Amazon Managed Service for Prometheus

User Guide

Amazon Managed Service for Prometheus: User Guide

Table of Contents

What is Amazon Managed Service for Prometheus?

Amazon Managed Service for Prometheus (AMP) is in open preview. The preview is open to all AWS accounts and you do not need to request access. Features may be added or changed before announcing General Availability.
The preview currently supports the following Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (Oregon)
- Europe (Frankfurt)
- Europe (Ireland)

Amazon Managed Service for Prometheus is a serverless, Prometheus-compatible monitoring service for container metrics that makes it easier to securely monitor container environments at scale. With AMP, you can use the same open-source Prometheus data model and query language that you use today to monitor the performance of your containerized workloads, and also enjoy improved scalability, availability, and security without having to manage the underlying infrastructure.

AMP automatically scales the ingestion, storage, and querying of operational metrics as workloads scale up and down. AMP integrates with AWS security services to enable fast and secure access to data.

AMP is designed to be highly available using multiple Availability Zone (Multi-AZ) deployments. Data ingested into a AMP workspace is replicated across three Availability Zones in the same Region.

AMP works with container clusters that run on Amazon Elastic Kubernetes Service and self-managed Kubernetes environments.

With AMP, you use the same open-source Prometheus data model and PromQL query language that you use with Prometheus. Engineering teams can use PromQL to filter, aggregate, and alarm on metrics and quickly gain performance visibility without any code changes. AMP provides flexible query capabilities without the operational cost and complexity.

In the Preview release, retention duration for metrics is 150 days. Any metrics older than 150 days will automatically be deleted.

For information about pricing, see Amazon Managed Service for Prometheus Pricing.

Preview release limitations

This preview release supports ingesting Prometheus metrics into a workspace, and the storage and querying of those metrics. It also supports integration with AWS Identity and Access Management, Amazon Virtual Private Cloud, and AWS CloudTrail.

The following features are not supported in this Preview release, but are planned for future releases:

- Prometheus alert manager
- Prometheus alerting rules and recording rules

- Options for configuring how long metric data is stored
- AWS CloudFormation support
- Support for resource tagging for workspaces

Setting up

Amazon Managed Service for Prometheus (AMP) is in open preview. The preview is open to all AWS accounts and you do not need to request access. Features may be added or changed before announcing General Availability.
The preview currently supports the following Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (Oregon)
- Europe (Frankfurt)
- Europe (Ireland)

Complete the tasks in this section to get set up with AWS for the first time. If you already have an AWS account, skip ahead to Getting started (p. 6).

When you sign up for AWS, your AWS account automatically has access to all services in AWS, including Amazon Managed Service for Prometheus. However, you are charged only for the services that you use.

Get an AWS account and your root user credentials

To access AWS, you must sign up for an AWS account.

To sign up for an AWS account

1. Open https://portal.aws.amazon.com/billing/signup.
2. Follow the online instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.

AWS sends you a confirmation email after the sign-up process is complete. At any time, you can view your current account activity and manage your account by going to https://aws.amazon.com/ and choosing **My Account**.

Creating an IAM user

If your account already includes an IAM user with full AWS administrative permissions, you can skip this section.

When you first create an Amazon Web Services (AWS) account, you begin with a single sign-in identity. That identity has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user*. When you sign in, enter the email address and password that you used to create the account.

Important
We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the best practice of using the root user only to create

your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks. To view the tasks that require you to sign in as the root user, see Tasks that require root user credentials.

To create an administrator user for yourself and add the user to an administrators group (console)

1. Sign in to the IAM console as the account owner by choosing **Root user** and entering your AWS account email address. On the next page, enter your password.

   Note
   We strongly recommend that you adhere to the best practice of using the **`Administrator`** IAM user that follows and securely lock away the root user credentials. Sign in as the root user only to perform a few account and service management tasks.
2. In the navigation pane, choose **Users** and then choose **Add user**.
3. For **User name**, enter **`Administrator`**.
4. Select the check box next to **AWS Management Console access**. Then select **Custom password**, and then enter your new password in the text box.
5. (Optional) By default, AWS requires the new user to create a new password when first signing in. You can clear the check box next to **User must create a new password at next sign-in** to allow the new user to reset their password after they sign in.
6. Choose **Next: Permissions**.
7. Under **Set permissions**, choose **Add user to group**.
8. Choose **Create group**.
9. In the **Create group** dialog box, for **Group name** enter **`Administrators`**.
10. Choose **Filter policies**, and then select **AWS managed - job function** to filter the table contents.
11. In the policy list, select the check box for **AdministratorAccess**. Then choose **Create group**.

    Note
    You must activate IAM user and role access to Billing before you can use the `AdministratorAccess` permissions to access the AWS Billing and Cost Management console. To do this, follow the instructions in step 1 of the tutorial about delegating access to the billing console.
12. Back in the list of groups, select the check box for your new group. Choose **Refresh** if necessary to see the group in the list.
13. Choose **Next: Tags**.
14. (Optional) Add metadata to the user by attaching tags as key-value pairs. For more information about using tags in IAM, see Tagging IAM entities in the *IAM User Guide*.
15. Choose **Next: Review** to see the list of group memberships to be added to the new user. When you are ready to proceed, choose **Create user**.

You can use this same process to create more groups and users and to give your users access to your AWS account resources. To learn about using policies that restrict user permissions to specific AWS resources, see Access management and Example policies.

Signing in as an IAM user

Sign in to the IAM console by choosing **IAM user** and entering your AWS account ID or account alias. On the next page, enter your IAM user name and your password.

Note
For your convenience, the AWS sign-in page uses a browser cookie to remember your IAM user name and account information. If you previously signed in as a different user, choose the sign-in

link beneath the button to return to the main sign-in page. From there, you can enter your AWS account ID or account alias to be redirected to the IAM user sign-in page for your account.

Creating IAM user access keys

Access keys consist of an access key ID and secret access key, which are used to sign programmatic requests that you make to AWS. If you don't have access keys, you can create them from the AWS Management Console. As a best practice, do not use the AWS account root user access keys for any task where it's not required. Instead, create a new administrator IAM user with access keys for yourself.

The only time that you can view or download the secret access key is when you create the keys. You cannot recover them later. However, you can create new access keys at any time. You must also have permissions to perform the required IAM actions. For more information, see Permissions required to access IAM resources in the *IAM User Guide*.

To create access keys for an IAM user

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose **Users**.
3. Choose the name of the user whose access keys you want to create, and then choose the **Security credentials** tab.
4. In the **Access keys** section, choose **Create access key**.
5. To view the new access key pair, choose **Show**. You will not have access to the secret access key again after this dialog box closes. Your credentials will look something like this:

   - Access key ID: AKIAIOSFODNN7EXAMPLE
   - Secret access key: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
6. To download the key pair, choose **Download .csv file**. Store the keys in a secure location. You will not have access to the secret access key again after this dialog box closes.

   Keep the keys confidential in order to protect your AWS account and never email them. Do not share them outside your organization, even if an inquiry appears to come from AWS or Amazon.com. No one who legitimately represents Amazon will ever ask you for your secret key.
7. After you download the `.csv` file, choose **Close**. When you create an access key, the key pair is active by default, and you can use the pair right away.

Related topics

- What is IAM? in the *IAM User Guide*
- AWS security credentials in *AWS General Reference*

Getting started

Amazon Managed Service for Prometheus (AMP) is in open preview. The preview is open to all AWS accounts and you do not need to request access. Features may be added or changed before announcing General Availability.
The preview currently supports the following Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (Oregon)
- Europe (Frankfurt)
- Europe (Ireland)

This section explains how to create Amazon Managed Service for Prometheus workspaces, set up the ingestion of Prometheus metrics to those workspaces, and query those metrics.

Topics

Create a workspace

A *workspace* is a logical and isolated Prometheus server dedicated to Prometheus resources such as metrics. A workspace supports fine-grained access control for authorizing its management such as update, list, describe, and delete, and the ingestion and querying of metrics. You can have one or more workspaces in each Region in your account.

To set up a workspace, follow these steps.

To create a workspace using the AWS CLI

1. Enter the following command to create the workspace. This example creates a workspace named `my-first-workspace`, but you can use a different alias if you want. Workspace aliases are friendly names that help you identify your workspaces. They do not have to be unique. Two workspaces could have the same alias, but all workspaces will have unique workspace IDs, which are generated by AMP.

   ```
   aws amp create-workspace [ --alias my-first-workspace]
   ```

   This command returns the following data:

   - `workspaceId` is the unique ID for this workspace. Make a note of this ID.
   - `arn` is the ARN for this workspace.
   - `status` is the current status of the workspace. Immediately after you create the workspace, this will probably be `CREATING`.

2. If your `create-workspace` command returns a status of `CREATING`, you can then enter the following command to find when the workspace is ready. Replace *`my-workspace-id`* with the value that the `create-workspace` command returned for `workspaceId`

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

   When the `describe-workspace` command returns `ACTIVE` for `status`, the workspace is ready to use.

To create a workspace using the AMP console

1. Open the AMP console at https://console.aws.amazon.com/prometheus/.
2. For **Workspace alias**, enter an alias for the new workspace.

   Workspace aliases are friendly names that help you identify your workspaces. They do not have to be unique. Two workspaces could have the same alias, but all workspaces will have unique workspace IDs, which are generated by AMP.
3. Choose **Create**.

   The workspace details page appears. This displays information including the status, ARN, workspace ID, and endpoint URLs for this workspace for both remote write and queries.

   Initially, the status is probably **CREATING**. Wait until the status is **ACTIVE before you move on to setting up your metric ingestion.**

   Make notes of the URLs displayed for **Endpoint - remote write URL** and **Endpoint - query URL**. You'll need them when you configure your Prometheus server to remote write metrics to this workspace and when you query those metrics.

Ingest Prometheus metrics to the workspace

This section explains how to set up the ingestion of metrics into your workspace.

Ingesting metrics is done using Prometheus remote write. Remote write enables the sending of samples to a remote storage destination. For more details about the remote write configurations, see remote_write in the Prometheus documentation.

Topics

Secure the ingestion of your metrics

Amazon Managed Service for Prometheus provides ways of helping you secure the ingestion of your metrics.

Using AWS PrivateLink with AMP

The network traffic of ingesting the metrics into AMP can be done over a public internet endpoint, or by a VPC endpoint through AWS PrivateLink. Using AWS PrivateLink ensures that the network traffic from your VPCs is secured within the AWS network without going over the public internet. To create a AWS PrivateLink VPC endpoint for AMP, see Using AMP with interface VPC endpoints (p. 47).

