

---

# Amazon Managed Grafana

## User Guide



## **Amazon Managed Grafana: User Guide**

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

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.

# Table of Contents

What is Amazon Managed Grafana? .....	1
Supported Regions .....	1
Upgrade from preview release .....	1
Setting up .....	2
Get an AWS account and your root user credentials .....	2
Creating an IAM user .....	2
Signing in as an IAM user .....	3
Creating IAM user access keys .....	3
User authentication .....	5
SAML .....	5
AWS SSO .....	6
Connecting to your identity provider .....	7
Azure Active Directory .....	7
CyberArk .....	8
Okta .....	10
OneLogin .....	11
Ping Identity .....	12
AWS SSO .....	14
Getting started .....	15
User authentication .....	15
Necessary permissions .....	15
Create your first workspace .....	16
Managing workspaces, users, and policies .....	19
Creating a workspace .....	19
User authentication in a workspace .....	19
Necessary permissions .....	15
Managing user and group access to Amazon Managed Grafana .....	23
Managing permissions for data sources and notification channels .....	24
Deleting a workspace .....	25
Working in your Grafana workspace .....	26
Users, teams, and permissions .....	26
Users .....	26
User roles .....	27
Managing teams .....	27
Using permissions .....	29
Getting started in your Grafana workspace console .....	32
What is Grafana? .....	32
Explore metrics and logs .....	32
Alerts .....	32
Annotations .....	32
Dashboard variables .....	33
Creating a dashboard .....	33
Data sources .....	39
How Amazon Managed Grafana works with AWS Organizations for AWS data source access .....	39
Built-in data sources .....	41
Data sources available in Grafana Enterprise .....	138
Panels .....	188
Adding or editing a panel .....	188
Queries .....	189
Transformations .....	192
Field options and overrides .....	199
Panel editor .....	205
Library panels .....	206
Visualizations .....	208

Dashboards .....	237
Manage dashboards .....	237
Rows .....	237
Annotations .....	238
Dashboard folders .....	239
Playlist .....	240
Dashboard search .....	243
Sharing a dashboard .....	244
Sharing a panel .....	244
Time range controls .....	244
Exporting and importing dashboards .....	247
Dashboard version history .....	247
Keyboard shortcuts .....	248
Dashboard JSON model .....	248
Scripted dashboards .....	253
Explore .....	254
Start exploring .....	254
Splitting and comparing .....	254
Sharing a shortened link .....	255
Query history .....	255
Prometheus-specific features .....	256
Logs integration .....	257
Tracing integration .....	259
Navigating between Explore and a dashboard .....	260
Query inspector .....	260
Linking .....	260
Which link should you use? .....	260
Controlling time range using the URL .....	261
Dashboard links .....	261
Panel links .....	262
Data links .....	263
Data link variables .....	264
Templates and variables .....	265
Templates .....	266
Variable best practices .....	266
Variable syntax .....	266
Variable types .....	266
Other variable options .....	276
Alerts .....	282
Alert configuration .....	282
Clustering .....	283
Notifications .....	283
Alert execution .....	283
Alert notifications .....	283
Creating alerts .....	286
Pausing an alert rule .....	289
Viewing existing alert rules .....	289
Notification templating .....	289
Troubleshooting alerts .....	290
Change your preferences .....	290
Edit your Amazon Managed Grafana profile .....	290
Edit your preferences .....	290
View your Amazon Managed Grafana sessions .....	291
Using Grafana HTTP APIs .....	292
Alerting API .....	293
Get alerts .....	293
Get alert by Id .....	294

Pause alert by Id .....	295
Alerting Notification Channels API .....	296
Get all notification channels .....	296
Get all notification channels (lookup) .....	297
Get all notification channels by UID .....	297
Get all notification channels by Id .....	298
Create notification channel .....	298
Update notification channel by UID .....	299
Update notification channel by Id .....	300
Delete notification channel by UID .....	301
Delete notification channel by Id .....	301
Test notification channel .....	301
Annotations API .....	302
Find annotations .....	302
Create annotation .....	303
Create annotation in graphite format .....	304
Update annotation .....	305
Patch annotation .....	305
Delete annotation by Id .....	306
Authentication API .....	306
Get API keys .....	306
Create API key .....	307
Delete API key .....	307
Dashboard API .....	308
Create/Update dashboard .....	308
Get dashboard by uid .....	312
Delete dashboard by uid .....	313
Gets the home dashboard .....	313
Get dashboard tags .....	314
Dashboard Permissions API .....	315
Get permissions for a dashboard .....	315
Update permissions for a dashboard .....	316
Dashboard Versions API .....	317
Get all dashboard versions .....	317
Get dashboard version .....	318
Restore dashboard .....	320
Compare dashboard versions .....	321
Data Source API .....	322
Get all data sources .....	323
Get a single data source by Id .....	323
Get a single data source by UID .....	324
Get a single data source by name .....	325
Get data source Id by name .....	325
Create a data source .....	326
Update an existing data source .....	328
Delete data source by Id .....	329
Delete data source by UID .....	329
Delete data source by name .....	330
Data source proxy calls .....	330
Query data source by Id .....	330
Data Source Permissions API .....	332
Enable permissions for a data source .....	332
Disable permissions for a data source .....	333
Get permissions for a data source .....	334
Add permission for a data source .....	335
Remove permission for a data source .....	336
External Group Synchronization API .....	336

Get external groups .....	336
Add external group .....	337
Remove external group .....	337
Folder API .....	338
Create folder .....	338
Update folder .....	339
Get all folders .....	341
Get folder by uid .....	341
Get folder by id .....	341
Delete folder by uid .....	343
Folder/Dashboard Search API .....	343
Search folders and dashboards .....	343
Folder Permissions API .....	345
Get permissions for a folder .....	345
Update permissions for a folder .....	346
Organization API .....	347
Get current organization .....	347
Get all users within the current organization .....	348
Get all users within the current organization (lookup) .....	348
Updates the given user .....	349
Deletes user in current organization .....	349
Update the current organization .....	350
Add user to the current organization .....	350
Playlist API .....	351
Search playlist .....	351
Get one playlist .....	351
Get playlist items .....	352
Get playlist dashboards .....	353
Create a playlist .....	353
Update a playlist .....	354
Delete a playlist .....	355
Preferences API .....	355
Get current user preferences .....	355
Update current user preferences .....	356
Get current org preferences .....	356
Update current org preferences .....	356
Snapshot API .....	357
Create new shapshot .....	357
Get list of snapshots .....	358
Get snapshot by key .....	359
Delete snapshot by key .....	360
Delete snapshot by deleteKey .....	360
Team API .....	361
Team search with pagination .....	361
Get team by Id .....	362
Add a team .....	362
Update team .....	363
Delete team by Id .....	363
Get team members .....	364
Add team member .....	365
Remove member from team .....	365
Get team preferences .....	366
Update team preferences .....	366
User API .....	367
Get teams that the user is a member of .....	367
Get list of snapshots .....	367
Unstar a dashboard .....	368

Get auth tokens of the actual user .....	368
Revoke an auth token of the actual user .....	369
Upgrade a workspace to Grafana Enterprise .....	370
Canceling Grafana Enterprise .....	372
Security .....	373
Data protection .....	373
Data protection in Amazon Managed Grafana .....	374
Identity and Access Management .....	374
Audience .....	375
Authenticating with identities .....	375
Managing access using policies .....	377
How Amazon Managed Grafana works with IAM .....	378
Identity-based policy examples .....	383
Troubleshooting .....	388
Amazon Managed Grafana permissions and policies for AWS data sources and notification channels ...	390
Service-managed permissions for a single account .....	391
Service-managed permissions for an organization .....	393
Customer-managed permissions .....	396
IAM permissions .....	397
Amazon Managed Grafana permissions .....	397
Compliance Validation .....	398
Resilience .....	399
Infrastructure Security .....	399
CloudTrail logs .....	399
Amazon Managed Grafana information in CloudTrail .....	400
Understanding Amazon Managed Grafana log file entries .....	400
Understanding Grafana API log file entries .....	403
Security best practices .....	415
Use short-lived API keys .....	415
Migrating from self-managed Grafana .....	416
Service quotas .....	417
Document history .....	418
AWS glossary .....	419

# What is Amazon Managed Grafana?

Amazon Managed Grafana is a fully managed and secure data visualization service that you can use to instantly query, correlate, and visualize operational metrics, logs, and traces from multiple sources. Amazon Managed Grafana makes it easy to deploy, operate, and scale Grafana, a widely deployed data visualization tool that is popular for its extensible data support.

