一部が削除されたことを意味します。

実行するアクション

メッセージ属性の設定に使用しているテンプレートを再評価し、有効な SNS メッセージ属性に解決されることを確認します。Amazon SNS トピックに送信するメッセージの検証の詳細については、「SNS トピックの検証」を参照してください。

メッセージの制限に関する警告

ログに次の警告が含まれている場合

```
{
   "workspaceId": "ws-abcd1234-ef56-78ab-cd90-1234abcd0000",
   "message": {
       "log": "Message has been truncated because it exceeds size limit,
 originSize=266K, truncatedSize=12K"
       "level": "WARN"
   },
   "component": "alertmanager"
}
```

これは、メッセージサイズの一部が大きすぎることを意味します。

実行するアクション

アラートレシーバーのメッセージテンプレートを確認し、サイズ制限に収まるように変更します。

リソースベースのポリシーがないことによるエラー

ログに次のエラーが含まれている場合

```
{
   "workspaceId": "ws-abcd1234-ef56-78ab-cd90-1234abcd0000",
   "message": {
       "log": "Notify for alerts failed, AMP is not authorized to perform: SNS:Publish
 on resource: arn:aws:sns:us-west-2:12345:testSnsReceiver because no resource-based
 policy allows the SNS:Publish action"
       "level": "ERROR"
   },
   "component": "alertmanager"
}
```

これは、指定された SNS トピックにアラートを送信するためのアクセス許可が Amazon Managed Service for Prometheus にないことを意味します。

実行するアクション

トピックに SNS メッセージを送信する許可が、Amazon SNS トピックのアクセスポリシーによって Amazon Managed Service for Prometheus に与えられていることを確認します。Amazon SNS トピックへのアクセスをサービス `aps.amazonaws.com` (Amazon Managed Service for Prometheus) に許可する SNS アクセスポリシーを作成します。SNS アクセスポリシーの詳細については、「Amazon Simple Notification Service デベロッパーガイド」の「アクセスポリシー言語の使用」と「Amazon SNS アクセスコントロールのケース例」を参照してください。

非 ASCII 文字に関する警告

ログに次の警告が含まれている場合

```
{
   "workspaceId": "ws-abcd1234-ef56-78ab-cd90-1234abcd0000",
   "message": {
       "log": "Subject has been modified because it contains control or non-ASCII
 characters."
       "level": "WARN"
   },
   "component": "alertmanager"
```

```
}
```

これは、件名に非 ASCII 文字が含まれていることを意味します。

実行するアクション

テンプレートの件名フィールドから、非 ASCII 文字を含む可能性のあるラベルへの参照を削除します。

KMS を呼び出す権限がありません

ログに次の AWS KMS エラーが含まれている場合

```
{
   "workspaceId": "ws-abcd1234-ef56-78ab-cd90-1234abcd0000",
   "message": {
       "log": "Notify for alerts failed, AMP is not authorized to call KMS",
       "level": "ERROR"
   },
   "component": "alertmanager"
}
```

実行するアクション

Amazon SNS トピックの暗号化に使用するキーのキーポリシーで、Amazon Managed Service for Prometheus サービスプリンシパル `aps.amazonaws.com` が `kms:GenerateDataKey*` アクションと `kms:Decrypt` アクションを実行することが許可されていることを検証します。詳細については、「SNS トピックに対するAWS KMS アクセス許可」を参照してください。

テンプレートエラー

ログに次のエラーが含まれている場合

```
              {
   "workspaceId": "ws-efdc5b42-b051-11ec-b123-4567ac120002",
   "message": {
       "log": "Notify for alerts failed. There is an error in a receiver that is using
 templates in the AlertManager definition. Make sure that the syntax is correct and
 only template functions and variables that exist are used in the receiver 'default',
 sns_configs position #2, section 'attributes'"
```

```
        "level": "ERROR"
    },
    "component": "alertmanager"
}
```

これは、AlertManager 定義で使用されているテンプレートにエラーがあることを意味します。エラーエントリには、レシーバー、sns_configs 内の位置、エラーを含むプロパティに関する指示が含まれています。

実行するアクション

アラートマネージャーの定義を検証します。構文が正しく、存在するテンプレート変数と関数を参照していることを確認します。詳細については、Prometheus オープンソースドキュメントの「Notification Template Reference」を参照してください。

Amazon Managed Service for Prometheus ワークスペースのログ記録とモニタリング

Amazon Managed Service for Prometheus は、Amazon CloudWatch を使用してオペレーションに関するデータを提供します。CloudWatch メトリクスを使用して、リソースの使用状況と Amazon Managed Service for Prometheus ワークスペースへのリクエストを確認できます。CloudWatch Logs サポートを有効にして、ワークスペースで発生するイベントのログを取得できます。

以下のトピックでは、CloudWatch の使用方法について詳しく説明します。

CloudWatch メトリクスを使用して Amazon Managed Service for Prometheus のリソースモニタリングする

Amazon Managed Service for Prometheus は、使用状況メトリクスを CloudWatch に提供します。これらのメトリクスにより、ワークスペースの使用状況が可視化されます。提供されたメトリクスは、CloudWatch の `AWS/Usage` 名前空間と `AWS/Prometheus` 名前空間で確認できます。これらのメトリクスは、CloudWatch で無料で利用できます。使用状況メトリクスの詳細については、「CloudWatch の使用状況メトリクス」を参照してください。

CloudWatch メトリクス名	リソース名	CloudWatch 名前空間	説明
ResourceCount*	CreateAlertManagerAlertsTPS	`AWS/Usage`	ワークスペースごとの 1 秒あたりの `CreateAlertManagerAlerts` API オペレーションの最大数。
ResourceCount*	DeleteAlertManagerSilencesTPS	`AWS/Usage`	ワークスペースごとの 1 秒あたりの `DeleteAlertManagerSilences` API オペレーションの最大数。

CloudWatch メトリクス名	リソース名	CloudWatch 名前空間	説明
ResourceCount*	GetAlertManagerSilenceTPS	AWS/Usage	ワークスペースごとの 1 秒あたりの GetAlertManagerSilence API オペレーションの最大数。
ResourceCount*	GetAlertManagerStatusTPS	AWS/Usage	ワークスペースごとの 1 秒あたりの GetAlertManagerStatus API オペレーションの最大数。
ResourceCount*	GetLabelsTPS	AWS/Usage	ワークスペースごとの 1 秒あたりの GetLabels API オペレーションの最大数。
ResourceCount*	GetMetricMetadataTPS	AWS/Usage	ワークスペースごとの 1 秒あたりの GetMetricMetadata API オペレーションの最大数。
ResourceCount*	GetSeriesTPS	AWS/Usage	ワークスペースごとの 1 秒あたりの GetSeries API オペレーションの最大数。
ResourceCount	InhibitionRulesInAlertManagerDefinition	AWS/Usage	アラートマネージャー定義ファイル内の禁止ルールの最大数。
ResourceCount*	ListAlertManagerAlertGroupInfosTPS	AWS/Usage	ワークスペースごとの 1 秒あたりの ListAlertManagerAlertGroupInfos API オペレーションの最大数。

CloudWatch メトリクス名	リソース名	CloudWatch 名前空間	説明
ResourceCount*	ListAlertManagerAlertGroupsTPS	AWS/Usage	ワークスペースごとの 1 秒あたりの ListAlertManagerAlertGroups API オペレーションの最大数。
ResourceCount*	ListAlertManagerAlertsTPS	AWS/Usage	ワークスペースごとの 1 秒あたりの ListAlertManagerAlerts API オペレーションの最大数。
ResourceCount*	ListAlertManagerReceiversTPS	AWS/Usage	ワークスペースごとの 1 秒あたりの ListAlertManagerReceivers API オペレーションの最大数。
ResourceCount*	ListAlertManagerSilencesTPS	AWS/Usage	ワークスペースごとの 1 秒あたりの ListAlertManagerSilences API オペレーションの最大数。
ResourceCount*	ListAlertsTPS	AWS/Usage	ワークスペースごとの 1 秒あたりの ListAlerts API オペレーションの最大数。
ResourceCount*	ListRulesTPS	AWS/Usage	ワークスペースごとの 1 秒あたりの ListRules API オペレーションの最大数。
ResourceCount*	PutAlertManagerSilencesTPS	AWS/Usage	ワークスペースごとの 1 秒あたりの PutAlertManagerSilences API オペレーションの最大数。

CloudWatch メトリクス名	リソース名	CloudWatch 名前空間	説明
ResourceCount	HAReplicaGroupCount	AWS/Usage	高可用性レプリカグループの数
ResourceCount*	QueryMetricsTPS	AWS/Usage	1 秒あたりのオペレーション数
ResourceCount*	RemoteWriteTPS	AWS/Usage	1 秒あたりのリモート書き込みオペレーション数
ResourceCount	ActiveAlerts	AWS/Usage	ワークスペースごとのアクティブなアラートの数 単位: カウント 有効な統計: Average、Minimum、Maximum
ResourceCount	ActiveSeries	AWS/Usage	ワークスペースごとのアクティブなシリーズの数 単位: カウント 有効な統計: Average、Minimum、Maximum
ResourceCount	AlertAggregationGroupSize	AWS/Usage	アラートマネージャー定義ファイル内のアラート集約グループの最大サイズ。group_by のラベル値の組み合わせごとに集約グループが作成されます。
ResourceCount	AlertManagerDefinitionSizeBytes	AWS/Usage	アラートマネージャー定義ファイルの最大サイズ (バイト単位)。

CloudWatch メトリクス名	リソース名	CloudWatch 名前空間	説明
ResourceCount	AllSilences	AWS/Usage	ワークスペースあたりの、期限切れ、アクティブ、保留中の無音を含む最大サイレンス数。
ResourceCount	IngestionRate	AWS/Usage	サンプルの取り込みレート 単位: カウント/秒 有効な統計: Average、Minimum、Maximum
ResourceCount	RuleEvaluationInterval	AWS/Usage	最小ルール評価間隔
ResourceCount	RuleGroupNamespaceDefinitionSizeBytes	AWS/Usage	ルールグループ名前空間定義ファイルの最大サイズ (バイト単位)。
ResourceCount	TemplatesInAlertManagerDefinition	AWS/Usage	アラートマネージャー定義ファイル内のテンプレートの最大数。
ResourceCount	WorkspaceCount	AWS/Usage	アカウントあたりのリージョンごとりのワークスペースの最大数。
ResourceCount	SizeOfAlerts	AWS/Usage	ワークスペース内のすべてのアラートの合計サイズ (バイト単位) 単位: バイト 有効な統計: Average、Minimum、Maximum

CloudWatch メトリクス名	リソース名	CloudWatch 名前空間	説明
ResourceCount	SuppressedAlerts	AWS/Usage	ワークスペースごとの抑制状態にあるアラートの数。アラートは、無音や禁止にすることで抑制できます。 単位: カウント 有効な統計: Average、Minimum、Maximum
ResourceCount	UnprocessedAlerts	AWS/Usage	ワークスペースごとの未処理状態のアラートの数。アラートは AlertManager が受信すると未処理状態になりますが、次の集約グループの評価を待っているアラートです。 単位: カウント 有効な統計: Average、Minimum、Maximum
ResourceCount	AllAlerts	AWS/Usage	ワークスペースあたりの任意の状態のアラートの数 単位: カウント 有効な統計: Average、Minimum、Maximum

CloudWatch メトリクス名	リソース名	CloudWatch 名前空間	説明
ResourceCount	AllRules	AWS/Usage	ワークスペースあたりの任意の状態のルールの数 単位: カウント 有効な統計: Average、Minimum、Maximum
ActiveSeriesPerLabelSet	-	AWS/Prometheus	各ユーザー定義ラベルセットの現在のアクティブなシリーズの使用状況 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum
ActiveSeriesLimitPerLabelSet	-	AWS/Prometheus	各ユーザー定義ラベルセットの現在のアクティブなシリーズの制限値 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum
AlertManagerAlertsReceived	-	AWS/Prometheus	アラートマネージャーが受信した正常なアラートの合計数 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum

CloudWatch メトリクス名	リソース名	CloudWatch 名前空間	説明
AlertManagerNotificationsFailed	-	AWS/Prometheus	失敗したアラート配信の数 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum
AlertManagerNotificationsThrottled	-	AWS/Prometheus	スロットリングされたアラートの数 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum
AnomalyDetectors	WorkspaceId	AWS/Prometheus	特定のワークスペースの異常ディテクターの総数 単位: カウント 有効な統計: Average、Minimum、Maximum
AnomalyDetectorEvaluations	WorkspaceId、AnomalyDetectorId	AWS/Prometheus	異常ディテクター評価の合計数 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum
AnomalyDetectorEvaluationFailures	WorkspaceId、AnomalyDetectorId	AWS/Prometheus	間隔での異常ディテクターの失敗の数 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum

CloudWatch メトリクス名	リソース名	CloudWatch 名前空間	説明
AnomalyDetectorLastEvaluationDuration	WorkspaceId、AnomalyDetectorId	AWS/Prometheus	異常ディテクターの最後の評価の期間 単位: 秒 有効な統計: Average、Minimum、Maximum、Sum
AnomalyDetectorMissedEvaluations	WorkspaceId、AnomalyDetectorId	AWS/Prometheus	間隔での欠落した異常ディテクター評価の数 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum
Discarded Samples**	-	AWS/Prometheus	破棄されたサンプルの数 (理由別) 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum
Discarded Series**	-	AWS/Prometheus	理由によって破棄されたサンプルを含む系列の数 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum

CloudWatch メトリクス名	リソース名	CloudWatch 名前空間	説明
DiscardedSamplesPerLabelSet	-	AWS/Prometheus	ユーザー定義ラベルセットごとに破棄されたサンプルの数 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum
DiscardedSeriesPerLabelSet	-	AWS/Prometheus	ユーザー定義ラベルセットごとに破棄されたサンプルを含む系列の数 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum
IngestionRatePerLabelSet	-	AWS/Prometheus	ユーザー定義の各ラベルセットの取り込みレート 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum
QuerySamplesProcessed	-	AWS/Prometheus	処理されたクエリサンプルの数。 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum

CloudWatch メトリクス名	リソース名	CloudWatch 名前空間	説明
RuleEvaluations	-	AWS/Prometheus	ルール評価の合計数 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum
RuleEvaluationFailures	-	AWS/Prometheus	特定の間隔におけるルール評価の失敗の数 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum
RuleGroupIterationsMissed	-	AWS/Prometheus	特定の間隔における欠落したルールグループイテレーションの数。 単位: カウント 有効な統計: Average、Minimum、Maximum、Sum
RuleGroupLastEvaluationDuration	-	AWS/Prometheus	ルールグループの最後の評価の期間。 単位: 秒 有効な統計: Average、Minimum、Maximum、Sum

*TPS メトリクスは 1 分ごとに生成され、その 1 分間での 1 秒あたりの平均です。短いバースト期間は TPS メトリクスにキャプチャされません。

**サンプルが破棄される理由には、次のようなものがあります。以下のすべての理由が DiscardedSeries メトリクスに表示されるわけではありません。

Reason	意味
greater_than_max_sample_age	1 時間を超えた古いサンプルを破棄します。
new-value-for-timestamp	重複したサンプルは、前のサンプルと同じタイムスタンプで送信されますが、値は異なります。
per_labelset_series_limit	ユーザーがラベルセットあたりのアクティブなシリーズの合計数の上限に達しました。
per_metric_series_limit	ユーザーがメトリクスごとのアクティブなシリーズ数の上限に達しました。
per_user_series_limit	ユーザーがアクティブなシリーズの合計数の上限に達しました。
rate_limited	取り込みレートが制限されました。
sample-out-of-order	サンプルが順不同で送信されたため、処理できません。
label_value_too_long	ラベル値の長さが許容される文字数の上限を超えています。
max_label_names_per_series	ユーザーがメトリクスごとのラベル名の上限数に達しました。
missing_metric_name	メトリクス名が指定されていません。
metric_name_invalid	無効なメトリクス名が指定されました。
label_invalid	無効なラベルが指定されました。
duplicate_label_names	重複するラベル名が指定されました。

> **Note**
>
> メトリクスがない場合は、そのメトリクスの値が 0 であることと同じ意味になります。

> **Note**
>
> RuleGroupIterationsMissed、RuleEvaluations、RuleEvaluationFailures、RuleGroup には、次の構造の RuleGroup ディメンションがあります。
> *RuleGroupNamespace*;*RuleGroup*

Prometheus から提供されるメトリクスへの CloudWatch アラームの設定

CloudWatch アラームを使用して、Prometheus リソースの使用状況をモニタリングできます。

Prometheus で ActiveSeries の数に対するアラームを設定するには

1. [グラフ化したメトリクス] タブを選択し、[ActiveSeries] ラベルまでスクロールします。

   [グラフ化したメトリクス] ビューには、現在取り込まれているメトリクスのみが表示されます。
2. [アクション] 列の [通知] アイコンを選択します。
3. [メトリクスと条件の指定] で、[条件値] フィールドにしきい値の条件を入力し、[次へ] を選択します。
4. [アクションの設定] で、通知の送信先となる既存の SNS トピックを選択するか、新しいトピックを作成します。
5. [名前と説明を追加] に、アラームの名前と、必要に応じて説明を追加します。
6. [アラームの作成] を選択します。

CloudWatch Logs で Amazon Managed Service for Prometheus イベントをモニタリングする

Amazon Managed Service for Prometheus は、アラートマネージャーとルーラーのエラーおよび警告イベントのログを Amazon CloudWatch Logs のロググループに記録します。アラートマネージャーとルーラーの詳細については、このガイドの「アラートマネージャー」トピックを参照してください。ワークスペースのログデータは、CloudWatch Logs のログストリームに発行できます。モニタリングするログは、Amazon Managed Service for Prometheus コンソールまたは AWS CLI を使用して構成できます。これらのログは、CloudWatch コンソールで表示したりクエリを実行したりできます。コンソールで CloudWatch Logs ログストリームを表示する方法の詳細については、「CloudWatch ユーザーガイド」の「CloudWatch でのロググループとログストリームの操作」を参照してください。

CloudWatch の無料利用枠では、最大 5 GB のログを CloudWatch Logs に発行できます。無料利用枠を超えるログは、CloudWatch 料金プランに基づいて課金されます。

トピック

- CloudWatch Logs の構成

CloudWatch Logs の構成

Amazon Managed Service for Prometheus は、アラートマネージャーとルーラーのエラーおよび警告イベントのログを Amazon CloudWatch Logs のロググループに記録します。

`create-logging-configuration` API リクエストを呼び出す AWS CLI ことで、Amazon Managed Service for Prometheus コンソールまたは で CloudWatch Logs ログ記録設定を設定できます。

前提条件

`create-logging-configuration` を呼び出す前に、以下のポリシーまたは同等のアクセス許可を、CloudWatch Logs の設定に使用する ID またはロールにアタッチします。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:UpdateLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries",
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups",
                "aps:CreateLoggingConfiguration",
                "aps:UpdateLoggingConfiguration",
                "aps:DescribeLoggingConfiguration",
                "aps:DeleteLoggingConfiguration"
```

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

CloudWatch Logs を構成するには

AWS コンソールまたは を使用して、Amazon Managed Service for Prometheus でログ記録を設定できます AWS CLI。

Console

Amazon Managed Service for Prometheus コンソールでログ記録を構成するには

1. ワークスペースの詳細パネルの [ログ] タブに移動します。
2. [ログ] パネルの右上にある [ログを管理] を選択します。
3. [ログレベル] ドロップダウンリストで、[すべて] を選択します。
4. [ロググループ] ドロップダウンリストで、ログを発行する先のロググループを選択します。

   CloudWatch コンソールで新しいロググループを作成することもできます。
5. [Save changes] (変更の保存) をクリックします。

AWS CLI

ログ記録設定は、 を使用して設定できます AWS CLI。

を使用してログ記録を設定するには AWS CLI

- を使用して AWS CLI、次のコマンドを実行します。

```
aws amp create-logging-configuration --workspace-id my_workspace_ID
                                --log-group-arn my-log-group-arn
```

制限事項

- すべてのイベントが記録されるわけではない

Amazon Managed Service for Prometheus は、`warning` または `error` レベルのイベントのみをログに記録します。

- ポリシーサイズの制限

  CloudWatch Logs リソースポリシーは 5120 文字に制限されています。CloudWatch Logs は、ポリシーがこのサイズ制限に近づいていることを検出すると、`/aws/vendedlogs/` で始まるロググループを自動的に有効にします。

  ログ記録を有効にしたステートマシンを作成するとき、Amazon Managed Service for Prometheus は、指定されたロググループで CloudWatch Logs リソースポリシーを更新する必要があります。CloudWatch Logs リソースポリシーのサイズ制限に達しないようにするには、CloudWatch Logs ロググループ名の先頭に `/aws/vendedlogs/` というプレフィックスを付けてください。Amazon Managed Service for Prometheus コンソールでロググループを作成する場合は、ロググループ名に `/aws/vendedlogs/` プレフィックスが付けられます。詳細については、CloudWatch Logs ユーザーガイド」の「特定の AWS サービスからのログ記録の有効化」を参照してください。

Amazon Managed Service for Prometheus でのクエリコストの管理

Amazon Managed Service for Prometheus では、1 つのクエリで使用できるクエリサンプル処理 (QSP) の量を制限することにより、クエリコストを制限できます。QSP には、警告とエラーの 2 種類のしきい値を設定して、クエリコストを効果的に管理および制御できます。

クエリが警告しきい値に達すると、API クエリレスポンスに警告メッセージが表示されます。Amazon Managed Grafana を介して表示されるクエリの場合、警告は Amazon Managed Grafana UI に表示され、ユーザーが高価なクエリを識別するのに役立ちます。エラーしきい値に達したクエリは課金されず、エラーで拒否されます。

Amazon Managed Service for Prometheus は、クエリスロットリングに加えて、クエリパフォーマンスデータを CloudWatch Logs に記録する機能を提供します。この機能を使用すると、クエリを詳細に分析できるため、Amazon Managed Service for Prometheus クエリを最適化し、コストをより効果的に管理できます。クエリログ記録は、指定されたクエリサンプル処理 (QSP) しきい値を超えるクエリに関する情報をキャプチャします。このデータは CloudWatch Logs に発行されるため、クエリのパフォーマンスを調査および分析できます。ログ記録されたクエリには、API クエリとルールクエリの両方が含まれます。デフォルトでは、不要な CloudWatch Logs の使用を最小限に抑えるた

めに、クエリログ記録は無効になっています。この機能は、クエリ分析に必要なときに有効にできます。

トピック

クエリログ記録の設定

`create-query-logging-configuration` API リクエストを呼び出すことで、Amazon Managed Service for Prometheus コンソールまたは AWS CLI でクエリログを設定できます。この API 本文には送信先のリストが含まれていますが、現時点では、送信先として CloudWatch Logs のみがサポートされており、送信先には CloudWatch 設定で 1 つの要素のみを含める必要があります。

前提条件

`logGroup` が既に作成されていることを確認します。設定に使用される ID またはロールには、次のポリシーまたは同等のアクセス許可が必要です。

JSON

```
{
"Version":"2012-10-17",
    "Statement": [
        {
"Effect": "Allow",
            "Action": [
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:UpdateLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries",
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups",
                "aps:CreateQueryLoggingConfiguration",
                "aps:UpdateQueryLoggingConfiguration",
```

```
                "aps:DescribeQueryLoggingConfiguration",
                "aps:DeleteQueryLoggingConfiguration"
            ],
            "Resource": "*"
        }
    ]
}
```

CloudWatch Logs を設定する。

CloudWatch Logs を設定するには、 AWS マネジメントコンソール または を使用して Amazon Managed Service for Prometheus にログインします AWS CLI。

Amazon Managed Service for Prometheus コンソールを使用してクエリログ記録を設定するには

1. ワークスペースの詳細パネルの [ログ] タブに移動します。
2. [クエリインサイト] で、[作成] を選択します。
3. [ロググループ] のドロップダウンを選択し、ログを発行するロググループを選択します。

   CloudWatch コンソールで新しいロググループを作成することもできます。
4. [しきい値 (QSP)] を入力します。
5. [保存] を選択します。

を使用してクエリログを設定するには、 コマンド AWS CLI を使用します。

```
aws amp create-query-logging-configuration \
--workspace-id my_workspace_ID \
--destinations '[{"cloudWatchLogs":{"logGroupArn":"$my-log-group-arn"},"filters":
{"qspThreshold":$qspThreshold}}]'
```

オペレーションを更新、削除、記述する方法については、「Amazon Managed Service for Prometheus API リファレンス」を参照してください。

クエリスロットリングしきい値の設定

QSP しきい値を設定するには、QueryMetrics API でクエリパラメータを指定する必要があります。

- max_samples_processed_warning_threshold – 処理されたクエリサンプルの警告しきい値を設定します

- max_samples_processed_error_threshold – 処理されたクエリサンプルのエラーしきい値を設定します

Amazon Managed Grafana ユーザーの場合、grafana データソース設定を使用して、データソースからのすべてのクエリに制限を適用できます。

1. Amazon Managed Grafana の Amazon Managed Service for Prometheus データソースを参照します。
2. [カスタムクエリパラメータ] で、しきい値ヘッダーを追加します。
3. [保存] を選択します。

ログの内容

ルールからのクエリの場合、CloudWatch Logs にクエリに関する次の情報が表示されます。

```
{
    workspaceId: "workspace_id",
    message: {
        query: "avg(rate(go_goroutines[1m])) > 1",
        name: "alert_rule",
        kind: "alerting",
        group: "test-alert",
        namespace: "test",
        samples: "59321",
    },
    component: "ruler"
}
```

API コールから発生するクエリの場合、CloudWatch Logs にクエリに関する次の情報が表示されます。

```
{
    workspaceId: "ws-5e7658c2-7ccf-4c30-9de9-2ab26fa30639",
    message: {
        query: "sum by (instance) (go_memstats_alloc_bytes{job=\"node\"})",
        queryType: "range",
        start: "1683308700000",
        end: "1683913500000",
        step: "300000",
```