Authentication and authorization

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources. Amazon Managed Service for Prometheus integrates with IAM to help you keep your data secure. When you set up AMP, you'll need to create some IAM roles that enable AMP to ingest metrics from Prometheus servers, and that enable Grafana servers to query metrics stored in AMP workspaces. For more information about IAM, see What is IAM?.

Another AWS security feature that can help you set up AMP is the AWS Signature Version 4 signing process (AWS SigV4). Signature Version 4 is the process to add authentication information to AWS requests sent by HTTP. For security, most requests to AWS must be signed with an access key, which consists of an access key ID and secret access key. These two keys are commonly referred to as your security credentials. For more information about SigV4, see Signature Version 4 signing process.

Ingestion methods and how to set them up

There are two options for ingesting metrics into your AMP workspace. Both methods use remote write.

- Ingest metrics from a Prometheus server.

  For this Preview release, we focus on documenting how to set up Prometheus on Amazon Elastic Kubernetes Service. Ingesting from self-managed Kubernetes servers running on Amazon EC2 instances is also supported.
- Use AWS Distro for OpenTelemetry.

Topics

Set up ingestion from an existing Prometheus server in Kubernetes

Amazon Managed Service for Prometheus supports ingesting metrics from Prometheus servers in clusters running Amazon EKS and in self-managed Kubernetes clusters running on Amazon EC2. The detailed instructions in this section are for a Prometheus server in an Amazon EKS cluster. The steps for a self-managed Kubernetes cluster on Amazon EC2 are the same, except that you will need to set up the OIDC provider and IAM roles for service accounts yourself in the Kubernetes cluster.

The instructions in this section use Helm as the Kubernetes package manager.

Topics

Step 1: Set up IAM roles for service accounts

For the method of onboarding that we are documenting, you need to use IAM roles for service accounts in the Amazon EKS cluster where the Prometheus server is running. These roles are also called *service roles*.

With service roles, you can associate an IAM role with a Kubernetes service account. This service account can then provide AWS permissions to the containers in any pod that uses that service account. For more information, see IAM roles for service accounts.

If you have not already set up these roles, follow the instructions at Set up service roles for the ingestion of metrics from Amazon EKS clusters (p. 43) to set up the roles.

Step 2: Upgrade your existing Prometheus server using Helm

The instructions in this section includes setting up remote write and sigv4 to authenticate and authorize the Prometheus server to remote write to your AMP workspace.

Using Prometheus Helm chart

Follow these steps if you are using a Helm chart with Prometheus Server image of version 2.26.0 or later.

To set up remote write from a Prometheus server using Helm chart

1. Create a new remote write section in your Helm configuration file:

   - Replace `${IAM_PROXY_PROMETHEUS_ROLE_ARN}` with the ARN of the **amp-iamproxy-ingest-role** that you created in Step 1: Set up IAM roles for service accounts (p. 8). The role ARN should have the format of `arn:aws:iam::`*`your account ID`*`:role/amp-iamproxy-ingest-role`.
   - Replace `${WORKSPACE_ID}` with your AMP workspace ID.
   - Replace `${AWS_REGION}` with the Region of the AMP workspace (i.e. `us-west-2`).

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

2. Update your existing Prometheus Server configuration using Helm:

   - Replace `prometheus-chart-name` with your Prometheus release name.
   - Replace `prometheus-namespace` with the Kubernetes namespace where your Prometheus Server is installed.
   - Replace `my_prometheus_values_yaml` with the path to your Helm configuration file.
   - Replace `current_helm_chart_version` with the current version of your Prometheus Server Helm chart. You can find the current chart version using helm list command.

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

Set up metrics ingestion using AWS Distro for Open Telemetry

This section describes how to configure the AWS Distro for OpenTelemetry (ADOT) Collector to scrape from a Prometheus-instrumented application, and send the metrics to Amazon Managed Service for Prometheus. For more information about the ADOT Collector, see AWS Distro for OpenTelemetry.

Collecting Prometheus metrics with ADOT involves two OpenTelemetry components: the Prometheus Receiver and the AWS Prometheus Remote Write Exporter.

You can configure the Prometheus Receiver using your existing Prometheus configuration to perform service discovery and metric scraping. The Prometheus Receiver scrapes metrics in the Prometheus exposition format. Any applications or endpoints that you want to scrape should be configured with the Prometheus client library. The Prometheus Receiver supports the full set of Prometheus scraping and re-labeling configurations described in Configuration in the Prometheus documentation. You can paste these configurations directly into your ADOT Collector configurations.

The AWS Prometheus Remote Write Exporter uses the `remote_write` endpoint to send the scraped metrics to your management portal workspace. The HTTP requests to export data will be signed with AWS SigV4, the AWS protocol for secure authentication. For more information, see Signature Version 4 signing process.

The collector automatically discovers Prometheus metrics endpoints on Amazon EKS and uses the configuration found in <kubernetes_sd_config>.

The following demo is an example of this configuration on a cluster running Amazon Elastic Kubernetes Service or self-managed Kubernetes. To perform these steps, you must have AWS credentials from any of the potential options in the default AWS credentials chain. For more information, see Configuring the AWS SDK for Go. This demo use a sample app that is used for integration tests of the process. The sample app exposes metrics at the `/metrics` endpoint, just as the Prometheus client library does.

Prerequisites

Before you begin the ingestion steup steps below, you must set up the your IAM role for service account and trust policy.

To set up the IAM role for service account and trust policy

1. Create the IAM role for the service account by following the steps in Set up service roles for the ingestion of metrics from Amazon EKS clusters (p. 43).

   The ADOT Collector will use this role when it scrapes and exports metrics.
2. Next, edit the trust policy. Open the IAM console at at https://console.aws.amazon.com/iam/.
3. In the left navigation pane choose **Roles** and find the **amp-iamproxy-ingest-role** that you created in step 1.
4. Choose the **Trust relationships** tab and choose **Edit trust relationship**.
5. In the trust relationship policy JSON, replace `aws-amp` with `adot-col` and then choose **Update Trust Policy**. Your resulting trust policy should look like the following:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::account-id:oidc-provider/
oidc.eks.aws_region.amazonaws.com/id/openid"
```

```
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.eks.aws_region.amazonaws.com/id/openid:sub":
 "system:serviceaccount:adot-col:amp-iamproxy-ingest-service-account"
        }
      }
    }
  ]
}
```

6. Choose the **Permissions** tab and make sure that the following permissions policy is attached to the role.

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

Enabling Prometheus metric collection

To enable Prometheus collection on an Amazon EKS or Kubernetes cluster

1. Fork and clone the sample app from the repository at aws-otel-community.

   Then run the following commands.

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

2. Push this image to a registry such as Amazon ECR or DockerHub.
3. Deploy the sample app in the cluster by copying this Kubernetes configuration and applying it. Change the image to the image that you just pushed by replacing `{{PUBLIC_SAMPLE_APP_IMAGE}}` in the `prometheus-sample-app.yaml` file.

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

4. Enter the following command to verify that the sample app has started. In the output of the command, you will see `prometheus-sample-app` in the `NAME` column.

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

5. Start a default instance of the ADOT Collector. To do so, first enter the following command to pull the Kubernetes configuration for ADOT Collector.

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

Then edit the template file, substituting the **remote_write** endpoint for your AMP workspace for `YOUR_ENDPOINT` and your Region for `YOUR_REGION`. Use the **remote_write** endpoint that is displayed in the AMP console when you look at your workspace details.

You'll also need to change `YOUR_ACCOUNT_ID` in the service account section of the Kubernetes configuration to your AWS account ID.

In this example, the ADOT Collector configuration uses an annotation (`scrape=true`) to tell which target endpoints to scrape. This allows the ADOT Collector to distinguish the sample app endpoint from kube-system endpoints in your cluster. You can remove this from the re-label configurations if you want to scrape a different sample app.

6. Enter the following command to deploy the sample app.

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

7. Enter the following command to verify that the ADOT collector has started. Look for `adot-col` in the `NAMESPACE` column.

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

8. Verify that the pipeline works by using the logging exporter. Our example template is already integrated with the logging exporter. Enter the following commands.

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

Some of the scraped metrics from the sample app will look like the following example.

```
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. To test whether AMP received the metrics, use `awscurl`. This tool enables you to send HTTP requests through the command line with AWS Sigv4 authentication, so you must have AWS credentials set up locally with the correct permissions to query from AMP For instructions on installing `awscurl`, see awscurl.

In the following command, replace `AMP_REGION`, and `AMP_ENDPOINT` with the information for your AMP workspace.

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

If you receive a metric as the response, that means your pipeline setup has been successful, and the metric has successfully propagated from the sample app into AMP.

Cleaning up

To clean up this demo, enter the following commands.

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

Advanced configuration

The Prometheus Receiver supports the full set of Prometheus scraping and re-labeling configurations described in Configuration in the Prometheus documentation. You can paste these configurations directly into your ADOT Collector configurations.

The configuration for the Prometheus Receiver includes your service discovery, scraping configurations, and re-labeling configurations. The receiver configurations looks like the following.

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

The following is an example configuration.

```
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
```

If you have existing an Prometheus configuration, you must replace the `$` characters with `$$` to avoid having the values replaced with environment variables. *This is especially important for the replacement value of the relabel_configurations. For example, if you start with the following relabel_configuration:

```
relabel_configs:
- source_labels:
 [__meta_kubernetes_ingress_scheme,__address__,__meta_kubernetes_ingress_path]
  regex: (.+);(.+);(.+)
```

```
  replacement: ${1}://${2}${3}
  target_label: __param_target
```

It would become the following:

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

AWS Prometheus remote write exporter

The configuration for the AWS Prometheus Remote Write Exporter is simpler than the Prometheus receiver. At this stage in the pipeline, metrics have already been ingested, and we're ready to export this data to AMP. The minimum requirement for a successful configuration to communicate with AMP is shown in the following example.

```
exporters:
  awsprometheusremotewrite:
    endpoint: "https://aws-managed-prometheus-endpoint/v1/api/remote_write"
    aws_auth:
      service: "aps"
      region: "user-region"
```

This configuration sends an HTTPS request that is signed by AWS SigV4 using AWS credentials from the default AWS credentials chain. For more information, see Configuring the AWS SDK for Go. You must specify the service to be `aps`.

Regardless of the method of deployment, the ADOT collector must have access to one of the listed options in the default AWS credentials chain. The AWS Prometheus Remote Write Exporter depends on the AWS SDK for Go AWS Go SDK and uses it to fetch credentials and authenticate. You must ensure that these credentials have remote write permissions for AMP.

Send high-availability data from your Kubernetes cluster (Deduplication)

Amazon Managed Service for Prometheus supports the use of multiple Prometheus instances that monitor the same set of metrics and send them to a single AMP workspace, for high-availability purposes. AMP deduplicates the metrics. Each Prometheus instance can be either a Prometheus server or an AWS Open Distro for Telemetry agent.