With Amazon Managed Grafana, you create logically isolated Grafana servers called *workspaces*. Then, you can create Grafana dashboards and visualizations to analyze your metrics, logs, and traces without having to build, package, or deploy any hardware to run your Grafana servers.

Amazon Managed Grafana manages the provisioning, setup, scaling, and maintenance of your logical Grafana servers so that you don't have to do these tasks yourself. Amazon Managed Grafana also provides built-in security features for compliance with corporate governance requirements, including single sign-on, data access control, and audit reporting.

Amazon Managed Grafana is integrated with AWS data sources that collect operational data, such as Amazon CloudWatch, Amazon OpenSearch Service, AWS X-Ray, AWS IoT SiteWise, Amazon Timestream, and Amazon Managed Service for Prometheus. Amazon Managed Grafana includes a permission provisioning feature for adding supported AWS services as data sources. Amazon Managed Grafana also supports many popular open-source, third-party, and other cloud data sources.

For user authentication and authorization, Amazon Managed Grafana can integrate with identity providers (IdPs) that support SAML 2.0 and also can integrate with AWS Single Sign-On.

Amazon Managed Grafana is priced per active user in a workspace. For information about pricing, see [Amazon Managed Grafana Pricing](#).

## Supported Regions

Amazon Managed Grafana currently supports the following Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (Oregon)
- Asia Pacific (Seoul)
- Asia Pacific (Singapore)
- Asia Pacific (Sydney)
- Asia Pacific (Tokyo)
- Europe (Frankfurt)
- Europe (Ireland)
- Europe (London)

## Upgrade from preview release

Upgrade from the preview release to the general availability release of Amazon Managed Grafana is automatic and does not require you to take any action. All workspaces are automatically upgraded to Grafana 8.0, and all new Amazon Managed Grafana features are available to you.



# Setting up

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 with Amazon Managed Grafana \(p. 15\)](#).

When you sign up for AWS, your AWS account automatically has access to all services in AWS, including Amazon Managed Grafana. 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*

# User authentication in Amazon Managed Grafana

Users are authenticated to use the Grafana console in an Amazon Managed Grafana workspace by single sign-on using your organization's identity provider, instead of by IAM. Each workspace can use one or both of the following authentication methods:

- User credentials stored in identity providers (IdPs) that support Security Assertion Markup Language 2.0 (SAML 2.0)
- AWS Single Sign-On

For each of your workspaces, you can use SAML, AWS SSO, or both. If you begin by using one method, you can switch to using the other.

## Topics

- [Using SAML with your Amazon Managed Grafana workspace \(p. 5\)](#)
- [Using AWS SSO with your Amazon Managed Grafana workspace \(p. 14\)](#)

## Using SAML with your Amazon Managed Grafana workspace

SAML authentication support enables you to use your existing identity provider to offer single sign-on for logging into the Grafana console of your Amazon Managed Grafana workspaces. Rather than authenticating through IAM, SAML authentication for Amazon Managed Grafana lets you use third-party identity providers to log in to Kibana, manage access control, search your data, and build visualizations. Amazon Managed Grafana supports identity providers that use the SAML 2.0 standard and have built and tested integration applications with Azure AD, CyberArk, Okta, OneLogin, and Ping Identity.

In the SAML authentication flow, an Amazon Managed Grafana workspace acts as the service provider (SP), and interacts with the IdP to obtain user information. For more information about SAML, see [Security Assertion Markup Language](#).

You can map groups in your IdP to teams in the Amazon Managed Grafana workspace, and set fine-grained access permissions on those teams. You can also map organization roles that are defined in the IdP to roles in the Amazon Managed Grafana workspace. For example, if you have a **Developer** role defined in the IdP, you can map that role to the **Grafana Admin** role in the Amazon Managed Grafana workspace.

To sign in to the Amazon Managed Grafana workspace, a user visits the workspace's Grafana console home page and chooses **Log in using SAML**. The workspace reads the SAML configuration and redirects the user to the IdP for authentication. The user enters their user name and password in the IdP portal, and if they are a valid user, the IdP issues a SAML assertion and redirects the user back to the Amazon Managed Grafana workspace. Amazon Managed Grafana verifies that the SAML assertion is valid, and the user is signed in and can use the workspace.

Amazon Managed Grafana supports the following SAML 2.0 bindings:

- From the service provider (SP) to the identity provider (IdP):
  - HTTP-POST binding
  - HTTP-Redirect binding
- From the identity provider (IdP) to the service provider (SP):
  - HTTP-POST binding

Amazon Managed Grafana supports signed and encrypted assertions, but does not support signed or encrypted requests.

Amazon Managed Grafana supports SP-initiated requests, and does not support IdP-initiated requests.

## Assertion mapping

During the SAML authentication flow, Amazon Managed Grafana receives the assertion consumer service (ACS) callback. The callback contains all relevant information for the user being authenticated, embedded in the SAML response. Amazon Managed Grafana parses the response to create (or update) the user within its internal database.

When Amazon Managed Grafana maps the user information, it looks at the individual attributes within the assertion. You can think of these attributes as key-value pairs, although they contain more information than that.

Amazon Managed Grafana provides configuration options so that you can modify which keys to look at for these values.

You can use the Amazon Managed Grafana console to map the following SAML assertion attributes to values in Amazon Managed Grafana:

- For **Assertion attribute role**, specify the name of the attribute within the SAML assertion to use as the user roles.
- For **Assertion attribute name**, specify the name of the attribute within the SAML assertion to use for the user full "friendly" names for SAML users.
- For **Assertion attribute login**, specify the name of the attribute within the SAML assertion to use for the user sign-in names for SAML users.
- For **Assertion attribute email**, specify the name of the attribute within the SAML assertion to use for the user email names for SAML users.
- For **Assertion attribute organization**, specify the name of the attribute within the SAML assertion to use for the "friendly" name for user organizations.
- For **Assertion attribute groups**, specify the name of the attribute within the SAML assertion to use for the "friendly" name for user groups.
- For **Allowed organizations**, you can limit user access to only the users who are members of certain organizations in the IdP.
- For **Editor role values**, specify the user roles from your IdP who should all be granted the `Editor` role in the Amazon Managed Grafana workspace.

### Required IAM permissions to create a workspace that uses SAML

When you create an Amazon Managed Grafana workspace that uses an IdP and SAML for authorization, you must be signed on to an IAM principal that has the **AWSGrafanaAccountAdministrator** policy attached.

## Connecting to your identity provider

The following external identity providers have been tested with Amazon Managed Grafana and provide applications directly in their app directories or galleries to help you configure Amazon Managed Grafana with SAML.

### Topics

- [Azure Active Directory \(p. 7\)](#)
- [CyberArk \(p. 8\)](#)
- [Okta \(p. 10\)](#)
- [OneLogin \(p. 11\)](#)
- [Ping Identity \(p. 12\)](#)

## Azure Active Directory

Use the following steps to configure Amazon Managed Grafana to use Azure Active Directory as an identity provider. These steps assume that you have already created your Amazon Managed Grafana workspace and you have made a note of the workspace's ID, URLs, and Region.

### Step 1: Steps to complete in Azure Active Directory

Complete the following steps in Azure Active Directory.