```
        samples: "11496",
        userAgent: "AWSPrometheusDPJavaClient/2.0.436.0 ",
        dashboardUid: "11234",
        panelId: "12"
    },
    component: "query-frontend"
}
```

制限事項

ポリシーサイズ制限 – CloudWatch Logs リソースポリシーは 5120 文字に制限されています。CloudWatch Logs は、ポリシーがサイズ制限に近づいていることを検出すると、/aws/vendedlogs/ で始まるロググループを自動的に有効にします。ログ記録を有効にする場合、Amazon Managed Service for Prometheus は、指定されたロググループで CloudWatch Logs リソースポリシーを更新する必要があります。CloudWatch Logs リソースポリシーのサイズ制限に達しないようにするには、CloudWatch Logs ロググループ名の先頭に /aws/vendedlogs/ というプレフィックスを付けてください。

Amazon Managed Service for Prometheus のコストを理解して最適化する

以下のよくある質問とその回答は、Amazon Managed Service for Prometheus に関連するコストを理解して最適化するために役立つ可能性があります。

コストに影響する要因は何ですか?

ほとんどのお客様にとって、コストの大部分はメトリクスの取り込みに由来します。クエリの使用量が多いお客様の場合、処理されたクエリサンプル数に基づくコストも発生しますが、メトリクスのストレージがコスト全体に及ぼす影響は小規模です。これらのそれぞれにかかる料金の詳細については、Amazon Managed Service for Prometheus の製品ページの「料金」を参照してください。

コストを削減する最善の方法は何ですか? どうすれば取り込みコストを下げることができますか?

ほとんどのお客様にとって、コストの大部分は取り込みレートです (メトリクスのストレージではありません)。取り込みレートを下げるには、収集頻度を低くする (収集間隔を大きくする) か、取り込むアクティブなシリーズの数を少なくします。

コレクションエージェントからコレクション (スレイピング) 間隔を増やすことができます。Prometheus サーバー (エージェントモードで実行) と AWS Distro for OpenTelemetry (ADOT) コレクターの両方が`scrape_interval`設定をサポートしています。例えば、収集間隔を 30 秒から 60 秒に増やすと、取り込みの使用量が半減します。

`<relabel_config>` を使用して、Amazon Managed Service for Prometheus に送信されるメトリクスをフィルタリングすることもできます。Prometheus エージェントの設定における再ラベル付けの詳細については、Prometheus ドキュメントの https://prometheus.io/docs/prometheus/latest/configuration/configuration/#relabel_config を参照してください。

クエリコストを削減する最善の方法は何ですか?

クエリコストは、処理されたサンプルの数に基づきます。クエリの頻度を低くすると、クエリコストを削減できます。

クエリコストに最も寄与しているクエリをより詳細に把握するには、「」を参照してくださいAmazon Managed Service for Prometheus でのクエリコストの管理。

メトリクスの保持期間を短くした場合、合計請求額の削減につながりますか?

保持期間を短縮することはできますが、それによってコストが大幅に削減される可能性はほとんどありません。

ワークスペースの保持期間を設定する方法については、「ワークスペースの設定」を参照してください。

アラートクエリのコストを低く抑えるにはどうすればよいですか?

アラートにより、データに対するクエリが作成されるため、クエリのコストが増大します。アラートのクエリを最適化し、コストを削減するために使用できる戦略をいくつか紹介します。

- Amazon Managed Service for Prometheus アラートの使用 - Amazon Managed Service for Prometheus 外のアラートシステムの場合、外部サービスは複数のアベイラビリティーゾーンやリージョンのメトリクスをクエリするため、耐障害性または高可用性を追加するために追加のクエリが必要になる場合があります。これには、高可用性のための Grafana でのアラートが含まれます。これにより、コストが 3 倍以上に増える場合があります。Amazon Managed Service for Prometheus のアラートは最適化されており、最小限のクエリ数で高可用性と耐障害性を実現します。

  外部アラートシステムではなく、Amazon Managed Service for Prometheus のネイティブアラートを使用することをお勧めします。
- アラート間隔の最適化 - アラートのクエリを最適化する 1 つの簡単な方法は、自動更新間隔を長くすることです。1 分ごとにクエリを実行するアラートがあっても、アラートが 5 分ごとにのみ必要である場合、自動更新間隔を長くすると、アラートのクエリコストを 5 分の 1 に縮小できます。
- 最適なルックバックの使用 - クエリのルックバックウィンドウが大きいほど、より多くのデータが取得されるため、クエリのコストが増えます。PromQL クエリのルックバックウィンドウが、アラートを必要とするデータに適切なサイズであることを確認します。例えば、次のルールでは、式に 10 分のルックバックウィンドウが含まれています。

```
    - alert: metric:alerting_rule
      expr: avg(rate(container_cpu_usage_seconds_total[10m])) > 0
```

```
      for: 2m
```

`expr` を `avg(rate(container_cpu_usage_seconds_total[5m])) > 0` に変更すると、クエリのコストを削減できます。

一般的に、アラートルールを確認し、サービスに最適なメトリクスでアラートが発生していることを確認します。特に時間の経過とともにアラートを追加する場合、同じメトリクスまたは同じ情報を提供する同じメトリクスや複数のアラートに重複するアラートを作成してしまうことがよくあります。アラートのグループが同時に発生することが多い場合は、アラートを最適化し、すべてのアラートは含めないようにすることができます。

以上の推奨事項は、コストを削減するのに役立ちます。最終的には、システムの状態を理解するための適切なアラートセットを作成することおよびコストとのバランスを取る必要があります。

Amazon Managed Service for Prometheus でのアラートの詳細については、「アラートマネージャーを使用して Amazon Managed Service for Prometheus でアラートを管理および転送する」を参照してください。

請求書はいつでも確認できますか?

は AWS 使用状況 AWS Cost and Usage Report を追跡し、請求期間内にアカウントに関連付けられた推定請求額を提供します。詳細については、AWS 「コストと使用状況レポートとは」を参照してください。 AWS 「 コストと使用状況レポートユーザーガイド」の「」

コストのモニタリングにはどのようなメトリクスを使用できますか?

取り込むメトリクスサンプルは、Amazon Managed Service for Prometheus の主なコストドライバーです。直接取り込まれるサンプルの数によって月額料金が決まるため、取り込みパターンをモニタリングして理解することが不可欠です。

AWS Cost Explorer は、Amazon Managed Service for Prometheus のコストをモニタリングするための信頼できる情報源です。Cost Explorer をモニタリングして、取り込まれたサンプルを含む複数のディメンションにわたる Amazon Managed Service for Prometheus のコストの履歴とday-by-day傾向を確認できます。AWS コスト異常検出を使用すると、支出パターンの予期しない変化をモニタリングすることもできます。

IngestionRate メトリクスを使用すると、コストに直接関係する取り込みの傾向をモニタリングするための補助方法が提供されます。を追加のメトリクスIngestionRateとして使用する利点は次のとおりです。

- ワークスペースレベルの追跡 – アカウントレベルだけでなく、ワークスペースごとに取り込みをモニタリングします。
- きめ細かな可視性 – リアルタイムのインサイトを得るために、取り込みパターンを時間単位またはminute-by-minute追跡します。
- プロアクティブモニタリング – CloudWatch アラームを設定して、請求に表示される前に使用量の急増を検出します。

> **Note**
>
> IngestionRate は、ワークスペースあたりのコストと傾向、または属性コストの見積もりに使用できますが、100% 正確ではありません。は 1 分間隔でサンプリングされた平均レートIngestionRateを報告するため、このレートに時間をかけると、取り込まれたサンプルの正確な数ではなく、近似値が得られます。さらに、Amazon CloudWatch のデータ保持ポリシーは、履歴クエリで使用できる詳細度に影響し、63 日以上経過したデータは 1 時間間隔に制限されます。

CloudWatch における Amazon Managed Service for Prometheus メトリクスのモニタリングの詳細については、「CloudWatch メトリクスを使用して Amazon Managed Service for Prometheus のリソースモニタリングする」を参照してください。

でコストを表示するにはどうすればよいですか AWS Cost Explorer?

Amazon Managed Service for Prometheus コストの信頼できる情報源として、 は、月別およびリージョン別の過去の請求データを含め、取り込まれた Amazon Managed Service for Prometheus サンプルの実際の請求使用量と料金 AWS Cost Explorer を提供します。最終的な請求額とday-by-dayコスト傾向には Cost Explorer を使用します。

Amazon Managed Service for Prometheus のコストを表示するには:

アクセス AWS Cost Explorer

1. AWS マネジメントコンソールにサインインします。
2. Billing and Cost Management ダッシュボードに移動します。
3. 左側のナビゲーションメニューから Cost Explorer を選択します。
4. Cost Explorerーを起動 (初めて使用する場合) を選択します。

レポートを設定する

1. 時間範囲を希望の請求期間 (2025 年 3 月 ~ 2026 年 2 月など) に設定します。
2. フィルターで、以下を選択します。
   - サービス:「Amazon Managed Service for Prometheus」を選択します。
   - 使用タイプ: MetricSampleCount」をフィルタリングして、サンプルの取り込み料金を分離します。

データのグループ化と表示

1. Group by で、リージョンを選択して、リージョンごとのコストと使用状況データを表示します。
2. 任意の視覚化 (棒グラフ、折れ線グラフ、またはテーブル) を選択します。
3. 適用 を選択してレポートを生成します。

データのエクスポート (オプション)

1. 右上隅にあるダウンロード CSV を選択してデータをエクスポートします。
2. CSV ファイルには、請求期間、リージョン、使用タイプ、請求額、使用数量 (請求されたサンプル数) が含まれます。

> **Note**
>
> Cost Explorer データには通常 24 時間の遅延があります。最新の請求期間中、データは翌日まで利用できない場合があります。

1 か月に取り込まれたサンプルの数を計算するにはどうすればよいですか?

を使用して、Amazon CloudWatch のIngestionRateメトリクスを使用して取り込まれたサンプルのおおよその数を計算できます AWS Command Line Interface。これは、毎月の請求書を確認し、ワークスペース全体の使用パターンを理解するのに役立ちます。

取り込みデータを取得するには:

```
aws cloudwatch get-metric-data \
    --region your-region \
    --start-time start-timestamp \
    --end-time end-timestamp \
    --metric-data-queries '[
        {
            "Id": "e1",
            "Expression": "SUM(METRICS())",
            "Period": 3600
        },
        {
            "Id": "ws1",
            "MetricStat": {
                "Metric": {
                    "Namespace": "AWS/Usage",
                    "MetricName": "ResourceCount",
                    "Dimensions": [
                        {"Name": "Service",    "Value": "Prometheus"},
                        {"Name": "Resource",   "Value": "IngestionRate"},
                        {"Name": "Type",       "Value": "Resource"},
                        {"Name": "Class",      "Value": "None"},
                        {"Name": "ResourceId", "Value": "YOUR_AMP_WORKSPACE_ID"}
                    ]
                },
                "Period": 3600,
                "Stat": "Average"
            }
        }
    ]'
```

コマンドはIngestionRate、1 秒あたりのサンプル数で測定された 1 時間あたりの平均値を返します。1 か月に取り込まれたサンプルのおおよその数を計算するには、1 時間あたりのデータポイント

に 3600 (1 時間あたりの秒数) を掛けて、その時間に取り込まれたサンプルを取得し、1 か月のすべての 1 時間あたりの合計を合計します。

```
Monthly samples ≈ Σ (hourly IngestionRate average × 3600)
```

たとえば、1 時間で 1 秒あたり平均 IngestionRate 500 サンプルが返された場合、その時間は約 500 × 3600 = 1,800,000 サンプルに寄与しました。これを月の 1 時間ごとに繰り返し、結果を合計しておおよその月間取り込み数を取得します。

主要パラメータ:

- Period: 3600 (1 時間/秒)
- StartTime: 月初 (例: 2026-02-01T00:00:00Z)
- EndTime: 月末 (例: 2026-03-01T00:00:00Z)
- Stat: 平均

ワークスペース IDsを検索するには:

```
aws amp list-workspaces --region your-region
```

ワークスペース ID を使用してメトリクスをフィルタリングし、リージョン内のすべての Prometheus リソースを集約するのではなく、指定されたワークスペースのデータのみを表示します。

履歴コスト分析にはどのようなデータの詳細度を使用できますか?

Amazon CloudWatch のデータ保持ポリシーは、履歴クエリで使用できる詳細度に影響します。

- 15 日未満のデータ: 1 分間隔でクエリする (Period: 60)
- 15 ~ 63 日経過のデータ: 5 分間隔でクエリを実行する (Period: 300)
- 63 日を超えるデータ: 1 時間間隔に制限 (Period: 3600)

63 日を超える履歴分析の場合、CloudWatch は自動的にデータを最低 1 時間にダウンサンプリングします。63 日以上経過した月の請求を確認する場合は、時間単位の集計データを使用する必要があります。月別サンプル計算では、これらの時間平均データポイントを使用し、各値を合計して 1 か月全体で 3600 を掛けます。

この粒度の低下は、古いデータの正確な数ではなく が見積りIngestionRateを提供する理由にさらに役立ちます。信頼できる請求額については、常に Cost Explorer を参照してください。

CloudWatch メトリクスの保持の詳細については、Amazon CloudWatch ユーザーガイド」の「メトリクスの保持」を参照してください。

Amazon Managed Service for Prometheus のコストをモニタリングするためのベストプラクティスは何ですか?

Amazon Managed Service for Prometheus の支出を効果的に管理および最適化するには、次のモニタリングプラクティスの実装を検討してください。

- Cost Explorer を定期的にモニタリングして実際の支出傾向を追跡し、取り込まれたサンプルを含む複数のディメンションにわたるコスト異常を特定します。
- AWS コスト異常検出を有効にして、Amazon Managed Service for Prometheus の支出の予期しないコスト増加に関するアラートを受信します。
- ワークスペースレベルのモニタリングと取り込みスパイクの早期検出IngestionRateのために、で CloudWatch アラームを設定します。
- Cost Explorer データを長期のコスト分析とレポートのために定期的にエクスポートします。

月初めの請求額が月末よりも高いのはなぜですか?

Amazon Managed Service for Prometheus では、取り込み量に応じた階層型の価格モデルが採用されているため、初期の使用量のコストの方が高い結果となります。使用量が上位の取り込み階層に達するにつれて、コストが下がります。取り込み階層を含む料金の詳細については、Amazon Managed Service for Prometheus の製品ページの「料金」を参照してください。

> **Note**
>
> - 階層は、リージョン内での使用を対象としており、リージョン間での使用は対象外です。より低いレートを使用するには、リージョン内の使用量が次の階層に達する必要があります。
> - の組織では AWS Organizations、階層の使用状況は、アカウントごとではなく、支払いアカウントごとに集計されます (支払いアカウントは常に組織管理アカウントです) 。組織

内のすべてのアカウントで取り込まれたメトリクスの合計 (リージョン内) が次の階層に達すると、すべてのアカウントの課金レートが低くなります。

Amazon Managed Service for Prometheus のすべてのワークスペースを削除しましたが、まだ料金が請求されているようです。どうなっているのでしょうか?

この場合の 1 つの可能性は、削除したワークスペースにメトリクスを送信するように設定された AWS マネージドスクレイパーがまだあることです。「スクレイパーの検出と削除」の手順に従ってください。

他の AWS のサービスとの統合

Amazon Managed Service for Prometheus は、他の AWS のサービスと統合されます。このセクションでは、Amazon Elastic Kubernetes Service (Amazon EKS) コストモニタリング (Kubecost を使用) との統合と、Amazon Data Firehose を使用して CloudWatch からメトリクスを取り込む方法について説明します。また、AWS Observability Accelerator の Terraform モジュールまたは AWS Controllers for Kubernetes を使用して、Amazon Managed Service for Prometheus をセットアップおよび管理する方法についても説明します。

トピック

- Amazon EKS コストモニタリングとの統合
- AWS Observability Accelerator で Amazon Managed Service for Prometheus をセットアップする
- AWS Controllers for Kubernetes を使用して Amazon Managed Service for Prometheus を管理する
- CloudWatch メトリクスと Amazon Managed Service for Prometheus の統合

Amazon EKS コストモニタリングとの統合

Amazon Managed Service for Prometheus は Amazon Elastic Kubernetes Service (Amazon EKS) のコストモニタリング (Kubecost を使用) と統合され、コスト配分を計算し、Kubernetes クラスターの最適化に関するインサイトを提供します。Amazon Managed Service for Prometheus を Kubecost と共に使用すると、信頼性の高い方法でコストモニタリングをスケールして、より大規模なクラスターをサポートできます。

Kubecost との統合により、Amazon EKS クラスターのコストが細かく可視化されます。コンテナレベルからクラスターレベル、さらにはマルチクラスターレベルまで、Kubernetes コンテキストの大部分でコストを集計できます。複数のコンテナやクラスターにまたがるレポートを生成して、ショーバックまたはチャージバックの目的でコストを追跡できます。

単一クラスターまたはマルチクラスターのシナリオで Kubecost と統合する手順を以下に示します。

- 単一クラスター統合 - Amazon EKS コストモニタリングを単一のクラスターと統合する方法については、AWS ブログ記事の「Integrating Kubecost with Amazon Managed Service for Prometheus」を参照してください。
- マルチクラスター統合 - Amazon EKS コストモニタリングを複数のクラスターと統合する方法については、AWS ブログ記事の「Multi-cluster cost monitoring for Amazon EKS using Kubecost and Amazon Managed Service for Prometheus」を参照してください。

> **Note**
>
> Kubecost の使用方法の詳細については、「Amazon EKS ユーザーガイド」の「コストモニタリング」を参照してください。

AWS Observability Accelerator で Amazon Managed Service for Prometheus をセットアップする

AWS は、Amazon Elastic Kubernetes Service (Amazon EKS) プロジェクトのモニタリング、ログ記録、アラート、ダッシュボードなどのオブザーバビリティツールを提供します。これには、Amazon Managed Service for Prometheus、Amazon Managed Grafana、AWS Distro for OpenTelemetry、その他のツールが含まれます。これらのツールを一緒に使用しやすくするために、 は、オブザーバビリティAWS アクセラレーターと呼ばれるこれらのサービスでオブザーバビリティを設定する Terraform モジュール AWS を提供します。

AWS Observability Accelerator には、Amazon Managed Service for Prometheus 用の 2 つのコレクタープロファイルが用意されています。

- マネージドメトリクス (エージェントレス) – Amazon Managed Service for Prometheus コレクターを使用します。これは、クラスターの外部で実行されるフルマネージド型のエージェントレススクレイパーです。管理するコレクターポッドはありません。メトリクスのみ。
- セルフマネージド – クラスターに Helm 経由で OpenTelemetry Collector をデプロイします。メトリクス、トレース (AWS X-Ray)、ログ (Amazon CloudWatch) をサポートします。

このセクションでは、推奨されるエージェントレスアプローチから始めて、両方のオプションについて説明します。

Terraform テンプレートと詳細な手順については、AWS Observability Accelerator for Terraform の GitHub ページを参照してください。

前提条件

AWS Observability Accelerator を使用するには、既存の Amazon EKS クラスターと、以下の前提条件が必要です。

- AWS CLI – コマンドラインから AWS 機能を呼び出すために使用されます。

- kubectl - コマンドラインから EKS クラスターを制御するために使用されます。
- Terraform (>= 1.5.0) – このソリューションのリソースの作成を自動化するために使用されます。 AWS アカウント内で Amazon Managed Service for Prometheus、Amazon Managed Grafana、IAM を作成および管理するためのアクセス権を持つ IAM ロールを AWS プロバイダーに設定する必要があります。Terraform 用の AWS プロバイダーを構成する方法の詳細については、Terraform ドキュメントの「AWS Provider」を参照してください。

マネージドメトリクス (エージェントレス) の例を使用する

この例では、Amazon Managed Service for Prometheus コレクターを使用して、コレクターポッドをデプロイせずに Amazon EKS クラスターから Prometheus メトリクスをスクレイプします。コレクターには、2 つの異なるアベイラビリティーゾーンに少なくとも 2 つのサブネットが必要です。詳細については、GitHub の eks-amp-managed の例を参照してください。

エージェントレスインフラストラクチャモニタリング Terraform モジュールを使用するには

1. プロジェクトを作成するフォルダーから、次のコマンドを使用してリポジトリをクローンします。

```
git clone https://github.com/aws-observability/terraform-aws-observability-
accelerator.git
```

2. 次のコマンドを実行して Terraform を初期化します。

```
cd examples/eks-amp-managed

terraform init
```

3. 新しい `terraform.tfvars` ファイルを作成し、以下の例を記述します。Amazon EKS クラスターの AWS リージョン、クラスター ID、VPC ネットワークの詳細を使用します。コレクターには、2 つの異なるアベイラビリティーゾーンに少なくとも 2 つのサブネットが必要です。

```
# (mandatory) AWS Region where your resources will be located
aws_region = "eu-west-1"

# (mandatory) EKS Cluster name
eks_cluster_id = "my-eks-cluster"

# (mandatory) Subnets for the managed scraper (>= 2 AZs)
scraper_subnet_ids = ["subnet-aaa", "subnet-bbb"]
```

```
# (mandatory) Security group allowing scraper access to the EKS API
scraper_security_group_ids = ["sg-xxx"]
```

4. 使用する Amazon Managed Grafana ワークスペースがない場合は作成します。新しいワークスペースを作成する方法については、「Amazon Managed Grafana User Guide」の「Create your first workspace」を参照してください。
5. コマンドラインで次のコマンドを実行して、Terraform で Grafana ワークスペースを使用するために必要な 2 つの変数を作成します。*grafana-workspace-id* は、Grafana ワークスペースの ID に置き換えます。

```
export TF_VAR_managed_grafana_workspace_id=grafana-workspace-id
export TF_VAR_grafana_api_key=`aws grafana create-workspace-api-key --key-name
 "observability-accelerator-$(date +%s)" --key-role ADMIN --seconds-to-live 1200 --
workspace-id $TF_VAR_managed_grafana_workspace_id --query key --output text`
```

6. (オプション) 既存の Amazon Managed Service for Prometheus ワークスペースを使用するには、次の例のように `terraform.tfvars` ファイルに ID を追加します。*prometheus-workspace-id* は Prometheus ワークスペース ID に置き換えます。既存のワークスペースを指定しない場合は、新しい Prometheus ワークスペースが自動的に作成されます。

```
# (optional) Leave it empty for a new workspace to be created
managed_prometheus_workspace_id = "prometheus-workspace-id"
```

7. 次のコマンドを使用してソリューションをデプロイします。

```
terraform apply -var-file=terraform.tfvars
```

これにより、以下を含むリソースが AWS アカウントに作成されます。

- 新しい Amazon Managed Service for Prometheus ワークスペース (既存のワークスペースを使用する場合を除く)。
- Amazon EKS クラスターから Prometheus メトリクスをスクレイプするように設定された Amazon Managed Service for Prometheus コレクター (エージェントレススクレイパー) 。
- Amazon Managed Service for Prometheus ワークスペースでの Prometheus の記録とアラートルール。
- kube-state-metrics と node-exporter は、インフラストラクチャメトリクス用に Amazon EKS クラスターにデプロイされます。

- Amazon Managed Grafana の現在のワークスペース内の新しいデータソースとダッシュボード。ダッシュボードは EKS Monitoring の下に表示されます。

代替: セルフマネージド OpenTelemetry Collector

トレース、ログ、またはコレクションパイプラインを完全に制御する必要がある場合は、セルフマネージドプロファイルを使用します。これにより、Amazon EKS クラスターに Helm 経由で OpenTelemetry Collector がデプロイされ、Prometheus メトリクスをスクレイプして Amazon Managed Service for Prometheus にリモート書き込みするように設定されます。また、トレース (AWS X-Ray) とログ (Amazon CloudWatch) もサポートしています。詳細については、GitHub の eks-amp-otel の例を参照してください。

セルフマネージド Terraform モジュールを使用するには

1. リポジトリをクローンし、Terraform を初期化します。

```
git clone https://github.com/aws-observability/terraform-aws-observability-
accelerator.git
cd examples/eks-amp-otel
terraform init
```

2. 新しい `terraform.tfvars` ファイルを作成し、以下の例を記述します。

```
# (mandatory) AWS Region where your resources will be located
aws_region = "eu-west-1"

# (mandatory) EKS Cluster name
eks_cluster_id = "my-eks-cluster"
```

3. マネージドメトリクスの例と同じステップを使用して、Amazon Managed Grafana ワークスペースと API キーを設定します (上記のステップ 4 ~ 6)。
4. 次のコマンドを使用してソリューションをデプロイします。

```
terraform apply -var-file=terraform.tfvars
```

これにより、 AWS アカウントに次のリソースが作成されます (エージェントレスアプローチとは異なり、コレクターはクラスター内で実行されます) 。

- Amazon Managed Service for Prometheus ワークスペース (指定されていない場合) 。

- データソースとダッシュボードを備えた Amazon Managed Grafana ワークスペース。
- Amazon EKS クラスターの Helm 経由でデプロイされた OpenTelemetry Collector。Prometheus メトリクスをスクレイプし、Amazon Managed Service for Prometheus にリモート書き込みするように設定されています。
- OpenTelemetry Collector のサービスアカウント (IRSA) の IAM ロール。
- パイプラインを AWS X-Ray にトレースします (デフォルトで有効) 。
- Amazon CloudWatch にパイプラインをログに記録します (デフォルトで有効) 。

ダッシュボードの表示。

新しいダッシュボードを表示するには、Amazon Managed Grafana ワークスペースでそのダッシュボードを開きます。インフラストラクチャダッシュボードは Terraform によって自動的にプロビジョニングされます。Amazon Managed Grafana の使用方法の詳細については、「Amazon Managed Grafana User Guide」の「Working in your Grafana workspace」を参照してください。

AWS Controllers for Kubernetes を使用して Amazon Managed Service for Prometheus を管理する

Amazon Managed Service for Prometheus は、AWS Controllers for Kubernetes (ACK) と統合され、Amazon EKS のワークスペース、アラートマネージャー、ルーラーリソースの管理をサポートします。 AWS Controllers for Kubernetes カスタムリソース定義 (CRDs) とネイティブ Kubernetes オブジェクトは、クラスター外のリソースを定義しなくても使用できます。

このセクションでは、既存の Amazon EKS クラスターで AWS Controllers for Kubernetes と Amazon Managed Service for Prometheus を設定する方法について説明します。

AWS Controllers for Kubernetes を紹介し、Amazon Managed Service for Prometheus 用の ACK コントローラーを紹介するブログ記事を読むこともできます。

前提条件

AWS Controllers for Kubernetes と Amazon Managed Service for Prometheus を Amazon EKS クラスターと統合する前に、次の前提条件が必要です。

- Amazon Managed Service for Prometheus AWS アカウント および IAM ロールをプログラムで作成するには、既存の および アクセス許可が必要です。

- OpenID Connect (OIDC) を有効にした既存の Amazon EKS クラスターが必要です。

  OIDC が有効でない場合、次のコマンドを使用して有効にすることができます。*YOUR_CLUSTER_NAME* と *AWS_REGION* は、アカウントに応じた適切な値に置き換えてください。

  ```
  eksctl utils associate-iam-oidc-provider \
      --cluster ${YOUR_CLUSTER_NAME} --region ${AWS_REGION} \
      --approve
  ```

  Amazon EKS で OIDC を使用する方法の詳細については、「Amazon EKS ユーザーガイド」の「OpenID Connect アイデンティティプロバイダーからクラスターのユーザーを認証する」と「クラスター用の IAM OIDC プロバイダーの作成」を参照してください。
- Amazon EKS クラスターに Amazon EBS CSI ドライバーがインストールされている必要があります。
- AWS CLI がインストールされている必要があります。 AWS CLI は、コマンドラインから AWS 機能を呼び出すために使用されます。
- Kubernetes のパッケージマネージャーである Helm がインストールされている必要があります。
- Amazon EKS クラスターで、Prometheus のコントロールプレーンメトリクスがセットアップされている必要があります。
- 新しいワークスペースからのアラートの送信先となる Amazon Simple Notification Service (Amazon SNS) トピックが必要です。トピックにメッセージを送信するためのアクセス許可が Amazon Managed Service for Prometheus に付与されていることを確認してください。

Amazon EKS クラスターが正しく構成されたら、`kubectl get --raw /metrics` を呼び出して、Prometheus 用にフォーマットされたメトリクスを確認できます。これで、 AWS Controllers for Kubernetes サービスコントローラーをインストールし、それを使用して Amazon Managed Service for Prometheus リソースをデプロイする準備が整いました。

AWS Controllers for Kubernetes を使用したワークスペースのデプロイ

新しい Amazon Managed Service for Prometheus ワークスペースをデプロイするには、 AWS Controllers for Kubernetes コントローラーをインストールし、それを使用してワークスペースを作成します。

AWS Controllers for Kubernetes を使用して新しい Amazon Managed Service for Prometheus ワークスペースをデプロイするには

1. 以下のコマンドを実行し、Helm を使用して Amazon Managed Service for Prometheus サービスコントローラーをインストールします。詳細については、GitHub の Controllers for Kubernetes ドキュメントの「Install an ACK AWS Controller」を参照してください。*region* には、us-east-1 など、システムに適したリージョンを使用してください。