AMP sets one instance as the leader replica and ingests from only that replica. If AMP stops receiving samples from the leader for a 30-second period, it automatically switches to another leader and begins ingesting metrics from the new leader.

To set up a high-availability configuration, you need to specify external labels on all of the instances of a high-availability group to identify them to AMP. Use the `cluster` label to identify a Prometheus instance agent as part of a high-availability group, and use the `__replica__` label to separately identify each replica in the group. For example, on one replica of a pair called `prom-team1`, you would include the following external labels.

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

On the other, you would use the following.

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

Note
In the `__replica__` external labels, there are two underscore characters before the word `replica` and two underscore characters after it.

When AMP stores samples from the high-availability replicas with these labels, it strips the replica label when the sample is accepted. This means that you'll only have a 1:1 series mapping for your current series instead of a series per replica. The cluster label is kept.

Query your Prometheus metrics

Now that metrics are being ingested to the workspace, you can query them. You can use a service such as Grafana to query the metrics, or you can use Amazon Managed Service for Prometheus APIs.

You perform your queries using the standard Prometheus query language, PromQL. For more information about PromQL and its syntax, see Querying Prometheus in the Prometheus documentation.

Topics

Securing your metric queries

Amazon Managed Service for Prometheus provides ways of helping you secure the querying of your metrics.

Using AWS PrivateLink with AMP

The network traffic of ingesting the metrics into AMP can be done over a public internet endpoint, or by a VPC endpoint through AWS PrivateLink. Using AWS PrivateLink ensures that the network traffic from your VPCs is secured within the AWS network without going over the public internet. To create a AWS PrivateLink VPC endpoint for AMP, see Using AMP with interface VPC endpoints (p. 47).

Authentication and authorization

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources. Amazon Managed Service for Prometheus integrates with IAM to help you keep your data secure. When you set up AMP, you'll need to create some IAM roles that enable Grafana servers to query metrics stored in AMP workspaces. For more information about IAM, see What is IAM?.

Another AWS security feature that can help you set up AMP is the AWS Signature Version 4 signing process (AWS SigV4). Signature Version 4 is the process to add authentication information to AWS requests sent by HTTP. For security, most requests to AWS must be signed with an access key, which consists of an access key ID and secret access key. These two keys are commonly referred to as your security credentials. For more information about SigV4, see Signature Version 4 signing process.

Set up Grafana open source or Grafana Enterprise for use with AMP

Amazon Managed Service for Prometheus supports the use of Grafana version 7.3.5 and later to query metrics in a AMP workspace. Versions 7.3.5 and later include support for AWS Signature Version 4 (SigV4) authentication.

For instructions for setting up a standalone Grafana using the tar.gz or zip file, see Install Grafana in the Grafana documentation. If you install a new standalone Grafana, you will be prompted for username and password. The default is **`admin/admin`**. You will be prompted to change the password after you log in for the first time. For more information, see Getting started with Grafana in the Grafana documentation.

To check your Grafana version, enter the following command.

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

To set up Grafana to work with AMP, you must be logged on to an account that has the **AmazonPrometheusQueryAccess** policy or the `aps:QueryMetrics` and `aps:GetMetricMetaData` permissions. For more information, see IAM permissions and policies (p. 39).

Set up AWS SigV4

Amazon Managed Service for Prometheus is integrated with AWS Identity and Access Management (IAM) to ensure that all calls to Prometheus APIs, such as query and ingest, are secured with IAM credentials. By default, the Prometheus data source in Grafana assumes that Prometheus requires no authentication. To enable Grafana to take advantage of AMP authentication and authorization capabilities, you will need to enable SigV4 authentication support in the Grafana data source. Follow the steps on this page when you are using a self-managed Grafana open-source or a Grafana enterprise server. If you are using Amazon Managed Service for Grafana, SIGv4 authentication is fully automated. For more information about Amazon Managed Service for Grafana, see What is Amazon Managed Service for Grafana?.

To enable SigV4 on Grafana, start Grafana with the `AWS_SDK_LOAD_CONFIG` and `GF_AUTH_SIGV4_AUTH_ENABLED` environment variables set to `true`. The `GF_AUTH_SIGV4_AUTH_ENABLED` environment variable overrides the default configuration for Grafana to enable SigV4 support. For more information, see Configuration in the Grafana documentation.

Linux

To enable SigV4 on a standalone Grafana server on Linux, enter the following commands.

```
export AWS_SDK_LOAD_CONFIG=true
```

```
export GF_AUTH_SIGV4_AUTH_ENABLED=true
```

```
cd grafana_install_directory
```

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

Windows

To enable SigV4 on a standalone Grafana on Windows using the Windows command prompt, enter the following commands.

```
set AWS_SDK_LOAD_CONFIG=true
```

```
set GF_AUTH_SIGV4_AUTH_ENABLED=true
```

```
cd grafana_install_directory
```

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

Add the Prometheus data source in Grafana

The following steps explain how to set up the Prometheus data source in Grafana to query your AMP metrics.

To add the Prometheus data source in your Grafana server

1. Open the Grafana console.
2. Under **Configurations**, choose **Data sources**.
3. Choose **Add data source**.
4. Choose **Prometheus**.
5. For the HTTP URL, specify the **Endpoint - query URL** displayed in the AMP workspace details page in the AMP console.
6. In the HTTP URL that you just specified, remove the `/api/v1/query` string that is appended to the URL, because the Prometheus data source will automatically append it.

   The correct URL should look similar to **https://aps-workspaces.us-west-2.amazonaws.com/workspaces/ws-1234a5b6-78cd-901e-2fgh-3i45j6k178l9**.
7. Under **Auth**, select the toggle for **SigV4 Auth** to enable it.
8. You can either configure SigV4 authorization by specifying your long-term credentials directly in Grafana, or by using a default provider chain. Specifying your long-term credentials directly gets you started quicker, and the following steps give those instructions first. Once you are more familiar with using Grafana with AMP, we recommend that you use a default provider chain, because it provides better flexibility and security. For more information about setting up your default provider chain, see Specifying Credentials.
   - To use your long-term credentials directly, do the following:
     a. Under **SigV4 Auth Details**, for **Authentication Provider** choose **Access & secret key**.
     b. For **Access Key ID**, enter your AWS access key ID.
     c. For **Secret Access Key**, enter your AWS secret access key.
     d. Leave the **Assume Role ARN** and **External ID** fields blank.
     e. For **Default Region**, choose the Region of your AMP workspace. This Region should match the Region contained in the URL that you listed in step 5.
     f. Choose **Save & Test**.

        You should see the following message: **Data source is working**
   - To use a default provider chain instead (recommended for a production environment), do the following:
     a. Under **SigV4 Auth Details**, for **Authentication Provider** choose **AWS SDK Default**.
     b. Leave the **Assume Role ARN** and **External ID** fields blank.
     c. For **Default Region**, choose the Region of your AMP workspace. This Region should match the Region contained in the URL that you listed in step 5.

d. Choose **Save & Test**.

You should see the following message: **Data source is working**

9. Test a PromQL query against the new data source:

a. Choose **Explore**.

b. Run a sample PromQL query such as:

```
prometheus_tsdb_head_series
```

Troubleshooting if Save & Test doesn't work

In the previous procedure, if you see an error when you choose **Save & Test**, check the following.

HTTP Error Not Found

Make sure that the workspace ID in the URL is correct.

HTTP Error Forbidden

This error means that the credentials are not valid. Check the following:

- Check that the Region specified in **Default Region** is correct.
- Check your credential for typos.
- Make sure that the credential that you are using has the **AmazonPrometheusQueryAccess** policy. For more information, see IAM permissions and policies (p. 39).
- Make sure that the credential that you are using has access to this AMP workspace.

HTTP Error Bad Gateway

Look at the Grafana server log to troubleshoot this error. For more information, see Troubleshooting in the Grafana documentation.

If you see **`Error http: proxy error: NoCredentialProviders: no valid providers in chain`**, the default credential provider chain was not able to find a valid AWS credential to use. Make sure you have set up your credentials as documented in Specifying Credentials. If you want to use a shared configuration, make sure that the `AWS_SDK_LOAD_CONFIG` environment is set to `true`.

Query using Grafana running in an Amazon EKS cluster

Amazon Managed Service for Prometheus supports the use of Grafana version 7.3.5 and later to query metrics in a AMP workspace. Versions 7.3.5 and later include support for AWS Signature Version 4 (SigV4) authentication.

To set up a Grafana server to work with AMP, you must be logged on to an account that has the **AmazonPrometheusQueryAccess** policy or the `aps:QueryMetrics` and `aps:GetMetricMetaData` permissions. For more information, see IAM permissions and policies (p. 39).

Set up AWS SigV4

Grafana has added a new feature to support AWS Signature Version 4 (SigV4) authentication. For more information, see Signature Version 4 signing process. This feature is not enabled by default on Grafana

servers. The following instructions for enabling this feature assume that you are using Helm to deploy Grafana on a Kubernetes cluster.

To enable SigV4 on your Grafana 7.3.5 or later server

1. Create a new update file to override your Grafana configuration, and name it `amp_query_override_values.yaml`.
2. Enter the following content into the file, and save the file. Replace *account-id* with the AWS account ID where the Grafana server is running.

```
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
```

   In that YAML file content, `amp-iamproxy-query-role` is the name of the role that you will create in the next section, Set up IAM roles for service accounts (p. 19). You can replace this role with your own role name if you already have a role created for querying your AMP workspace.

You will use this file later, in Upgrade the Grafana server using Helm (p. 20).

Set up IAM roles for service accounts

If you are using a Grafana server in an Amazon EKS cluster, we recommend that you use IAM roles for service accounts, also known as service roles, for your access control. When you do this to associate an IAM role with a Kubernetes service account, the service account can then provide AWS permissions to the containers in any pod that uses that service account. For more information, see IAM roles for service accounts.

If you have not already set up these service roles for querying, follow the instructions at Set up IAM roles for service accounts for the querying of metrics (p. 45) to set up the roles.

You then need to add the Grafana service account in the conditions of the trust relationship.

To add the Grafana service account in the conditions of the trust relationship

1. From a terminal window, determine the namespace and the service account name for your Grafana server. For example, you could use the following command.

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

2. In the Amazon EKS console, open the IAM role for service accounts that is associated with the EKS cluster.
3. Choose **Edit trust relationship**.
4. Update the **Condition** to include the Grafana namespace and the Grafana service account name that you found in the output of the command in step 1. The following is an example.