#### To set up Azure Active Directory as an identity provider for Amazon Managed Grafana

1. Sign in to the Azure console as an admin.
2. Choose **Azure Active Directory**.
3. Choose **Enterprise Applications**.
4. Search for **Amazon Managed Grafana SAML2.0**, and select it.
5. Select the application and choose **Setup**.
6. In the Azure Active Directory application configuration, choose **Users and groups**.
7. Assign the application to the users and groups that you want.
8. Choose **Single sign-on**.
9. Choose **Next** to get to the SAML configuration page.
10. Specify your SAML settings:
  - For **Identifier (Entity ID)**, paste in your **Service provider identifier** URL from the Amazon Managed Grafana workspace.
  - For **Reply URL (Assertion Consumer Service URL)**, paste in your **Service provider reply** from the Amazon Managed Grafana workspace.
  - Make sure that **Sign Assertion** is selected and that **Encrypt Assertion** is not selected.
11. In the **User Attributes & Claims** section, make sure that these attributes are mapped. They are case sensitive.
  - **mail** is set with **user.userprincipalname**.
  - **displayName** is set with **user.displayname**.
  - **Unique User Identifier** is set with **user.userprincipalname**.
  - Add any other attributes that you would to pass. For more information about the attributes that you can pass to Amazon Managed Grafana in the assertion mapping, see [Assertion mapping \(p. 6\)](#).

12. Copy the **SAML Metadata URL**. You will use it in the Amazon Managed Grafana workspace configuration.

## Step 2: Steps to complete in Amazon Managed Grafana

Complete the following steps in the Amazon Managed Grafana console.

### To finishg setting up Azure Active Directory as an identity provider for Amazon Managed Grafana

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
  2. In the navigation pane, choose the menu icon.
  3. Choose **All workspaces**.
  4. Choose the name of the workspace.
  5. In the **Authentication** tab, choose **Setup SAML configuration**.
  6. Under **Import the metadata**, choose **Upload or copy/paste** and paste the Azure Active Directory URL that you copied from **SAML Metadata URL** in the previous section.
  7. Under **Assertion mapping**, do the following:
    - Make sure that **I want to opt-out of assigning admins to my workspace** is not selected.
- Note**  
If you choose **I want to opt-out of assigning admins to my workspace**, you won't be able to use the Amazon Managed Grafana workspace console to administer the workspace, including tasks such as managing data sources, users, and dashboard permissions. You can make administrative changes to the workspace only by using Grafana APIs.
- Set **Assertion attribute role** to the attribute name that you chose.
  - Set **Admin role values** to value corresponding to your admin users' roles.
  - (Optional) If you changed the default attributes in your Azure Active Directory application, expand **Additional settings - optional** and then set the new attribute names.
- By default, the Azure **displayName** attribute will be passed as the **Name** attribute and the Ping Identity **mail** attribute will be passed to both the **email** and **login** attributes.
8. Choose **Save SAML Configuration**.

## CyberArk

Use the following steps to configure Amazon Managed Grafana to use CyberArk as an identity provider. These steps assume that you have already created your Amazon Managed Grafana workspace and you have made a note of the workspace's ID, URLs, and Region.

## Step 1: Steps to complete in CyberArk

Complete the following steps in CyberArk.

### To set up CyberArk as an identity provider for Amazon Managed Grafana

1. Sign in to the CyberArk Identity Admin Portal.
2. Choose **Apps, Web Apps**.
3. Choose **Add Web App**.
4. Search for **Amazon Managed Grafana for SAML2.0**, and choose **Add**.
5. In the CyberArk application configuration, go to the **Trust** section.

6. Under **Identity Provider Configuration**, choose **Metadata**.
7. Choose **Copy URL** and save the URL to use later in these steps.
8. Under **Service Provider Configuration**, choose **Manual Configuration**.
9. Specify your SAML settings:
  - For **SP Entity ID**, paste in your **Service provider identifier** URL from the Amazon Managed Grafana workspace.
  - For **Assertion Consumer Service (ACS) URL**, paste in your **Service provider reply** from the Amazon Managed Grafana workspace.
  - Set **Sign Response Assertion** to **Assertion**.
  - Make sure that **NameID Format** is **emailAddress**.
10. Choose **Save**.
11. In the **SAML Response** section, make sure that the Amazon Managed Grafana attribute is in **Application Name** and that the CyberArk attribute is in **Attribute Value**. Then make sure that the following attributes are mapped. They are case sensitive.
  - **displayName** is set with **LoginUser.DisplayName**.
  - **mail** is set with **LoginUser.Email**.
  - Add any other attributes that you would to pass. For more information about the attributes that you can pass to Amazon Managed Grafana in the assertion mapping, see [Assertion mapping \(p. 6\)](#).
12. Choose **Save**.
13. In the **Permissions** section, choose which users and groups to assign this application to, and then choose **Save**.

## Step 2: Steps to complete in Amazon Managed Grafana

Complete the following steps in the Amazon Managed Grafana console.

### To finishg setting up CyberArk as an identity provider for Amazon Managed Grafana

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
  2. In the navigation pane, choose the menu icon.
  3. Choose **All workspaces**.
  4. Choose the name of the workspace.
  5. In the **Authentication** tab, choose **Setup SAML configuration**.
  6. Under **Import the metadata**, choose **Upload or copy/paste** and paste the CyberArk URL that you copied in the previous procedure.
  7. Under **Assertion mapping**, do the following:
    - Make sure that **I want to opt-out of assigning admins to my workspace** is not selected.
- Note**  
If you choose **I want to opt-out of assigning admins to my workspace**, you won't be able to use the Amazon Managed Grafana workspace console to administer the workspace, including tasks such as managing data sources, users, and dashboard permissions. You can make administrative changes to the workspace only by using Grafana APIs.
- Set **Assertion attribute role** to the attribute name that you chose.
  - Set **Admin role values** to value corresponding to your admin users' roles.
  - (Optional) If you changed the default attributes in your CyberArk application, expand **Additional settings - optional** and then set the new attribute names.



By default, the CyberArk **displayName** attribute will be passed to the **name** attribute and the CyberArk **mail** attribute will be passed to both the **email** and **login** attributes.

8. Choose **Save SAML Configuration**.

## Okta

Use the following steps to configure Amazon Managed Grafana to use Okta as an identity provider. These steps assume that you have already created your Amazon Managed Grafana workspace and you have made a note of the workspace's ID, URLs, and Region.

### Step 1: Steps to complete in Okta

Complete the following steps in Okta.

#### To set up Okta as an identity provider for Amazon Managed Grafana

1. Sign in to the Okta console as an admin.
2. In the left panel, choose **Applications, Applications**.
3. Choose **Browse App Catalog** and search for **Amazon Managed Grafana**.
4. Choose **Amazon Managed Grafana** and choose **Add, Done**.
5. Choose the application to start setting it up.
6. In the **Sign On** tab, choose **Edit**.
7. Under **Advanced Sign-on Settings**, enter your Amazon Managed Grafana workspace name and your Region in the **Name Space** and **Region** fields. Your Amazon Managed Grafana workspace name format is **workspace-name.grafana-workspace.Region.amazonaws.com**.
8. Choose **Save**.
9. Under **SAML 2.0**, copy the URL for **Identity Provider metadata**. You will use this later in this procedure in the Amazon Managed Grafana console.
10. In the **Assignments** tab, choose the **People** and **Groups** that you want to be able to use Amazon Managed Grafana.

### Step 2: Steps to complete in Amazon Managed Grafana

Complete the following steps in the Amazon Managed Grafana console.

#### To finishg setting up Okta as an identity provider for Amazon Managed Grafana

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the navigation pane, choose the menu icon.
3. Choose **All workspaces**.
4. Choose the name of the workspace.
5. In the **Authentication** tab, choose **Complete Setup**.
6. Under **Import the metadata**, choose **Upload or copy/paste** and paste the Okta URL that you copied in the previous procedure.
7. Under **Assertion mapping**, do the following:
  - Make sure that **I want to opt-out of assigning admins to my workspace** is not selected.

#### Note

If you choose **I want to opt-out of assigning admins to my workspace**, you won't be able to use the Amazon Managed Grafana workspace console to administer the workspace,

including tasks such as managing data sources, users, and dashboard permissions. You can make administrative changes to the workspace only by using Grafana APIs.

- Set **Assertion attribute role** to the attribute name that you chose.
- Set **Admin role values** to value corresponding to your admin users' roles.
- (Optional) If you changed the default attributes in your Okta application, expand **Additional settings - optional** and then set the new attribute names.

By default, the Okta **displayName** attribute will be passed to the **name** attribute and the Okta **mail** attribute will be passed to both the **email** and **login** attributes.