```
export SERVICE=prometheusservice
export RELEASE_VERSION=`curl -sL https://api.github.com/repos/aws-controllers-k8s/
$SERVICE-controller/releases/latest | jq -r '.tag_name | ltrimstr("v")'`
export ACK_SYSTEM_NAMESPACE=ack-system
export AWS_REGION=region

aws ecr-public get-login-password --region us-east-1 | helm registry login --
username AWS --password-stdin public.ecr.aws
helm install --create-namespace -n $ACK_SYSTEM_NAMESPACE ack-$SERVICE-controller \
  oci://public.ecr.aws/aws-controllers-k8s/$SERVICE-chart --version=
$RELEASE_VERSION --set=aws.region=$AWS_REGION
```

   しばらくすると、成功を示す次のようなレスポンスが表示されます。

```
You are now able to create Amazon Managed Service for Prometheus (AMP) resources!
The controller is running in "cluster" mode.
The controller is configured to manage AWS resources in region: "us-east-1"
```

   オプションで、次のコマンドを使用して Controllers for Kubernetes AWS コントローラーが正常にインストールされたことを確認できます。

```
helm list --namespace $ACK_SYSTEM_NAMESPACE -o yaml
```

   これにより、コントローラー ack-prometheusservice-controller に関する情報 (status: deployed など) が返されます。

2. workspace.yaml という名前のファイルを作成し、次のテキストを記述します。これは、作成するワークスペースの設定として使用されます。

```
apiVersion: prometheusservice.services.k8s.aws/v1alpha1
kind: Workspace
metadata:
```

```
  name: my-amp-workspace
spec:
  alias: my-amp-workspace
  tags:
    ClusterName: EKS-demo
```

3. 次のコマンドを実行してワークスペースを作成します (このコマンドでは、ステップ 1 で設定したシステム変数が使用されます)。

```
kubectl apply -f workspace.yaml -n $ACK_SYSTEM_NAMESPACE
```

しばらくすると、アカウントに my-amp-workspace という新しいワークスペースが表示されます。

次のコマンドを実行して、ワークスペース ID などのワークスペースの詳細とステータスを確認します。または、Amazon Managed Service for Prometheus コンソールで新しいワークスペースを確認することもできます。

```
kubectl describe workspace my-amp-workspace -n $ACK_SYSTEM_NAMESPACE
```

> **Note**
>
> 新しいワークスペースを作成する代わりに、既存のワークスペースを使用することもできます。

4. この後で作成する Rulegroups と AlertManager の設定として、2 つの新しい yaml ファイルを以下に示す内容で作成します。

次の設定を rulegroup.yaml として保存します。*WORKSPACE-ID* は、前のステップで確認したワークスペース ID に置き換えます。

```
apiVersion: prometheusservice.services.k8s.aws/v1alpha1
kind: RuleGroupsNamespace
metadata:
  name: default-rule
spec:
  workspaceID: WORKSPACE-ID
  name: default-rule
  configuration: |
    groups:
```

```
    - name: example
      rules:
      - alert: HostHighCpuLoad
        expr: 100 - (avg(rate(node_cpu_seconds_total{mode="idle"}[2m])) * 100) > 60
        for: 5m
        labels:
          severity: warning
          event_type: scale_up
        annotations:
          summary: Host high CPU load (instance {{ $labels.instance }})
          description: "CPU load is > 60%\n  VALUE = {{ $value }}\n  LABELS =
 {{ $labels }}"
      - alert: HostLowCpuLoad
        expr: 100 - (avg(rate(node_cpu_seconds_total{mode="idle"}[2m])) * 100) < 30
        for: 5m
        labels:
          severity: warning
          event_type: scale_down
        annotations:
          summary: Host low CPU load (instance {{ $labels.instance }})
          description: "CPU load is < 30%\n  VALUE = {{ $value }}\n  LABELS =
 {{ $labels }}"
```

次の設定を alertmanager.yaml として保存します。*WORKSPACE-ID* は、前のステップで確認したワークスペース ID に置き換えます。*TOPIC-ARN* を通知を送信する Amazon SNS トピックの ARN に置き換え、*REGION* を使用中の に置き換え AWS リージョン ます。Amazon Managed Service for Prometheus に、Amazon SNS トピックへのアクセス許可が必要であることを忘れないでください。

```
apiVersion: prometheusservice.services.k8s.aws/v1alpha1
kind: AlertManagerDefinition
metadata:
  name: alert-manager
spec:
  workspaceID: WORKSPACE-ID
  configuration: |
    alertmanager_config: |
      route:
         receiver: default_receiver
      receivers:
        - name: default_receiver
          sns_configs:
```

```
      - topic_arn: TOPIC-ARN
        sigv4:
          region: REGION
        message: |
          alert_type: {{ .CommonLabels.alertname }}
          event_type: {{ .CommonLabels.event_type }}
```

> **Note**
>
> これらの設定ファイルの形式の詳細については、「RuleGroupsNamespaceData」および「AlertManagerDefinitionData」を参照してください。

5. 次のコマンドを実行して、ルールグループとアラートマネージャーの設定を作成します (このコマンドでは、ステップ 1 で設定したシステム変数が使用されます)。

```
kubectl apply -f rulegroup.yaml -n $ACK_SYSTEM_NAMESPACE
kubectl apply -f alertmanager.yaml -n $ACK_SYSTEM_NAMESPACE
```

しばらくすると変更が有効になります。

> **Note**
>
> リソースを作成するのではなく更新する場合は、yaml ファイルを更新し、kubectl apply コマンドを再実行するだけです。
> リソースを削除するには、次のコマンドを実行します。*ResourceType* は、削除するリソースのタイプとして、Workspace、AlertManagerDefinition、RuleGroupNamespace のいずれかに置き換えます。*ResourceName* は、削除するリソースの名前に置き換えます。
>
> ```
> kubectl delete ResourceType ResourceName -n $ACK_SYSTEM_NAMESPACE
> ```

これで、新しいワークスペースのデプロイは完了です。次のセクションでは、このワークスペースにメトリクスを送信するようにクラスターを構成する方法を説明します。

Amazon Managed Service for Prometheus ワークスペースに書き込むための Amazon EKS クラスターの構成

このセクションでは、Helm を使用して、Amazon EKS クラスターで実行されている Prometheus を構成し、前のセクションで作成した Amazon Managed Service for Prometheus ワークスペースへのメトリクスのリモートで書き込みを行う方法について説明します。

この手順では、メトリクスの取り込みに使用するために作成した IAM ロールの名前が必要です。まだ作成していない場合は、「Amazon EKS クラスターからメトリクスを取り込むためのサービスロールの設定」を参照して、詳細と手順を確認してください。これらの手順に従うと、amp-iamproxy-ingest-role という IAM ロールが作成されます。

Amazon EKS クラスターをリモート書き込み用に構成するには

1. 次のコマンドを使用して、ワークスペースの prometheusEndpoint を取得します。*WORKSPACE-ID* は、前のセクションで確認したワークスペース ID に置き換えます。

   ```
   aws amp describe-workspace --workspace-id WORKSPACE-ID
   ```

   返される結果には prometheusEndpoint が含まれ、次のような形式になります。

   ```
   https://aps-workspaces.us-west-2.amazonaws.com/workspaces/ws-a1b2c3d4-a123-b456-c789-ac1234567890/
   ```

   後のステップで使用するために、この URL を保存しておきます。

2. 次のテキストで新しいファイルを作成し、prometheus-config.yaml という名前を付けます。*account* は自分のアカウント ID に、*workspaceURL/* は先ほど確認した URL に、*region* はシステムの適切な AWS リージョン に置き換えます。

   ```
   serviceAccounts:
           server:
               name: "amp-iamproxy-ingest-service-account"
               annotations:
                   eks.amazonaws.com/role-arn: "arn:aws:iam::account:role/amp-iamproxy-ingest-role"
   server:
       remoteWrite:
           - url: workspaceURL/api/v1/remote_write
             sigv4:
   ```

```
            region: region
          queue_config:
            max_samples_per_send: 1000
            max_shards: 200
            capacity: 2500
```

3. 次の Helm コマンドを使用して、Prometheus のチャート名、名前空間の名前、チャートのバージョンを確認します。

```
helm ls --all-namespaces
```

ここまでの手順に基づくと、Prometheus チャートと名前空間にはどちらも prometheus という名前が付いていて、チャートのバージョンは 15.2.0 のようになります。

4. 前のステップで確認した *PrometheusChartName*、*PrometheusNamespace*、*PrometheusChartVersion* を使用して、次のコマンドを実行します。

```
helm upgrade PrometheusChartName prometheus-community/prometheus -
n PrometheusNamespace -f prometheus-config.yaml --version PrometheusChartVersion
```

数分後に、アップグレードが成功したことを示すメッセージが表示されます。

5. 必要に応じて、awscurl を使用して Amazon Managed Service for Prometheus エンドポイントにクエリを実行して、メトリクスが正常に送信されていることを確認します。*Region* を使用中 AWS リージョン の に、*workspaceURL/* をステップ 1 で見つけた URL に置き換えます。

```
awscurl --service="aps" --region="Region" "workspaceURL/api/v1/query?
query=node_cpu_seconds_total"
```

これで、YAML ファイルを設定として使用して、Amazon Managed Service for Prometheus ワークスペースを作成し、そのワークスペースに Amazon EKS クラスターから接続することができました。これらのファイルはカスタムリソース定義 (CRD) と呼ばれ、Amazon EKS クラスター内に配置されます。 AWS Controllers for Kubernetes コントローラーを使用して、クラスターからすべての Amazon Managed Service for Prometheus リソースを直接管理できます。

CloudWatch メトリクスと Amazon Managed Service for Prometheus の統合

すべてのメトリクスを 1 か所にまとめると便利です。Amazon Managed Service for Prometheus は、Amazon CloudWatch メトリクスを自動的には取り込みません。ただし、Amazon Data Firehose と AWS Lambda を使用すると、CloudWatch メトリクスを Amazon Managed Service for Prometheus にプッシュできます。

このセクションでは、Amazon CloudWatch メトリクスストリームをインストルメント化し、Amazon Data Firehose と AWS Lambda を使用して Amazon Managed Service for Prometheus にメトリクスを取り込む方法について説明します。

シナリオ全体をデモンストレーションするために、AWS Cloud Development Kit (CDK) を使用してスタックをセットアップし、Firehose 配信ストリーム、Lambda、Amazon S3 バケットを作成します。

インフラストラクチャ

まず、このレシピのインフラストラクチャをセットアップする必要があります。

CloudWatch メトリクスストリームを使用すると、ストリーミングメトリクスデータを HTTP エンドポイントまたは Amazon S3 バケットに転送できます。

インフラストラクチャのセットアップは、次の 4 つのステップで構成されます。

- 前提条件を構成する
- Amazon Managed Service for Prometheus ワークスペースを作成する
- 依存関係をインストールする
- スタックをデプロイする

前提条件

- AWS CLI が環境にインストールされ、構成されている。
- AWS CDK TypeScript が環境にインストールされている。
- Node.js と Go が環境にインストールされている。
- AWS オブザーバビリティ CloudWatch メトリクスエクスポーターの github リポジトリ (`CWMetricsStreamExporter`) がローカルマシンに複製されている。

Amazon Managed Service for Prometheus ワークスペースを作成するには

1. このレシピのデモアプリケーションは、Amazon Managed Service for Prometheus 上で実行されます。次のコマンドを使用して、Amazon Managed Service for Prometheus Workspace ワークスペースを作成します。

```
aws amp create-workspace --alias prometheus-demo-recipe
```

2. 次のコマンドを使用して、ワークスペースが作成されたことを確認します。

```
aws amp list-workspaces
```

Amazon Managed Service for Prometheus の詳細については、「Amazon Managed Service for Prometheus ユーザーガイド」を参照してください。

依存関係をインストールするには

1. 依存関係のインストール

aws-o11y-recipes リポジトリのルートから、次のコマンドを使用してディレクトリを CWMetricStreamExporter に変更します。

```
cd sandbox/CWMetricStreamExporter
```

以降では、このディレクトリをリポジトリのルートと見なします。

2. 次のコマンドを実行して、ディレクトリを /cdk に変更します。

```
cd cdk
```

3. 次のコマンドを実行して、CDK の依存関係をインストールします。

```
npm install
```

4. ディレクトリをリポジトリのルートに戻してから、次のコマンドを使用してディレクトリを /lambda に変更します。

```
cd lambda
```

5. /lambda フォルダに移動したら、次のコマンドを使用して Go の依存関係をインストールします。

```
go get
```

これですべての依存関係がインストールされました。

スタックをデプロイするには

1. リポジトリのルートで config.yaml を開き、Amazon Managed Service for Prometheus ワークスペース URL を変更して、{workspace} を新しく作成したワークスペース ID に置き換えます。さらに、リージョンを変更して、Amazon Managed Service for Prometheus ワークスペースのあるリージョンを指定します。

例えば、以下の部分を変更します。

```
AMP:
    remote_write_url: "https://aps-workspaces.us-east-2.amazonaws.com/workspaces/
{workspaceId}/api/v1/remote_write"
    region: us-east-2
```

Firehose 配信ストリームと Amazon S3 バケットの名前を好みに合わせて変更します。

2. リポジトリのルートで次のコマンドを実行して、AWS CDK と Lambda コードをビルドします。

```
npm run build
```

このビルドステップにより、Go Lambda バイナリがビルドされ、CDK が CloudFormation にデプロイされます。

3. スタックに必要とされる IAM の変更を確認して承認し、デプロイを完了します。
4. (オプション) 次のコマンドを実行すると、スタックが作成されたことを確認できます。

```
aws cloudformation list-stacks
```

リストに CDK Stack という名前のスタックが表示されます。

Amazon CloudWatch ストリームの作成

これでメトリクスを処理する Lambda 関数が設定されたので、Amazon CloudWatch からメトリクスストリームを作成できます。

CloudWatch メトリクスストリームを作成するには

1. CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/home#metric-streams:streamsList) に移動し、[メトリクスストリームの作成] を選択します。
2. 必要なメトリクスを選択します。すべてのメトリクスを選択することも、特定の名前空間からのメトリクスのみを選択することもできます。
3. Configuration で、[アカウントが所有している既存の Firehose を選択] を選択します。
4. CDK によって以前に作成された Firehose を使用します。[Kinesis Data Firehose ストリームを選択] ドロップダウンで、以前に作成したストリームを選択します。これは CdkStack-KinesisFirehoseStream123456AB-sample1234 のような名前になります。
5. 出力形式を [JSON] に変更します。
6. メトリクスストリームにわかりやすい名前を付けます。
7. [メトリクスストリームの作成] を選択します。
8. (オプション) Lambda 関数の呼び出しを検証するには、Lambda コンソールに移動して KinesisMessageHandler 関数を選択します。[モニタリング] タブと [ログ] サブタブを選択すると、[最近の呼び出し] に、トリガーされている Lambda 関数のエントリが表示されます。

> **Note**
>
> 呼び出しが [モニタリング] タブに表示されるようになるまでに、最大で 5 分ほどかかることがあります。

これで、Amazon CloudWatch から Amazon Managed Service for Prometheus にメトリクスがストリーミングされるようになりました。

クリーンアップ

この例で使用したリソースのクリーンアップが必要になる場合があります。以下の手順では、その方法を説明します。これにより、作成したメトリクスストリームが停止します。

リソースをクリーンアップするには

1. まず、次のコマンドを使用して CloudFormation スタックを削除します。

```
cd cdk
cdk destroy
```

2. Amazon Managed Service for Prometheus ワークスペースを削除します。

```
aws amp delete-workspace --workspace-id \
    `aws amp list-workspaces --alias prometheus-sample-app --query
 'workspaces[0].workspaceId' --output text`
```

3. 最後に、Amazon CloudWatch コンソールを使用して Amazon CloudWatch メトリクスストリームを削除します。

Amazon Managed Service for Prometheus でのセキュリティ

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を活用できます。

セキュリティは、 AWS お客様とお客様の間の責任共有です。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 – クラウドで AWS 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 AWS 責任があります。 AWS また、 では、安全に使用できるサービスも提供しています。サードパーティーの監査者は、AWS コンプライアンスプログラムコンプライアンスプログラムの一環として、当社のセキュリティの有効性を定期的にテストおよび検証。Amazon Managed Service for Prometheus 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AWS プログラムによる対象範囲内のサービスコンプライアンス」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、Amazon Managed Service for Prometheus を使用する際に責任共有モデルを適用する方法を理解するために役立ちます。以下のトピックでは、セキュリティとコンプライアンスの目標を達成するように Amazon Managed Service for Prometheus を構成する方法を説明します。また、Amazon Managed Service for Prometheus リソースのモニタリングと保護に役立つ他の AWS サービスの使用方法についても説明します。

トピック

- サービスアカウントの IAM ロールの設定
- インターフェイス VPC エンドポイントでの Amazon Managed Service for Prometheus の使用

Amazon Managed Service for Prometheus でのデータ保護

責任 AWS 共有モデル、Amazon Managed Service for Prometheus 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。ユーザーは、このインフラストラクチャでホストされ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 のサービス 」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、「Data Privacy FAQChina」を参照してください。 欧州におけるデータ保護に関する情報については、General Data Protection Regulation (GDPR) Center 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 アイデンティティセンターまたは AWS Identity and Access Management (IAM) 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。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。TLS 1.2 は必須ですが、TLS 1.3 を推奨します。
- で 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。CloudTrail 証跡を使用して AWS アクティビティをキャプチャする方法については、「 AWS CloudTrail ユーザーガイド」のCloudTrail 証跡の使用」を参照してください。
- AWS 暗号化ソリューションと、その中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 のサービス。
- Amazon Macie などの高度な管理された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 API AWS を介して にアクセスするときに FIPS 140-3 検証済み暗号化モジュールが必要な場合は、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を、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含めないことを強くお勧めします。これは、コンソール、API、または SDK を使用して Amazon Managed Service for Prometheus AWS CLIまたは他の AWS のサー

ビス を使用する場合も同様です。 AWS SDKs タグ、または名前に使用される自由記述のテキストフィールドに入力したデータは、請求または診断ログに使用される場合があります。外部サーバーへの URL を提供する場合は、そのサーバーへのリクエストを検証するための認証情報を URL に含めないように強くお勧めします。

トピック

Amazon Managed Service for Prometheus によって収集されるデータ

Amazon Managed Service for Prometheus は、アカウントで実行されている Prometheus サーバーから Amazon Managed Service for Prometheus に送信するように構成された運用メトリクスを収集して保存します。このデータには以下が含まれています。

- メトリクス値
- データの識別と分類に役立つメトリクスラベル (任意のキーと値のペア)
- データサンプルのタイムスタンプ

一意のテナント ID により、さまざまな顧客からのデータが分離されます。これらの ID は、どの顧客データにアクセスできるかを制限します。顧客がテナント ID を変更することはできません。

Amazon Managed Service for Prometheus は、 AWS Key Management Service (AWS KMS) キーで保存するデータを暗号化します。これらのキーは Amazon Managed Service for Prometheus によって管理されます。

> **Note**
>
> Amazon Managed Service for Prometheus は、データを暗号化するためのカスタマーマネージドキーの作成をサポートしています。Amazon Managed Service for Prometheus がデフォルトで使用するキーと、独自のカスタマーマネージドキーの使用方法の詳細については、「保管中の暗号化」を参照してください。

転送中のデータは HTTPS で自動的に暗号化されます。Amazon Managed Service for Prometheus は、HTTPS を使用して AWS リージョン内のアベイラビリティーゾーン間の接続を内部的に保護します。

保管中の暗号化

デフォルトでは、Amazon Managed Service for Prometheus は保管時の暗号化を自動的に提供し、AWS 所有の暗号化キーを使用してこれを行います。

- AWS 所有キー – Amazon Managed Service for Prometheus は、これらのキーを使用して、ワークスペースにアップロードされたデータを自動的に暗号化します。 AWS 所有キーを表示、管理、使用したり、その使用を監査したりすることはできません。ただし、データを暗号化するキーを保護するためのアクションの実施やプログラムの変更を行う必要はありません。詳細については、AWS Key Management Service デベロッパーガイドの「AWS 所有キー」を参照してください。

保管中のデータの暗号化は、個人を特定できる情報など、顧客の機密データを保護するにあたって伴う運用上のオーバーヘッドと複雑さを軽減するために役立ちます。これにより、厳格な暗号化のコンプライアンスと規制要件に対応する安全なアプリケーションを構築できます。

ワークスペースの作成時にカスタマーマネージドキーを使用することもできます。

- カスタマーマネージドキー — Amazon Managed Service for Prometheus では、ワークスペース内のデータを暗号化するために、ユーザーが作成、所有、管理する対称型カスタマーマネージドキーの使用をサポートします。この暗号化は完全に制御できるため、次のようなタスクを実行できます。
  - キーポリシーの策定と維持
  - IAM ポリシーとグラントの策定と維持
  - キーポリシーの有効化と無効化
  - キー暗号化マテリアルのローテーション
  - タグを追加する
  - キーエイリアスの作成
  - 削除のためのキースケジューリング

  詳細については、「AWS Key Management Service デベロッパーガイド」の「カスタマーマネージドキー」を参照してください。

カスタマーマネージドキーと AWS 所有キーのどちらを慎重に使用するかを選択します。カスタマーマネージドキーで作成されたワークスペースは、後で (またはその逆で) AWS 所有キーを使用するように変換することはできません。

> **Note**
>
> Amazon Managed Service for Prometheus は、 AWS 所有キーを使用して保管時の暗号化を自動的に有効にし、データを無料で保護します。
> ただし、カスタマーマネージドキーの使用には AWS KMS 料金が適用されます。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。

詳細については AWS KMS、「 とは」を参照してください AWS Key Management Service。

> **Note**
>
> カスタマーマネージドキーで作成されたワークスペースは、取り込み用にAWS マネージドコレクターを使用することはできません。

Amazon Managed Service for Prometheus が で許可を使用する方法 AWS KMS

Amazon Managed Service for Prometheus には、カスタマーマネージドキーを使用するための許可が 3 つ必要です。

カスタマーマネージドキーで暗号化された Amazon Managed Service for Prometheus ワークスペースを作成すると、Amazon Managed Service for Prometheus は CreateGrant リクエストを送信することで、ユーザーに代わって 3 つの許可を作成します AWS KMS。の許可 AWS KMS は、ユーザーに代わって直接呼び出されない場合でも (Amazon EKS クラスターからスクレイピングされたメトリクスデータを保存する場合などに)、Amazon Managed Service for Prometheus にアカウントの KMS キーへのアクセスを許可するために使用されます。

Amazon Managed Service for Prometheus は、以下の内部オペレーションのためにユーザーのカスタマーマネージドキーを使用する許可を必要とします。

- DescribeKey リクエストを に送信 AWS KMS して、ワークスペースの作成時に指定された対称カスタマーマネージド KMS キーが有効であることを確認します。
- GenerateDataKey リクエストを に送信 AWS KMS して、カスタマーマネージドキーによって暗号化されたデータキーを生成します。
- に Decrypt リクエストを送信 AWS KMS して、暗号化されたデータキーを復号し、データの暗号化に使用できるようにします。

Amazon Managed Service for Prometheus は、Amazon Managed Service for Prometheus がユーザーに代わって AWS KMS キーを使用できるようにする 3 つの許可をキーに作成します。キーポリシーを変更するか、キーを無効にするか、または許可を取り消すことで、キーへのアクセスを削除できます。これらのアクションを実行する前に、その結果を理解しておく必要があります。これにより、ワークスペース内のデータが失われる可能性があります。

何らかの方法で許可へのアクセスを削除すると、Amazon Managed Service for Prometheus は、カスタマーマネージドキーによって暗号化されたすべてのデータにアクセスできなくなり、ワークスペースに送信された新しいデータを保存することもできなくなります。これにより、そのデータに依存するオペレーションが影響を受けます。ワークスペースに送信された新しいデータにはアクセスできなくなり、永久に失われる可能性があります。

⚠ Warning

- キーを無効にするか、キーポリシーで Amazon Managed Service for Prometheus へのアクセスを削除すると、ワークスペースデータにはアクセスできなくなります。ワークスペースに送信される新しいデータにはアクセスできなくなり、永久に失われる可能性があります。

  Amazon Managed Service for Prometheus のキーへのアクセスを復元することで、ワークスペースデータにアクセスできるようになり、新しいデータの受信を再開できます。
- 許可を取り消すと、再作成することはできず、ワークスペース内のデータは永久に失われます。

ステップ 1：カスタマーマネージドキーを作成する

対称カスタマーマネージドキーは AWS マネジメントコンソール、 または AWS KMS APIs を使用して作成できます。以下に説明するように、ポリシーを通じて正しいアクセスを提供している限り、キーは Amazon Managed Service for Prometheus ワークスペースと同じアカウントにある必要はありません。

対称カスタマーマネージドキーを作成するには

AWS Key Management Service デベロッパーガイド にある 対称カスタマーマネージドキーの作成 ステップに従います。

キーポリシー

キーポリシーは、カスタマーマネージドキーへのアクセスを制御します。すべてのカスタマーマネージドキーには、キーポリシーが 1 つだけ必要です。このポリシーには、そのキーを使用できるユーザーとその使用方法を決定するステートメントが含まれています。カスタマーマネージドキーを作成する際に、キーポリシーを指定することができます。詳細については、AWS Key Management Service デベロッパーガイド の「カスタマーマネージドキーへのアクセスの管理」を参照してください。

Amazon Managed Service for Prometheus でカスタマーマネージドキーを使用するには、キーポリシーで次の API オペレーションを許可する必要があります。

- kms:CreateGrant - カスタマーマネージドキーに許可を追加します。指定された KMS キーへのアクセスを付与します。これにより、Amazon Managed Service for Prometheus が必要とする許可オペレーションへのアクセスを許可します。詳細については、「AWS Key Management Service デベロッパーガイド」の「許可の使用」を参照してください。

  これにより、Amazon Managed Service for Prometheus は以下を実行できるようになります。

  - GenerateDataKey を呼び出して、暗号化されたデータキーを生成して保存します。データキーは暗号化にすぐには使用されないからです。
  - Decrypt を呼び出して、保存された暗号化データキーを使用して暗号化データにアクセスします。
- kms:DescribeKey — カスタマーマネージドキーの詳細を指定し、Amazon Managed Service for Prometheus がキーを検証できるようにします。

Amazon Managed Service for Prometheus に追加できるポリシーステートメントの例を以下に示します。

```
 "Statement" : [
   {
     "Sid" : "Allow access to Amazon Managed Service for Prometheus principal within
your account",
     "Effect" : "Allow",
     "Principal" : {
       "AWS" : "*"
     },
     "Action" : [
       "kms:DescribeKey",
       "kms:CreateGrant",
       "kms:GenerateDataKey",
       "kms:Decrypt"
```