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

```
        "Federated": "arn:aws:iam::account-id:oidc-provider/
oidc.eks.aws_region.amazonaws.com/id/openid"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.eks.aws_region.amazonaws.com/id/openid:sub": [
            "system:serviceaccount:aws-amp:amp-iamproxy-query-service-account",
            "system:serviceaccount:grafana-namespace:grafana-service-account-name"
          ]
        }
      }
    }
  ]
}
```

5. Choose **Update trust policy**.

Upgrade the Grafana server using Helm

This step upgrades the Grafana server to use the entries that you added to the `amp_query_override_values.yaml` file in the previous section.

Run the following commands. For more information about Helm charts for Grafana, see Grafana Community Kubernetes Helm Charts.

```
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
```

Add the Prometheus data source in Grafana

The following steps explain how to set up the Prometheus data source in Grafana to query your AMP metrics.

To add the Prometheus data source in your Grafana server

1. Open the Grafana console.
2. Under **Configurations**, choose **Data sources**.
3. Choose **Add data source**.
4. Choose **Prometheus**.
5. For the HTTP URL, specify the **Endpoint - query URL** displayed in the AMP workspace details page in the AMP console.
6. In the HTTP URL that you just specified, remove the `/api/v1/query` string that is appended to the URL, because the Prometheus data source will automatically append it.
7. Under **Auth**, select the toggle for **SigV4 Auth** to enable it.

   Leave the **Assume Role ARN** and **External ID** fields blank. Then for **Default Region**, select the Region where your AMP workspace is.
8. Choose **Save & Test**.

   You should see the following message: **Data source is working**
9. Test a PromQL query against the new data source:

   a. Choose **Explore**.

b. Run a sample PromQL query such as:

```
prometheus_tsdb_head_series
```

Query using Prometheus-compatible APIs

Amazon Managed Service for Prometheus supports several Prometheus-compatible APIs that you can use to query your metrics. For more information about all the available Prometheus-compatible APIs, see Prometheus-compatible APIs (p. 61).

When you use these APIs to query your metrics, the requests must be signed with the AWS Signature Version 4 signing process. You can set up AWS Signature Version 4 as a sidecar or as a pod. For more information, see aws-sigv4-proxy.

Managing workspaces

Amazon Managed Service for Prometheus (AMP) is in open preview. The preview is open to all AWS accounts and you do not need to request access. Features may be added or changed before announcing General Availability.
The preview currently supports the following Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (Oregon)
- Europe (Frankfurt)
- Europe (Ireland)

Use the procedures in this section to create and manage your Amazon Managed Service for Prometheus workspaces.

Topics

Create a workspace

Follow these steps to create a AMP workspace.

To create a workspace using the AWS CLI

1. Enter the following command to create the workspace. This example creates a workspace named `my-first-workspace`, but you can use a different alias if you want. Workspace aliases are friendly names that help you identify your workspaces. They do not have to be unique. Two workspaces can have the same alias, but all workspaces have unique workspace IDs, which are generated by AMP.

   ```
   aws amp create-workspace [ --alias my-first-workspace]
   ```

   This command returns the following data:

   - `workspaceId` is the unique ID for this workspace. Make a note of this ID.
   - `arn` is the ARN for this workspace.
   - `status` is the current status of the workspace. Immediately after you create the workspace, this will probably be `CREATING`.
2. If your `create-workspace` command returns a status of `CREATING`, you can then enter the following command to determine when the workspace is ready. Replace *`my-workspace-id`* with the value that the `create-workspace` command returned for `workspaceId`.

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

When the `describe-workspace` command returns `ACTIVE` for `status`, the workspace is ready to use.

To create a workspace using the AMP console

1. Open the AMP console at https://console.aws.amazon.com/prometheus/.
2. For **Workspace alias**, enter an alias for the new workspace.

   Workspace aliases are friendly names that help you identify your workspaces. They do not have to be unique. Two workspaces can have the same alias, but all workspaces have unique workspace IDs, which are generated by AMP.
3. Choose **Create**.

   The workspace details page appears. This displays information including the status, ARN, workspace ID, and endpoint URLs for this workspace for both remote write and queries.

   Initially, the status is probably **CREATING**. Wait until the status is **ACTIVE** before you move on to setting up your metric ingestion.

   Make note of the URLs that are displayed for **Endpoint - remote write URL** and **Endpoint - query URL**. You'll need them when you configure your Prometheus server to remote write metrics to this workspace and when you query those metrics.

For information about how to ingest metrics into the workspace, see Ingest Prometheus metrics to the workspace (p. 7).

Edit a workspace

You can edit a workspace to change its alias. To change the workspace alias using the AWS CLI, enter the following command.

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

To edit a workspace using the AMP console

1. Open the AMP console at https://console.aws.amazon.com/prometheus/.
2. In the upper left corner of the page, choose the menu icon and then choose **All workspaces**.
3. Choose the workspace ID of the workspace that you want to edit, and then choose **Edit**.
4. Enter a new alias for the workspace and then choose **Save**.

Find your workspace ARN

You can find the ARN of your AMP workspace by using either the console or the AWS CLI.

To find your workspace ARN using the AMP console

1. Open the AMP console at https://console.aws.amazon.com/prometheus/.
2. In the upper left corner of the page, choose the menu icon and then choose **All workspaces**.
3. Choose the **Workspace ID** of the workspace.

The workspace ARN is displayed under **ARN**.

To use the AWS CLI to find your workspace ARN, enter the following command.

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

Find the value of `arn` in the results.

Delete a workspace

When you delete a workspace, the data that has been ingested into it is not immediately deleted. It will be permanently deleted within one month.

To delete a workspace using the AWS CLI, enter the following command.

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

To delete a workspace using the AMP console

1. Open the AMP console at https://console.aws.amazon.com/prometheus/.
2. In the upper left corner of the page, choose the menu icon and then choose **All workspaces**.
3. Choose the workspace ID of the workspace that you want to delete, and then choose **Delete**.
4. Enter **`delete`** in the confirmation box, and choose **Delete**.

Security in Amazon Managed Service for Prometheus

Amazon Managed Service for Prometheus (AMP) is in open preview. The preview is open to all AWS accounts and you do not need to request access. Features may be added or changed before announcing General Availability.
The preview currently supports the following Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (Oregon)
- Europe (Frankfurt)
- Europe (Ireland)

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from data centers and network architectures that are built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to Amazon Managed Service for Prometheus, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using AMP. The following topics show you how to configure AMP to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your AMP resources.

Topics

- Logging AMP API calls using AWS CloudTrail (p. 48)

Data protection in AMP

Amazon Managed Service for Prometheus conforms to the AWS shared responsibility model, which includes regulations and guidelines for data protection. AWS is responsible for protecting the global infrastructure that runs all the AWS services. AWS maintains control over data hosted on this infrastructure, including the security configuration controls for handling customer content and personal data. AWS customers and APN Partners, acting either as data controllers or data processors, are responsible for any personal data that they put in the AWS Cloud.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual user accounts with AWS Identity and Access Management (IAM), so that each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing personal data that is stored in Amazon S3.

We strongly recommend that you never put sensitive identifying information, such as your customers' account numbers, into free-form fields such as a **Name** field. This includes when you work with AMP or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into AMP or other services might get picked up for inclusion in diagnostic logs. When you provide a URL to an external server, don't include credentials information in the URL to validate your request to that server.

For more information about data protection, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

Data collected by AMP

Amazon Managed Service for Prometheus collects and stores operational metrics that you configure to be sent to AMP from Prometheus servers running in your account. This data includes the following:

- Metric values
- Labels for the metrics, which are arbitrary key-value pairs that help identify and classify the data
- Timestamps for the data samples

Data from different customers is isolated from other customer data by way of a unique tenant ID, which limits what data is accessible. The tenant ID can't be mutated by customers.

Customer data is stored on Amazon Elastic Block Store for short-term storage and forwarded to Amazon Simple Storage Service for long-term storage.

The data that AMP stores is encrypted with AWS Key Management Service. The key is managed by AMP. In this Preview release, AMP does not support customer master keys (CMKs).

Data in transit is automatically encrypted with HTTPS. AMP uses internal HTTPS to secure connections between Availability Zones within a Region.

Identity and Access Management for Amazon Managed Service for Prometheus

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use AMP resources. IAM is an AWS service that you can use with no additional charge.

Topics

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in AMP.

Service user – If you use the AMP service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more AMP features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in AMP, see Troubleshooting Amazon Managed Service for Prometheus identity and access (p. 37).

Service administrator – If you're in charge of AMP resources at your company, you probably have full access to AMP. It's your job to determine which AMP features and resources your employees should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with AMP, see How Amazon Managed Service for Prometheus works with IAM (p. 31).

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to AMP. To view example AMP identity-based policies that you can use in IAM, see Identity-based policy examples for Amazon Managed Service for Prometheus (p. 35).

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. For more information about signing in using the AWS Management Console, see Signing in to the AWS Management Console as an IAM user or root user in the *IAM User Guide*.

You must be *authenticated* (signed in to AWS) as the AWS account root user, an IAM user, or by assuming an IAM role. You can also use your company's single sign-on authentication or even sign in using Google or Facebook. In these cases, your administrator previously set up identity federation using IAM roles. When you access AWS using credentials from another company, you are assuming a role indirectly.

To sign in directly to the AWS Management Console, use your password with your root user email address or your IAM user name. You can access AWS programmatically using your root user or IAM users access keys. AWS provides SDK and command line tools to cryptographically sign your request

using your credentials. If you don't use AWS tools, you must sign the request yourself. Do this using *Signature Version 4*, a protocol for authenticating inbound API requests. For more information about authenticating requests, see Signature Version 4 signing process in the *AWS General Reference*.

Regardless of the authentication method that you use, you might also be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Using multi-factor authentication (MFA) in AWS in the *IAM User Guide*.

AWS account root user

When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you do not use the root user for your everyday tasks, even the administrative ones. Instead, adhere to the best practice of using the root user only to create your first IAM user. Then securely lock away the root user credentials and use them to perform only a few account and service management tasks.

IAM users and groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. An IAM user can have long-term credentials such as a user name and password or a set of access keys. To learn how to generate access keys, see Managing access keys for IAM users in the *IAM User Guide*. When you generate access keys for an IAM user, make sure you view and securely save the key pair. You cannot recover the secret access key in the future. Instead, you must generate a new access key pair.

An *IAM group* is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see When to create an IAM user (instead of a role) in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by switching roles. You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see Using IAM roles in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Temporary IAM user permissions** – An IAM user can assume an IAM role to temporarily take on different permissions for a specific task.
- **Federated user access** – Instead of creating an IAM user, you can use existing identities from AWS Directory Service, your enterprise user directory, or a web identity provider. These are known as *federated users*. AWS assigns a role to a federated user when access is requested through an identity provider. For more information about federated users, see Federated users and roles in the *IAM User Guide*.
- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access.