8. Choose **Save SAML Configuration**.

## OneLogin

Use the following steps to configure Amazon Managed Grafana to use OneLogin as an identity provider. These steps assume that you have already created your Amazon Managed Grafana workspace and you have made a note of the workspace's ID, URLs, and Region.

### Step 1: Steps to complete in OneLogin

Complete the following steps in OneLogin.

#### To set up OneLogin as an identity provider for Amazon Managed Grafana

1. Sign in to the OneLogin portal as an administrator.
2. Choose **Applications, Applications, Add app**.
3. Search for **Amazon Managed Service for Grafana**.
4. Assign a **Display name** of your choice and choose **Save**.
5. Navigate to **Configuration** and enter the Amazon Managed Grafana workspace ID in **Namespace**, and enter the Region of your Amazon Managed Grafana workspace.
6. In the **Configuration** tab, enter your Amazon Managed Grafana workspace URL.
7. You can leave the **adminRole** parameter as the default **No Default** and populate it using the **Rules** tab, if an admin requires a corresponding value in AMG. In this example, the **Assertion attribute role** would be set to **adminRole** in Amazon Managed Grafana, with a value of **true**. You can point this value to any attribute in your tenant. Click the **+** to add and configure parameters to meet your organization's requirements.
8. Choose the **Rules** tab, choose **Add Rule**, and give your Rule a name. In the **Conditions** field (the if statement), we add **Email contains [email address]**. In the **Actions** field (the then statement), we select **Set AdminRole in Amazon Managed Service** and we select **Macro** in the **Set adminRole** to dropdown, with a value of **true**. Your organization may choose different rules to resolve different use cases.
9. Choose **Save**. Go to **More Actions** and choose **Reapply entitlement mappings**. You must reapply mappings any time that you create or update rules.
10. Make a note of the **Issuer URL**, which you will use later in the configuration in the Amazon Managed Grafana console. Then choose **Save**.
11. Choose the **Access** tab to assign the OneLogin roles that are to access Amazon Managed Grafana and select an app security policy.

### Step 2: Steps to complete in Amazon Managed Grafana

Complete the following steps in the Amazon Managed Grafana console.

### To finishg setting up OneLogin as an identity provider for Amazon Managed Grafana

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the navigation pane, choose the menu icon.
3. Choose **All workspaces**.
4. Choose the name of the workspace.
5. In the **Authentication** tab, choose **Setup SAML configuration**.
6. Under **Import the metadata**, choose **Upload or copy/paste** and paste the OneLogin Issuer URL that you copied from the OneLogin console in the previous procedure.
7. Under **Assertion mapping**, do the following:

- Make sure that **I want to opt-out of assigning admins to my workspace** is not selected.

#### Note

If you choose **I want to opt-out of assigning admins to my workspace**, you won't be able to use the Amazon Managed Grafana workspace console to administer the workspace, including tasks such as managing data sources, users, and dashboard permissions. You can make administrative changes to the workspace only by using Grafana APIs.

- Set **Assertion attribute role** to the attribute name that you chose. The default value for OneLogin is **adminRole**.
- Set **Admin role values** to value corresponding to your admin users' roles.
- (Optional) If you changed the default attributes in your OneLogin application, expand **Additional settings - optional** and then set the new attribute names.

By default, the OneLogin **displayName** attribute will be passed to the **name** attribute and the OneLogin **mail** attribute will be passed to both the **email** and **login** attributes.

8. Choose **Save SAML Configuration**.

## Ping Identity

Use the following steps to configure Amazon Managed Grafana to use Ping Identity as an identity provider. These steps assume that you have already created your Amazon Managed Grafana workspace and you have made a note of the workspace's ID, URLs, and Region.

### Step 1: Steps to complete in Ping Identity

Complete the following steps in Ping Identity.

#### To set up Ping Identity as an identity provider for Amazon Managed Grafana

1. Sign in to the Ping Identity console as an admin.
2. Choose **Applications**.
3. Choose **Add Application, Search Application Catalog**.
4. Search for the **Amazon Managed Grafana for SAML** application, then choose it and choose **Setup**.
5. In the Ping Identity application, choose **Next** to get to the SAML configuration page. Then make the following SAML settings:
  - For **Assertion Consumer Service**, paste in your **Service provider reply URL** from the Amazon Managed Grafana workspace.
  - For **Entity ID**, paste in your **Service provider identifier** from the Amazon Managed Grafana workspace.
  - Make sure that **Sign Assertion** is selected and that **Encrypt Assertion** is not selected.

6. Choose **Continue to Next Step**.
7. In **SSO Attribute Mapping**, make sure that the Amazon Managed Grafana attribute is in **Application Attribute** and that the Ping Identity attribute is in the **Identity Bridge Attribute**. Then make the following settings:
  - **mail** must be **Email (Work)**.
  - **displayName** must be **Display Name**.
  - **SAML\_SUBJECT** must be **Email (Work)**. And then for this attribute, choose **Advanced**, set the **Name ID Format to send to SP** to **urn:oasis:names:tc:SAML:2.0:nameid-format:transient** and choose **Save**.
  - Add in any other attribute that you would like to pass.
  - Add any other attributes that you would like to pass. For more information about the attributes that you can pass to Amazon Managed Grafana in the assertion mapping, see [Assertion mapping \(p. 6\)](#).
8. Choose **Continue to Next Step**.
9. In **Group Access**, choose which groups to assign this application to.
10. Choose **Continue to Next Step**.
11. Copy the **SAML Metadata URL** which starts with `https://admin-api.pingone.com/latest/metadata/`. You will use this later in the configuration.
12. Choose **Finish**.

## Step 2: Steps to complete in Amazon Managed Grafana

Complete the following steps in the Amazon Managed Grafana console.

### To finishg setting up Ping Identity as an identity provider for Amazon Managed Grafana

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the navigation pane, choose the menu icon.
3. Choose **All workspaces**.
4. Choose the name of the workspace.
5. In the **Authentication** tab, choose **Setup SAML configuration**.
6. Under **Import the metadata**, choose **Upload or copy/paste** and paste the OneLogin URL that you copied in the previous procedure.
7. Under **Assertion mapping**, do the following:

- Make sure that **I want to opt-out of assigning admins to my workspace** is not selected.

#### Note

If you choose **I want to opt-out of assigning admins to my workspace**, you won't be able to use the Amazon Managed Grafana workspace console to administer the workspace, including tasks such as managing data sources, users, and dashboard permissions. You can make administrative changes to the workspace only by using Grafana APIs.

- Set **Assertion attribute role** to the attribute name that you chose.
- Set **Admin role values** to value corresponding to your admin users' roles.
- (Optional) If you changed the default attributes in your Ping Identity application, expand **Additional settings - optional** and then set the new attribute names.

By default, the Ping Identity **displayName** attribute will be passed to the **name** attribute and the Ping Identity **mail** attribute will be passed to both the **email** and **login** attributes.

8. Choose **Save SAML Configuration**.

# Using AWS SSO with your Amazon Managed Grafana workspace

Amazon Managed Grafana integrates with AWS SSO to provide identity federation for your workforce. Using Amazon Managed Grafana and AWS SSO, users are redirected to their existing company directory to sign in with their existing credentials. Then, they are seamlessly signed in to their Amazon Managed Grafana workspace. This ensures that security settings such as password policies and two-factor authentication are enforced. Using AWS SSO does not impact your existing IAM configuration.

If you do not have an existing user directory or prefer not to federate, AWS SSO offers an integrated user directory that you can use to create users and groups for Amazon Managed Grafana. Amazon Managed Grafana does not support the use of IAM users and roles to assign permissions within an Amazon Managed Grafana workspace.

For more information about AWS SSO, see [What is AWS Single Sign-On](#). For more information about getting started with AWS SSO, see [Getting started](#).

To use AWS SSO, you must also have AWS Organizations activated for the account. If needed, Amazon Managed Grafana can activate Organizations for you when you create your first workspace that is configured to use AWS SSO.