```
      ],
      "Resource" : "*",
      "Condition" : {
        "StringEquals" : {
          "kms:ViaService" : "aps.region.amazonaws.com",
          "kms:CallerAccount" : "111122223333"
        }
    },
    {
      "Sid": "Allow access for key administrators - not required for Amazon Managed
Service for Prometheus",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
       },
      "Action" : [
        "kms:*"
       ],
      "Resource": "arn:aws:kms:region:111122223333:key/key_ID"
    },
    <other statements needed for other non-Amazon Managed Service for Prometheus
scenarios>
 ]
```

- ポリシーでの許可の指定に関する詳細については、「AWS Key Management Service デベロッパーガイド」を参照してください。
- キーアクセスのトラブルシューティングに関する詳細については、「AWS Key Management Service デベロッパーガイド」を参照してください。

ステップ 2: Amazon Managed Service for Prometheus のカスタマーマネージドキーを指定する

ワークスペースを作成するときは、Amazon Managed Service for Prometheus がワークスペースに保存されているデータを暗号化するために使用する KMS キー ARN を入力して、カスタマーマネージドキーを指定できます。

ステップ 3: Amazon Managed Grafana などの他のサービスからデータにアクセスする

このステップはオプションです。別のサービスから Amazon Managed Service for Prometheus データにアクセスする必要がある場合にのみ使用します。

暗号化されたデータには、 AWS KMS キーを使用するためのアクセス権限がない限り、他の サービスからアクセスすることはできません。例えば、Amazon Managed Grafana を使用してデータに対するダッシュボードやアラートを作成する場合は、Amazon Managed Grafana にキーへのアクセス権を付与する必要があります。

Amazon Managed Grafana にカスタマーマネージドキーへのアクセス権を付与するには

1. Amazon Managed Grafana ワークスペースリストで、Amazon Managed Service for Prometheus へのアクセスを許可するワークスペースの名前を選択します。Amazon Managed Grafana ワークスペースの概要情報が表示されます。
2. ワークスペースで使用している IAM ロールの名前を書き留めます。名前の形式は `AmazonGrafanaServiceRole-<unique-id>` です。コンソールには、ロールの完全な ARN が表示されます。この名前は、後のステップで AWS KMS コンソールで指定します。
3. AWS KMS カスタマーマネージドキーリストで、Amazon Managed Service for Prometheus ワークスペースの作成時に使用したカスタマーマネージドキーを選択します。キー設定の詳細ページが開きます。
4. [キーユーザー] の横にある [追加] ボタンを選択します。
5. 名前のリストから、上で書き留めた Amazon Managed Grafana IAM ロールを選択します。簡単に見つけるには、名前で検索することもできます。
6. [追加] を選択して、IAM ロールをキーユーザーのリストに追加します。

これで、Amazon Managed Grafana ワークスペースから Amazon Managed Service for Prometheus ワークスペースのデータにアクセスできるようになりました。他のユーザーやロールをキーユーザーに追加して、他のサービスからワークスペースにアクセスできるようにすることができます。

Amazon Managed Service for Prometheus 暗号化コンテキスト

暗号化コンテキストは、データに関する追加のコンテキスト情報が含まれたキーバリューペアのオプションのセットです。

AWS KMS は、追加の認証済みデータとして暗号化コンテキストを使用して、認証済み暗号化をサポートします。データを暗号化するリクエストに暗号化コンテキストを含めると、 は暗号化コンテ

キストを暗号化されたデータに AWS KMS バインドします。データを復号化するには、そのリクエストに (暗号化時と) 同じ暗号化コンテキストを含めます。

Amazon Managed Service for Prometheus 暗号化コンテキスト

Amazon Managed Service for Prometheus は、すべての AWS KMS 暗号化オペレーションで同じ暗号化コンテキストを使用します。キーは `aws:amp:arn`で、値はワークスペースの Amazon リソースネーム (ARN) です。

Example

```
"encryptionContext": {
    "aws:aps:arn": "arn:aws:aps:us-west-2:111122223333:workspace/ws-sample-1234-
abcd-56ef-7890abcd12ef"
}
```

モニタリングに暗号化コンテキストを使用する

対称カスタマーマネージドキーを使用してワークスペースデータを暗号化する場合は、監査レコードとログで暗号化コンテキストを使用して、カスタマーマネージドキーがどのように使用されているかを特定することもできます。暗号化コンテキストは、 AWS CloudTrail または Amazon CloudWatch Logs によって生成されたログにも表示されます。

暗号化コンテキストを使用してカスタマーマネージドキーへのアクセスを制御する

対称カスタマーマネージドキー (CMK) へのアクセスを制御するための `conditions` として、キーポリシーと IAM ポリシー内の暗号化コンテキストを使用することができます。グラントに暗号化コンテキストの制約を使用することもできます。

Amazon Managed Service for Prometheus は、許可で暗号化コンテキスト制約を使用して、アカウントまたはリージョン内のカスタマーマネージドキーへのアクセスを制御します。グラントの制約では、指定された暗号化コンテキストの使用をグラントが許可するオペレーションが必要です。

Example

次に、特定の暗号化コンテキストのカスタマーマネージドキーへのアクセスを付与するキーポリシーステートメントの例を示します。このポリシーステートメントの条件では、暗号化コンテキストを指定する暗号化コンテキスト制約がグラントに必要です。

```
{
```

```
    "Sid": "Enable DescribeKey",
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/ExampleReadOnlyRole"
     },
     "Action": "kms:DescribeKey",
     "Resource": "*"
},
{
     "Sid": "Enable CreateGrant",
     "Effect": "Allow",
     "Principal": {
         "AWS": "arn:aws:iam::111122223333:role/ExampleReadOnlyRole"
     },
     "Action": "kms:CreateGrant",
     "Resource": "*",
     "Condition": {
         "StringEquals": {
             "kms:EncryptionContext:aws:aps:arn": "arn:aws:aps:us-
west-2:111122223333:workspace/ws-sample-1234-abcd-56ef-7890abcd12ef"
          }
     }
}
```

Amazon Managed Service for Prometheus の暗号化キーを監視

Amazon Managed Service for Prometheus ワークスペースで AWS KMS カスタマーマネージドキーを使用する場合、 AWS CloudTrailまたは Amazon CloudWatch Logs を使用して、Amazon Managed Service for Prometheus が送信するリクエストを追跡できます AWS KMS。

次の例はCreateGrant、カスタマーマネージドキーによって暗号化されたデータにアクセスDescribeKeyするために Amazon Managed Service for Prometheus によって呼び出される KMS オペレーションをモニタリングするための Decrypt、、、および GenerateDataKeyの AWS CloudTrail イベントです。

CreateGrant

AWS KMS カスタマーマネージドキーを使用してワークスペースを暗号化すると、Amazon Managed Service for Prometheus はユーザーに代わって 3 つのCreateGrantリクエストを送信して、指定した KMS キーにアクセスします。Amazon Managed Service for Prometheus が作成する許可は、 AWS KMS カスタマーマネージドキーに関連付けられたリソースに固有のものです。

以下のイベント例では CreateGrant オペレーションを記録しています。


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "TESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE-KEY-ID1",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "TESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-04-22T17:02:00Z"
            }
        },
        "invokedBy": "aps.amazonaws.com"
    },
    "eventTime": "2021-04-22T17:07:02Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "172.12.34.56",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "retiringPrincipal": "aps.region.amazonaws.com",
        "operations": [
            "GenerateDataKey",
            "Decrypt",
            "DescribeKey"
        ],
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "granteePrincipal": "aps.region.amazonaws.com"
    },
    "responseElements": {
```

```
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "111122223333"
}
```

GenerateDataKey

ワークスペースの AWS KMS カスタマーマネージドキーを有効にすると、Amazon Managed Service for Prometheus は一意のキーを作成します。リソースの AWS KMSカスタマーマネージドキー AWS KMS を指定するGenerateDataKeyリクエストを に送信します。

次に、GenerateDataKey オペレーションを記録するイベントの例を示します。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "aps.amazonaws.com"
    },
    "eventTime": "2021-04-22T17:07:02Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "172.12.34.56",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "encryptionContext": {
```

```
            "aws:aps:arn": "arn:aws:aps:us-west-2:111122223333:workspace/ws-
sample-1234-abcd-56ef-7890abcd12ef"
        },
        "keySpec": "AES_256",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "111122223333",
    "sharedEventID": "57f5dbee-16da-413e-979f-2c4c6663475e"
}
```

Decrypt

暗号化されたワークスペースでクエリが生成されると、Amazon Managed Service for Prometheus は `Decrypt` オペレーションを呼び出し、保存されている暗号化されたデータキーを使用して暗号化されたデータにアクセスします。

次に、`Decrypt` オペレーションを記録するイベントの例を示します。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "aps.amazonaws.com"
    },
    "eventTime": "2021-04-22T17:10:51Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
```

```
    "awsRegion": "us-west-2",
    "sourceIPAddress": "172.12.34.56",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "encryptionContext": {
            "aws:aps:arn": "arn:aws:aps:us-west-2:111122223333:workspace/ws-
sample-1234-abcd-56ef-7890abcd12ef"
        },
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "111122223333",
    "sharedEventID": "dc129381-1d94-49bd-b522-f56a3482d088"
}
```

DescribeKey

Amazon Managed Service for Prometheus は、DescribeKey オペレーションを使用して、ワークスペースに関連付けられている AWS KMS カスタマーマネージドキーがアカウントとリージョンに存在するかどうかを確認します。

以下のイベント例では、DescribeKey オペレーションを記録しています。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
```

```
        "principalId": "TESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE-KEY-ID1",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "TESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-04-22T17:02:00Z"
            }
        },
        "invokedBy": "aps.amazonaws.com"
    },
    "eventTime": "2021-04-22T17:07:02Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DescribeKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "172.12.34.56",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "keyId": "00dd0db0-0000-0000-ac00-b0c000SAMPLE"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
```

```
    "recipientAccountId": "111122223333"
}
```

詳細情報

次のリソースは、保管時のデータ暗号化についての詳細を説明しています。

- AWS Key Management Service 基本概念の詳細については、「AWS Key Management Service デベロッパーガイド」を参照してください。
- のセキュリティのベストプラクティスの詳細については、 AWS Key Management Service「 AWS Key Management Service デベロッパーガイド」を参照してください。

Amazon Managed Service for Prometheus の Identity and Access Management

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 のサービス 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 Amazon Managed Service for Prometheus リソースの使用を認可する (アクセス許可を付与する) かを制御します。IAM は、追加料金なしで使用できる AWS のサービス です。

トピック

- オーディエンス
- アイデンティティを使用した認証
- ポリシーを使用したアクセスの管理
- Amazon Managed Service for Prometheus と IAM の連携
- Amazon Managed Service for Prometheus のアイデンティティベースのポリシーの例
- Amazon Managed Service for Prometheus のアイデンティティとアクセスに関するトラブルシューティング

オーディエンス

AWS Identity and Access Management (IAM) の使用方法は、ロールによって異なります。

- サービスユーザー - 機能にアクセスできない場合は、管理者にアクセス許可をリクエストします (「Amazon Managed Service for Prometheus のアイデンティティとアクセスに関するトラブルシューティング」を参照)。
- サービス管理者 - ユーザーアクセスを決定し、アクセス許可リクエストを送信します (「Amazon Managed Service for Prometheus と IAM の連携」を参照)
- IAM 管理者 - アクセスを管理するためのポリシーを作成します (「Amazon Managed Service for Prometheus のアイデンティティベースのポリシーの例」を参照)

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。、IAM ユーザー 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で認証される必要があります。

AWS IAM アイデンティティセンター (IAM Identity Center)、シングルサインオン認証、Google/Facebook 認証情報などの ID ソースからの認証情報を使用して、フェデレーティッド ID としてサインインできます。サインインの詳細については、「AWS サインイン ユーザーガイド」の「AWS アカウントにサインインする方法」を参照してください。

プログラムによるアクセスの場合、 は SDK と CLI AWS を提供してリクエストを暗号化して署名します。詳細については、「IAM ユーザーガイド」の「API リクエストに対するAWS 署名バージョン 4」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、まず、すべての AWS のサービス および リソースへの完全なアクセス権を持つ AWS アカウント root ユーザーと呼ばれる 1 つのサインインアイデンティティから始めます。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。ルートユーザー認証情報を必要とするタスクについては、「IAM ユーザーガイド」の「ルートユーザー認証情報が必要なタスク」を参照してください。

フェデレーテッドアイデンティティ

ベストプラクティスとして、人間のユーザーが一時的な認証情報 AWS のサービス を使用して にアクセスするには、ID プロバイダーとのフェデレーションを使用する必要があります。

フェデレーティッド ID は、エンタープライズディレクトリ、ウェブ ID プロバイダー、または ID Directory Service ソースの認証情報 AWS のサービス を使用して にアクセスするユーザーです。フェデレーテッドアイデンティティは、一時的な認証情報を提供するロールを引き受けます。

アクセスを一元管理する場合は、 AWS IAM アイデンティティセンターをお勧めします。詳細については、「AWS IAM アイデンティティセンター ユーザーガイド」の「IAM アイデンティティセンターとは」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、特定の個人やアプリケーションに対する特定のアクセス許可を持つアイデンティティです。長期認証情報を持つ IAM ユーザーの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。詳細については、IAM ユーザーガイドの「ID プロバイダーとのフェデレーションを使用して にアクセスすることを人間 AWS のユーザーに要求する」を参照してください。

IAM グループは、IAM ユーザーの集合を指定し、大量のユーザーに対するアクセス許可の管理を容易にします。詳細については、「IAM ユーザーガイド」の「IAM ユーザーに関するユースケース」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を持つアイデンティであり、一時的な認証情報を提供します。ユーザーから IAM ロール (コンソール) に切り替えるか、 または API オペレーションを呼び出すことで、ロールを引き受けることができます。 AWS CLI AWS 詳細については、「IAM ユーザーガイド」の「ロールを引き受けるための各種方法」を参照してください。

IAM ロールは、フェデレーションユーザーアクセス、一時的な IAM ユーザーのアクセス許可、クロスアカウントアクセス、クロスサービスアクセス、および Amazon EC2 で実行するアプリケーションに役立ちます。詳細については、IAM ユーザーガイド の IAM でのクロスアカウントリソースアクセス 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、ID AWS またはリソースにアタッチします。ポリシーは、ID またはリソースに関連付けられたときにアクセス許可を定義します。 は、プリンシパル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。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詳細については、「IAM ユーザーガイド」の「JSON ポリシー概要」を参照してください。

管理者は、ポリシーを使用して、どのプリンシパルがどのリソースに対して、どのような条件でアクションを実行できるかを定義することで、誰が何にアクセスできるかを指定します。

デフォルトでは、ユーザーやロールにアクセス許可はありません。IAM 管理者は IAM ポリシーを作成してロールに追加し、このロールをユーザーが引き受けられるようにします。IAM ポリシーは、オペレーションの実行方法を問わず、アクセス許可を定義します。

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は、アイデンティティ (ユーザー、グループ、またはロール) 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、アイデンティティがどのリソースに対してどのような条件下でどのようなアクションを実行できるかを制御します。アイデンティティベースポリシーの作成方法については、IAM ユーザーガイド の カスタマー管理ポリシーでカスタム IAM アクセス許可を定義する 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、インラインポリシー (単一の ID に直接埋め込む) または管理ポリシー (複数の ID にアタッチされたスタンドアロンポリシー) にすることができます。管理ポリシーとインラインポリシーのいずれ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いずれかを選択する」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、リソースに添付する JSON ポリシードキュメントです。例としては、IAM ロール信頼ポリシーや 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制御できます。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。

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では、IAM の AWS マネージドポリシーを使用できません。

その他のポリシータイプ

AWS は、より一般的なポリシータイプによって付与されるアクセス許可の最大数を設定できる追加のポリシータイプをサポートしています。

- アクセス許可の境界 – アイデンティティベースのポリシーで IAM エンティティに付与することのできるアクセス許可の数の上限を設定します。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 (SCP) - AWS Organizations内の組織または組織単位の最大のアクセス許可を指定します。詳細については、「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。

- リソースコントロールポリシー (RCP) – は、アカウント内のリソースで利用できる最大数のアクセス許可を定義します。詳細については、「AWS Organizations ユーザーガイド」の「リソースコントロールポリシー (RCP)」を参照してください。
- セッションポリシー – ロールまたはフェデレーションユーザーの一時セッションを作成する際にパラメータとして渡される高度なポリシーです。詳細については、「IAM ユーザーガイド」の「セッションポリシー」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アクセス許可を理解するのがさらに難しくなります。が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、「IAM ユーザーガイド」の「ポリシー評価ロジック」を参照してください。

Amazon Managed Service for Prometheus と IAM の連携

IAM を使用して Amazon Managed Service for Prometheus へのアクセスを管理する前に、Amazon Managed Service for Prometheus で利用できる IAM の機能を理解しておく必要があります。

Amazon Managed Service for Prometheus で使用できる IAM 機能

IAM 機能	Amazon Managed Service for Prometheus でのサポート
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	はい
ポリシーアクション	あり
ポリシーリソース	あり
ポリシー条件キー	いいえ
ACL	なし
ABAC (ポリシー内のタグ)	あり
一時的な認証情報	あり

IAM 機能	Amazon Managed Service for Prometheus でのサポート
転送アクセスセッション (FAS)	いいえ
サービスロール	いいえ
サービスリンクロール	はい

Amazon Managed Service for Prometheus 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については、IAM ユーザーガイドのAWS 「IAM と連携する のサービス」を参照してください。

Amazon Managed Service for Prometheus 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

アイデンティティベースポリシーは、IAM ユーザー、ユーザーグループ、ロールなど、アイデンティティにアタッチできる JSON 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ポリシーの作成方法については、「IAM ユーザーガイド」の「カスタマー管理ポリシーでカスタム IAM アクセス許可を定義する」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、許可または拒否するアクションとリソース、およびアクションを許可または拒否する条件を指定できます。JSON ポリシーで使用できるすべての要素については、「IAM ユーザーガイド」の「IAM JSON ポリシーの要素のリファレンス」を参照してください。

Amazon Managed Service for Prometheus のアイデンティティベースのポリシーの例

Amazon Managed Service for Prometheus のアイデンティティベースのポリシーの例については、「Amazon Managed Service for Prometheus のアイデンティティベースのポリシーの例」を参照してください。

Amazon Managed Service for Prometheus 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: あり

リソースベースのポリシーは、リソースに添付する 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には例として、IAM ロールの信頼ポリシーや Amazon S3 バケットポリシーがあげ

られます。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、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 のサービス。

クロスアカウントアクセスを有効にするには、全体のアカウント、または別のアカウントの IAM エンティティを、リソースベースのポリシーのプリンシパルとして指定します。詳細については、IAM ユーザーガイドのIAM でのクロスアカウントリソースアクセスを参照してください。

Amazon Managed Service for Prometheus 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 JSON AWS ポリシーを使用して、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` 要素にはポリシー内の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が記述されます。ポリシーにアクションを含めることで、関連するオペレーションを実行するためのアクセス許可を付与します。

Amazon Managed Service for Prometheus アクションの一覧については、「サービス認可リファレンス」の「Amazon Managed Service for Prometheus によって定義されるアクション」を参照してください。

Amazon Managed Service for Prometheus のポリシーアクションでは、アクションの前に次のプレフィックスが使用されます。

```
aps
```

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": [
      "aps:action1",
      "aps:action2"
         ]
```

Amazon Managed Service for Prometheus のアイデンティティベースのポリシーの例については、「Amazon Managed Service for Prometheus のアイデンティティベースのポリシーの例」を参照してください。

Amazon Managed Service for Prometheus 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 JSON AWS ポリシーを使用して、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。

`Resource` JSON ポリシー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。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。リソースレベルのアクセス許可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": "*"
```

Amazon Managed Service for Prometheus リソースタイプとその ARN の一覧については、「サービス認可リファレンス」の「Amazon Managed Service for Prometheus で定義されるリソース」を参照してください。どのアクションで各リソースの ARN を指定できるかについては、「Amazon Managed Service for Prometheus で定義されるアクション」を参照してください。

Amazon Managed Service for Prometheus のアイデンティティベースのポリシーの例については、「Amazon Managed Service for Prometheus のアイデンティティベースのポリシーの例」を参照してください。

Amazon Managed Service for Prometheus のポリシー条件キー

サービス固有のポリシー条件キーへのサポート: なし

管理者は JSON AWS ポリシーを使用して、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` 要素は、定義された基準に基づいてステートメントが実行される時期を指定します。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して、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。すべての AWS グローバル条件キーを確認するには、「IAM ユーザーガイド」のAWS 「グローバル条件コンテキストキー」を参照してください。

Amazon Managed Service for Prometheus 条件キーの一覧については、「サービス認可リファレンス」の「Amazon Managed Service for Prometheus によって定義される条件キー」を参照してください。どのアクションおよびリソースで条件キーを使用できるかについては、「Amazon Managed Service for Prometheus で定義されるアクション」を参照してください。

Amazon Managed Service for Prometheus のアイデンティティベースのポリシーの例については、「Amazon Managed Service for Prometheus のアイデンティティベースのポリシーの例」を参照してください。

Amazon Managed Service for Prometheus のアクセスコントロールリスト (ACL)

ACL のサポート: なし

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするためのアクセス許可を持つかを制御します。ACL 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は使用しません。

Amazon Managed Service for Prometheus での属性ベースのアクセス制御 (ABAC)

ABAC (ポリシー内のタグ) のサポート: あり

属性ベースのアクセス制御 (ABAC) は、タグと呼ばれる属性に基づいてアクセス許可を定義する認可戦略です。IAM エンティティと AWS リソースにタグをアタッチし、プリンシパルのタグがリソースのタグと一致するときにオペレーションを許可するように ABAC ポリシーを設計できます。

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。

ABAC の詳細については、「IAM ユーザーガイド」の「ABAC 認可でアクセス許可を定義する」を参照してください。ABAC をセットアップする手順を説明するチュートリアルについては、「IAM 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」を参照してください。

Amazon Managed Service for Prometheus 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

一時的な認証情報は、 AWS リソースへの短期的なアクセスを提供し、フェデレーションまたはスイッチロールの使用時に自動的に作成されます。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 AWS することをお勧めします。詳細については、「IAM ユーザーガイド」の「IAM の一時的な認証情報」および「AWS のサービス と IAM との連携」を参照してください。

Amazon Managed Service for Prometheus の転送アクセスセッション

転送アクセスセッション (FAS) のサポート: なし

転送アクセスセッション (FAS) は、 を呼び出すプリンシパルのアクセス許可と AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストをリクエストする を使用します。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。

Amazon Managed Service for Prometheus のサービスロール

サービスロールのサポート: なし

サービスロールとは、サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、IAM ユーザーガイドの AWS のサービスに許可を委任するロールを作成する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、Amazon Managed Service for Prometheus の機能が破損する可能性があります。Amazon Managed Service for Prometheus 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。

Amazon Managed Service for Prometheus のサービスにリンクされたロール

サービスリンクロールのサポート: あり

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 のサービス。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、サービスリンク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 Managed Service for Prometheus サービスリンクロールの作成または管理の詳細については、「Amazon Managed Service for Prometheus のサービスリンクロールの使用」を参照してください。

Amazon Managed Service for Prometheus のアイデンティティベースのポリシーの例

デフォルトでは、ユーザーおよびロールには、Amazon Managed Service for Prometheus リソースを作成または変更する許可はありません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。

これらのサンプルの JSON ポリシードキュメントを使用して IAM 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、「IAM ユーザーガイド」の「IAM ポリシーを作成する (コンソール)」を参照してください。

Amazon Managed Service for Prometheus で定義されるアクションとリソースタイプ (リソースタイプごとの ARN の形式を含む) の詳細については、「サービス認可リファレンス」の「Amazon Managed Service for Prometheus 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

- ポリシーに関するベストプラクティス
- Amazon Managed Service for Prometheus コンソールの使用
- 自分の権限の表示をユーザーに許可する

ポリシーに関するベストプラクティス

アイデンティティベースのポリシーは、ユーザーのアカウントで誰かが Amazon Managed Service for Prometheus 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では、 AWS アカウントに費用が発生する場合があります。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の付与を開始するには、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することで、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、IAM ユーザーガイド の AWS マネージドポリシー または ジョブ機能のAWS マネージドポリシー を参照してください。

- 最小特権を適用する – 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 を使用して許可を適用する方法の詳細については、IAM ユーザーガイド の IAM でのポリシーとアクセス許可 を参照してください。
- IAM ポリシーで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、アクションやリソースへのアクセスを制限できます。たとえば、ポリシー条件を記述して、すべてのリクエストを SSL を使用して送信するように指定できます。条件を使用して、サービスアクションが などの特定の を通じて使用されている場合に AWS のサービス、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 CloudFormation。詳細については、IAM ユーザーガイド の IAM JSON ポリシー要素:条件 を参照してください。
- IAM アクセスアナライザー を使用して IAM ポリシーを検証し、安全で機能的な権限を確保する - IAM アクセスアナライザー は、新規および既存のポリシーを検証して、ポリシーが IAM ポリシー言語 (JSON) および IAM のベストプラクティスに準拠するようにします。IAM アクセスアナライザーは 100 を超えるポリシーチェックと実用的な推奨事項を提供し、安全で機能的なポリシーの作成をサポートします。詳細については、IAM ユーザーガイド の IAM Access Analyzer でポリシーを検証する を参照してください。
- 多要素認証 (MFA) を要求する – 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、MFA をオンにしてセキュリティを強化します。API オペレーションが呼び出されるときに MFA を必須にするには、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 を使用した安全な API アクセス を参照してください。

IAM で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 IAM でのセキュリティのベストプラクティス を参照してください。

Amazon Managed Service for Prometheus コンソールの使用

Amazon Managed Service for Prometheus コンソールにアクセスするには、最小限のアクセス許可のセットが必要です。これらのアクセス許可で、 AWS アカウント内の Amazon Managed Service for Prometheus リソースの一覧表示と詳細表示を許可する必要があります。最小限必要な許可よりも制限が厳しいアイデンティティベースのポリシーを作成すると、そのポリシーを持つエンティティ (ユーザーまたはロール) に対してコンソールが意図したとおりに機能しません。

AWS CLI または AWS API のみを呼び出すユーザーには、最小限のコンソールアクセス許可を付与する必要はありません。代わりに、実行しようとしている API オペレーションに一致するアクションのみへのアクセスを許可します。

ユーザーとロールが引き続き Amazon Managed Service for Prometheus コンソールを使用できるようにするには、Amazon Managed Service for Prometheus `ConsoleAccess`または `ReadOnly` AWS 管理ポリシーもエンティティにアタッチします。詳細については、「IAM ユーザーガイド」の「ユーザーへのアクセス許可の追加」を参照してください。

自分の権限の表示をユーザーに許可する

この例では、ユーザーアイデンティティにアタッチされたインラインおよびマネージドポリシーの表示を IAM ユーザーに許可するポリシーの作成方法を示します。このポリシーには、コンソールで、または AWS CLI または AWS API を使用してプログラムでこのアクションを実行するアクセス許可が含まれています。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
```

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

Amazon Managed Service for Prometheus のアイデンティティとアクセスに関するトラブルシューティング

以下の情報は、Amazon Managed Service for Prometheus と IAM の使用時に発生する可能性のある一般的な問題の診断と修復に役立ちます。

トピック

- Amazon Managed Service for Prometheus でアクションを実行する権限がない
- iam:PassRole を実行する権限がありません
- AWS アカウント以外のユーザーに Amazon Managed Service for Prometheus リソースへのアクセスを許可したい

Amazon Managed Service for Prometheus でアクションを実行する権限がない

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、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。

次のエラー例は、mateojackson IAM ユーザーがコンソールを使用して、ある *my-example-widget* リソースに関する詳細情報を表示しようとしたことを想定して、その際に必要な aps:*GetWidget* アクセス許可を持っていない場合に発生するものです。

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

この場合、aps:*GetWidget* アクションを使用して *my-example-widget* リソースへのアクセスを許可するように、mateojackson ユーザーのポリシーを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:PassRole を実行する権限がありません

`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 Amazon Managed Service for Prometheus にロールを渡せるようにする必要があります。