However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see How IAM roles differ from resource-based policies in the *IAM User Guide*.

- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.
  - **Principal permissions** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. Policies grant permissions to a principal. When you use some services, you might perform an action that then triggers another action in a different service. In this case, you must have permissions to perform both actions. To see whether an action requires additional dependent actions in a policy, see Actions, resources, and condition keys for Amazon Managed Service for Prometheus in the *Service Authorization Reference*.
  - **Service role** – A service role is an IAM role that a service assumes to perform actions on your behalf. Service roles provide access only within your account and cannot be used to grant access to services in other accounts. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Creating a role to delegate permissions to an AWS service in the *IAM User Guide*.
  - **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your IAM account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances in the *IAM User Guide*.

To learn whether to use IAM roles or IAM users, see When to create an IAM role (instead of a user) in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to IAM identities or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. You can sign in as the root user or an IAM user, or you can assume an IAM role. When you then make a request, AWS evaluates the related identity-based or resource-based policies. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

Every IAM entity (user or role) starts with no permissions. In other words, by default, users can do nothing, not even change their own password. To give a user permission to do something, an administrator must attach a permissions policy to a user. Or the administrator can add the user to a group that has the intended permissions. When an administrator gives permissions to a group, all users in that group are granted those permissions.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Creating IAM policies in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see Choosing between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Access control lists (ACLs)

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see Access control list (ACL) overview in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see How SCPs work in the *AWS Organizations User Guide*.
- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's

permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

How Amazon Managed Service for Prometheus works with IAM

Before you use IAM to manage access to AMP, learn what IAM features are available to use with AMP.

IAM features you can use with Amazon Managed Service for Prometheus

IAM feature	AMP support
Identity-based policies (p. 31)	Yes
Resource-based policies (p. 32)	No
Policy actions (p. 32)	Yes
Policy resources (p. 33)	Yes
Policy condition keys (p. 33)	No
ACLs (p. 34)	No
ABAC (tags in policies) (p. 34)	No
Temporary credentials (p. 34)	Yes
Principal permissions	No
Service roles (p. 35)	No
Service-linked roles (p. 35)	No

To get a high-level view of how AMP and other AWS services work with most IAM features, see AWS services that work with IAM in the *IAM User Guide*.

Identity-based policies for AMP

Supports identity-based policies	Yes

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Creating IAM policies in the *IAM User Guide*.

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. You can't specify the principal in an identity-based policy because it applies to the user or role to which it is attached. To learn about all of the elements that you can use in a JSON policy, see IAM JSON policy elements reference in the *IAM User Guide*.

Identity-based policy examples for AMP

To view examples of AMP identity-based policies, see Identity-based policy examples for Amazon Managed Service for Prometheus (p. 35).

Resource-based policies within AMP

Supports resource-based policies	No

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. Adding a cross-account principal to a resource-based policy is only half of establishing the trust relationship. When the principal and the resource are in different AWS accounts, an IAM administrator in the trusted account must also grant the principal entity (user or role) permission to access the resource. They grant permission by attaching an identity-based policy to the entity. However, if a resource-based policy grants access to a principal in the same account, no additional identity-based policy is required. For more information, see How IAM roles differ from resource-based policies in the *IAM User Guide*.

Policy actions for AMP

Supports policy actions	Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Policy actions usually have the same name as the associated AWS API operation. There are some exceptions, such as *permission-only actions* that don't have a matching API operation. There are also some operations that require multiple actions in a policy. These additional actions are called *dependent actions*.

Include actions in a policy to grant permissions to perform the associated operation.

To see a list of AMP actions, see Actions defined by Amazon Managed Service for Prometheus in the *Service Authorization Reference*.

Policy actions in AMP use the following prefix before the action:

```
aps
```

To specify multiple actions in a single statement, separate them with commas.

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

To view examples of AMP identity-based policies, see Identity-based policy examples for Amazon Managed Service for Prometheus (p. 35).

Policy resources for AMP

Supports policy resources	Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. Statements must include either a `Resource` or a `NotResource` element. As a best practice, specify a resource using its Amazon Resource Name (ARN). You can do this for actions that support a specific resource type, known as *resource-level permissions*.

For actions that don't support resource-level permissions, such as listing operations, use a wildcard (*) to indicate that the statement applies to all resources.

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

To see a list of AMP resource types and their ARNs, see Resources defined by Amazon Managed Service for Prometheus in the *Service Authorization Reference*. To learn with which actions you can specify the ARN of each resource, see Actions defined by Amazon Managed Service for Prometheus.

To view examples of AMP identity-based policies, see Identity-based policy examples for Amazon Managed Service for Prometheus (p. 35).

Policy condition keys for AMP

Supports policy condition keys	No

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Condition` element (or `Condition` *block*) lets you specify conditions in which a statement is in effect. The `Condition` element is optional. You can create conditional expressions that use condition operators, such as equals or less than, to match the condition in the policy with values in the request.

If you specify multiple `Condition` elements in a statement, or multiple keys in a single `Condition` element, AWS evaluates them using a logical `AND` operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical `OR` operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see IAM policy elements: variables and tags in the *IAM User Guide*.

AWS supports global condition keys and service-specific condition keys. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

To see a list of AMP condition keys, see Condition keys for Amazon Managed Service for Prometheus in the *Service Authorization Reference*. To learn with which actions and resources you can use a condition key, see Actions defined by Amazon Managed Service for Prometheus.

To view examples of AMP identity-based policies, see Identity-based policy examples for Amazon Managed Service for Prometheus (p. 35).

Access control lists (ACLs) in AMP

Supports ACLs	No

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Attribute-based access control (ABAC) with AMP

Supports ABAC (tags in policies)	No

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes. In AWS, these attributes are called *tags*. You can attach tags to IAM entities (users or roles) and to many AWS resources. Tagging entities and resources is the first step of ABAC. Then you design ABAC policies to allow operations when the principal's tag matches the tag on the resource that they are trying to access.

ABAC is helpful in environments that are growing rapidly and helps with situations where policy management becomes cumbersome.

To control access based on tags, you provide tag information in the condition element of a policy using the `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys.

For more information about ABAC, see What is ABAC? in the *IAM User Guide*. To view a tutorial with steps for setting up ABAC, see Use attribute-based access control (ABAC) in the *IAM User Guide*.

Using Temporary credentials with AMP

Supports temporary credentials	Yes

Some AWS services don't work when you sign in using temporary credentials. For additional information, including which AWS services work with temporary credentials, see AWS services that work with IAM in the *IAM User Guide*.

You are using temporary credentials if you sign in to the AWS Management Console using any method except a user name and password. For example, when you access AWS using your company's single sign-on (SSO) link, that process automatically creates temporary credentials. You also automatically create temporary credentials when you sign in to the console as a user and then switch roles. For more information about switching roles, see Switching to a role (console) in the *IAM User Guide*.

You can manually create temporary credentials using the AWS CLI or AWS API. You can then use those temporary credentials to access AWS. AWS recommends that you dynamically generate temporary

credentials instead of using long-term access keys. For more information, see Temporary security credentials in IAM.

Service roles for AMP

Supports service roles	No

A service role is an IAM role that a service assumes to perform actions on your behalf. Service roles provide access only within your account and cannot be used to grant access to services in other accounts. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Creating a role to delegate permissions to an AWS service in the *IAM User Guide*.

> **Warning**
> Changing the permissions for a service role might break AMP functionality. Edit service roles only when AMP provides guidance to do so.

Service-linked roles for AMP

Supports service-linked roles	No

A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your IAM account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

For details about creating or managing service-linked roles, see AWS services that work with IAM. Find a service in the table that includes a `Yes` in the **Service-linked role** column. Choose the **Yes** link to view the service-linked role documentation for that service.

Identity-based policy examples for Amazon Managed Service for Prometheus

By default, IAM users and roles don't have permission to create or modify AMP resources. They also can't perform tasks using the AWS Management Console, AWS CLI, or AWS API. An IAM administrator must create IAM policies that grant users and roles permission to perform actions on the resources that they need. The administrator must then attach those policies to the IAM users or groups that require those permissions.

To learn how to create an IAM identity-based policy using these example JSON policy documents, see Creating policies on the JSON tab in the *IAM User Guide*.

Topics

Policy best practices

Identity-based policies are very powerful. They determine whether someone can create, access, or delete AMP resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started using AWS managed policies** – To start using AMP quickly, use AWS managed policies to give your employees the permissions they need. These policies are already available in your account and are maintained and updated by AWS. For more information, see Get started using permissions with AWS managed policies in the *IAM User Guide*.
- **Grant least privilege** – When you create custom policies, grant only the permissions required to perform a task. Start with a minimum set of permissions and grant additional permissions as necessary. Doing so is more secure than starting with permissions that are too lenient and then trying to tighten them later. For more information, see Grant least privilege in the *IAM User Guide*.
- **Enable MFA for sensitive operations** – For extra security, require IAM users to use multi-factor authentication (MFA) to access sensitive resources or API operations. For more information, see Using multi-factor authentication (MFA) in AWS in the *IAM User Guide*.
- **Use policy conditions for extra security** – To the extent that it's practical, define the conditions under which your identity-based policies allow access to a resource. For example, you can write conditions to specify a range of allowable IP addresses that a request must come from. You can also write conditions to allow requests only within a specified date or time range, or to require the use of SSL or MFA. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.

Using the AMP console

To access the Amazon Managed Service for Prometheus console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the AMP resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (IAM users or roles) with that policy.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that you're trying to perform.

To ensure that users and roles can still use the AMP console, also attach the AMP `ConsoleAccess` or `ReadOnly` AWS managed policy to the entities. For more information, see Adding permissions to a user in the *IAM User Guide*.

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

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

Troubleshooting Amazon Managed Service for Prometheus identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with AMP and IAM.

Topics

I am not authorized to perform an action in AMP

If the AWS Management Console tells you that you're not authorized to perform an action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a fictional *`my-example-widget`* resource but does not have the fictional `aps:GetWidget` permissions.

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

In this case, Mateo asks his administrator to update his policies to allow him to access the *`my-example-widget`* resource using the `aps:GetWidget` action.

I am not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your user name and password. Ask that person to update your policies to allow you to pass a role to AMP.

Some AWS services allow you to pass an existing role to that service, instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in AMP. However, the action requires the service to have permissions granted by a service role. Mary does not have permissions to pass the role to the service.

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

In this case, Mary asks her administrator to update her policies to allow her to perform the `iam:PassRole` action.

I want to view my access keys

After you create your IAM user access keys, you can view your access key ID at any time. However, you can't view your secret access key again. If you lose your secret key, you must create a new access key pair.