## Required IAM permissions to create a workspace that uses AWS SSO

If you use AWS SSO for authorization, the following prerequisite applies when you create your first Amazon Managed Grafana workspace in an account that is not a member of an organization:

- You must be signed on to an IAM principal that has at least the following policies attached:
  - **AWSGrafanaAccountAdministrator**
  - **AWSSSOMemberAccountAdministrator**
  - **AWSSSODirectoryAdministrator**

For more information, see [Create and manage Amazon Managed Grafana workspaces and users in a single standalone account using AWS SSO](#) (p. 385).

# Getting started with Amazon Managed Grafana

This tutorial helps you get started with Amazon Managed Grafana (Amazon Managed Grafana). Create your first workspace, and then connect to the Grafana console in that workspace.

A *workspace* is a logical Grafana server. You can have as many as five workspaces in each Region in your account.

## Topics

- [User authentication \(p. 15\)](#)
- [Necessary permissions \(p. 15\)](#)
- [Create your first workspace \(p. 16\)](#)

## User authentication

For user authentication, Amazon Managed Grafana supports the following options:

- User credentials stored in identity providers (IdPs), with authentication by Security Assertion Markup Language 2.0 (SAML 2.0)
- AWS Single Sign-On

### SAML

If you use SAML, your users must already be created in an identity provider. Amazon Managed Grafana supports identity providers that support SAML 2.0. For more information, see [Using SAML with your Amazon Managed Grafana workspace \(p. 5\)](#).

### AWS SSO

When you create a workspace and choose to use AWS SSO for authentication, Amazon Managed Grafana activates AWS SSO in your account if you are not already using it. For more information about AWS SSO, see [What is AWS Single Sign-On](#).

To use AWS SSO with Amazon Managed Grafana, you must also have AWS Organizations activated in your account. If you don't have it activated already, Amazon Managed Grafana activates it when it activates AWS SSO. If Amazon Managed Grafana enables Organizations, it also creates an organization for you. For more information about Organizations, see [What is AWS Organizations](#).

#### Note

To create a workspace in an account that is already a member of an AWS organization, AWS SSO must be enabled in the management account of the organization. If you enabled AWS SSO in the management account before November 25, 2019, you must also enable AWS SSO-integrated applications in the management account. For more information, see [AWS SSO-integrated applications](#).

## Necessary permissions

To create a workspace that uses an IdP and SAML for authorization, you must be signed on to an IAM principal that has the **AWSGrafanaAccountAdministrator** policy attached.

To create your first workspace that uses AWS SSO for authorization, you must be signed on to an IAM principal that has at least the following policies attached:

- **AWSGrafanaAccountAdministrator**
- **AWSSSOMemberAccountAdministrator**
- **AWSSSODirectoryAdministrator**

For more information, see [Create and manage Amazon Managed Grafana workspaces and users in a single standalone account using AWS SSO \(p. 385\)](#).

## Create your first workspace

Use the following steps to create your first workspace.

### To create a workspace in Amazon Managed Grafana

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. Choose **Create workspace**.
3. For **Workspace name**, enter a name for the workspace.

Optionally, enter a description for the workspace.

4. Choose **Next**.
5. For **Authentication access**, select **AWS Single Sign-On (AWS SSO)**, **Security Assertion Markup Language (SAML)**, or both.

- **AWS SSO**— If you select AWS SSO and you have not already enabled AWS Single Sign-On in your account, you are prompted to enable it by creating your first AWS SSO user. AWS SSO handles user management for access to Amazon Managed Grafana workspaces.

To enable AWS SSO, follow these steps:

- a. Choose **Create user**.
- b. Enter an email address, first name, and last name for the user, and choose **Create user**. For this tutorial, use the name and email address of the account that you want to use to try out Amazon Managed Grafana. You will receive an email message prompting you to create a password for this account for AWS SSO.

### Important

The user that you create does not automatically have access to your Amazon Managed Grafana workspace. You provide the user with access to the workspace in the workspace details page in a later step.

- **SAML**— If you select **SAML**, you will complete the SAML setup after the workspace is created.
6. Choose **Next**.
  7. For this first workspace, confirm that **Service managed** is selected for **Permission type**. This selection enables Amazon Managed Grafana to automatically provision the permissions you need for the AWS data sources that you choose to use for this workspace.
  8. For this tutorial, choose **Current account**.
  9. (Optional) Select the data sources that you want to query in this workspace. For this getting started tutorial, you do not need to select any data sources. However, if you plan to use this workspace with any of the listed data sources, select them here.

Selecting data sources enables Amazon Managed Grafana to create AWS Identity and Access Management (IAM) policies for each of the data sources so that Amazon Managed Grafana has permission to read their data. This does not completely set up these services as data sources for the Grafana workspace. You can do that within the Grafana workspace console.

10. (Optional) If you want Grafana alerts from this workspace to be sent to an Amazon Simple Notification Service (Amazon SNS) notification channel, select **Amazon SNS**. This enables Amazon Managed Grafana to create an IAM policy to publish to the Amazon SNS topics in your account with `TopicName` values that start with `grafana`. This does not completely set up Amazon SNS as a notification channel for the workspace. You can do that within the Grafana console in the workspace.
11. Choose **Next**.
12. Confirm the workspace details, and choose **Create workspace**.

The workspace details page appears.

Initially, the **Status** is **CREATING**.

**Important**

Wait until the status is **ACTIVE** before doing either of the following:

- Completing the SAML setup, if you are using SAML.
- Assigning your AWS SSO users access to the workspace, if you are using AWS SSO.

You might need to refresh your browser to see the current status.

13. If you are using AWS SSO, do the following:
  - a. In the **Authentication** tab, choose **Assign new user or group**.
  - b. Select the check box next to the user that you want to grant workspace access to, and choose **Assign user**.
  - c. Select the check box next to the user, and choose **Make admin**.

**Important**

Assign at least one user as `Admin` for each workspace, in order to sign in to the Grafana workspace console to manage the workspace.

14. If you are using SAML, do the following:
  - a. In the **Authentication** tab, under **Security Assertion Markup Language (SAML)**, choose **Complete setup**.
  - b. For **Import method**, do one of the following:
    - Choose **URL** and enter the URL of the IdP metadata.
    - Choose **Upload or copy/paste**. If you are uploading the metadata, choose **Choose file** and select the metadata file. Or, if you are using copy and paste, copy the metadata into **Import the metadata**.
  - c. For **Assertion attribute role**, enter the name of the SAML assertion attribute from which to extract role information.
  - d. For **Admin role values**, either enter the user roles from your IdP who should all be granted the `Admin` role in the Amazon Managed Grafana workspace, or select **I want to opt-out of assigning admins to my workspace**.

**Note**

If you choose **I want to opt-out of assigning admins to my workspace**, you won't be able to use the Grafana workspace console to administer the workspace, including tasks such as managing data sources, users, and dashboard permissions. You can make administrative changes to the workspace only by using Grafana APIs.



- e. (Optional) To enter additional SAML settings, choose **Additional settings** and do one or more of the following. All of these fields are optional.
- For **Assertion attribute name**, specify the name of the attribute within the SAML assertion to use for the user full "friendly" names for SAML users.
  - For **Assertion attribute login**, specify the name of the attribute within the SAML assertion to use for the user sign-in names for SAML users.
  - For **Assertion attribute email**, specify the name of the attribute within the SAML assertion to use for the user email names for SAML users.
  - For **Login validity duration (in minutes)**, specify how long a SAML user's sign-in is valid before the user must sign in again.
  - For **Assertion attribute organization**, specify the name of the attribute within the SAML assertion to use for the "friendly" name for user organizations.
  - For **Assertion attribute groups**, specify the name of the attribute within the SAML assertion to use for the "friendly" name for user groups.
  - For **Allowed organizations**, you can limit user access to only the users who are members of certain organizations in the IdP. Enter one or more organizations to allow, separating them with commas.
  - For **Editor role values**, enter the user roles from your IdP who should all be granted the `Editor` role in the Amazon Managed Grafana workspace. Enter one or more roles, separated by commas.

**Note**

Any users that are not specifically assigned an Admin or Editor role will be assigned as Viewers.

- f. Choose **Save SAML configuration**.
15. In the workspace details page, choose the URL displayed under **Grafana workspace URL**.
16. Choosing the workspace URL takes you to the landing page for the Grafana workspace console. Do one of the following:
- Choose **Sign in with SAML**, and enter the name and password.
  - Choose **Sign in with AWS SSO**, and enter the email address and password of the user that you created earlier in this procedure. These credentials only work if you have responded to the email from Amazon Managed Grafana that prompted you to create a password for AWS SSO.