一部の AWS のサービス 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アクセス許可が必要です。

次の例は、`marymajor` という IAM ユーザーがコンソールを使用して Amazon Managed Service for Prometheus でアクションを実行しようとした場合に発生するエラーを示しています。ただし、このアクションをサービスで実行するには、サービスロールから付与されたアクセス許可が必要です。Mary には、ロールをサービスに渡すアクセス許可がありません。

```
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。サインイン資格情報を提供した担当者が管理者です。

AWS アカウント以外のユーザーに Amazon Managed Service for Prometheus リソース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 (ACL) をサポートするサービスの場合、それらのポリシーを使用して、リソースへのアクセスを付与できます。

詳細については、以下を参照してください:

- Amazon Managed Service for Prometheus 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、「Amazon Managed Service for Prometheus と IAM の連携」を参照してください。
- 所有 AWS アカウント 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、「IAM 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」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、IAM 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」を参照してください。
- ID フェデレーションを介し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 外部で認証されたユーザー (ID フェデレーション) へのアクセスの許可 を参照してください。
- クロスアカウントアクセスにおけるロールとリソースベースのポリシーの使用方法の違いについては、「IAM ユーザーガイド」の「IAM でのクロスアカウントのリソースへのアクセス」を参照してください。

IAM のアクセス許可とポリシー

Amazon Managed Service for Prometheus のアクションとデータにアクセスするには、認証情報が必要です。これらの認証情報には、アクションを実行し、クラウド AWS リソースに関する Amazon Managed Service for Prometheus データを取得するなど、リソースにアクセスするためのアクセス許可が必要です。以下のセクションでは、 AWS Identity and Access Management (IAM) と Amazon Managed Service for Prometheus を使用して、リソースにアクセスできるユーザーを制御することで、リソースを保護する方法について詳しく説明します。詳細については、「IAM でのポリシーとアクセス許可」を参照してください。

Amazon Managed Service for Prometheus のアクセス許可

可能な Amazon Managed Service for Prometheus アクションのリスト、リソースタイプ、条件キーについては、「Amazon Managed Service for Prometheus 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。

IAM ポリシーのサンプル

このセクションでは、ユーザーが作成できるセルフマネージドポリシーの例を示します。

次の IAM ポリシーでは、Amazon Managed Service for Prometheus へのフルアクセスを許可するとともに、ユーザーが Amazon EKS クラスターを検出してその詳細を確認できるようにします。

JSON

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

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "aps:*",
                "eks:DescribeCluster",
                "eks:ListClusters"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon Managed Service for Prometheus のコンプライアンス検証

AWS のサービス 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AWS のサービス プログラムによるスコープ」の「コンプライアンス」を参照して、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、「Downloading Reports in AWS Artifact」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 のサービス 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。を使用する際のコンプライアンス責任の詳細については AWS のサービス、AWS「 セキュリティドキュメント」を参照してください。

Amazon Managed Service for Prometheus の耐障害性

AWS のグローバルインフラストラクチャは、AWS リージョンとアベイラビリティーゾーンを中心として構築されています。AWSリージョンには、低レイテンシー、高いスループット、そして高度の冗長ネットワークで接続されている複数の物理的に独立および隔離されたアベイラビリティーゾーンがあります。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、耐障害性、および拡張性が優れています。

AWS リージョンと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。

AWS のグローバルインフラストラクチャに加えて、Amazon Managed Service for Prometheus にも、高可用性データのサポートなど、データの耐障害性とバックアップのニーズに対応するための機能が用意されています。

Amazon Managed Service for Prometheus 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 Managed Service for Prometheus 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 のセキュリティサービスと、AWS がインフラストラクチャをどのように保護するかについては、「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「インフラストラクチャ保護」を参照してください。

ネットワーク経由で Amazon Managed Service for Prometheus にアクセスするには、AWS が公開している API コールを使用します。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)。TLS 1.2 が必須で、TLS 1.3 をお勧めします。
- DHE (楕円ディフィー・ヘルマン鍵共有) や ECDHE (楕円曲線ディフィー・ヘルマン鍵共有) などの完全前方秘匿性 (PFS) による暗号スイート。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。

Amazon Managed Service for Prometheus のサービスリンクロールの使用

Amazon Managed Service for Prometheus は AWS Identity and Access Management 、(IAM) サービスにリンクされたロールを使用します。サービスリンクロールは、Amazon Managed Service for Prometheus に直接リンクされた特殊な IAM ロールです。サービスリンクロールは Amazon Managed Service for Prometheus によって事前に定義されており、サービスがユーザーに代わって他の AWS のサービスを呼び出すために必要な、すべてのアクセス許可が含まれています。

必要なアクセス許可を手動で追加する必要がないため、サービスリンクロールは Amazon Managed Service for Prometheus のセットアップを容易にします。サービスリンクロールのアクセス許可は

Amazon Managed Service for Prometheus が定義し、別段の定義がない限り、Amazon Managed Service for Prometheus のみがそのロールを引き受けることができます。定義される許可は信頼ポリシーと許可ポリシーに含まれており、その許可ポリシーを他の IAM エンティティにアタッチすることはできません。

EKS からメトリクスをスクレイピングするためのロールの使用

Amazon Managed Service for Prometheus マネージドコレクターを使用してメトリクスを自動的にスクレイピングする場合、AWSServiceRoleForAmazonPrometheusScraper サービスリンクロールを使用すると、必要なアクセス許可を手動で追加する必要がないため、マネージドコレクターを簡単にセットアップできます。Amazon Managed Service for Prometheus がアクセス許可を定義し、Amazon Managed Service for Prometheus のみが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をサポートする他のサービスについては、「IAM と連携するAWS のサービス」を参照し、サービスリンクロール列内ではいと表記されたサービスを確認してください。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、リンクで [はい] を選択します。

Amazon Managed Service for Prometheus 用のサービスリンクロールのアクセス許可

Amazon Managed Service for Prometheus は、AWSServiceRoleForAmazonPrometheusScraper というプレフィックスが付いた名前のサービスリンクロールを使用して、Amazon Managed Service for Prometheus が Amazon EKS クラスター内のメトリクスを自動的にスクレイピングできるようにします。

AWSServiceRoleForAmazonPrometheusScraper サービスリンクロールは、以下のサービスを信頼してロールを引き受けます。

- `scraper.aps.amazonaws.com`

このロールの AmazonPrometheusScraperServiceRolePolicy というアクセス許可ポリシーは、指定したリソースに対して以下のアクションを実行することを Amazon Managed Service for Prometheus に許可します。

- Amazon EKS クラスターを含むネットワークに接続するためのネットワーク設定を準備し、変更します。
- Amazon EKS クラスターからメトリクスを読み取り、Amazon Managed Service for Prometheus ワークスペースにメトリクスを書き込みます。

ユーザー、グループ、ロールなどがサービスリンクロールを作成できるようにするには、アクセス許可を設定する必要があります。詳細についてはIAM ユーザーガイド の「サービスにリンクされた役割のアクセス許可」を参照してください。

Amazon Managed Service for Prometheus のサービスリンクロールの作成

サービスリンクロールを手動で作成する必要はありません。、 AWS マネジメントコンソール、または AWS CLI AWS API で Amazon EKS または Amazon Managed Service for Prometheus を使用してマネージドコレクターインスタンスを作成すると、Amazon Managed Service for Prometheus によってサービスにリンクされたロールが作成されます。

> **Important**
>
> このサービスリンクロールはこのロールでサポートされている機能を使用する別のサービスでアクションが完了した場合にアカウントに表示されます。詳細については、「新しいロールが AWS アカウント」を参照してください。

このサービスリンクロールを削除した後で再度作成する必要が生じた場合は同じ方法でアカウントにロールを再作成できます。Amazon EKS または Amazon Managed Service for Prometheus を使用してマネージドコレクターインスタンスを作成する際に、Amazon Managed Service for Prometheus によってサービスリンクロールが再作成されます。

Amazon Managed Service for Prometheus のサービスリンクロール

Amazon Managed Service for Prometheus では、AWSServiceRoleForAmazonPrometheusScraper サービスリンクロールを編集することはできません。サービスリンクロールの作成後は、さまざまなエンティティがロールを参照する可能性があるため、ロール名を変更することはできません。ただし、IAM を使用してロールの説明を編集することはできます。詳細については、「IAM ユーザーガイド」の「サービスリンクロールの編集」を参照してください。

Amazon Managed Service for Prometheus のサービスリンクロールの削除

AWSServiceRoleForAmazonPrometheusScraper の手動削除は不要です。 AWS マネジメントコンソール、、または AWS API のロールに関連付けられているすべてのマネージドコレクターインスタンスを削除すると、Amazon Managed Service for Prometheus はリソースをクリーンアップし AWS CLI、サービスにリンクされたロールを削除します。

Amazon Managed Service for Prometheus のサービスリンクロールがサポートされているリージョン

Amazon Managed Service for Prometheus は、サービスが利用可能なすべてのリージョンでサービスリンクロールの使用をサポートします。詳細については、「サポート対象のリージョン」を参照してください。

を使用した Amazon Managed Service for Prometheus API コールのログ記録 AWS CloudTrail

Amazon Managed Service for Prometheus は、ユーザー、ロール、または AWS のサービスで実行したアクションを記録するサービスである AWS CloudTrail と統合されています。CloudTrail は、Amazon Managed Service for Prometheus のすべての API コールをイベントとしてキャプチャします。キャプチャされるコールには、Amazon Managed Service for Prometheus コンソールからのコールと、Amazon Managed Service for Prometheus API オペレーションへのコードコールが含まれます。CloudTrail で収集した情報を使用して、Amazon Managed Service for Prometheus に対して行われたリクエスト、リクエスト元の IP アドレス、リクエストの日時、その他の詳細を確認できます。

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。

- ルートユーザーまたはユーザー認証情報のどちらを使用してリクエストが送信されたか。
- リクエストが IAM Identity Center ユーザーに代わっ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 のサービスによって送信されたかどうか。

CloudTrail は、アカウントを作成する AWS アカウント と でアクティブになり、CloudTrail イベント履歴に自動的にアクセスできます。CloudTrail の [イベント履歴] では、 AWS リージョンで過去 90 日間に記録された管理イベントの表示、検索、およびダウンロードが可能で、変更不可能な記録を確認できます。詳細については、「AWS CloudTrail ユーザーガイド」の「CloudTrail イベント履歴の使用」を参照してください。[イベント履歴] の閲覧には CloudTrail の料金はかかりません。

AWS アカウント 過去 90 日間のイベントの継続的な記録については、証跡または CloudTrail Lake イベントデータストアを作成します。

CloudTrail 証跡

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。を使用して作成されたすべての証跡 AWS マネジメントコンソール はマルチリージョンです。 AWS CLIを使用する際は、単一リージョンまたは複数リージョンの証跡を作成できます。アカウント AWS リージョン 内のすべての でアクティビティをキャプチャするため、マルチリージョン証跡を作成することをお勧めします。単一リージョンの証跡を作成する場合、証跡の AWS リージョンに記録されたイベントのみを表示できます。証跡の詳細については、「AWS CloudTrail ユーザーガイド」の「AWS アカウントの証跡の作成」および「組織の証跡の作成」を参照してください。

証跡を作成すると、進行中の管理イベントのコピーを 1 つ無料で CloudTrail から Amazon S3 バケットに配信できますが、Amazon S3 ストレージには料金がかかります。CloudTrail の料金の詳細については、「AWS CloudTrail の料金」を参照してください。Amazon S3 の料金に関する詳細については、「Amazon S3 の料金」を参照してください。

CloudTrail Lake イベントデータストア

[CloudTrail Lake] を使用すると、イベントに対して SQL ベースのクエリを実行できます。CloudTrail Lake は、行ベースの JSON 形式の既存のイベントを Apache ORC 形式に変換します。ORC は、データを高速に取得するために最適化された単票ストレージ形式です。イベントは、イベントデータストアに集約されます。イベントデータストアは、高度なイベントセレクタを適用することによって選択する条件に基づいた、イベントのイミュータブルなコレクションです。どのイベントが存続し、クエリに使用できるかは、イベントデータストアに適用するセレクタが制御します。CloudTrail Lake の詳細については、 AWS CloudTrail ユーザーガイドのAWS CloudTrail 「Lake の使用」を参照してください。

CloudTrail Lake のイベントデータストアとクエリにはコストがかかります。イベントデータストアを作成する際に、イベントデータストアに使用する料金オプションを選択します。料金オプションによって、イベントの取り込みと保存にかかる料金、および、そのイベントデータストアのデフォルトと最長の保持期間が決まります。CloudTrail の料金の詳細については、「AWS CloudTrail の料金」を参照してください。

CloudTrail での Amazon Managed Service for Prometheus 管理イベント

管理イベントは、 のリソースで実行される管理オペレーションに関する情報を提供します AWS アカウント。これらのイベントは、コントロールプレーンオペレーションとも呼ばれます。CloudTrail は、デフォルトで管理イベントをログ記録します。

Amazon Managed Service for Prometheus は、すべての Amazon Managed Service for Prometheus コントロールプレーンオペレーションを管理イベントとしてログに記録します。Amazon Managed Service for Prometheus が CloudTrail にログとして記録する Amazon Managed Service for Prometheus コントロールプレーンオペレーションのリストについては、「Amazon Managed Service for Prometheus API Reference」を参照してください。

Amazon Managed Service for Prometheus イベントの例

各イベントは任意の送信元からの単一のリクエストを表し、リクエストされた API オペレーション、オペレーションの日時、リクエストパラメータなどに関する情報を含みます。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、イベントは特定の順序で表示されません。

例: CreateWorkspace

次の例は、CreateWorkspace アクションを示す CloudTrail ログエントリです。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLE123EXAMPLE123-1234567890616",
        "arn": "arn:aws:sts::123456789012:assumed-role/Admin/admin",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Admin"
            },
            "webIdFederationData": {

            },
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2020-11-30T23:39:29Z"
            }
        }
    },
```

```
    "eventTime": "2020-11-30T23:43:21Z",
    "eventSource": "aps.amazonaws.com",
    "eventName": "CreateWorkspace",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "203.0.113.1",
    "userAgent": "aws-cli/1.11.167 Python/2.7.10 Darwin/16.7.0 botocore/1.7.25",
    "requestParameters": {
        "alias": "alias-example",
        "clientToken": "12345678-1234-abcd-1234-12345abcd1"
    },
    "responseElements": {
        "Access-Control-Expose-Headers": "x-amzn-errortype,x-amzn-requestid,x-amzn-
trace-id,x-amzn-errormessage,x-amz-apigw-id,date",
        "arn": "arn:aws:aps:us-west-2:123456789012:workspace/ws-abc123456-
abcd-1234-5678-1234567890",
        "status": {
            "statusCode": "CREATING"
        },
        "workspaceId": "ws-12345678-1234-abcd-1234-1234567890"
    },
    "requestID": "890b8639-e51f-11e7-b038-EXAMPLE",
    "eventID": "874f89fa-70fc-4798-bc00-EXAMPLE",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123456789012"
}
```

例: CreateAlertManagerDefinition

次の例は、CreateAlertManagerDefinition アクションを示す CloudTrail ログエントリです。

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLE123EXAMPLE123-1234567890616",
        "arn": "arn:aws:sts::123456789012:assumed-role/Admin/admin",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
```

```
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Admin"
            },
            "webIdFederationData": {

            },
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2021-09-23T20:20:14Z"
            }
        }
    },
    "eventTime": "2021-09-23T20:22:43Z",
    "eventSource": "aps.amazonaws.com",
    "eventName": "CreateAlertManagerDefinition",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "203.0.113.1",
    "userAgent": "Boto3/1.17.46 Python/3.6.14 Linux/4.14.238-182.422.amzn2.x86_64 exec-
env/AWS_ECS_FARGATE Botocore/1.20.46",
    "requestParameters": {
        "data":
 "YWxlcnRtYW5hZ2VyX2NvbmZpZzogfAogIGdsb2JhbDoKICAgIHNtdHBfc21hcnRob3N0OiAnbG9jYWxob3N0OjI1JwogI
        "clientToken": "12345678-1234-abcd-1234-12345abcd1",
        "workspaceId": "ws-12345678-1234-abcd-1234-1234567890"
    },
    "responseElements": {
        "Access-Control-Expose-Headers": "x-amzn-errortype,x-amzn-requestid,x-amzn-
trace-id,x-amzn-errormessage,x-amz-apigw-id,date",
        "status": {
            "statusCode": "CREATING"
        }
    },
    "requestID": "890b8639-e51f-11e7-b038-EXAMPLE",
    "eventID": "874f89fa-70fc-4798-bc00-EXAMPLE",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123456789012"
}
```

例: CreateRuleGroupsNamespace

次の例は、CreateRuleGroupsNamespace アクションを示す CloudTrail ログエントリです。


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLE123EXAMPLE123-1234567890616",
        "arn": "arn:aws:sts::123456789012:assumed-role/Admin/admin",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Admin"
            },
            "webIdFederationData": {

            },
            "attributes": {
                "creationDate": "2021-09-23T20:22:19Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2021-09-23T20:25:08Z",
    "eventSource": "aps.amazonaws.com",
    "eventName": "CreateRuleGroupsNamespace",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "34.212.33.165",
    "userAgent": "Boto3/1.17.63 Python/3.6.14 Linux/4.14.238-182.422.amzn2.x86_64 exec-
env/AWS_ECS_FARGATE Botocore/1.20.63",
    "requestParameters": {
        "data":
 "Z3JvdXBzOgogIC0gbmFtZTogdGVzdFJ1bGVHcm91cHNOYW1lc3BhY2UKICAgIHJ1bGVzOgogICAgLSBhbGVydDogdGVzd
        "clientToken": "12345678-1234-abcd-1234-12345abcd1",
        "name": "exampleRuleGroupsNamespace",
        "workspaceId": "ws-12345678-1234-abcd-1234-1234567890"
    },
    "responseElements": {
```

```
        "Access-Control-Expose-Headers": "x-amzn-errortype,x-amzn-requestid,x-amzn-
trace-id,x-amzn-errormessage,x-amz-apigw-id,date",
        "name": "exampleRuleGroupsNamespace",
        "arn": "arn:aws:aps:us-west-2:492980759322:rulegroupsnamespace/ws-
ae46a85c-1609-4c22-90a3-2148642c3b6c/exampleRuleGroupsNamespace",
        "status": {
            "statusCode": "CREATING"
        },
        "tags": {}
    },
    "requestID": "890b8639-e51f-11e7-b038-EXAMPLE",
    "eventID": "874f89fa-70fc-4798-bc00-EXAMPLE",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "123456789012"
}
```


CloudTrail レコードの内容については、「AWS CloudTrail ユーザーガイド」の「CloudTrail record contents」を参照してください。

サービスアカウントの IAM ロールの設定

サービスアカウントの IAM ロールを使用すると、IAM ロールを Kubernetes サービスアカウントに関連付けることができます。このサービスアカウントは、そのサービスアカウントを使用する任意のポッドのコンテナにアクセス AWS 許可を付与できます。詳細については、「サービスアカウントの IAM ロール」を参照してください。

サービスアカウントの IAM ロールは、サービスロールとも呼ばれます。

Amazon Managed Service for Prometheus でサービスロールを使用すると、Amazon Managed Service for Prometheus、Prometheus サーバー、Grafana サーバー間での認証と認可に必要なロールを取得するために役立ちます。

前提条件

このページの手順では、 AWS CLI と EKSCTL コマンドラインインターフェイスがインストールされている必要があります。

Amazon EKS クラスターからメトリクスを取り込むためのサービスロールの設定

サービスロールを設定して、Amazon Managed Service for Prometheus で Amazon EKS クラスター内の Prometheus サーバーからメトリクスを取り込めるようにするには、次のアクセス許可を持つアカウントにログオンする必要があります。

- `iam:CreateRole`
- `iam:CreatePolicy`
- `iam:GetRole`
- `iam:AttachRolePolicy`
- `iam:GetOpenIDConnectProvider`

Amazon Managed Service for Prometheus への取り込みのためのサービスロールを設定するには

1. `createIRSA-AMPIngest.sh` という名前のファイルを作成し、次の内容を記述します。`<my_amazon_eks_clustername>` はクラスターの名前に、`<my_prometheus_namespace>` は Prometheus 名前空間に置き換えます。

```
#!/bin/bash -e
CLUSTER_NAME=<my_amazon_eks_clustername>
SERVICE_ACCOUNT_NAMESPACE=<my_prometheus_namespace>
AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
OIDC_PROVIDER=$(aws eks describe-cluster --name $CLUSTER_NAME --query
 "cluster.identity.oidc.issuer" --output text | sed -e "s/^https:\/\///")
SERVICE_ACCOUNT_AMP_INGEST_NAME=amp-iamproxy-ingest-service-account
SERVICE_ACCOUNT_IAM_AMP_INGEST_ROLE=amp-iamproxy-ingest-role
SERVICE_ACCOUNT_IAM_AMP_INGEST_POLICY=AMPIngestPolicy
#
# Set up a trust policy designed for a specific combination of K8s service account
 and namespace to sign in from a Kubernetes cluster which hosts the OIDC Idp.
#
cat <<EOF > TrustPolicy.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
```

```
        "Federated": "arn:aws:iam::${AWS_ACCOUNT_ID}:oidc-provider/
${OIDC_PROVIDER}"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "${OIDC_PROVIDER}:sub": "system:serviceaccount:
${SERVICE_ACCOUNT_NAMESPACE}:${SERVICE_ACCOUNT_AMP_INGEST_NAME}"
        }
      }
    }
  ]
}
EOF
#
# Set up the permission policy that grants ingest (remote write) permissions for
 all AMP workspaces
#
cat <<EOF > PermissionPolicyIngest.json
{
  "Version": "2012-10-17",
   "Statement": [
       {"Effect": "Allow",
        "Action": [
           "aps:RemoteWrite",
           "aps:GetSeries",
           "aps:GetLabels",
           "aps:GetMetricMetadata"
        ],
        "Resource": "*"
      }
   ]
}
EOF

function getRoleArn() {
  OUTPUT=$(aws iam get-role --role-name $1 --query 'Role.Arn' --output text 2>&1)

  # Check for an expected exception
  if [[ $? -eq 0 ]]; then
    echo $OUTPUT
  elif [[ -n $(grep "NoSuchEntity" <<< $OUTPUT) ]]; then
    echo ""
  else
```

```
    >&2 echo $OUTPUT
    return 1
  fi
}

#
# Create the IAM Role for ingest with the above trust policy
#
SERVICE_ACCOUNT_IAM_AMP_INGEST_ROLE_ARN=$(getRoleArn
 $SERVICE_ACCOUNT_IAM_AMP_INGEST_ROLE)
if [ "$SERVICE_ACCOUNT_IAM_AMP_INGEST_ROLE_ARN" = "" ];
then
  #
  # Create the IAM role for service account
  #
  SERVICE_ACCOUNT_IAM_AMP_INGEST_ROLE_ARN=$(aws iam create-role \
  --role-name $SERVICE_ACCOUNT_IAM_AMP_INGEST_ROLE \
  --assume-role-policy-document file://TrustPolicy.json \
  --query "Role.Arn" --output text)
  #
  # Create an IAM permission policy
  #
  SERVICE_ACCOUNT_IAM_AMP_INGEST_ARN=$(aws iam create-policy --policy-name
 $SERVICE_ACCOUNT_IAM_AMP_INGEST_POLICY \
  --policy-document file://PermissionPolicyIngest.json \
  --query 'Policy.Arn' --output text)
  #
  # Attach the required IAM policies to the IAM role created above
  #
  aws iam attach-role-policy \
  --role-name $SERVICE_ACCOUNT_IAM_AMP_INGEST_ROLE \
  --policy-arn $SERVICE_ACCOUNT_IAM_AMP_INGEST_ARN
else
    echo "$SERVICE_ACCOUNT_IAM_AMP_INGEST_ROLE_ARN IAM role for ingest already
 exists"
fi
echo $SERVICE_ACCOUNT_IAM_AMP_INGEST_ROLE_ARN
#
# EKS cluster hosts an OIDC provider with a public discovery endpoint.
# Associate this IdP with AWS IAM so that the latter can validate and accept the
 OIDC tokens issued by Kubernetes to service accounts.
# Doing this with eksctl is the easier and best approach.
#
```

```
eksctl utils associate-iam-oidc-provider --cluster $CLUSTER_NAME --approve
```

2. 次のコマンドを入力して、スクリプトに必要な権限を付与します。

```
chmod +x createIRSA-AMPIngest.sh
```

3. スクリプトを実行します。

メトリクスのクエリを実行するためのサービスアカウントの IAM ロールの設定

サービスアカウントの IAM ロール (サービスロール) を設定して、Amazon Managed Service for Prometheus ワークスペースからメトリクスのクエリを実行できるようにするには、次のアクセス許可を持つアカウントにログオンする必要があります。

- `iam:CreateRole`
- `iam:CreatePolicy`
- `iam:GetRole`
- `iam:AttachRolePolicy`
- `iam:GetOpenIDConnectProvider`

Amazon Managed Service for Prometheus メトリクスのクエリを実行するためのサービスロールを設定するには

1. `createIRSA-AMPQuery.sh` という名前のファイルを作成し、次の内容を記述します。<my_amazon_eks_clustername> はクラスターの名前に置き換え、<my_prometheus_namespace> は Prometheus 名前空間に置き換えます。

```
#!/bin/bash -e
CLUSTER_NAME=<my_amazon_eks_clustername>
SERVICE_ACCOUNT_NAMESPACE=<my_prometheus_namespace>
AWS_ACCOUNT_ID=$(aws sts get-caller-identity --query "Account" --output text)
OIDC_PROVIDER=$(aws eks describe-cluster --name $CLUSTER_NAME --query
 "cluster.identity.oidc.issuer" --output text | sed -e "s/^https:\/\///")
SERVICE_ACCOUNT_AMP_QUERY_NAME=amp-iamproxy-query-service-account
SERVICE_ACCOUNT_IAM_AMP_QUERY_ROLE=amp-iamproxy-query-role
SERVICE_ACCOUNT_IAM_AMP_QUERY_POLICY=AMPQueryPolicy
#
```

```
# Setup a trust policy designed for a specific combination of K8s service account
 and namespace to sign in from a Kubernetes cluster which hosts the OIDC Idp.
#
cat <<EOF > TrustPolicy.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::${AWS_ACCOUNT_ID}:oidc-provider/
${OIDC_PROVIDER}"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "${OIDC_PROVIDER}:sub": "system:serviceaccount:
${SERVICE_ACCOUNT_NAMESPACE}:${SERVICE_ACCOUNT_AMP_QUERY_NAME}"
        }
      }
    }
  ]
}
EOF
#
# Set up the permission policy that grants query permissions for all AMP workspaces
#
cat <<EOF > PermissionPolicyQuery.json
{
  "Version": "2012-10-17",
   "Statement": [
       {"Effect": "Allow",
        "Action": [
           "aps:QueryMetrics",
           "aps:GetSeries",
           "aps:GetLabels",
           "aps:GetMetricMetadata"
        ],
        "Resource": "*"
      }
   ]
}
EOF
```

```
function getRoleArn() {
  OUTPUT=$(aws iam get-role --role-name $1 --query 'Role.Arn' --output text 2>&1)

  # Check for an expected exception
  if [[ $? -eq 0 ]]; then
    echo $OUTPUT
  elif [[ -n $(grep "NoSuchEntity" <<< $OUTPUT) ]]; then
    echo ""
  else
    >&2 echo $OUTPUT
    return 1
  fi
}

#
# Create the IAM Role for query with the above trust policy
#
SERVICE_ACCOUNT_IAM_AMP_QUERY_ROLE_ARN=$(getRoleArn
 $SERVICE_ACCOUNT_IAM_AMP_QUERY_ROLE)
if [ "$SERVICE_ACCOUNT_IAM_AMP_QUERY_ROLE_ARN" = "" ];
then
  #
  # Create the IAM role for service account
  #
  SERVICE_ACCOUNT_IAM_AMP_QUERY_ROLE_ARN=$(aws iam create-role \
  --role-name $SERVICE_ACCOUNT_IAM_AMP_QUERY_ROLE \
  --assume-role-policy-document file://TrustPolicy.json \
  --query "Role.Arn" --output text)
  #
  # Create an IAM permission policy
  #
  SERVICE_ACCOUNT_IAM_AMP_QUERY_ARN=$(aws iam create-policy --policy-name
 $SERVICE_ACCOUNT_IAM_AMP_QUERY_POLICY \
  --policy-document file://PermissionPolicyQuery.json \
  --query 'Policy.Arn' --output text)
  #
  # Attach the required IAM policies to the IAM role create above
  #
  aws iam attach-role-policy \
  --role-name $SERVICE_ACCOUNT_IAM_AMP_QUERY_ROLE \
  --policy-arn $SERVICE_ACCOUNT_IAM_AMP_QUERY_ARN
else
    echo "$SERVICE_ACCOUNT_IAM_AMP_QUERY_ROLE_ARN IAM role for query already
 exists"
```

```
fi
echo $SERVICE_ACCOUNT_IAM_AMP_QUERY_ROLE_ARN
#
# EKS cluster hosts an OIDC provider with a public discovery endpoint.
# Associate this IdP with AWS IAM so that the latter can validate and accept the
 OIDC tokens issued by Kubernetes to service accounts.
# Doing this with eksctl is the easier and best approach.
#
eksctl utils associate-iam-oidc-provider --cluster $CLUSTER_NAME --approve
```