Access keys consist of two parts: an access key ID (for example, `AKIAIOSFODNN7EXAMPLE`) and a secret access key (for example, `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Like a user name and password, you must use both the access key ID and secret access key together to authenticate your requests. Manage your access keys as securely as you do your user name and password.

> **Important**
> Do not provide your access keys to a third party, even to help find your canonical user ID. By doing this, you might give someone permanent access to your account.

When you create an access key pair, you are prompted to save the access key ID and secret access key in a secure location. The secret access key is available only at the time you create it. If you lose your secret access key, you must add new access keys to your IAM user. You can have a maximum of two access keys. If you already have two, you must delete one key pair before creating a new one. To view instructions, see Managing access keys in the *IAM User Guide*.

I'm an administrator and want to allow others to access AMP

To allow others to access AMP, you must create an IAM entity (user or role) for the person or application that needs access. They will use the credentials for that entity to access AWS. You must then attach a policy to the entity that grants them the correct permissions in AMP.

To get started right away, see Creating your first IAM delegated user and group in the *IAM User Guide*.

I want to allow people outside of my AWS account to access my AMP resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether AMP supports these features, see How Amazon Managed Service for Prometheus works with IAM (p. 31).
- To learn how to provide access to your resources across AWS accounts that you own, see Providing access to an IAM user in another AWS account that you own in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see Providing access to AWS accounts owned by third parties in the *IAM User Guide*.
- To learn how to provide access through identity federation, see Providing access to externally authenticated users (identity federation) in the *IAM User Guide*.

- To learn the difference between using roles and resource-based policies for cross-account access, see How IAM roles differ from resource-based policies in the *IAM User Guide*.

IAM permissions and policies

Access to Amazon Managed Service for Prometheus actions and data requires credentials. Those credentials must have permissions to perform the actions and to access the AWS resources, such as retrieving AMP data about your cloud resources. The following sections provide details about how you can use AWS Identity and Access Management (IAM) and AMP to help secure your resources, by controlling who can access them. For more information, see Policies and permissions in IAM.

AMP permissions

The following table displays possible AMP actions and their required permissions:

Action	Required permission
Create an AMP workspace. A workspace is the conceptual location where you import, store, and work with your Prometheus metrics, isolated from other AMP workspaces.	`aps:CreateWorkspace`
Delete an AMP workspace.	`aps:DeleteWorkspace`
Retrieve detailed information about an AMP workspace.	`aps:DescribeWorkspace`
Retrieve labels.	`aps:GetLabels`
Retrieve metadata for AMP metrics.	`aps:GetMetricMetadata`
Retrieve time series data.	`aps:GetSeries`
Retrieve a list of the AMP workspaces that exist in the account.	`aps:ListWorkspaces`
Run a query on AMP metrics.	`aps:QueryMetrics`
Perform a remote write operation to initiate the streaming of metrics from a Prometheus server to AMP.	`aps:RemoteWrite`
Modify the aliases of existing workspaces.	`aps:UpdateWorkspaceAlias`

Built-in AWS-managed policies

AWS provides several built-in, AWS-managed policies for AMP.

AmazonPrometheusFullAccess

This policy provides full access to all AMP actions and resources.

```
{
```

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

AmazonPrometheusConsoleFullAccess

This policy provides access to all AMP console actions.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "aps:CreateWorkspace",
                "aps:DescribeWorkspace",
                "aps:UpdateWorkspaceAlias",
                "aps:DeleteWorkspace",
                "aps:ListWorkspaces"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

AmazonPrometheusQueryAccess

This policy provides access to query the metrics stored in all AMP workspaces in the account.

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

AmazonPrometheusRemoteWriteAccess

This policy provides permissiont to remote write metrics into all AMP workspaces in the account.

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

```
            "Action": [
                "aps:RemoteWrite"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Sample IAM policies

This section provides examples of other self-managed policies that you can create.

The following IAM policy grants full access to AMP and also enables a user to discover Amazon EKS clusters and see the details about them.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "aps:*",
                "eks:DescribeCluster",
                "eks:ListClusters"
            ],
            "Resource": "*"
        }
    ]
}
```

Compliance Validation for Amazon Managed Service for Prometheus

Third-party auditors assess the security and compliance of Amazon Managed Service for Prometheus as part of multiple AWS compliance programs. These include SOC, PCI, FedRAMP, HIPAA, and others.

To learn whether AMP or other AWS services are in scope of specific compliance programs, see AWS Services in Scope by Compliance Program. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying baseline environments on AWS that are security and compliance focused.
- Architecting for HIPAA Security and Compliance Whitepaper – This whitepaper describes how companies can use AWS to create HIPAA-compliant applications.

  Note
  Not all services are compliant with HIPAA.

- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.
- Evaluating Resources with Rules in the *AWS Config Developer Guide* – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- AWS Security Hub – This AWS service provides a comprehensive view of your security state within AWS that helps you check your compliance with security industry standards and best practices.
- AWS Audit Manager – This AWS service helps you continuously audit your AWS usage to simplify how you manage risk and compliance with regulations and industry standards.

Resilience in Amazon Managed Service for Prometheus

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

In addition to the AWS global infrastructure, AMP offers several features to help support your data resiliency and backup needs.

Infrastructure Security in Amazon Managed Service for Prometheus

As a managed service, Amazon Managed Service for Prometheus is protected by the AWS global network security procedures that are described in the Amazon Web Services: Overview of Security Processes whitepaper.

You use AWS published API calls to access AMP through the network. Clients must support Transport Layer Security (TLS) 1.2 or later. Clients must also support cipher suites with perfect forward secrecy (PFS) such as Ephemeral Diffie-Hellman (DHE) or Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

Set up IAM roles for service accounts

With IAM roles for service accounts, you can associate an IAM role with a Kubernetes service account. This service account can then provide AWS permissions to the containers in any pod that uses that service account. For more information, see IAM roles for service accounts.

IAM roles for service accounts are also known as *service roles*.

In Amazon Managed Service for Prometheus, using service roles can help you get the roles you need to authorize and authenticate between AMP, Prometheus servers, and Grafana servers.

Set up service roles for the ingestion of metrics from Amazon EKS clusters

To set up the service roles to enable AMP to ingest metrics from Prometheus servers in Amazon EKS clusters, you must be logged on to an account with the following permissions:

- `iam:CreateRole`
- `iam:CreatePolicy`
- `iam:GetRole`
- `iam:AttachRolePolicy`
- `iam:GetOpenIDConnectProvider`

To set up the service role for ingestion into AMP

1. Create a file named `createIRSA-AMPIngest.sh` with the following content. Replace `<my_amazon_eks_clustername>` with the name of your cluster, and replace `<my_prometheus_namespace>` with your Prometheus namespace.

```
#!/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. Enter the following command to give the script the necessary privileges.

```
chmod +x createIRSA-AMPIngest.sh
```

3. Run the script.

Set up IAM roles for service accounts for the querying of metrics

To set up the IAM role for service account (service role) to enable the querying of metrics from AMP workspaces, you must be logged on to an account with the following permissions:

- `iam:CreateRole`
- `iam:CreatePolicy`
- `iam:GetRole`
- `iam:AttachRolePolicy`
- `iam:GetOpenIDConnectProvider`

To set up service roles for the querying of AMP metrics;

1. Create a file named `createIRSA-AMPQuery.sh` with the following content. Replace <my_amazon_eks_clustername> with the name of your cluster, and replace <my_prometheus_namespace> with your Prometheus namespace.

```
#!/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. Enter the following command to give the script the necessary privileges.

```
chmod +x createIRSA-AMPQuery.sh
```

3. Run the script.

Using AMP with interface VPC endpoints

Amazon Managed Service for Prometheus (AMP) is in open preview. The preview is open to all AWS accounts and you do not need to request access. Features may be added or changed before announcing General Availability.
The preview currently supports the following Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (Oregon)
- Europe (Frankfurt)
- Europe (Ireland)

If you use Amazon Virtual Private Cloud (Amazon VPC) to host your AWS resources, you can establish a private connection between your VPC and Amazon Managed Service for Prometheus. You can use these connections to enable AMP to communicate with your resources on your VPC without going through the public internet.

Amazon VPC is an AWS service that you can use to launch AWS resources in a virtual network that you define. With a VPC, you have control over your network settings, such the IP address range, subnets, route tables, and network gateways. To connect your VPC to AMP, you define an *interface VPC endpoint* to connect your VPC to AWS services. The endpoint provides reliable, scalable connectivity to AMP without requiring an internet gateway, a network address translation (NAT) instance, or a VPN connection. For more information, see What Is Amazon VPC in the *Amazon VPC User Guide*.

Interface VPC endpoints are powered by AWS PrivateLink, an AWS technology that enables private communication between AWS services using an elastic network interface with private IP addresses. For more information, see the New – AWS PrivateLink for AWS Services blog post.

The following steps are for users of Amazon VPC. For more information, see Getting Started in the *Amazon VPC User Guide*.

Creating a VPC endpoint for AMP

To begin using AMP with your VPC, create an interface VPC endpoint for AMP. The service name to choose is `com.amazonaws.`*`region`*`.aps-workspaces`. For more information, see Creating an Interface Endpoint in the *Amazon VPC User Guide*.

You do not need to change the settings for AMP. AMP calls other AWS services using either public endpoints or private interface VPC endpoints, whichever are in use. For example, if you create an

interface VPC endpoint for AMP, and you already have metrics flowing to AMP from resources located on your VPC, these metrics begin flowing through the interface VPC endpoint by default.

Logging AMP API calls using AWS CloudTrail

Amazon Managed Service for Prometheus is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, a role, or an AWS service in AMP. CloudTrail captures all API calls for AMP as events. The calls that are captured include calls from the AMP console and code calls to the AMP API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for AMP. If you don't configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was made to AMP, the IP address from which the request was made, who made the request, when it was made, and additional details.

To learn more about CloudTrail, see the AWS CloudTrail User Guide.

AMP information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in AMP, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing events with CloudTrail Event history.

For an ongoing record of events in your AWS account, including events for AMP, create a trail. A *trail* enables CloudTrail to deliver log files to an Amazon S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition, and delivers the log files to the Amazon S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data that's collected in CloudTrail logs. For more information, see the following:

- Overview for creating a trail
- CloudTrail supported services and integrations
- Configuring Amazon SNS notifications for CloudTrail
- Receiving CloudTrail log files from multiple regions and Receiving CloudTrail log files from multiple accounts

All AMP actions are logged by CloudTrail and are documented in the API reference (p. 53). For example, calls to the `CreateWorkspace`, `DeleteWorkspace`, and `ListWorkspaces` actions generate entries in the CloudTrail log files.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root or AWS Identity and Access Management (IAM) user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the CloudTrail userIdentity element.