You are now in your Grafana workspace, or logical Grafana server. You can start adding data sources to query, visualize, and analyze data. For more information, see [Working in your Grafana workspace \(p. 26\)](#).

# Managing workspaces, users, and policies

The topics in this section explain how to manage your workspaces, users, and policies in Amazon Managed Grafana.

## Topics

- [Creating a workspace \(p. 19\)](#)
- [Managing user and group access to Amazon Managed Grafana \(p. 23\)](#)
- [Managing permissions for data sources and notification channels \(p. 24\)](#)
- [Deleting a workspace \(p. 25\)](#)

## Creating a workspace

A *workspace* is a logical Grafana server. You can have as many as five workspaces in each Region in your account.

## User authentication in a workspace

For user authentication, Amazon Managed Grafana supports the following options:

- User credentials stored in identity providers (IdPs), with authentication by Security Assertion Markup Language 2.0 (SAML 2.0)
- AWS Single Sign-On

### SAML

If you use SAML, your users must already be created in an identity provider. Amazon Managed Grafana supports any identity provider that supports SAML 2.0. For more information, see [Using SAML with your Amazon Managed Grafana workspace \(p. 5\)](#).

### AWS SSO

When you create a workspace and choose to use AWS SSO for authentication, Amazon Managed Grafana activates AWS SSO in your account if you are not already using it. For more information about AWS SSO, see [What is AWS Single Sign-On](#).

To use AWS SSO with Amazon Managed Grafana, you must also have AWS Organizations activated in your account. If you don't have it activated already, Amazon Managed Grafana activates it when it activates AWS SSO. If Amazon Managed Grafana enables Organizations, it also creates an organization for you. For more information about Organizations, see [What is AWS Organizations](#).

### Note

To create a workspace in an account that is already a member of an AWS organization, AWS SSO must be enabled in the management account of the organization. If you enabled AWS SSO in the management account before November 25, 2019, you must also enable AWS SSO-

integrated applications in the management account. For more information, see [AWS SSO-integrated applications](#).

## Necessary permissions

To create a workspace that uses an IdP and SAML for authorization, you must be signed on to an AWS Identity and Access Management (IAM) principal that has the **AWSGrafanaAccountAdministrator** policy attached.

To create your first workspace that uses AWS SSO for authorization, you must be signed on to an IAM principal that has at least the following policies attached:

- **AWSGrafanaAccountAdministrator**
- **AWSSSOMemberAccountAdministrator**
- **AWSSSODirectoryAdministrator**

For more information, see [Create and manage Amazon Managed Grafana workspaces and users in a single standalone account using AWS SSO](#) (p. 385).

To create a workspace, follow these steps.

### To create a workspace in Amazon Managed Grafana

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. Choose **Create workspace**.
3. For **Workspace name**, enter a name for the workspace.

Optionally, enter a description for the workspace.

4. Choose **Next**.
5. For **Authentication access**, select **AWS Single Sign-On (AWS SSO)**, **Security Assertion Markup Language (SAML)**, or both.

- **AWS SSO** — If you select AWS SSO and you have not already enabled AWS Single Sign-On in your account, you are prompted to enable it by creating your first AWS SSO user. AWS SSO handles user management for access to Amazon Managed Grafana workspaces.

To enable AWS SSO, follow these steps:

- a. Choose **Create user**.
- b. Enter an email address, first name, and last name for the user, and choose **Create user**. For this tutorial, use the name and email address of the account that you want to use to try out Amazon Managed Grafana. You will receive an email message prompting you to create a password for this account for AWS SSO.

### Important

The user that you create does not automatically have access to your Amazon Managed Grafana workspace. You provide the user with access to the workspace in the workspace details page in a later step.

- **SAML** — If you select **SAML**, you will complete the SAML setup after the workspace is created.
6. Choose **Service managed** or **Customer managed**, and then choose **Next**.

If you choose **Service managed**, Amazon Managed Grafana automatically creates the IAM roles and provisions the permissions that you need for the AWS data sources in this account that you choose to use for this workspace.

If you want to manage these roles and permissions yourself, choose **Customer managed**.

If you are creating a workspace in a member account of an organization, to be able to choose **Service managed** the member account must be a delegated administrator account in an organization. For more information about delegated administrator accounts, see [Register a delegated administrator](#).

7. If you chose **Service managed**, choose **Current account** to have Amazon Managed Grafana automatically create policies and permissions that allow it to read AWS data only in the current account.

If you are creating a workspace in the management account or a delegated administrator account in an organization, you can choose **Organization** to have Amazon Managed Grafana automatically create policies and permissions that allow it to read AWS data in other accounts in the organizational units that you specify. For more information about delegated administrator accounts, see [Register a delegated administrator](#).

**Note**

Creating resources such as Amazon Managed Grafana workspaces in the management account of an organization is against AWS security best practices.

- a. If you chose **Organization**, and you are prompted to enable AWS CloudFormation StackSets, choose **Enable trusted access**. Then, add the AWS Organizations organizational units (OUs) that you want Amazon Managed Grafana to read data from. Amazon Managed Grafana can then read data from all accounts in each OU that you choose.
- b. If you chose **Organization**, choose **Data sources and notification channels - optional**.
8. Select the AWS data sources that you want to query in this workspace. Selecting data sources enables Amazon Managed Grafana to create IAM roles and permissions that allow Amazon Managed Grafana to read data from these sources. You must still add the data sources in the Grafana workspace console.
9. (Optional) If you want Grafana alerts from this workspace to be sent to an Amazon Simple Notification Service (Amazon SNS) notification channel, select **Amazon SNS**. This enables Amazon Managed Grafana to create an IAM policy to publish to the Amazon SNS topics in your account with `TopicName` values that start with `grafana`. This does not completely set up Amazon SNS as a notification channel for the workspace. You can do that within the Grafana console in the workspace.
10. Choose **Next**.
11. Confirm the workspace details, and choose **Create workspace**.

The workspace details page appears.

Initially, the **Status** is **CREATING**.

**Important**

Wait until the status is **ACTIVE** before doing either of the following:

- Completing the SAML setup, if you are using SAML.
- Assigning your AWS SSO users access to the workspace, if you are using AWS SSO.

You might need to refresh your browser to see the current status.

12. If you are using AWS SSO, do the following:
  - a. In the **Authentication** tab, choose **Assign new user or group**.
  - b. Select the check box next to the user that you want to grant workspace access to, and choose **Assign user**.
  - c. Select the check box next to the user, and choose **Make admin**.

### Important

Assign at least one user as `Admin` for each workspace, in order to sign in to the Grafana workspace console to manage the workspace.

13. If you are using SAML, do the following:
    - a. In the **Authentication** tab, under **Security Assertion Markup Language (SAML)**, choose **Complete setup**.
    - b. For **Import method**, do one of the following:
      - Choose **URL** and enter the URL of the IdP metadata.
      - Choose **Upload or copy/paste**. If you are uploading the metadata, choose **Choose file** and select the metadata file. Or, if you are using copy and paste, copy the metadata into **Import the metadata**.
    - c. For **Assertion attribute role**, enter the name of the SAML assertion attribute from which to extract role information.
    - d. For **Admin role values**, either enter the user roles from your IdP who should all be granted the `Admin` role in the Amazon Managed Grafana workspace, or select **I want to opt-out of assigning admins to my workspace**.
- Note**  
If you choose **I want to opt-out of assigning admins to my workspace**, you won't be able to use the console to administer the workspace, including tasks such as managing data sources, users, and dashboard permissions. You can make administrative changes to the workspace only by using Amazon Managed Grafana APIs.
- e. (Optional) To enter additional SAML settings, choose **Additional settings** and do one or more the following. All of these fields are optional.
    - For **Assertion attribute name**, specify the name of the attribute within the SAML assertion to use for the user full "friendly" names for SAML users.
    - For **Assertion attribute login**, specify the name of the attribute within the SAML assertion to use for the user sign-in names for SAML users.
    - For **Assertion attribute email**, specify the name of the attribute within the SAML assertion to use for the user email names for SAML users.
    - For **Login validity duration (in minutes)**, specify how long a SAML user's sign-in is valid before the user must sign in again. The default is 1 day, and the maximum is 30 days.
    - For **Assertion attribute organization**, specify the name of the attribute within the SAML assertion to use for the "friendly" name for user organizations.
    - For **Assertion attribute groups**, specify the name of the attribute within the SAML assertion to use for the "friendly" name for user groups.
    - For **Allowed organizations**, you can limit user access to only the users who are members of certain organizations in the IdP. Enter one or more organizations to allow, separating them with commas.
    - For **Editor role values**, enter the user roles from your IdP who should all be granted the `Editor` role in the Amazon Managed Grafana workspace. Enter one or more roles, separated by commas.
  - f. Choose **Save SAML configuration**.
14. In the workspace details page, choose the URL displayed under **Grafana workspace URL**.
  15. Choosing the workspace URL takes you to the landing page for the Grafana workspace console. Do one of the following:
    - Choose **Sign in with SAML**, and enter the name and password.
    - Choose **Sign in with AWS SSO**, and enter the email address and password of the user that you created earlier in this procedure. These credentials only work if you have responded to the email from Amazon Managed Grafana that prompted you to create a password for AWS SSO.