2. 次のコマンドを入力して、スクリプトに必要な権限を付与します。

```
chmod +x createIRSA-AMPQuery.sh
```

3. スクリプトを実行します。

インターフェイス VPC エンドポイントでの Amazon Managed Service for Prometheus の使用

Amazon Virtual Private Cloud (Amazon VPC) を使用して AWS リソースをホストする場合は、VPC と Amazon Managed Service for Prometheus 間のプライベート接続を確立できます。これらの接続を使用すると、Amazon Managed Service for Prometheus はパブリックインターネットを経由せずに VPC のリソースと通信できます。

Amazon VPC は、定義した仮想ネットワークで AWS リソースを起動するために使用できる AWS サービスです。VPC を使用することで、IP アドレス範囲、サブネット、ルートテーブル、ネットワークゲートウェイなどのネットワーク設定を制御できます。VPC を Amazon Managed Service for Prometheus に接続するには、VPC を AWS のサービスに接続するためのインターフェイス VPC エンドポイントを定義します。このエンドポイントは、インターネットゲートウェイ、ネットワークアドレス変換 (NAT) インスタンス、または VPN 接続を必要とすることなく、Amazon Managed Service for Prometheus へのスケーラブルで信頼性の高い接続を提供します。詳細については、「Amazon VPC ユーザーガイド」の「Amazon VPC とは」を参照してください。

インターフェイス VPC エンドポイントは AWS PrivateLink、プライベート IP アドレスを持つ Elastic Network Interface を使用して AWS サービス間のプライベート通信を可能にする AWS テクノロジーを利用しています。詳細については、「New – AWS PrivateLink for AWS Services」ブログ記事を参照してください。

以下の情報は Amazon VPC ユーザーを対象としています。詳細については、「Amazon VPC ユーザーガイド」の「開始方法」を参照してください。

Amazon Managed Service for Prometheus 用のインターフェイス VPC エンドポイントの作成

インターフェイス VPC エンドポイントを作成して、Amazon Managed Service for Prometheus の使用を開始します。次のいずれかのサービス名エンドポイントを選択します。

- `com.amazonaws.region.aps-workspaces`

  Prometheus 互換 API を使用するには、このサービス名を選択します。詳細については、「Amazon Managed Service for Prometheus ユーザーガイド」の「Prometheus 互換 API」を参照してください。

- `com.amazonaws.region.aps`

  ワークスペースの管理タスクを実行するには、このサービス名を選択します。詳細については、「Amazon Managed Service for Prometheus ユーザーガイド」の「Amazon Managed Service for Prometheus API」を参照してください。

> **Note**
>
> 直接インターネットアクセスのない VPC で remote_write を使用している場合は、sigv4 がエンドポイントを経由できるように AWS Security Token Service、 のインターフェイス VPC エンドポイントも作成する必要があります。の VPC エンドポイントの作成の詳細については AWS STS、「 AWS Identity and Access Management AWS STS ユーザーガイド」の「インターフェイス VPC エンドポイントの使用」を参照してください。リージョン化されたエンドポイントを使用する AWS STS ように を設定する必要があります。

インターフェイス VPC エンドポイントの作成手順を含む詳細については、「Amazon VPC ユーザーガイド」の「インターフェイスエンドポイントの作成」を参照してください。

> **Note**
>
> VPC エンドポイントポリシーを使用すると、Amazon Managed Service for Prometheus インターフェイス VPC エンドポイントへのアクセスを制御できます。詳細については、次のセクションを参照してください。

Amazon Managed Service for Prometheus のインターフェイス VPC エンドポイントを作成済みで、VPC に配置されたワークスペースに既にデータが流れている場合、メトリクスはデフォルトでインターフェイス VPC エンドポイントを通じて送信されます。Amazon Managed Service for Prometheus は、パブリックエンドポイントまたはプライベートインターフェイスエンドポイント (どちらか使用中のもの) を使用してこのタスクを実行します。

Amazon Managed Service for Prometheus VPC エンドポイントへのアクセスの制御

VPC エンドポイントポリシーを使用すると、Amazon Managed Service for Prometheus インターフェイス VPC エンドポイントへのアクセスを制御できます。VPC エンドポイントポリシーは、エンドポイントの作成時または変更時にエンドポイントに加える IAM リソースポリシーです。エンドポイントの作成時にポリシーをアタッチしない場合、サービスへのフルアクセスを許可するデフォルトのポリシーが Amazon VPC によって自動的にアタッチされます。エンドポイントポリシーは、IAM アイデンティティベースのポリシーやサービス固有のポリシーを上書きしたり置き換えたりするものではありません。これは、評価項目から指定されたサービスへのアクセスを制御するための別のポリシーです。

詳細については、「Amazon VPC ユーザーガイド」の「VPC エンドポイントによるサービスのアクセス制御」を参照してください。

Amazon Managed Service for Prometheus のエンドポイントポリシーの例を次に示します。このポリシーは、`PromUser` というロールを持ち、VPC 経由で Amazon Managed Service for Prometheus に接続するユーザーに、ワークスペースとルールグループの表示を許可しますが、例えば、ワークスペースの作成や削除は許可しません。

JSON

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

```
            "Effect": "Allow",
            "Action": [
                "aps:DescribeWorkspace",
                "aps:DescribeRuleGroupsNamespace",
                "aps:ListRuleGroupsNamespaces",
                "aps:ListWorkspaces"
            ],
            "Resource": "arn:aws:aps:*:*:/workspaces*",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::111122223333:role/PromUser"
                ]
            }
        }
    ]
}
```

次の例は、指定した VPC 内の指定した IP アドレスから送信されたリクエストのみが成功するように許可するポリシーを示しています。他の IP アドレスからのリクエストは失敗します。

```
{
    "Statement": [
        {
            "Action": "aps:*",
            "Effect": "Allow",
            "Principal": "*",
            "Resource": "*",
            "Condition": {
                "IpAddress": {
                    "aws:VpcSourceIp": "192.0.2.123"
                },
        "StringEquals": {
                    "aws:SourceVpc": "vpc-555555555555"
                }
            }
        }
    ]
}
```

Amazon Managed Service for Prometheus のエラーのトラブルシューティング

Amazon Managed Service for Prometheus に関する問題のトラブルシューティングを行うには、以下のセクションが役立ちます。

トピック

429 または制限超過エラー

次の例のような 429 エラーが表示される場合は、リクエストが Amazon Managed Service for Prometheus の取り込みのクォータを超えています。

```
ts=2020-10-29T15:34:41.845Z caller=dedupe.go:112 component=remote level=error
 remote_name=e13b0c
url=http://iamproxy-external.prometheus.uswest2-prod.eks:9090/workspaces/workspace_id/
api/v1/remote_write
msg="non-recoverable error" count=500 err="server returned HTTP status 429
Too Many Requests: ingestion rate limit (6666.666666666667) exceeded while adding 499
 samples and 0 metadata
```

次の例のような 429 エラーが表示される場合は、リクエストが Amazon Managed Service for Prometheus のワークスペースあたりのアクティブなメトリクス数のクォータを超えています。

```
ts=2020-11-05T12:40:33.375Z caller=dedupe.go:112 component=remote level=error
 remote_name=aps
url=http://iamproxy-external.prometheus.uswest2-prod.eks:9090/workspaces/workspace_id/
api/v1/remote_write
msg="non-recoverable error" count=500 err="server returned HTTP status 429 Too Many
 Requests: user=accountid_workspace_id:
```

```
per-user series limit (local limit: 0 global limit: 3000000 actual local limit: 500000)
 exceeded
```

次の例のような 429 エラーが表示された場合、リクエストは、RemoteWrite Prometheus 互換 API を使用してワークスペースにデータを送信できるレート (1 秒あたりのトランザクション数) に対する Amazon Managed Service for Prometheus クォータを超えています。

```
ts=2024-03-26T16:50:21.780708811Z caller=dedupe.go:112 component=remote level=error
 remote_name=ab123c
url=https://aps-workspaces.us-east-1.amazonaws.com/workspaces/workspace_id/api/v1/
remote_write
msg="non-recoverable error" count=1000 exemplarCount=0 err="server returned HTTP status
 429 Too Many Requests: {\"message\":\"Rate exceeded\"}"
```

次の例のような 400 エラーが表示された場合、リクエストはアクティブな時系列の Amazon Managed Service for Prometheus のクォータを超えています。アクティブな時系列クォータの処理方法の詳細については、「アクティブシリーズのデフォルトのクォータ」を参照してください。

```
ts=2024-03-26T16:50:21.780708811Z caller=push.go:53 level=warn
url=https://aps-workspaces.us-east-1.amazonaws.com/workspaces/workspace_id/api/v1/
remote_write
msg="non-recoverable error" count=500 exemplarCount=0
err="server returned HTTP status 400 Bad Request: maxFailure (quorum) on a given error
 family, rpc error: code = Code(400)
desc = addr=10.1.41.23:9095 state=ACTIVE zone=us-east-1a, rpc error: code = Code(400)
desc = user=accountid_workspace_id: per-user series limit of 10000000 exceeded,
Capacity from 2,000,000 to 10,000,000 is automatically adjusted based on the last 30
 min of usage.
If throttled above 10,000,000 or in case of incoming surges, please contact
 administrator to raise it.
(local limit: 0 global limit: 10000000 actual local limit: 92879)"
```

Amazon Managed Service for Prometheus のサービスクォータの詳細については、「Amazon Managed Service for Prometheus のサービスクォータ」を参照してください。

サンプルが重複している

高可用性の Prometheus グループを使用している場合は、Prometheus インスタンスで外部ラベルを使用して重複排除を設定する必要があります。詳細については、「Amazon Managed Service for Prometheus に送信される高可用性メトリクスの重複排除」を参照してください。

重複データに関するその他の問題については、次のセクションで説明します。

サンプルのタイムスタンプに関するエラーが表示される

Amazon Managed Service for Prometheus はデータを順番に取り込み、各サンプルのタイムスタンプは前のサンプルより後になるものと想定しています。

データが順番に到着しない場合、`out-of-order samples`、`duplicate sample for timestamp`、または `samples with different value but same timestamp` に関するエラーが表示されます。これらの問題は、通常、Amazon Managed Service for Prometheus にデータを送信しているクライアントの設定が正しくないことが原因で発生します。使用している Prometheus クライアントがエージェントモードで実行している場合は、シリーズ名やターゲットが重複しているルールの設定を確認します。メトリクスがタイムスタンプを直接提供している場合は、タイムスタンプが順番どおりであることを確認します。

この仕組みやセットアップの確認方法の詳細については、Prom Labs のブログ記事「Understanding Duplicate Samples and Out-of-order Timestamp Errors in Prometheus」を参照してください。

制限に関するエラーメッセージが表示される

> **Note**
>
> Amazon Managed Service for Prometheus は、Prometheus のリソース使用状況をモニタリングするための CloudWatch 使用状況メトリクスを提供しています。CloudWatch 使用状況メトリクスのアラーム機能を使用すると、Prometheus のリソースと使用状況をモニタリングして制限エラーを回避できます。

次のいずれかのエラーメッセージが表示される場合は、Amazon Managed Service for Prometheus のいずれかのクォータの引き上げをリクエストすることで問題を解決できます。詳細については、「Amazon Managed Service for Prometheus のサービスクォータ」を参照してください。

- ユーザーあたりのシリーズ制限 *<value>* を超えました。管理者に連絡して制限を引き上げてください
- メトリクスあたりのシリーズ制限 *<value>* を超えました。管理者に連絡して制限を引き上げてください
- ingestion rate limit (...) exceeded
- series has too many labels (...) series: '%s'

- the query time range exceeds the limit (query length: xxx, limit: yyy)
- the query hit the max number of chunks limit while fetching chunks from ingesters
- Limit exceeded. Maximum workspaces per account.

ローカル Prometheus サーバーの出力が制限を超えている

Amazon Managed Service for Prometheus には、ワークスペースが Prometheus サーバーから受信できるデータ量に対するサービスクォータがあります。Prometheus サーバーが Amazon Managed Service for Prometheus に送信しているデータ量を特定するには、Prometheus サーバーに対して以下のクエリを実行します。Prometheus の出力が Amazon Managed Service for Prometheus の制限を超えていると判明した場合は、対応するサービスクォータの引き上げをリクエストできます。詳細については、「Amazon Managed Service for Prometheus のサービスクォータ」を参照してください。

出力制限を確認するためにローカル自己実行 Prometheus サーバーに対して実行するクエリ

データの種類	使用するクエリ
現在のアクティブなシリーズ数	`prometheus_tsdb_head_series`
現在の取り込みレート	`rate(prometheus_tsdb_head_samples_appended_total[5m])`
メトリクス名ごとのアクティブなシリーズ数の降順リスト	`sort_desc(count by(__name__)({__name__!=""}))`
メトリクスシリーズごとのラベル数	`group by(mylabe`

データの種類	使用するクエリ	
	lname) ({__name__! =""})	

一部のデータが表示されない

Amazon Managed Service for Prometheus に送信されたデータは、さまざまな理由で破棄される場合があります。次の表は、データが取り込まれずに破棄される場合の理由を示しています。

Amazon CloudWatch を使用して、データが破棄される量と理由を追跡できます。詳細については、「CloudWatch メトリクスを使用して Amazon Managed Service for Prometheus のリソースモニタリングする」を参照してください。

Reason	意味
greater_than_max_sample_age	現在時刻より古い口グ行は破棄されます
new-value-for-timestamp	重複したサンプルは、前のサンプルと同じタイムスタンプで送信されますが、値は異なります。
per_metric_series_limit	ユーザーがメトリクスごとのアクティブなシリーズ数の上限に達しました
per_user_series_limit	ユーザーがアクティブなシリーズの合計数の上限に達しました
rate_limited	取り込みレートが制限されました
sample-out-of-order	サンプルが順不同で送信されたため、処理できません
label_value_too_long	ラベル値の長さが許容される文字数の上限を超えています
max_label_names_per_series	ユーザーがメトリクスごとのラベル名の数の上限に達しました
missing_metric_name	メトリクス名が指定されていません

Reason	意味
metric_name_invalid	無効なメトリクス名が指定されました
label_invalid	無効なラベルが指定されました
duplicate_label_names	重複するラベル名が指定されました

Amazon Managed Service for Prometheus でのタグ付け

タグは、 AWS リソース AWS に割り当てるカスタム属性ラベルです。各 AWS タグには 2 つの部分があります。

- タグキー (CostCenter、Environment、Project、Secret など)。タグキーでは、大文字と小文字が区別されます。
- タグ値と呼ばれるオプションのフィールド (111122223333、Production、チーム名など)。タグ値を省略すると、空の文字列を使用した場合と同じになります。タグキーと同様に、タグ値では大文字と小文字が区別されます。

これらを合わせて、キーと値のペアと呼ばれます。各ワークスペースには、最大 50 個のタグを割り当てることができます。

タグは、 AWS リソースの識別と整理に役立ちます。多くの AWS サービスはタグ付けをサポートしているため、異なるサービスのリソースに同じタグを割り当てて、リソースが関連していることを示すことができます。例えば、Amazon S3 バケットに割り当てたものと同じタグを、Amazon Managed Service for Prometheus ワークスペースに割り当てることができます。タグ付け戦略の詳細については、「Tagging AWS Resources」を参照してください。

Amazon Managed Service for Prometheus では、ワークスペースとルールグループ名前空間の両方にタグを付けることができます。コンソール、 AWS CLI、APIs、または SDKs を使用して、これらのリソースのタグを追加、管理、削除できます。タグは、ワークスペースやルールグループの識別、整理、追跡に使用できるほか、IAM ポリシーで使用すると、Amazon Managed Service for Prometheus リソースを表示および操作できるユーザーを制御するために役立ちます。

タグの制限

タグには以下のベーシックな制限があります。

- 各リソースには、最大 50 個のタグを設定できます。
- タグキーは、リソースごとにそれぞれ一意である必要があります。また、各タグキーに設定できる値は 1 つのみです。
- タグキーの最大長は UTF-8 で 128 Unicode 文字です。
- タグ値の最大長は UTF-8 で 256 Unicode 文字です。

- タグ付けスキーマが複数の AWS サービスやリソースで使用されている場合は、他のサービスで許可される文字に制限がある場合があることに注意してください。一般的に使用できる文字は、UTF-8 で表現できる英字、数字、スペースと、. : + = @ _ / - (ハイフン) です。
- タグのキーと値では、大文字と小文字が区別されます。ベストプラクティスとして、タグでの大文字の使用方針を決定し、その方針をすべてのリソースタイプで一貫して実装することをお勧めします。例えば、Costcenter、costcenter、CostCenter のいずれを使用するかを決定し、すべてのタグに同じ規則を使用します。大文字と小文字の扱いについて、同様のタグに整合性のない規則を使用することは避けてください。
- キーまたは値のプレフィックスとして、aws:、AWS:、またはその大文字と小文字の組み合わせを変えたものは使用しないでください。これらは AWS 使用のためにのみ予約されています。このプレフィックスを持つタグのキーや値を編集または削除することはできません。このプレフィックスを持つタグは、リソースごとのタグ数の制限にはカウントされません。

トピック

Amazon Managed Service for Prometheus ワークスペースのタグ付け

タグとは、リソースに割り当てることができるカスタムラベルです。タグには、一意のキーとオプションの値 (キーと値のペア) が含まれます。タグは、 AWS リソースの識別や整理に役立ちます。Amazon Managed Service for Prometheus では、ワークスペース (およびルールグループ名前空間) にタグを付けることができます。コンソール、CLI、APIs、または SDKs AWS を使用して、これらのリソースのタグを追加、管理、削除できます。タグは、ワークスペースの識別、整理、追跡に使用できるほか、IAM ポリシーで使用すると、Amazon Managed Service for Prometheus のリソースを誰が表示および操作できるかを制御するのに役立ちます。

Amazon Managed Service for Prometheus ワークスペースのタグを操作するには、このセクションの手順を使用します。

トピック

- ワークスペースからのタグの削除

ワークスペースへのタグの追加

Amazon Managed Service for Prometheus ワークスペースにタグを追加すると、 AWS リソースの識別と整理、アクセスの管理に役立ちます。まず、ワークスペースに 1 つ以上のタグ (キーと値のペア) を追加します。タグを追加したら、それらのタグに基づいてワークスペースへのアクセスを管理する IAM ポリシーを作成できます。コンソールまたは を使用して AWS CLI 、Amazon Managed Service for Prometheus ワークスペースにタグを追加できます。

> **Important**
>
> ワークスペースにタグを追加すると、そのワークスペースへのアクセスに影響が生じる可能性があります。ワークスペースにタグを追加する前に、タグを使用してリソースへのアクセスを制御している可能性のある IAM ポリシーをすべて確認してください。

Amazon Managed Service for Prometheus ワークスペースの作成時にタグを追加する方法の詳細については、「Amazon Managed Service for Prometheus ワークスペースを作成する」を参照してください。

トピック

- ワークスペースへのタグの追加 (コンソール)
- ワークスペースへのタグの追加 (AWS CLI)

ワークスペースへのタグの追加 (コンソール)

コンソールを使用して、Amazon Managed Service for Prometheus ワークスペースに 1 つ以上のタグを追加できます。

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ナビゲーションペインで、メニューアイコンを選択します。
3. [すべての WorkSpaces] を選択します。
4. 管理するワークスペースのワークスペース ID を選択します。
5. [タグ] タブを選択します。

6. Amazon Managed Service for Prometheus ワークスペースにタグが追加されていない場合は、[タグを作成] を選択します。それ以外の場合は、[タグを管理] を選択します。
7. [キー] にタグの名前を入力します。[値] では、任意でタグに値を追加できます。
8. (オプション) 別のタグを追加するには、[タグを追加] を再度選択します。
9. タグの追加が完了したら、[変更を保存] を選択します。

ワークスペースへのタグの追加 (AWS CLI)

を使用して Amazon Managed Service for Prometheus ワークスペースにタグ AWS CLI を追加するには、次の手順に従います。ワークスペースの作成時にタグを追加するには、「Amazon Managed Service for Prometheus ワークスペースを作成する」を参照してください。

これらのステップでは、 の最新バージョンが既にインストールされているか、最新バージョンに AWS CLI 更新されていることを前提としています。詳細については、「AWS Command Line Interfaceのインストール」を参照してください。

ターミナルまたはコマンドラインで、tag-resource コマンドを実行して、タグを追加するワークスペースの Amazon リソースネーム (ARN) と、追加するタグのキーおよび値を指定します。ワークスペースには 1 つ以上のタグを追加できます。例えば、My-Workspace という名前の Amazon Managed Service for Prometheus ワークスペースを、タグキーが *Status* でタグ値が *Secret* のタグと、タグキーが *Team* でタグ値が *My-Team* のタグの 2 つでタグ付けするには、次のように入力します。

```
aws amp tag-resource --resource-arn arn:aws:aps:us-
west-2:123456789012:workspaces/IDstring
--tags Status=Secret,Team=My-Team
```

成功した場合、このコマンドは何も返しません。

ワークスペースのタグの表示

タグは、 AWS リソースを識別して整理し、リソースへのアクセスを管理するのに役立ちます。タグ付け戦略の詳細については、AWS 「リソースのタグ付け」を参照してください。

Amazon Managed Service for Prometheus ワークスペースのタグの表示 (コンソール)

コンソールを使用して、Amazon Managed Service for Prometheus ワークスペースに関連付けられているタグを表示できます。

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ナビゲーションペインで、メニューアイコンを選択します。
3. [すべての WorkSpaces] を選択します。
4. 管理するワークスペースのワークスペース ID を選択します。
5. [タグ] タブを選択します。

Amazon Managed Service for Prometheus ワークスペースのタグの表示 (AWS CLI)

を使用してワークスペースの AWS タグ AWS CLI を表示するには、次の手順に従います。タグが追加されていない場合、返されるリストは空になります。

ターミナルまたはコマンドラインで、list-tags-for-resource コマンドを実行します。例えば、ワークスペースのタグキーとタグ値のリストを表示するには、次のように入力します。

```
aws amp list-tags-for-resource --resource-arn arn:aws:aps:us-
west-2:123456789012:workspace/IDstring
```

成功した場合、このコマンドは次のような情報を返します。

```
{
    "tags": {
        "Status": "Secret",
        "Team": "My-Team"
    }
}
```

ワークスペースのタグの編集

ワークスペースに関連付けられているタグの値を変更できます。キーの名前を変更することもできます。これは、現在のタグを削除し、新しい名前を使用して、元のキーと同じ値を持つタグを追加することと同等です。

> **Important**
>
> Amazon Managed Service for Prometheus ワークスペースのタグを編集すると、そのワークスペースへのアクセスに影響が生じる可能性があります。ワークスペースのタグの名前

(キー) または値を編集する前に、タグのキーや値を使用してリポジトリなどのリソースへのアクセスを制御する可能性のある IAM ポリシーを必ず確認してください。

Amazon Managed Service for Prometheus ワークスペースのタグの編集 (コンソール)

コンソールを使用して、Amazon Managed Service for Prometheus ワークスペースに関連付けられているタグを編集できます。

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ナビゲーションペインで、メニューアイコンを選択します。
3. [すべての WorkSpaces] を選択します。
4. 管理するワークスペースのワークスペース ID を選択します。
5. [タグ] タブを選択します。
6. ワークスペースにタグが追加されていない場合は、[タグを作成] を選択します。それ以外の場合は、[タグを管理] を選択します。
7. [キー] にタグの名前を入力します。[値] では、任意でタグに値を追加できます。
8. (オプション) 別のタグを追加するには、[タグを追加] を再度選択します。
9. タグの追加が完了したら、[変更を保存] を選択します。

Amazon Managed Service for Prometheus ワークスペースのタグの編集 (AWS CLI)

を使用してワークスペースのタグ AWS CLI を更新するには、次の手順に従います。既存のキーの値を変更したり、別のキーを追加したりできます。

ターミナルまたはコマンドラインで、tag-resource コマンドを実行して、タグを更新する Amazon Managed Service for Prometheus ワークスペースの Amazon リソースネーム (ARN) と、タグキーおよびタグ値を指定します。

```
aws amp tag-resource --resource-arn arn:aws:aps:us-
west-2:123456789012:workspace/IDstring --tags Team=New-Team
```

ワークスペースからのタグの削除

ワークスペースに関連付けられている 1 つ以上のタグを削除できます。タグを削除しても、そのタグに関連付けられている他の AWS リソースからタグは削除されません。

> **Important**
>
> Amazon Managed Service for Prometheus ワークスペースのタグを削除すると、そのワークスペースへのアクセスに影響が生じる可能性があります。ワークスペースからタグを削除する前に、タグのキーや値を使用してリポジトリなどのリソースへのアクセスを制御する可能性のある IAM ポリシーを必ず確認してください。

Amazon Managed Service for Prometheus ワークスペースからのタグの削除 (コンソール)

コンソールを使用して、タグとワークスペースとの関連付けを解除できます。

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ナビゲーションペインで、メニューアイコンを選択します。
3. [すべての WorkSpaces] を選択します。
4. 管理するワークスペースのワークスペース ID を選択します。
5. [タグ] タブを選択します。
6. [タグを管理] を選択します。
7. 削除するタグを見つけ、[削除] を選択します。

Amazon Managed Service for Prometheus ワークスペースからのタグの削除 (AWS CLI)

を使用してワークスペースからタグ AWS CLI を削除するには、次の手順に従います。タグを削除してもそのタグがなくなるわけではありません。タグとワークスペースとの関連付けが解除されるだけです。

> **Note**
>
> Amazon Managed Service for Prometheus ワークスペースを削除すると、削除されたワークスペースからタグの関連付けがすべて解除されます。ワークスペースを削除する前にタグを削除する必要はありません。

ターミナルまたはコマンドラインで、untag-resource コマンドを実行して、タグを削除するワークスペースの Amazon リソースネーム (ARN) と、削除するタグのタグキーを指定します。例えば、My-Workspace という名前のワークスペースから *Status* というタグキーを持つタグを削除するには、次のように入力します。

```
aws amp untag-resource --resource-arn arn:aws:aps:us-
west-2:123456789012:workspace/IDstring --tag-keys Status
```