Understanding AMP log file entries

A trail is a configuration that enables delivery of events as log files to an Amazon S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from

any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following example shows a CloudTrail log entry that demonstrates the CreateWorkspace action.

```
{
    "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"
}
```

Troubleshooting

Amazon Managed Service for Prometheus (AMP) is in open preview. The preview is open to all AWS accounts and you do not need to request access. Features may be added or changed before announcing General Availability.
The preview currently supports the following Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (Oregon)
- Europe (Frankfurt)
- Europe (Ireland)

Use the following sections to help troubleshoot issues with Amazon Managed Service for Prometheus.

429 errors

If you see a 429 error similar to the following example, your requests have exceeded Amazon Managed Service for Prometheus ingestion quotas.

```
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
```

If you see a 429 error similar to the following example, your requests have exceeded the Amazon Managed Service for Prometheus quota for the number of active metrics in a workspace.

```
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
```

For more information about AMP service quotas and about how to request increases, see Amazon Managed Service for Prometheus service quotas (p. 51)

I see duplicate samples

If you are using a high-availability Prometheus group, you need to use external labels on your Prometheus instances to set up deduplication. For more information, see Send high-availability data from your Kubernetes cluster (Deduplication) (p. 14).

Amazon Managed Service for Prometheus service quotas

Amazon Managed Service for Prometheus (AMP) is in open preview. The preview is open to all AWS accounts and you do not need to request access. Features may be added or changed before announcing General Availability.
The preview currently supports the following Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (Oregon)
- Europe (Frankfurt)
- Europe (Ireland)

AMP has the following quotas for series, labels, and API requests.

Resource	Default quota
Active series (metrics that have reported data in the past 5 minutes) per workspace	1,000,000. You can request a quota increase.
Active series per metric name	200,000. You can request a quota increase.
Ingestion rate	70,000 samples per second. You can request a quota increase.
Ingestion burst size	1,000,000 samples. You can request a quota increase.
Labels per metric series.	70. You can request a quota increase.
Query time range	32 days between the start time and the end time of a PromQL query. This quota cannot be changed.
Query size	2,000,000 *chunks* that can be fetched in a single query. This quota cannot be changed. A *chunk* is a set of samples for one time series that holds up to 120 samples.
Retention time for ingested data	150 days. Data older than this is deleted from the workspace. For this Preview release, you cannot request an increase.
Workspaces per Region per account	10. You can request a quota increase.
Workspace management API operations, including CreateWorkspace, DeleteWorkspace, DescribeWorkspaces, ListWorkspaces, and UpdateWorkspaceAlias	5 transactions per second (TPS), with a burst limit of 5 TPS. You can request a quota increase.

Additional quotas on ingested data

AMP also has additional quotas for data ingested into the AMP workspace. None of these quotas can be changed in this Preview release.

- Metric samples older than 1 hour are refused from being ingested.
- Every sample and metadata must have a metric name.
- Maximum length accepted for label names: 1024 bytes
- Maximum length accepted for label value: 2048 bytes
- Maximum number of metadata per metric: 10
- Maximum size of ingestion or query request: 1 MB
- Maximum length accepted for metric metadata, which includes metric name, HELP, and UNIT: 1024 bytes
- Maximum number of active metrics with metadata per workspace: 20,000
- Maximum retention time for ingested metrics: 150 days. Data older than this is deleted from the workspace.

API reference

This section lists the API operations and data structures supported by Amazon Managed Service for Prometheus.

> Amazon Managed Service for Prometheus (AMP) is in open preview. The preview is open to all AWS accounts and you do not need to request access. Features may be added or changed before announcing General Availability.
> The preview currently supports the following Regions:
>
> - US East (Ohio)
> - US East (N. Virginia)
> - US West (Oregon)
> - Europe (Frankfurt)
> - Europe (Ireland)

Amazon Managed Service for Prometheus APIs

The following AMP API operations and data structures are supported.

Topics

CreateWorkspace

The `CreateWorkspace` operation creates a *workspace*. A workspace is a logical and isolated Prometheus server dedicated to Prometheus resources such as metrics. You can have one or more workspaces in each Region in your account.

Request parameters

- *alias*—(Optional) An alias that you assign to this workspace to help you identify it. It does not need to be unique.

  The alias can be up to 100 characters and can include any type of characters. AMP will automatically strip any blank spaces from the beginning and end of the alias that you specify.
- *clientToken*—(Optional) A unique, case-sensitive identifier that you can provide to ensure the idempotency of the request.

Response elements

- *workspaceId*—The unique ID for the new workspace.
- *arn*—The ARN for the new workspace.
- *status*—The current status of the new workspace. Immediately after you create the workspace, the status is usually `CREATING`.

Sample request

```
POST /workspaces HTTP/1.1
Content-Length: 79,
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": "myWorkspace",
    "clientToken": "6e5b2726-4bb0-4010-a2e3-de91980cf783"
}
```

Sample response

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 185
Access-Control-Allow-Methods: OPTION,GET,POST,DELETE
Access-Control-Expose-Headers: x-amzn-errortype,x-amzn-requestid,x-amzn-trace-id,x-amzn-
errormessage,x-amz-apigw-id,date
Strict-Transport-Security: strict-transport-security: max-age=31536000; includeSubDomains
x-amz-apigw-id: 1234EXAMPLE=
X-Amzn-Trace-Id: Root=1-12345678-39c98b574fd600282a1a0f21
Connection: keep-alive
Cache-Control: no-store,no-cache,must-revalidate
Date: Tue, 01 Dec 2020 19:37:25 GMT
Access-Control-Allow-Origin: *
Access-Control-Allow-Headers: *,Authorization,Date,X-Amz-Date,content-type,x-amz-content-
sha256,x-amz-user-agent,x-amzn-platform-id,x-amzn-trace-id,X-Amz-Security-Token,X-Amz-
Target
Content-Type: application/x-amz-json-1.1

{
    "arn": "arn:aws:aps:us-west-2:123456789012:workspace/ws-12345678-1234-46a9-933a-
ac50479a5568",
    "status": {
        "statusCode": "CREATING"
    },
    "workspaceId": "ws-b226cc2a-a446-46a9-933a-ac50479a5568"
}
```

DescribeWorkspace

The `DescribeWorkspace` operation displays information about an existing workspace.

Request parameters

- *workspaceId*—The ID of the workspace to describe.

Response elements

- *workspace*—A `WorkspaceDescription` structure that contains details about the workspace. For more information, see WorkspaceDescription structure (p. 58).

Sample request

```
GET /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568 HTTP/1.1
Content-Length: 79,
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": "myWorkspace",
    "clientToken": "6e5b2726-4bb0-4010-a2e3-de91980cf783"
}
```

Sample response

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 185
Access-Control-Allow-Methods: OPTION,GET,POST,DELETE
Access-Control-Expose-Headers: x-amzn-errortype,x-amzn-requestid,x-amzn-trace-id,x-amzn-
errormessage,x-amz-apigw-id,date
Strict-Transport-Security: strict-transport-security: max-age=31536000; includeSubDomains
x-amz-apigw-id: 1234EXAMPLE=
X-Amzn-Trace-Id: Root=1-12345678-39c98b574fd600282a1a0f21
Connection: keep-alive
Cache-Control: no-store,no-cache,must-revalidate
Date: Tue, 01 Dec 2020 19:37:25 GMT
Access-Control-Allow-Origin: *
Access-Control-Allow-Headers: *,Authorization,Date,X-Amz-Date,content-type,x-amz-content-
sha256,x-amz-user-agent,x-amzn-platform-id,x-amzn-trace-id,X-Amz-Security-Token,X-Amz-
Target
Content-Type: application/x-amz-json-1.1

{
    "workspace": {
        "alias": "myWorkspace",
        "arn": "arn:aws:aps:us-west-2:123456789012:workspace/ws-12345678-a446-46a9-933a-
ac50479a5568",
        "createdAt": 1.606851445922E9,
        "prometheusEndpoint": "https://aps-workspaces.us-west-2.amazonaws.com/workspaces/
ws-b226cc2a-a446-46a9-933a-ac50479a5568/",
        "status": {
            "statusCode": "ACTIVE"
        },
        "workspaceId": "ws-12345678-a446-46a9-933a-ac50479a5568"
    }
}
```

ListWorkspaces

The `ListWorkspaces` operation lists all of the AMP workspaces in your account. This includes workspaces being created or deleted.

Request parameters

- *alias*—(Optional) If this is included, it filters the results to only the workspaces with names that start with the value that you specify here.

AMP will automatically strip any blank spaces from the beginning and end of the alias that you specify.

- *maxResults*—(Optional) The maximum number of workspaces to return in one `ListWorkspaces` operation. The range is 1–1000.

  If you omit this parameter, the default of 100 is used.
- *nextToken*—(Optional) The token for the next set of items to return. (You received this token from a previous call.)

Response elements

- *workspaces*—An array of `WorkspaceSummary` structures. For more information, see WorkspaceSummary structure (p. 59).
- *nextToken*—A token indicating that there are more results to retrieve. You can use this token as part of your next `ListWorkspaces` operation to retrieve those results.

Sample request

```
GET /workspaces?alias=myWorkspace HTTP/1.1
Content-Length: 79,
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": "myWorkspace",
    "clientToken": "6e5b2726-4bb0-4010-a2e3-de91980cf783"
}
```

Sample response

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 185
Access-Control-Allow-Methods: OPTION,GET,POST,DELETE
Access-Control-Expose-Headers: x-amzn-errortype,x-amzn-requestid,x-amzn-trace-id,x-amzn-
errormessage,x-amz-apigw-id,date
Strict-Transport-Security: strict-transport-security: max-age=31536000; includeSubDomains
x-amz-apigw-id: 1234EXAMPLE=
X-Amzn-Trace-Id: Root=1-12345678-39c98b574fd600282a1a0f21
Connection: keep-alive
Cache-Control: no-store,no-cache,must-revalidate
Date: Tue, 01 Dec 2020 19:37:25 GMT
Access-Control-Allow-Origin: *
Access-Control-Allow-Headers: *,Authorization,Date,X-Amz-Date,content-type,x-amz-content-
sha256,x-amz-user-agent,x-amzn-platform-id,x-amzn-trace-id,X-Amz-Security-Token,X-Amz-
Target
Content-Type: application/x-amz-json-1.1

{
    "nextToken": null,
    "workspaces": [
        {
            "alias": "myWorkspace",
            "arn": "arn:aws:aps:us-west-2:123456789012:workspace/ws-b226cc2a-
a446-46a9-933a-ac50479a5568",
            "createdAt": 1.606851445922E9,
            "status": {
```