You are now in your Grafana workspace, or logical Grafana server. You can start adding data sources to query, visualize, and analyze data. For more information, see [Working in your Grafana workspace \(p. 26\)](#).

## Managing user and group access to Amazon Managed Grafana

To grant a user or a user group access to Amazon Managed Grafana workspaces, the user or group must first be provisioned in an IdP or in AWS Single Sign-On. For more information, see [User authentication in Amazon Managed Grafana \(p. 5\)](#).

To manage user and group access, you must be signed in to an account that has the AWS Identity and Access Management (IAM) policy **AWSGrafanaWorkspacePermissionManagement**, or equivalent permissions. If you are managing users with AWS SSO, you must also have the **AWSSSOMemberAccountAdministrator** and **AWSSSODirectoryReadOnly** IAM policies, or equivalent permissions. For more information, see [Assign and unassign users access to Amazon Managed Grafana \(p. 385\)](#).

### To manage user access to the Amazon Managed Grafana workspace

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the navigation pane, choose the menu icon.
3. Choose **All workspaces**.
4. Choose the name of the workspace that you want to manage.
5. Choose the **Authentication** tab.
6. If you are using AWS SSO in this workspace, choose **Configure users and user groups** and do one or more of the following:
  - To give a user access to the Amazon Managed Grafana workspace, select the check box next to the user, and choose **Assign user**.
  - To make a user an Admin of the workspace, choose **Make admin**.
  - To remove workspace access for a user, choose **Unassign user**.
  - To add groups of users such as an LDAP group, choose the **Assigned user groups** tab. Then, do one of the following:
    - To give all members of a group access to the Amazon Managed Grafana workspace, select the check box next to the group, and choose **Assign group**.
    - To give all members of a group the Admin role in the workspace, choose **Make admin**.
    - To remove workspace access for all members of a group, choose **Unassign group**.
7. If you are using SAML in this workspace, choose **SAML configuration** and do one or more of the following:
  - For **Import method**, do one of the following:
    - Choose **URL** and enter the URL of the IdP metadata.
    - Choose **Upload or copy/paste**. If you are uploading the metadata, choose **Choose file** and select the metadata file. Or, if you are using copy and paste, copy the metadata into **Import the metadata**.
  - For **Assertion attribute role**, enter the name of the SAML assertion attribute from which to extract role information.

- For **Admin role values**, either enter the user roles from your IdP who should all be granted the `Admin` role in the Amazon Managed Grafana workspace, or select **I want to opt-out of assigning admins to my workspace**.

**Note**

If you choose **I want to opt-out of assigning admins to my workspace**, you won't be able to use the console to administer the workspace, including tasks such as managing data sources, users, and dashboard permissions. You can make administrative changes to the workspace only by using Amazon Managed Grafana APIs.

- (Optional) To enter additional SAML settings, choose **Additional settings** and do one or more the following, and then choose **Save SAML configuration**. All of these fields are optional.
    - For **Assertion attribute name**, specify the name of the attribute within the SAML assertion to use for the user full "friendly" names for SAML users.
    - For **Assertion attribute login**, specify the name of the attribute within the SAML assertion to use for the user sign-in names for SAML users.
    - For **Assertion attribute email**, specify the name of the attribute within the SAML assertion to use for the user email names for SAML users.
    - For **Login validity duration (in minutes)**, specify how long a SAML user's sign-in is valid before the user must sign in again.
    - For **Assertion attribute organization**, specify the name of the attribute within the SAML assertion to use for the "friendly" name for user organizations.
    - For **Assertion attribute groups**, specify the name of the attribute within the SAML assertion to use for the "friendly" name for user groups.
    - For **Allowed organizations**, you can limit user access to only the users who are members of certain organizations in the IdP. Enter one or more organizations to allow, separating them with commas.
    - For **Editor role values**, enter the user roles from your IdP who should all be granted the `Editor` role in the Amazon Managed Grafana workspace. Enter one or more roles, separated by commas.
8. Alternatively, to add groups of users such as an LDAP group, choose the **User Group** tab. Then, do one of the following:
- To give all members of a group access to the Amazon Managed Grafana workspace, select the check box next to the group, and choose **Assign group**.
  - To give all members of a group the `Admin` role in the workspace, choose **Make admin**.
  - To remove workspace access for all members of a group, choose **Unassign group**.

## Managing permissions for data sources and notification channels

You can use the Amazon Managed Grafana console to have Amazon Managed Grafana create AWS Identity and Access Management (IAM) policies and permissions for the AWS data sources and notification channels that you want to use in the Amazon Managed Grafana workspace.

### To manage permissions and policies for data sources and notification channels

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the navigation pane, choose the menu icon.
3. Choose **All workspaces**.
4. Choose the name of the workspace that you want to manage.

5. To switch between using **Service managed** and **Customer managed** permissions, choose the edit icon for **IAM role** and then make your selection. For more information, see [Amazon Managed Grafana permissions and policies for AWS data sources and notification channels](#) (p. 390).

If you change from **Service managed** permissions to **Customer managed** permissions, the roles and policies that Amazon Managed Grafana created for you are not deleted in the current account. If you were using **Service managed** permissions for an organization, the roles and policies in other accounts in the organization are deleted.

6. Choose the **Data sources** tab.
7. If you are using **Service managed** permissions, you can choose **Edit** next to **IAM permission access settings** to change whether your **Service managed** permissions apply to only the current account or to an entire organization. For more information, see [Amazon Managed Grafana permissions and policies for AWS data sources and notification channels](#) (p. 390).

Under **Data sources**, select the AWS data sources that you want to query in this workspace. Selecting data sources enables Amazon Managed Grafana to create the IAM roles and permissions that allow Amazon Managed Grafana to read data from these sources. You must still add the data sources in the Grafana workspace console.

To manage AWS services that can be used as notification channels, choose **Notification channels**.

Select the AWS notification channel that you want to use in this workspace. Selecting a notification channel enables Amazon Managed Grafana to create IAM roles and permissions that allow Amazon Managed Grafana to use these services. You must still add the notification channels in the Grafana workspace console.

## Deleting a workspace

If you delete an Amazon Managed Grafana workspace, all the configuration data for that workspace is also deleted. This includes dashboards, data source configuration, alerts, and snapshots.

### To delete an Amazon Managed Grafana workspace

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the navigation pane, choose the menu icon.
3. Choose **All workspaces**.
4. Choose the name of the workspace that you want to delete.
5. Choose **Delete**.
6. To confirm the deletion, enter the name of the workspace and choose **Delete**.



# Working in your Grafana workspace

The topics in this section explain how to use your Amazon Managed Grafana workspace.

Before you work in your Amazon Managed Grafana workspace, you must connect to it.