成功した場合、このコマンドは何も返しません。ワークスペースに関連付けられているタグを確認するには、list-tags-for-resource コマンドを実行します。

ルールグループ名前空間のタグ付け

タグとは、リソースに割り当てることができるカスタムラベルです。タグには、一意のキーとオプションの値 (キーと値のペア) が含まれます。タグは、 AWS リソースの識別や整理に役立ちます。Amazon Managed Service for Prometheus では、ルールグループ名前空間 (およびワークスペース) にタグを付けることができます。コンソール、CLI、APIs、または SDKs AWS を使用して、これらのリソースのタグを追加、管理、削除できます。タグは、ルールグループ名前空間の識別、整理、追跡に使用できるほか、IAM ポリシーで使用すると、Amazon Managed Service for Prometheus のリソースを誰が表示および操作できるかを制御するのに役立ちます。

Amazon Managed Service for Prometheus のルールグループ名前空間のタグを操作するには、このセクションの手順を使用します。

トピック

- ルールグループ名前空間へのタグの追加
- ルールグループ名前空間のタグの表示
- ルールグループ名前空間のタグの編集
- ルールグループ名前空間からのタグの削除

ルールグループ名前空間へのタグの追加

Amazon Managed Service for Prometheus ルールグループ名前空間にタグを追加すると、 AWS リソースを識別して整理し、リソースへのアクセスを管理するのに役立ちます。まず、ルールグループ名前空間に 1 つ以上のタグ (キーと値のペア) を追加します。タグを追加した後、IAM ポリシーを作成して、それらのタグに基づいて名前空間へのアクセスを管理できます。コンソールまたは を使用

して AWS CLI 、Amazon Managed Service for Prometheus ルールグループ名前空間にタグを追加できます。

> **Important**
>
> ルールグループ名前空間にタグを追加すると、そのルールグループ名前空間へのアクセスに影響が生じる可能性があります。タグを追加する前に、タグを使用してリソースへのアクセスを制御している可能性のある IAM ポリシーをすべて確認してください。

ルールグループ名前空間の作成時にタグを追加する方法の詳細については、「ルールファイルを作成する」を参照してください。

トピック

- ルールグループ名前空間へのタグの追加 (コンソール)
- ルールグループ名前空間へのタグの追加 (AWS CLI)

ルールグループ名前空間へのタグの追加 (コンソール)

コンソールを使用して、Amazon Managed Service for Prometheus のルールグループ名前空間に 1 つ以上のタグを追加できます。

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ナビゲーションペインで、メニューアイコンを選択します。
3. [すべての WorkSpaces] を選択します。
4. 管理するワークスペースのワークスペース ID を選択します。
5. [ルール管理] タブを選択します。
6. 名前空間名の横にあるボタンを選択し、[編集] を選択します。
7. [タグを作成]、[新しいタグを追加] を選択します。
8. [キー] にタグの名前を入力します。[値] では、任意でタグに値を追加できます。
9. (オプション) 別のタグを追加するには、[新しいタグを追加] を再度選択します。
10. タグの追加が完了したら、[変更を保存] を選択します。

ルールグループ名前空間へのタグの追加 (AWS CLI)

を使用して Amazon Managed Service for Prometheus ルールグループ名前空間にタグ AWS CLI を追加するには、次の手順に従います。ルールグループ名前空間の作成時にタグを追加するには、「Amazon Managed Service for Prometheus にルール設定ファイルをアップロードする」を参照してください。

これらのステップでは、 の最新バージョンが既にインストールされているか、最新バージョンに AWS CLI 更新されていることを前提としています。詳細については、「AWS Command Line Interfaceのインストール」を参照してください。

ターミナルまたはコマンドラインで、tag-resource コマンドを実行して、タグを追加するルールグループ名前空間の Amazon リソースネーム (ARN) と、追加するタグのキーおよび値を指定します。ルールグループ名前空間には複数のタグを追加できます。例えば、My-Workspace という名前の Amazon Managed Service for Prometheus 名前空間を、タグキーが *Status* でタグ値が *Secret* のタグと、タグキーが *Team* でタグ値が *My-Team* のタグの 2 つでタグ付けするには、次のように入力します。

```
aws amp tag-resource \
    --resource-arn arn:aws:aps:us-
west-2:123456789012:rulegroupsnamespace/IDstring/namespace_name \
    --tags Status=Secret,Team=My-Team
```

成功した場合、このコマンドは何も返しません。

ルールグループ名前空間のタグの表示

タグは、 AWS リソースを識別して整理し、リソースへのアクセスを管理するのに役立ちます。タグ付け戦略の詳細については、AWS 「リソースのタグ付け」を参照してください。

Amazon Managed Service for Prometheus ルールグループ名前空間のタグの表示 (コンソール)

コンソールを使用して、Amazon Managed Service for Prometheus のルールグループ名前空間に関連付けられているタグを表示できます。

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ナビゲーションペインで、メニューアイコンを選択します。

3. [すべての WorkSpaces] を選択します。
4. 管理するワークスペースのワークスペース ID を選択します。
5. [ルール管理] タブを選択します。
6. 名前空間名を選択します。

Amazon Managed Service for Prometheus ワークスペースのタグの表示 (AWS CLI)

を使用してルールグループ名前空間の AWS タグ AWS CLI を表示するには、次の手順に従います。タグが追加されていない場合、返されるリストは空になります。

ターミナルまたはコマンドラインで、list-tags-for-resource コマンドを実行します。例えば、ルールグループ名前空間のタグキーとタグ値の一覧を表示するには、次のように入力します。

```
aws amp list-tags-for-resource --resource-arn rn:aws:aps:us-
west-2:123456789012:rulegroupsnamespace/IDstring/namespace_name
```

成功した場合、このコマンドは次のような情報を返します。

```
{
    "tags": {
        "Status": "Secret",
        "Team": "My-Team"
    }
}
```

ルールグループ名前空間のタグの編集

ルールグループ名前空間に関連付けられているタグの値を変更できます。キーの名前を変更することもできます。これは、現在のタグを削除し、新しい名前を使用して、元のキーと同じ値を持つタグを追加することと同等です。

> **Important**
>
> ルールグループ名前空間のタグを編集すると、その名前空間へのアクセスに影響が生じる可能性があります。リソースのタグの名前 (キー) または値を編集する前に、タグのキーや値を使用してリソースへのアクセスを制御する可能性のある IAM ポリシーを必ず確認してください。

Amazon Managed Service for Prometheus ルールグループ名前空間のタグの編集 (コンソール)

コンソールを使用して、Amazon Managed Service for Prometheus のルールグループ名前空間に関連付けられているタグを編集できます。

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ナビゲーションペインで、メニューアイコンを選択します。
3. [すべての WorkSpaces] を選択します。
4. 管理するワークスペースのワークスペース ID を選択します。
5. [ルール管理] タブを選択します。
6. 名前空間の名前を選択します。
7. [タグを管理] を選択し、[新しいタグを追加] を選択します。
8. 既存のタグの値を変更するには、[値] に新しい値を入力します。
9. 他のタグを追加するには、[新しいタグを追加] を選択します。
10. タグの追加と編集が完了したら、[変更を保存] を選択します。

Amazon Managed Service for Prometheus ルールグループ名前空間のタグの編集 (AWS CLI)

を使用してルールグループ名前空間のタグ AWS CLI を更新するには、次の手順に従います。既存のキーの値を変更したり、別のキーを追加したりできます。

ターミナルまたはコマンドラインで、tag-resource コマンドを実行して、タグを更新するリソースの Amazon リソースネーム (ARN) と、タグキーおよびタグ値を指定します。

```
aws amp tag-resource --resource-arn rn:aws:aps:us-
west-2:123456789012:rulegroupsnamespace/IDstring/namespace_name --tags Team=New-Team
```

ルールグループ名前空間からのタグの削除

ルールグループ名前空間に関連付けられている 1 つ以上のタグを削除できます。タグを削除しても、そのタグに関連付けられている他の AWS リソースからタグは削除されません。

> **Important**
>
> リソースのタグを削除すると、そのリソースへのアクセスに影響が生じる可能性があります。リソースからタグを削除する前に、タグのキーや値を使用してリポジトリなどのリソースへのアクセスを制御する可能性のある IAM ポリシーを必ず確認してください。

Amazon Managed Service for Prometheus ルールグループ名前空間のタグの削除 (コンソール)

コンソールを使用して、タグとルールグループ名前空間との関連付けを解除できます。

1. Amazon Managed Service for Prometheus コンソール (https://console.aws.amazon.com/prometheus/) を開きます。
2. ナビゲーションペインで、メニューアイコンを選択します。
3. [すべての WorkSpaces] を選択します。
4. 管理するワークスペースのワークスペース ID を選択します。
5. [ルール管理] タブを選択します。
6. 名前空間の名前を選択します。
7. [Manage tags (タグの管理)] を選択します。
8. 削除するタグの横にある [削除] を選択します。
9. 完了したら、[変更を保存] を選択します。

Amazon Managed Service for Prometheus ルールグループ名前空間のタグの削除 (AWS CLI)

を使用してルールグループ名前空間からタグ AWS CLI を削除するには、次の手順に従います。タグを削除してもそのタグがなくなるわけではありません。タグとルールグループ名前空間との関連付けが解除されるだけです。

> **Note**
>
> Amazon Managed Service for Prometheus のルールグループ名前空間を削除すると、削除された名前空間からタグの関連付けがすべて解除されます。名前空間を削除する前にタグを削除する必要はありません。

ターミナルまたはコマンドラインで、untag-resource コマンドを実行して、タグを削除するルールグループ名前空間の Amazon リソースネーム (ARN) と、削除するタグのタグキーを指定します。例えば、My-Workspace という名前のワークスペースから *Status* というタグキーを持つタグを削除するには、次のように入力します。

```
aws amp untag-resource --resource-arn rn:aws:aps:us-
west-2:123456789012:rulegroupsnamespace/IDstring/namespace_name --tag-keys Status
```

成功した場合、このコマンドは何も返しません。リソースに関連付けられているタグを確認するには、list-tags-for-resource コマンドを実行します。

Amazon Managed Service for Prometheus のサービスクォータ

以下の 2 つのセクションでは、Amazon Managed Service for Prometheus に関連付けられているクォータと制限について説明します。

Service Quotas

Amazon Managed Service for Prometheus には、以下に示すクォータがあります。Amazon Managed Service for Prometheus は、Prometheus のリソース使用状況をモニタリングするための CloudWatch 使用状況メトリクスを提供しています。Amazon CloudWatch 使用状況メトリクスのアラーム機能を使用すると、Prometheus のリソースと使用状況をモニタリングして制限エラーを回避できます。

プロジェクトやワークスペースの拡大に伴い、モニタリングや引き上げリクエストが必要になる最も一般的なクォータは、ワークスペースごとのアクティブシリーズ数、およびワークスペースごとの取り込みレートです。

すべての調整可能なクォータでは、[調整可能] 列のリンクを選択するか、クォータの引き上げをリクエストすることにより、クォータの引き上げをリクエストできます。

ワークスペースごとのアクティブシリーズ数の制限は動的に適用されます。詳細については、「アクティブシリーズのデフォルトのクォータ」を参照してください。ワークスペースあたりの取り込みレートクォータは、ワークスペースにデータを取り込む速度を決定します。詳細については、「取り込みスロットリング」を参照してください。

> **Note**
>
> 特に明記されていない限り、これらのクォータはワークスペースごとに設定されます。ワークスペースあたりのアクティブシリーズの最大値は 10 億です。

名前	デフォルト	引き上げ可能	説明
ワークスペースごとのメタデータを持つアクティブなメトリクス数	サポートされている各リージョン: 20,000	いいえ	ワークスペースごとの、メタデータを持つ一意のアクティブなメトリクスの数。注: 制限に達すると、メトリクスサンプルは記録されますが、制限を超えたメタデータは削除されます。
ワークスペースごとのアクティブなシリーズ数	サポートされている各リージョン: 50,000,000	あり	ワークスペースあたりの一意のアクティブシリーズの数 (最大 10 億)。過去 2 時間以内にサンプルが報告された場合、そのシリーズはアクティブです。2 M から 50 M までの容量は、過去 30 分間の使用状況に基づいて自動的に調整されます。
アラートマネージャー定義ファイル内のアラート集約グループのサイズ	サポートされている各リージョン: 1,000	あり	アラートマネージャー定義ファイル内のアラート集約グループの最大サイズ。group_by のラベル値の組み合わせごとに集約グループが作成されます。

名前	デフォルト	引き上げ可能	説明
アラートマネージャー定義ファイルのサイズ	サポートされている各リージョン: 1,000,000	いいえ	アラートマネージャー定義ファイルの最大サイズ (バイト単位)。
アラートマネージャーのアラートペイロードサイズ	サポートされている各リージョン: 20,000,000	いいえ	ワークスペースあたりのすべてのアラートマネージャーアラートの最大アラートペイロードサイズ。バイト単位。アラートのサイズは、ラベルと注釈に依存します。
アラートマネージャーのアラート数	サポートされている各リージョン: 1,000	あり	ワースペースごとの同時アラートマネージャーアラートの最大数。
HA トラッカーのクラスター数	サポートされている各リージョン: 500	いいえ	ワークスペースごとの、取り込まれたサンプルについて HA トラッカーが追跡するクラスターの最大数。
ワークスペースごとの取り込みレート	サポートされている各リージョン: 1,666,666	あり	ワークスペースごとの、1 秒あたりのメトリクスサンプルの取り込みレート。制限は、ワークスペースあたりのアクティブなシリーズの 1/30 に、最大 1,666,666 まで自動的に調整されます。

名前	デフォルト	引き上げ可能	説明
アラートマネージャー定義ファイル内の禁止ルール数	サポートされている各リージョン: 100	可能	アラートマネージャー定義ファイル内の禁止ルールの最大数。
ラベルサイズ	サポートされている各リージョン: 7	いいえ	1 つのシリーズで許容される、すべてのラベルとラベル値を合わせた最大サイズ (キロバイト単位)。
ワークスペースあたりの LabelSet の制限	サポートされている各リージョン: 100	可能	ワークスペースごとに作成できるラベルセット制限の最大数。
メトリクスシリーズごとのラベル数	サポートされている各リージョン: 150	あり	メトリクスシリーズごとのラベルの数。
メタデータの長さ	サポートされている各リージョン: 1	[いいえ]	メトリクスのメタデータに許容される最大長 (キロバイト単位)。メタデータとは、メトリクス名、タイプ、単位、ヘルプテキストを指します。
メトリクスごとのメタデータ数	サポートされている各リージョン: 10	いいえ	メトリクスごとのメタデータの数。注: 制限に達すると、メトリクスサンプルは記録されますが、制限を超えたメタデータは削除されます。

名前	デフォルト	引き上げ可能	説明
アラートマネージャーのルーティングツリー内のノード数	サポートされている各リージョン: 100	可能	アラートマネージャーのルーティングツリー内の最大ノード数。
リージョンごとの API オペレーション数 (トランザクション数/秒)	サポートされている各リージョン: 10	あり	ワークスペース CRUD API、タグ付け API、ルールグループ名前空間 CRUD API、アラートマネージャー定義 CRUD API など、すべての Amazon Managed Service for Prometheus API のリージョンごとの 1 秒あたりの API オペレーションの最大数。
ワークスペースごとの GetSeries 、GetLabels、GetMetricMetadata API オペレーション数 (トランザクション数/秒)	サポートされている各リージョン: 10	いいえ	ワークスペースごとの 1 秒あたりにおける GetSeries、GetLabels、GetMetricMetadata Prometheus 互換 API オペレーションの最大数。
ワークスペースごとの QueryMetrics API オペレーション数 (トランザクション数/秒)	サポートされている各リージョン: 300	いいえ	ワークスペースごとの 1 秒あたりにおける QueryMetrics Prometheus 互換 API オペレーションの最大数。

名前	デフォルト	引き上げ可能	説明
ワークスペースごとの RemoteWrite API オペレーション数 (トランザクション数/秒)	サポートされている各リージョン: 3,000	いいえ	ワークスペースごとの 1 秒あたりにおける RemoteWrite Prometheus 互換 API オペレーションの最大数。
ワークスペースごとの他の Prometheus 互換 API オペレーション数 (トランザクション数/秒)	サポートされている各リージョン: 100	いいえ	他のすべての Prometheus 互換 API (ListAlerts、ListRules など) におけるワークスペースごとの 1 秒あたりの API オペレーションの最大数。
ワークスペースあたりの順不同取り込みレート	サポートされている各リージョン: 83,333	あり	ワークスペースあたりの 1 秒あたりのサンプル取り込みレートが順不同です。上書きされない限り、制限はワークスペースあたりの取り込みレートの 5% になるように自動的に調整されます。
ワークスペースあたりの順序外の時間枠	サポートされている各リージョン: 600	あり	ワークスペースあたりの順序の異なるサンプルの最大時間枠を秒単位で指定します。

名前	デフォルト	引き上げ可能	説明
インスタントクエリのクエリバイト数	サポートされている各リージョン：5	いいえ	1 つのインスタントクエリでスキャンできる最大バイト数 (ギガバイト単位)。
範囲クエリのクエリバイト数	サポートされている各リージョン：5	いいえ	1 つの範囲クエリで 24 時間ごとにスキャンできる最大バイト数 (ギガバイト単位)。
クエリサンプル数	サポートされている各リージョン: 50,000,000	いいえ	1 つの範囲クエリまたは 1 つのインスタントクエリで 24 時間間隔ごとにスキャンできるサンプルの最大数。
フェッチされるクエリシリーズ数	サポートされている各リージョン: 12,000,000	いいえ	1 つの範囲クエリまたは 1 つのインスタントクエリで 24 時間間隔ごとにスキャンできるシリーズの最大数。
クエリ時間範囲の日数	サポートされている各リージョン: 95	いいえ	QueryMetrics、GetSeries、GetLabels API の最大時間範囲。
リクエストサイズ	サポートされている各リージョン: 1	[いいえ]	取り込みまたはクエリの最大リクエストサイズ (メガバイト単位)。

名前	デフォルト	引き上げ可能	説明
ルール評価間隔	サポートされている各リージョン: 30	あり	ワークスペースあたりの、ルールグループのルール評価間隔の最小値 (秒単位)。
ルールグループ名前空間定義ファイルのサイズ	サポートされている各リージョン: 1,000,000	いいえ	ルールグループ名前空間定義ファイルの最大サイズ (バイト単位)。
ワークスペースごとのルール数	サポートされている各リージョン: 2,000	あり	ワークスペースごとのルールの最大数。
ワークスペースあたりのサイレンス数	サポートされている各リージョン: 1,000	あり	ワークスペースあたりの、期限切れ、アクティブ、保留中の無音を含む最大サイレンス数。
アラートマネージャー定義ファイル内のテンプレート数	サポートされている各リージョン: 100	可能	アラートマネージャー定義ファイル内のテンプレートの最大数。
アカウントごとのリージョンあたりのワークスペース数	サポートされている各リージョン: 25	可能	リージョンあたりのワークスペースの最大数。

アクティブシリーズのデフォルトのクォータ

Amazon Managed Service for Prometheus のワークスペースは、取り込み使用量に自動的に適応します。使用量の増加に応じて、サービスは、デフォルトのクォータを上限として、時系列の容量を自動的に増やします。

Amazon Managed Service for Prometheus ワークスペースは、使用状況に基づいて、次の 2 つの方法で自動的にスケールされます。

1. 30 分の平均使用量が 500 万シリーズを下回ると、容量は 2 倍になります (例えば、350 万の使用量のワークスペースは 700 万の容量を取得します)。
2. 使用量が 500 万シリーズを超えると、ワークスペースは 1,000 万のバッファを追加します (例えば、使用量が 2,500 万のワークスペースは 3,500 万の容量を取得します)。

Amazon Managed Service for Prometheus は、クォータまで取り込みが増加すると、より多くの容量を自動的に割り当てます。これにより、ワークロードで持続的なスロットリングが発生しなくなります。ただし、過去 30 分間に計算された前のベースラインの 2 倍または 1,000 万を超えると、スロットリングが発生する可能性があります。スロットリングを避けるため、Amazon Managed Service for Prometheus では、以前のベースラインを超えて増加する場合は、取り込み量を徐々に増やすことを推奨しています。

> **Note**
>
> アクティブ時系列の最小容量は 200 万で、時系列が 200 万未満の場合、スロットリングは発生しません。
> デフォルトのクォータを超えるには、クォータの引き上げをリクエストできます。

デフォルトのクォータを超えるスケーリング

デフォルトのアクティブシリーズクォータを超えるクォータの引き上げをリクエストすると、Amazon Managed Service for Prometheus はそれに応じてワークスペース容量を調整します。増加した容量を十分に活用しない場合、サービスは時間の経過と共に未使用の部分を再利用します。使用量が増えると、ワークスペースは自動的に再度スケールアップします。

ただし、過去 2 時間から計算された以前のベースラインに対して 2 倍以上または 5,000 万を超えるアクティブ時系列がある場合、スロットリングが発生する可能性があります。例えば、次のようになります。

- クォータが 1 億でベースラインが 3,000 万の場合、スロットリングなしで 2 時間以内に 6,000 万までスケールアップできます。
- クォータが 1 億でベースラインが 5,000 万の場合、スロットリングなしで 2 時間以内に 1 億までスケールアップできます。

取り込みスロットリング

Amazon Managed Service for Prometheus は、現在の制限に基づいて、各ワークスペースの取り込みをスロットリングします。これは、ワークスペースのパフォーマンスを維持するのに役立ちます。制限を超えると、CloudWatch メトリクスに `DiscardedSamples` が (`rate_limited` の理由と共に) 表示されます。CloudWatch を使用して取り込みをモニタリングし、スロットリング制限に近づいたときに警告するアラームを作成できます。詳細については、「CloudWatch メトリクスを使用して Amazon Managed Service for Prometheus のリソースモニタリングする」を参照してください。

Amazon Managed Service for Prometheus は、トークンバケットアルゴリズムを使用して取り込みのスロットリングを実装します。このアルゴリズムでは、アカウントには、特定の数のトークンを保持するバケットがあります。バケット内のトークンの数は、特定の秒における取り込み制限を表します。

取り込まれたデータサンプルごとに、バケットから 1 つのトークンが削除されます。バケットサイズ (ワークスペースごとの取り込みレート) が 1,000,000 の場合、ワークスペースは 1 秒あたり 100 万個のデータサンプルを取り込むことができます。取り込むサンプルが 100 万個を超えると、スロットリングされ、それ以上のレコードは取り込まれません。余分なデータサンプルは破棄されます。

バケットは、設定したレートで自動的に補充されます。バケットが最大容量に達していない場合、最大容量に達するまで、設定した数のトークンが毎秒追加されます。補充トークンが到着したときにバケットが満杯である場合、補充トークンは破棄されます。バケットは最大数を超えてトークンを保持することはできません。サンプル取り込みの補充レートは、ワークスペースごとの取り込みレートの制限に従って設定します。ワークスペースごとの取り込みレートを 170,000 に設定すると、バケットの補充レートは 1 秒あたり 17 万トークンになります。

ワークスペースが 1 秒間に 100 万個のデータサンプルを取り込むと、バケット内のトークン数はすぐにゼロまで減ります。その後、バケットは最大容量の 100 万トークンに達するまで、毎秒 17 万トークンずつ補充されます。最大数を超えない限り、ゼロになったバケットは 6 秒で最大容量に戻ります。

> **Note**
>
> 取り込みはバッチリクエストで行われます。使用可能なトークンが 100 個あり、101 個のサンプルのリクエストを送信すると、リクエスト全体が拒否されます。Amazon Managed Service for Prometheus はリクエストを部分的に受け入れることはしません。コレクターを作成すると、再試行を管理できます (バッチを小さくするか、しばらく経ってから再試行します)。

バケットが満杯になるまで待たなくても、ワークスペースは追加のデータサンプルを取り込むことができます。トークンはバケットに追加されるとすぐに使用できます。補充トークンをすぐに使用すると、バケットが最大容量に達することはありません。例えば、バケットを使い果たしても、1 秒あたり 17 万個のデータサンプルを引き続き取り込むことができます。1 秒あたり 17 万未満のデータサンプルを取り込む場合のみ、バケットを最大容量まで補充できます。

取り込まれるデータに関する追加の制限

Amazon Managed Service for Prometheus には、ワークスペースに取り込まれるデータに関して次の要件があります。これらは調整できません。

- 1 時間以上経過したメトリクスサンプルは取り込まれません。
- すべてのサンプルとメタデータにメトリクス名が必要です。

Amazon Managed Service for Prometheus API Reference

Amazon Managed Service for Prometheus は、次の 2 種類の API を提供しています。

1. Amazon Managed Service for Prometheus API - これらの API を使用すると、ワークスペース、スクレイパー、アラートマネージャー定義、ルールグループ名前空間、ログ記録用のオペレーションなどを含め、Amazon Managed Service for Prometheus ワークスペースを作成および管理できます。これらの API を操作するには、さまざまなプログラミング言語で利用できる AWS SDK を使用します。
2. Prometheus 互換 API - Amazon Managed Service for Prometheus は、Prometheus と互換性のある HTTP API をサポートしています。これらの API を使用すると、カスタムアプリケーションの構築、ワークフローの自動化、他のサービスやツールとの統合、Prometheus クエリ言語 (PromQL) を使用したモニタリングデータのクエリと操作が可能になります。

このセクションでは、Amazon Managed Service for Prometheus でサポートされる API オペレーションとデータ構造の一覧を示します。

シリーズ、ラベル、API リクエストのクォータについては、「Amazon Managed Service for Prometheus ユーザーガイド」の「Amazon Managed Service for Prometheus のサービスクォータ」を参照してください。

トピック

- Amazon Managed Service for Prometheus API
- Prometheus 互換 API

Amazon Managed Service for Prometheus API

Amazon Managed Service for Prometheus は、Amazon Managed Service for Prometheus ワークスペースを作成および管理する API オペレーションを提供しています。これには、ワークスペース、スクレイパー、アラートマネージャー定義、ルールグループ名前空間、ログ記録用の API が含まれます。

Amazon Managed Service for Prometheus API の詳細については、「Amazon Managed Service for Prometheus API Reference」を参照してください。

AWS SDK での Amazon Managed Service for Prometheus の使用

AWS Software Development Kit (SDK) は、多くの一般的なプログラミング言語で使用できます。各 SDK には、デベロッパーが好みの言語で AWS アプリケーションを簡単に構築できるようにする API、コード例、およびドキュメントが提供されています。言語別の SDK とツールのリストについては、「AWS デベロッパーセンター」の「AWS での構築ツール」を参照してください。

> **SDK のバージョン**
>
> プロジェクトで使用する AWS SDK やその他の SDK は最新ビルドを使用し、SDK を最新の状態に保つことをお勧めします。AWS SDK には、最新の特長と機能に加え、セキュリティアップデートも含まれています。

Prometheus 互換 API

Amazon Managed Service for Prometheus では、以下の Prometheus 互換 API がサポートされています。

Prometheus 互換 API の使用方法の詳細については、「Prometheus 互換 API を使用したクエリ」を参照してください。

トピック

- CreateAlertManagerAlerts
- DeleteAlertManagerSilence
- GetAlertManagerStatus
- GetAlertManagerSilence
- GetLabels
- GetMetricMetadata
- GetSeries
- ListAlerts
- ListAlertManagerAlerts
- ListAlertManagerAlertGroups
- ListAlertManagerReceivers