```
                "statusCode": "ACTIVE"
            },
            "workspaceId": "ws-12345678-a446-46a9-933a-ac50479a5568"
        }
    ]
}
```

DeleteWorkspace

The `DeleteWorkspace` operation deletes an existing workspace.

When you delete a workspace, the data that has been ingested into it is not immediately deleted. It will be permanently deleted within one month.

Request parameters

- *workspaceId*—The ID of the workspace to delete.
- *clientToken*—(Optional) A unique, case-sensitive identifier that you can provide to ensure the idempotency of the request.

Response elements

- None

Sample request

```
DELETE /workspaces/ws-b226cc2a-a446-46a9-933a-ac50479a5568?
clientToken=99ce509d-014c-4911-8f71-9844d9df0a97 HTTP/1.1
Content-Length: 79,
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": "myWorkspace",
    "clientToken": "6e5b2726-4bb0-4010-a2e3-de91980cf783"
}
```

Sample response

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 185
Access-Control-Allow-Methods: OPTION,GET,POST,DELETE
Access-Control-Expose-Headers: x-amzn-errortype,x-amzn-requestid,x-amzn-trace-id,x-amzn-
errormessage,x-amz-apigw-id,date
Strict-Transport-Security: strict-transport-security: max-age=31536000; includeSubDomains
x-amz-apigw-id: 1234EXAMPLE=
X-Amzn-Trace-Id: Root=1-12345678-39c98b574fd600282a1a0f21
Connection: keep-alive
Cache-Control: no-store,no-cache,must-revalidate
Date: Tue, 01 Dec 2020 19:37:25 GMT
Access-Control-Allow-Origin: *
Access-Control-Allow-Headers: *,Authorization,Date,X-Amz-Date,content-type,x-amz-content-
sha256,x-amz-user-agent,x-amzn-platform-id,x-amzn-trace-id,X-Amz-Security-Token,X-Amz-
Target
Content-Type: application/x-amz-json-1.1
```

UpdateWorkspaceAlias

The `UpdateWorkspaceAlias` operation updates the alias of an existing workspace.

Request parameters

- *workspaceId*—The ID of the workspace to update.
- *alias*—The new alias for the workspace. It does not need to be unique.

  The alias can be up to 100 characters and can include any type of characters. AMP will automatically strip any blank spaces from the beginning and end of the alias that you specify.
- *clientToken*—(Optional) A unique, case-sensitive identifier that you can provide to ensure the idempotency of the request.

Response elements

- None

Sample request

```
POST /workspaces/ws-1234567890-a446-46a9-933a-ac50479a5568/alias HTTP/1.1
Content-Length: 79,
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": "myWorkspace",
    "clientToken": "6e5b2726-4bb0-4010-a2e3-de91980cf783"
}
```

Sample response

```
HTTP/1.1 200 OK
x-amzn-RequestId: 12345678-abcd-4442-b8c5-262b45e9b535
Content-Length: 185
Access-Control-Allow-Methods: OPTION,GET,POST,DELETE
Access-Control-Expose-Headers: x-amzn-errortype,x-amzn-requestid,x-amzn-trace-id,x-amzn-
errormessage,x-amz-apigw-id,date
Strict-Transport-Security: strict-transport-security: max-age=31536000; includeSubDomains
x-amz-apigw-id: 1234EXAMPLE=
X-Amzn-Trace-Id: Root=1-12345678-39c98b574fd600282a1a0f21
Connection: keep-alive
Cache-Control: no-store,no-cache,must-revalidate
Date: Tue, 01 Dec 2020 19:37:25 GMT
Access-Control-Allow-Origin: *
Access-Control-Allow-Headers: *,Authorization,Date,X-Amz-Date,content-type,x-amz-content-
sha256,x-amz-user-agent,x-amzn-platform-id,x-amzn-trace-id,X-Amz-Security-Token,X-Amz-
Target
Content-Type: application/x-amz-json-1.1
```

WorkspaceDescription structure

The `WorkspaceDescription` structure contains the full details about one AMP workspace in your account.

Structure parameters

- *workspaceId*—The unique ID for the workspace.
- *arn*—The ARN of the workspace.
- *status*—The current status of the workspace. The possible values are `CREATING`, `ACTIVE`, `UPDATING`, `DELETING`, and `CREATION_FAILED`.
- *prometheusEndpoint*—The Prometheus endpoint available for this workspace.
- *createdAt*—The date and time that the workspace was created.

WorkspaceSummary structure

The `WorkspaceSummary` structure contains information about one AMP workspace in your account.

Structure parameters

- *workspaceId*—The unique ID for the workspace.
- *arn*—The ARN of the workspace.
- *status*—The current status of the workspace. The possible values are `CREATING`, `ACTIVE`, `UPDATING`, `DELETING`, and `CREATION_FAILED`.
- *createdAt*—The date and time that the workspace was created.

Common errors

This section lists the errors common to the API actions of all AWS services. For errors specific to an API action for this service, see the topic for that API action.

AccessDeniedException

You do not have sufficient access to perform this action.

HTTP Status Code: 400

IncompleteSignature

The request signature does not conform to AWS standards.

HTTP Status Code: 400

InternalFailure

The request processing has failed because of an unknown error, exception or failure.

HTTP Status Code: 500

InvalidAction

The action or operation requested is invalid. Verify that the action is typed correctly.

HTTP Status Code: 400

InvalidClientTokenId

The X.509 certificate or AWS access key ID provided does not exist in our records.

HTTP Status Code: 403

InvalidParameterCombination

Parameters that must not be used together were used together.

HTTP Status Code: 400

InvalidParameterValue

An invalid or out-of-range value was supplied for the input parameter.

HTTP Status Code: 400

InvalidQueryParameter

The AWS query string is malformed or does not adhere to AWS standards.

HTTP Status Code: 400

MalformedQueryString

The query string contains a syntax error.

HTTP Status Code: 404

MissingAction

The request is missing an action or a required parameter.

HTTP Status Code: 400

MissingAuthenticationToken

The request must contain either a valid (registered) AWS access key ID or X.509 certificate.

HTTP Status Code: 403

MissingParameter

A required parameter for the specified action is not supplied.

HTTP Status Code: 400

NotAuthorized

You do not have permission to perform this action.

HTTP Status Code: 400

OptInRequired

The AWS access key ID needs a subscription for the service.

HTTP Status Code: 403

RequestExpired

The request reached the service more than 15 minutes after the date stamp on the request or more than 15 minutes after the request expiration date (such as for pre-signed URLs), or the date stamp on the request is more than 15 minutes in the future.

HTTP Status Code: 400

ServiceUnavailable

The request has failed due to a temporary failure of the server.

HTTP Status Code: 503

ThrottlingException

The request was denied due to request throttling.

HTTP Status Code: 400

ValidationError

The input fails to satisfy the constraints specified by an AWS service.

HTTP Status Code: 400

Prometheus-compatible APIs

Amazon Managed Service for Prometheus supports the following Prometheus-compatible APIs.

Topics

GetLabels

The `GetLabels` operation retrieves the labels associated with a time series.

Valid HTTP verbs:

`GET`, `POST`

Valid URIs:

`/workspaces/workspaceId/api/v1/labels`

`/workspaces/workspaceId/api/v1/label/label-name/values` This URI supports only GET requests.

URL query parameters:

`start=<rfc3339 | unix_timestamp>` Start timestamp. Optional

`end=<rfc3339 | unix_timestamp>` End timestamp. Optional

GetMetricMetadata

The `GetMetricMetadata` operation retrieves metadata about metrics that are currently being scraped from targets. It does not provide any target information.

The data section of the query result consists of an object where each key is a metric name and each value is a list of unique metadata objects, as exposed for that metric name across all targets.

Valid HTTP verbs:

`GET`

Valid URIs:

`/workspaces/workspaceId/api/v1/metadata`

URL query parameters:

`limit=<number>` The maximum number of metrics to return.

`metric=<string>` A metric name to filter metadata for. If you leave this empty, all metric metadata is retrieved.

GetSeries

The `GetSeries` operation retrieves list of time series that match a certain label set.

Valid HTTP verbs:

`GET`, `POST`

Valid URIs:

`/workspaces/workspaceId/api/v1/series`

URL query parameters:

`match[]=<series_selector>` Repeated series selector argument that selects the series to return. At least one `match[]` argument must be provided.

`start=<rfc3339 | unix_timestamp>` Start timestamp. Optional

`end=<rfc3339 | unix_timestamp>` End timestamp. Optional

QueryMetrics

The `QueryMetrics` operation evaluates an instant query at a single point in time or over a range of time.

Valid HTTP verbs:

`GET`, `POST`

Valid URIs:

`/workspaces/workspaceId/api/v1/query` This URI evaluates an instant query at a single point in time.

`/workspaces/workspaceId/api/v1/query_range` This URI evaluates an instant query over a range of time.

URL query parameters:

`query=<string>` A Prometheus expression query string. Used in both `query` and `query_range`.

`time=<rfc3339 | unix_timestamp>` (Optional) Evaluation timestamp if you are using the `query` for an instant query at a single point in time.

`timeout=<duration>` (Optional) Evaluation timeout. Defaults to and is capped by the value of the `-query.timeout` flag. Used in both `query` and `query_range`.

`start=<rfc3339 | unix_timestamp>` Start timestamp if you are using `query_range` to query for a range of time.

`end=<rfc3339 | unix_timestamp>` End timestamp if you are using `query_range` to query for a range of time.

`step=<duration | float>` Query resolution step width in `duration` format or as a `float` number of seconds. Use only if you are using `query_range` to query for a range of time, and required for such queries.

Duration

A `duration` in a Prometheus-compatible API is a number, followed immediately by one of the following units:

- `ms` milliseconds
- `s` seconds
- `m` minutes
- `h` hours
- `d` days, assuming a day always has 24h
- `w` weeks, assuming a week always has 7d
- `y` years, assuming a year always has 365d

RemoteWrite

The `RemoteWrite` operation writes metrics from a Prometheus server to a remote URL in a standardized format.

Valid HTTP verbs:

`POST`

Valid URIs:

`/workspaces/workspaceId/api/v1/remote_write`

URL query parameters:

None

Document History for Amazon Managed Service for Prometheus User Guide

The following table describes important documentation updates in the Amazon Managed Service for Prometheus User Guide. For notification about updates to this documentation, you can subscribe to an RSS feed.

update-history-change	update-history-description	update-history-date
Active series and ingestion rate quotas increased.	The active series quota increased to 1,000,000 and the ingestion rate quota increased to 70,000 samples per second.	February 22, 2021
Amazon Managed Service for Prometheus preview release.	The preview of Amazon Managed Service for Prometheus is released.	December 15, 2020

AWS glossary

For the latest AWS terminology, see the AWS glossary in the *AWS General Reference*.