- ListAlertManagerSilences
- ListRules
- PutAlertManagerSilences
- QueryMetrics
- RemoteWrite

CreateAlertManagerAlerts

`CreateAlertManagerAlerts` オペレーションは、ワークスペースにアラートを作成します。

有効な HTTP 動詞:

`POST`

有効な URI:

`/workspaces/workspaceId/alertmanager/api/v2/alerts`

URL クエリパラメータ:

`alerts` オブジェクトの配列。各オブジェクトは 1 つのアラートを表します。アラートブジェクトの例を以下に示します。

```
[
  {
    "startsAt": "2021-09-24T17:14:04.995Z",
    "endsAt": "2021-09-24T17:14:04.995Z",
    "annotations": {
      "additionalProp1": "string",
      "additionalProp2": "string",
      "additionalProp3": "string"
    },
    "labels": {
      "additionalProp1": "string",
      "additionalProp2": "string",
      "additionalProp3": "string"
    },
    "generatorURL": "string"
  }
]
```

リクエスト例

```
POST /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/alertmanager/api/v2/alerts
 HTTP/1.1
Content-Length: 203,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0

[
  {
    "labels": {
      "alertname": "test-alert"
    },
    "annotations": {
      "summary": "this is a test alert used for demo purposes"
    },
    "generatorURL": "https://www.amazon.com/"
  }
]
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 0
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
vary: Origin
```

DeleteAlertManagerSilence

`DeleteSilence` は、1 つのアラートサイレンスを削除します。

有効な HTTP 動詞:

`DELETE`

有効な URI:

`/workspaces/`*`workspaceId`*`/alertmanager/api/v2/silence/`*`silenceID`*

URL クエリパラメータ: なし

リクエスト例

```
DELETE /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/alertmanager/api/v2/silence/
d29d9df3-9125-4441-912c-70b05f86f973 HTTP/1.1
Content-Length: 0,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 0
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
vary: Origin
```

GetAlertManagerStatus

GetAlertManagerStatus は、アラートマネージャーのステータスに関する情報を取得します。

有効な HTTP 動詞:

GET

有効な URI:

/workspaces/*workspaceId*/alertmanager/api/v2/status

URL クエリパラメータ: なし

リクエスト例

```
GET /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/alertmanager/api/v2/status
 HTTP/1.1
Content-Length: 0,
```

```
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 941
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
vary: Origin

{
    "cluster": null,
    "config": {
        "original": "global:\n  resolve_timeout: 5m\n  http_config:\n
 follow_redirects: true\n  smtp_hello: localhost\n  smtp_require_tls: true\nroute:
\n  receiver: sns-0\n  group_by:\n  - label\n  continue: false\nreceivers:\n-
 name: sns-0\n  sns_configs:\n  - send_resolved: false\n    http_config:\n
      follow_redirects: true\n    sigv4: {}\n    topic_arn: arn:aws:sns:us-
west-2:123456789012:test\n    subject: '{{ template \"sns.default.subject\" . }}'\n
    message: '{{ template \"sns.default.message\" . }}'\n    workspace_arn:
 arn:aws:aps:us-west-2:123456789012:workspace/ws-58a6a446-5ec4-415b-9052-a449073bbd0a
\ntemplates: []\n"
    },
    "uptime": null,
    "versionInfo": null
}
```

GetAlertManagerSilence

GetAlertManagerSilence は、1 つのアラートサイレンスに関する情報を取得します。

有効な HTTP 動詞:

GET

有効な URI:

/workspaces/*workspaceId*/alertmanager/api/v2/silence/*silenceID*

URL クエリパラメータ: なし

リクエスト例

```
GET /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/alertmanager/api/v2/silence/
d29d9df3-9125-4441-912c-70b05f86f973 HTTP/1.1
Content-Length: 0,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 310
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
vary: Origin

{
    "id": "d29d9df3-9125-4441-912c-70b05f86f973",
    "status": {
        "state": "active"
    },
    "updatedAt": "2021-10-22T19:32:11.763Z",
    "comment": "hello-world",
    "createdBy": "test-person",
    "endsAt": "2023-07-24T01:05:36.000Z",
    "matchers": [
        {
            "isEqual": true,
            "isRegex": true,
            "name": "job",
            "value": "hello"
        }
    ],
    "startsAt": "2021-10-22T19:32:11.763Z"
}
```

GetLabels

GetLabels オペレーションは、時系列に関連付けられているラベルを取得します。

有効な HTTP 動詞:

GET, POST

有効な URI:

/workspaces/*workspaceId*/api/v1/labels

/workspaces/*workspaceId*/api/v1/label/*label-name*/values この URI は GET リクエストのみをサポートします。

URL クエリパラメータ:

match[]=<series_selector> ラベル名を読み取るシリーズを選択する、シリーズセレクターの繰り返しを含む引数。オプション。

start=<rfc3339 | unix_timestamp> 開始タイムスタンプ。オプション。

end=<rfc3339 | unix_timestamp> 終了タイムスタンプ。オプションです。

/workspaces/workspaceId/api/v1/labels のサンプルリクエスト

```
GET /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/api/v1/labels HTTP/1.1
Content-Length: 0,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0
```

/workspaces/workspaceId/api/v1/labels のサンプル応答

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 1435
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
vary: Origin
```

```
{
    "status": "success",
    "data": [
        "__name__",
        "access_mode",
        "address",
        "alertname",
        "alertstate",
        "apiservice",
        "app",
        "app_kubernetes_io_instance",
        "app_kubernetes_io_managed_by",
        "app_kubernetes_io_name",
        "area",
        "beta_kubernetes_io_arch",
        "beta_kubernetes_io_instance_type",
        "beta_kubernetes_io_os",
        "boot_id",
        "branch",
        "broadcast",
        "buildDate",
        ...
    ]
}
```

/workspaces/workspaceId/api/v1/label/label-name/values のサンプルリクエスト

```
GET /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/api/v1/label/access_mode/values
 HTTP/1.1
Content-Length: 0,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0
```

/workspaces/workspaceId/api/v1/label/label-name/values のサンプル応答

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 74
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
```

```
Server: amazon
vary: Origin

{
    "status": "success",
    "data": [
        "ReadWriteOnce"
    ]
}
```

GetMetricMetadata

GetMetricMetadata オペレーションは、現在ターゲットからスクレイピングされているメトリクスに関するメタデータを取得します。ターゲット情報は提供されません。

クエリ結果のデータセクションはオブジェクトで構成されます。各オブジェクトのキーはメトリクス名を表し、値には、そのメトリクス名で公開されている固有のメタデータオブジェクトが、すべてのターゲットにわたるリストとして含まれます。

有効な HTTP 動詞:

GET

有効な URI:

/workspaces/*workspaceId*/api/v1/metadata

URL クエリパラメータ:

limit=<number> 取得するメトリクスの最大数。

metric=<string> メタデータをフィルタリングするメトリクス名。空のままにすると、すべてのメトリクスメタデータが取得されます。

リクエスト例

```
GET /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/api/v1/metadata HTTP/1.1
Content-Length: 0,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
Transfer-Encoding: chunked

{
    "status": "success",
    "data": {
        "aggregator_openapi_v2_regeneration_count": [
            {
                "type": "counter",
                "help": "[ALPHA] Counter of OpenAPI v2 spec regeneration count broken
 down by causing APIService name and reason.",
                "unit": ""
            }
        ],
        ...
    }
}
```

GetSeries

GetSeries オペレーションは、特定のラベルセットに一致する時系列のリストを取得します。

有効な HTTP 動詞:

GET, POST

有効な URI:

/workspaces/*workspaceId*/api/v1/series

URL クエリパラメータ:

match[]=<series_selector> 取得するシリーズを選択する、シリーズセレクターの繰り返しを含む引数。少なくとも 1 つの match[] 引数を指定する必要があります。

start=<rfc3339 | unix_timestamp> 開始タイムスタンプ。オプションです。

end=<rfc3339 | unix_timestamp> 終了タイムスタンプ。オプションです。

リクエスト例

```
POST /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/api/v1/series --data-urlencode
 'match[]=node_cpu_seconds_total{app="prometheus"}' --data-urlencode 'start=1634936400'
 --data-urlencode 'end=1634939100' HTTP/1.1
Content-Length: 0,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
content-encoding: gzip

{
    "status": "success",
    "data": [
        {
            "__name__": "node_cpu_seconds_total",
            "app": "prometheus",
            "app_kubernetes_io_managed_by": "Helm",
            "chart": "prometheus-11.12.1",
            "cluster": "cluster-1",
            "component": "node-exporter",
            "cpu": "0",
            "heritage": "Helm",
            "instance": "10.0.100.36:9100",
            "job": "kubernetes-service-endpoints",
            "kubernetes_name": "servicesstackprometheuscf14a6d7-node-exporter",
            "kubernetes_namespace": "default",
            "kubernetes_node": "ip-10-0-100-36.us-west-2.compute.internal",
            "mode": "idle",
            "release": "servicesstackprometheuscf14a6d7"
        },
```

```
        {
            "__name__": "node_cpu_seconds_total",
            "app": "prometheus",
            "app_kubernetes_io_managed_by": "Helm",
            "chart": "prometheus-11.12.1",
            "cluster": "cluster-1",
            "component": "node-exporter",
            "cpu": "0",
            "heritage": "Helm",
            "instance": "10.0.100.36:9100",
            "job": "kubernetes-service-endpoints",
            "kubernetes_name": "servicesstackprometheuscf14a6d7-node-exporter",
            "kubernetes_namespace": "default",
            "kubernetes_node": "ip-10-0-100-36.us-west-2.compute.internal",
            "mode": "iowait",
            "release": "servicesstackprometheuscf14a6d7"
        },
        ...
    ]
}
```

ListAlerts

ListAlerts オペレーションは、ワークスペースで現在アクティブなアラートを取得します。

有効な HTTP 動詞:

```
GET
```

有効な URI:

```
/workspaces/workspaceId/api/v1/alerts
```

リクエスト例

```
GET /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/api/v1/alerts HTTP/1.1
Content-Length: 0,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 386
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
vary: Origin

{
  "status": "success",
  "data": {
    "alerts": [
      {
        "labels": {
          "alertname": "test-1.alert",
          "severity": "none"
        },
        "annotations": {
          "message": "message"
        },
        "state": "firing",
        "activeAt": "2020-12-01T19:37:25.429565909Z",
        "value": "1e+00"
      }
    ]
  },
  "errorType": "",
  "error": ""
}
```

ListAlertManagerAlerts

`ListAlertManagerAlerts` は、ワークスペースのアラートマネージャーで現在発生しているアラートに関する情報を取得します。

有効な HTTP 動詞:

```
GET
```

有効な URI:

```
/workspaces/workspaceId/alertmanager/api/v2/alerts
```

リクエスト例

```
GET /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/alertmanager/api/v2/alerts
 HTTP/1.1
Content-Length: 0,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 354
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
vary: Origin

[
    {
        "annotations": {
            "summary": "this is a test alert used for demo purposes"
        },
        "endsAt": "2021-10-21T22:07:31.501Z",
        "fingerprint": "375eab7b59892505",
        "receivers": [
            {
                "name": "sns-0"
            }
        ],
        "startsAt": "2021-10-21T22:02:31.501Z",
        "status": {
            "inhibitedBy": [],
            "silencedBy": [],
            "state": "active"
        },
        "updatedAt": "2021-10-21T22:02:31.501Z",
        "labels": {
            "alertname": "test-alert"
        }
    }
```

```
]
```

ListAlertManagerAlertGroups

ListAlertManagerAlertGroups オペレーションは、ワークスペースのアラートマネージャーで構成されているアラートグループのリストを取得します。

有効な HTTP 動詞:

GET

有効な URI:

/workspaces/*workspaceId*/alertmanager/api/v2/alerts/groups

URL クエリパラメータ:

active ブール値。true の場合、返されるリストにはアクティブなアラートが含まれます。デフォルトは True です。オプションです。

silenced ブール値。true の場合、返されるリストには無音のアラートが含まれます。デフォルトは True です。オプションです。

inhibited ブール値。true の場合、返されるリストには禁止されたアラートが含まれます。デフォルトは True です。オプションです。

filter 文字列の配列。アラートをフィルタリングするマッチャーのリスト。オプションです。

receiver 文字列。アラートをフィルタリングするレシーバーに一致する正規表現。オプションです。

リクエスト例

```
GET /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/alertmanager/api/v2/alerts/
groups HTTP/1.1
Content-Length: 0,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0
```

レスポンス例

```
HTTP/1.1 200 OK
```

```
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 443
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
vary: Origin

[
    {
        "alerts": [
            {
                "annotations": {
                    "summary": "this is a test alert used for demo purposes"
                },
                "endsAt": "2021-10-21T22:07:31.501Z",
                "fingerprint": "375eab7b59892505",
                "receivers": [
                    {
                        "name": "sns-0"
                    }
                ],
                "startsAt": "2021-10-21T22:02:31.501Z",
                "status": {
                    "inhibitedBy": [],
                    "silencedBy": [],
                    "state": "unprocessed"
                },
                "updatedAt": "2021-10-21T22:02:31.501Z",
                "generatorURL": "https://www.amazon.com/",
                "labels": {
                    "alertname": "test-alert"
                }
            }
        ],
        "labels": {},
        "receiver": {
            "name": "sns-0"
        }
    }
]
```

ListAlertManagerReceivers

`ListAlertManagerReceivers` オペレーションは、アラートマネージャーで構成されているレシーバーに関する情報を取得します。

有効な HTTP 動詞:

```
GET
```

有効な URI:

```
/workspaces/workspaceId/alertmanager/api/v2/receivers
```

URL クエリパラメータ: なし

リクエスト例

```
GET /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/alertmanager/api/v2/receivers
 HTTP/1.1
Content-Length: 0,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 19
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
vary: Origin

[
    {
        "name": "sns-0"
    }
]
```

ListAlertManagerSilences

`ListAlertManagerSilences` は、ワークスペースに構成されているアラートサイレンスに関する情報を取得します。

有効な HTTP 動詞:

```
GET
```

有効な URI:

```
/workspaces/workspaceId/alertmanager/api/v2/silences
```

リクエスト例

```
GET /workspaces/ws-58a6a446-5ec4-415b-9052-a449073bbd0a/alertmanager/api/v2/silences
 HTTP/1.1
Content-Length: 0,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 312
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
vary: Origin

[
    {
        "id": "d29d9df3-9125-4441-912c-70b05f86f973",
        "status": {
            "state": "active"
        },
        "updatedAt": "2021-10-22T19:32:11.763Z",
        "comment": "hello-world",
        "createdBy": "test-person",
```

```
        "endsAt": "2023-07-24T01:05:36.000Z",
        "matchers": [
            {
                "isEqual": true,
                "isRegex": true,
                "name": "job",
                "value": "hello"
            }
        ],
        "startsAt": "2021-10-22T19:32:11.763Z"
    }
]
```

ListRules

ListRules は、ワークスペースに構成されているルールに関する情報を取得します。

有効な HTTP 動詞:

GET

有効な URI:

/workspaces/*workspaceId*/api/v1/rules

リクエスト例

```
GET /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/api/v1/rules HTTP/1.1
Content-Length: 0,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 423
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
```

```
vary: Origin

{
    "status": "success",
    "data": {
        "groups": [
            {
                "name": "test-1.rules",
                "file": "test-rules",
                "rules": [
                    {
                        "name": "record:1",
                        "query": "sum(rate(node_cpu_seconds_total[10m:1m]))",
                        "labels": {},
                        "health": "ok",
                        "lastError": "",
                        "type": "recording",
                        "lastEvaluation": "2021-10-21T21:22:34.429565909Z",
                        "evaluationTime": 0.001005399
                    }
                ],
                "interval": 60,
                "lastEvaluation": "2021-10-21T21:22:34.429563992Z",
                "evaluationTime": 0.001010504
            }
        ]
    },
    "errorType": "",
    "error": ""
}
```

PutAlertManagerSilences

PutAlertManagerSilences オペレーションは、新しいアラートサイレンスの作成または既存のアラートサイレンスの更新を行います。

有効な HTTP 動詞:

POST

有効な URI:

/workspaces/*workspaceId*/alertmanager/api/v2/silences

URL クエリパラメータ:

silence サイレンスを表すオブジェクト。形式を次に示します。

```
{
  "id": "string",
  "matchers": [
    {
      "name": "string",
      "value": "string",
      "isRegex": Boolean,
      "isEqual": Boolean
    }
  ],
  "startsAt": "timestamp",
  "endsAt": "timestamp",
  "createdBy": "string",
  "comment": "string"
}
```

リクエスト例

```
POST /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/alertmanager/api/v2/silences
 HTTP/1.1
Content-Length: 281,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0

{
   "matchers":[
      {
         "name":"job",
         "value":"up",
         "isRegex":false,
         "isEqual":true
      }
   ],
   "startsAt":"2020-07-23T01:05:36+00:00",
   "endsAt":"2023-07-24T01:05:36+00:00",
   "createdBy":"test-person",
   "comment":"test silence"
```

```
}
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 53
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
vary: Origin

{
    "silenceID": "512860da-74f3-43c9-8833-cec026542b32"
}
```

QueryMetrics

QueryMetrics オペレーションは、特定の時点において、または一定期間にわたってインスタントクエリを評価します。

有効な HTTP 動詞:

GET, POST

有効な URI:

/workspaces/*workspaceId*/api/v1/query この URI は、特定の時点でインスタントクエリを評価します。

/workspaces/*workspaceId*/api/v1/query_range この URI は、一定期間にわたってインスタントクエリを評価します。

URL クエリパラメータ:

query=<string> Prometheus 式のクエリ文字列。query と query_range の両方で使用されます。

time=<rfc3339 | unix_timestamp> (オプション) query を使用して特定の時点でインスタントクエリを評価する場合、評価のタイムスタンプ。

timeout=<duration> (オプション) 評価のタイムアウト。デフォルトは -query.timeout フラグの値で、この値が上限になります。query と query_range の両方で使用されます。

start=<rfc3339 | unix_timestamp> query_range を使用して一定期間にわたってクエリを評価する場合、開始タイムスタンプ。

end=<rfc3339 | unix_timestamp> query_range を使用して一定期間にわたってクエリを評価する場合、終了タイムスタンプ。

step=<duration | float> クエリの解決ステップ幅 (duration 形式または float の秒数)。query_range を使用して一定期間にわたってクエリを評価するときに、そのクエリで必要とされる場合にのみ使用します。

max_samples_processed_warning_threshold=<integer> (オプション) 処理されたクエリサンプル (QSP) の警告しきい値を設定します。クエリがこのしきい値に達すると、API レスポンスに警告メッセージが表示されます。

max_samples_processed_error_threshold=<integer>> (オプション) 処理されたクエリサンプル (QSP) のエラーしきい値を設定します。このしきい値を超えるクエリはエラーで拒否され、課金されません。過剰なクエリコストを防ぐために使用されます。

duration

Prometheus 互換 API の duration には、数値に続けて以下の単位のいずれかを指定します。

- ms ミリ秒
- s 秒
- m 分
- h 時間
- d 日 (1 日は常に 24h と想定)
- w 週 (1 週間は常に 7d と想定)
- y 年 (1 年は常に 365d と想定)

リクエスト例

```
POST /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/api/v1/query?
query=sum(node_cpu_seconds_total) HTTP/1.1
Content-Length: 0,
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Grafana/8.1.0
```

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 132
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
content-encoding: gzip

{
    "status": "success",
    "data": {
        "resultType": "vector",
        "result": [
            {
                "metric": {},
                "value": [
                    1634937046.322,
                    "252590622.81000024"
                ]
            }
        ]
    }
}
```

RemoteWrite

RemoteWrite オペレーションは、Prometheus サーバーからリモート URL にメトリクスを標準化された形式で書き込みます。通常、このオペレーションを呼び出すには、Prometheus サーバーなどの既存のクライアントを使用します。

有効な HTTP 動詞:

POST

有効な URI:

/workspaces/*workspaceId*/api/v1/remote_write

URL クエリパラメータ:

なし

RemoteWrite での取り込みレートは 70,000 サンプル/秒、取り込みバーストサイズは 1,000,000 サンプルです。

リクエスト例

```
POST /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568/api/v1/remote_write --data-
binary "@real-dataset.sz" HTTP/1.1
Authorization: AUTHPARAMS
X-Amz-Date: 20201201T193725Z
User-Agent: Prometheus/2.20.1
Content-Type: application/x-protobuf
Content-Encoding: snappy
X-Prometheus-Remote-Write-Version: 0.1.0

body
```

> **Note**
>
> リクエスト本文の構文については、https://github.com/prometheus/prometheus/blob/1c624c58ca934f618be737b4995e22051f5724c1/prompb/remote.pb.go#L64 のプロトコルバッファの定義を参照してください。

レスポンス例

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length:0
Connection: keep-alive
Date: Tue, 01 Dec 2020 19:37:25 GMT
Content-Type: application/json
Server: amazon
vary: Origin
```

Amazon Managed Service for Prometheus ユーザーガイドのドキュメント履歴

次の表は、「Amazon Managed Service for Prometheus ユーザーガイド」のドキュメントの重要な更新情報をまとめたものです。このドキュメントの更新に関する通知を受け取るには、RSS フィードにサブスクライブできます。

変更	説明	日付
PagerDuty のサポートを開始	Amazon Managed Service for Prometheus に PagerDuty 統合のサポートが追加されます。これにより、自動化されたインシデント対応ワークフローが可能になり、重要なアラートが適切なチームメンバーに適切なタイミングで届くようになります。詳細については、「PagerDuty をアラートレシーバーとして使用する」を参照してください。	2025 年 8 月 29 日
リソースベースのポリシーのサポートを追加	以下の API アクションを使用できます。 • DeleteResourcePolicy • DescribeResourcePolicy • PutResourcePolicy	2025 年 8 月 15 日
AmazonPrometheusConsoleFullAccess マネージド IAM ポリシーを更新します。	AmazonPrometheusConsoleFullAccess ポリシーが更新されました。`aps:CreateQueryLoggingConfiguration` 、`aps:UpdateQueryLoggingConfiguration` 、`aps:Delet`	2025 年 5 月 5 日

	eQueryLoggingConfiguration 、aps:DescribeQueryLoggingConfiguration アクセス許可がポリシーに追加されました。	
コンソールでのルール定義ファイルとアラートマネージャー設定ファイルの編集を追加	Amazon Managed Service for Prometheus で、Amazon Managed Service for Prometheus コンソール内からアラートマネージャー設定ファイルとルール定義ファイルを編集するためのサポートを追加しました。	2024 年 5 月 16 日
Amazon EKS のアクセスエントリを使用して、よりシンプルな AWS マネージドコレクター設定を追加	Amazon Managed Service for Prometheus で、AWS マネージドコレクターの設定を簡素化するための Amazon EKS アクセスエントリのサポートを追加しました。AWS マネージドコレクターの AmazonPrometheusScraperServiceRolePolicy マネージドポリシーが更新され、使用されなくなったアクセスエントリを削除できるようになりました。	2024 年 5 月 2 日

AWS API を別の API リファレンスガイドに移動する	Amazon Managed Service for AWS APIsが、独自のリファレンスである Amazon Managed Service for Prometheus API リファレンスで利用可能になりました。Prometheus 互換 API については、「Amazon Managed Service for Prometheus ユーザーガイド」に引き続き記載されます。	2024 年 2 月 7 日
ワークスペース暗号化用のカスタマーマネージドキーを追加	Amazon Managed Service for Prometheus では、ワークスペース暗号化用のカスタマーマネージドキーのサポートが追加されました。詳細については、「保管時の暗号化」を参照してください。	2023 年 12 月 21 日
AmazonPrometheusFullAccess に新しいアクセス許可を追加	Amazon EKS クラスターのマネージドコレクターの作成をサポートするために、AmazonPrometheusFullAccess AWS 管理ポリシーに新しいアクセス許可を追加しました。	2023 年 11 月 26 日
AmazonPrometheusScraperServiceLinkedRolePolicy という新しい管理ポリシーを追加	Amazon EKS クラスターからメトリクスを収集するための AWS マネージドコレクター向けに、新しい管理ポリシー AmazonPrometheusScraperServiceLinkedRolePolicy を追加しました。	2023 年 11 月 26 日

取り込み方法として AWS マネージドコレクターを追加	Amazon Managed Service for Prometheus では、AWS マネージドコレクターのサポートが追加されました。	2023 年 11 月 26 日
Amazon Managed Grafana との統合のサポートが追加されました。	Amazon Managed Service for Prometheus に Amazon Managed Grafana アラートとの統合のサポートが追加されました。	2022 年 11 月 23 日
AmazonPrometheusConsoleFullAccess に新しいアクセス許可が追加されました。	AmazonPrometheusConsoleFullAccess マネージドポリシーに、CloudWatch Logs へのアラートマネージャーとルーラーイベントのログ記録をサポートする新しいアクセス許可が追加されました。	2022 年 10 月 24 日
Amazon EKS オブザーバビリティソリューションが追加されました。	Amazon Managed Service for Prometheus で AWS Observability Accelerator を使用した新しいソリューションが追加されました。詳細については、「AWS Observability Accelerator の使用」を参照してください。	2022 年 10 月 14 日
Amazon EKS コストモニタリングとの統合のサポートが追加されました。	Amazon Managed Service for Prometheus に、Amazon EKS コストモニタリングとの統合のサポートが追加されました。詳細については、「Amazon EKS コストモニタリングとの統合」を参照してください。	2022 年 9 月 22 日

Amazon CloudWatch Logs でのアラートマネージャーとルーラーのログのサポートが開始されました。	Amazon Managed Service for Prometheus で、Amazon CloudWatch Logs でのアラートマネージャーとルーラーのエラーログのサポートが開始されました。詳細については、「Amazon CloudWatch Logs」を参照してください。	2022 年 9 月 1 日
カスタムのストレージ保持のサポートが追加されました。	Amazon Managed Service for Prometheus で、ワークスペースのクォータを変更することにより、ワークスペースごとにカスタムのストレージ保持がサポートされるようになりました。Amazon Managed Service for Prometheus のクォータの詳細については、「サービスクォータ」を参照してください。	2022 年 8 月 12 日
Amazon CloudWatch に使用状況メトリクスが追加されました。	Amazon Managed Service for Prometheus に、Amazon CloudWatch への使用状況メトリクスの送信のサポートが追加されました。詳細については、「Amazon CloudWatch メトリクス」を参照してください。	2022 年 5 月 6 日
欧州 (ロンドン) リージョンのサポートが追加されました。	Amazon Managed Service for Prometheus に欧州 (ロンドン) リージョンのサポートが追加されました。	2022 年 5 月 4 日

Amazon Managed Service for Prometheus が一般公開され、ルールとアラートマネージャーのサポートが追加されました。	Amazon Managed Service for Prometheus が一般公開されました。また、ルールとアラートマネージャーのサポートも追加されました。詳細については、「記録ルールとアラートルール」および「アラートマネージャーとテンプレート」を参照してください。	2021 年 9 月 29 日
タグ付けのサポートが追加されました。	Amazon Managed Service for Prometheus で、Amazon Managed Service for Prometheus ワークスペースのタグ付けがサポートされました。	2021 年 9 月 7 日
アクティブなシリーズ数と取り込みレートのクォータが増加しました。	アクティブなシリーズ数のクォータが 1,000,000 に増加し、取り込みレートのクォータが 1 秒あたり 70,000 サンプルに増加しました。	2021 年 2 月 22 日
Amazon Managed Service for Prometheus のプレビューがリリースされました。	Amazon Managed Service for Prometheus のプレビューがリリースされました。	2020 年 12 月 15 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。