## To connect to your Grafana workspace console

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the upper left corner of the page, choose the menu icon and then choose **All workspaces**.
3. Choose the name of the workspace.
4. In the workspace details page, choose the URL displayed under **Grafana workspace URL**.
5. Choosing the workspace URL takes you to the landing page for the Grafana workspace console. Choose **Sign in with AWS SSO**, and enter the email address and password

## Topics

- [Users, teams, and permissions \(p. 26\)](#)
- [Getting started in your Grafana workspace console \(p. 32\)](#)
- [Data sources \(p. 39\)](#)
- [Panels \(p. 188\)](#)
- [Dashboards \(p. 237\)](#)
- [Explore \(p. 254\)](#)
- [Linking \(p. 260\)](#)
- [Templates and variables \(p. 265\)](#)
- [Alerts \(p. 282\)](#)
- [Change your preferences \(p. 290\)](#)

## Users, teams, and permissions

Use the information in these sections to manage your users, teams, and permissions in Amazon Managed Grafana.

## Topics

- [Users \(p. 26\)](#)
- [User roles \(p. 27\)](#)
- [Managing teams \(p. 27\)](#)
- [Using permissions \(p. 29\)](#)

## Users

In Amazon Managed Grafana, you don't add users in the Grafana workspace. Instead, you use a single-sign on (SSO) provider to grant users access to Amazon Managed Grafana workspaces. For more information, see [Managing user and group access to Amazon Managed Grafana \(p. 23\)](#).

## User roles

In Amazon Managed Grafana, each user enabled to use the Amazon Managed Grafana workspace are assigned to one of three roles.

- **Admin role**— Users with the Admin role can do the following:
  - Can add, edit, and delete data sources.
  - Can add and edit users and teams.
  - Can add, edit, and delete folders containing dashboards.
  - Can do everything allowed by the Editor role.
- **Editor role**— Users with the Editor role can do the following:
  - Can view, add, and edit dashboards, panels, and alert rules in dashboards they have access to. This can be disabled on specific folders and dashboards.
  - Can create, update, or delete playlists.
  - Can access Explore.
  - Can add, edit, and delete notification channels.
  - Cannot add, edit, or delete data sources.
  - Can do everything allowed by the Viewer role.
- **Viewer role**— Users with the Viewer role can do the following:
  - Can view any dashboard they have access to. This can be disabled on specific folders and dashboards.
  - Cannot create, update, or delete playlists.
  - Cannot access Explore.
  - Cannot add, edit, and delete notification channels.
  - Cannot add, edit, or delete data sources.
  - Cannot add, edit, or delete dashboards or panels.
  - Cannot manage other users or teams.

If your workspace uses AWS SSO for authentication, you can use the following steps to assign roles.

### To assign a role to a user

1. In the Grafana workspace console, choose the Configuration (gear) icon in the left navigation panel.
2. Choose **Users**.
3. Next to a user's name, select **Admin**, **Editor**, or **Viewer**.

If your workspace uses SAML for authentication, user roles are defined only by assertion attributes. If you make role assignments using the Grafana workspace console, the changes do not take effect. For more information, see [Assertion mapping \(p. 6\)](#).

## Managing teams

Using *teams* enables you to grant permissions to a group of users at the same time. You can also set up team sync to automatically synchronize team membership between your Grafana workspace and your authorization provider.

### Creating or removing a team

Create teams to manage users in groups.

### To create a team

1. In the sidebar, pause on the **Configuration**(gear) icon, and choose **Teams**.
2. Choose **New team**.
3. For **Name**, enter a name for the new team, and then choose **Create**.

### To remove a team

1. In the sidebar, pause on the **Configuration** (gear) icon, and choose **Teams**.
2. To the right of the team's name, choose **X**.
3. To confirm, choose **Delete**.

## Adding or removing a user from a team

Use these steps to add users to teams or remove them from teams.

### To add a user to a team

1. In the sidebar, pause on the **Configuration** (gear) icon, and choose **Teams**.
2. Choose the team that you want to add the user to.
3. Choose **Add member**.
4. In the **Add team member** box, select the user to add to the team, and then choose **Add to team**.

### To remove a user from a team

1. In the sidebar, pause on the **Configuration** (gear) icon, and choose **Teams**.
2. Choose the team that you want to remove the user from.
3. To the right of the user's name, choose **X**.
4. To confirm, choose **Delete**.

## Using team sync

With *team sync*, you can set up synchronization between your authorization provider's groups and the teams in Grafana. You can automatically add or remove LDAP users who are members of certain teams or groups that are synchronized with certain teams in Grafana workspaces in Amazon Managed Grafana. The synchronization happens only when a user logs in, unless LDAP is used together with active background synchronization.

Grafana workspaces keep track of all synchronized users in teams, and you can see which users have been synchronized in the team members list. This mechanism allows Amazon Managed Grafana to remove an existing synchronized user from a team when its LDAP group membership changes. This mechanism also enables you to manually add a user as member of a team, and the user is not removed when the user signs in. This gives you the flexibility to combine LDAP group memberships and Amazon Managed Grafana team memberships.

### To synchronize a Grafana team with an external group.

1. In the Grafana console, navigate to **Configuration, Teams**.
2. Select a team and on the **External group sync** tab, choose **Add group**.
3. Insert the value of the group that you want to sync with. This becomes the `GroupID` in the Grafana console.

4. Choose **Add group**.

## Using permissions

What you can do in a Grafana workspace in Amazon Managed Grafana is defined by the *permissions* that are associated with your user account.

Amazon Managed Grafana uses three types of permissions:

- Permissions granted as a Grafana admin
- Permissions associated with your membership on a team
- Permissions granted to a specific folder or dashboard

You can be granted permissions based on your admin status, dashboard or folder permissions assigned to your user account, and data source permissions.

## Dashboard and folder permissions overview

By using dashboard and folder permissions, you can remove the default role-based permissions for editors and viewers. You can then assign permissions to specific users and teams. For more information, see [Dashboard and folder permissions \(p. 29\)](#).

## Data source permissions overview

By default, a data source can be queried by any user. For example, a user with the `Viewer` role can issue any possible query to a data source, not just those queries that exist on dashboards they have access to.

Using data source permissions, you can change the default permissions for data sources and restrict query permissions to specific **Users** and **Teams**. For more information, see [Data source permissions \(p. 31\)](#).

## Dashboard and folder permissions

For dashboards and dashboard folders, you can use the **Permissions** page to remove the default role based permissions for **Editors** and **Viewers**. On this page, you can add and assign permissions to specific **Users** and **Teams**.

Amazon Managed Grafana provides the following permission levels:

- **Admin**: Can edit and create dashboards and edit permissions. Can also add, edit, and delete folders.
- **Edit**: Can edit and create dashboards. **Cannot** edit folder or dashboard permissions, or add, edit, or delete folders.
- **View**: Can only view existing dashboards and folders.

## Granting folder permissions

### To grant folder permissions

1. In the sidebar, pause on the **Dashboards** (squares) icon, and then choose **Manage**.
2. Pause on a folder, and then choose **Go to folder**.
3. On the **Permissions** tab, choose **Add Permission**.
4. In the **Add Permission For** dialog box, choose **User**, **Team**, or one of the role options.

5. In the second box, select the user or team to add permission for. Skip this step if you selected a role option in the previous step.
6. In the third box, select the permission that you want to add.
7. Choose **Save**.

## Granting dashboard permissions

### To grant dashboard permissions

1. In the top right corner of your dashboard, choose the cog icon to go to **Dashboard settings**.
2. On the **Permissions** tab, choose **Add Permission**.
3. In the **Add Permission For** dialog box, select **User**, **Team**, or one of the role options.
4. In the second box, select the user or team to add permission for. Skip this step if you selected a role option in the previous step.
5. In the third box, select the permission you that want to add.
6. Choose **Save**.

## Restricting access

The highest permission always wins.

- You cannot override permissions for users with the `Admin` role. Admins always have access to everything.
- A more specific permission with a lower permission level does not have any effect if a more general rule exists with a higher permission level. You need to remove or lower the permission level of the more general rule.

## How Amazon Managed Grafana resolves multiple permissions – examples

The following examples show how multiple permissions are resolved.

### Example 1: `user1` has the `Editor` role

Permissions for a dashboard:

- Everyone with the `Editor` role can edit.
- `user1` can view.

Result: `user1` has Edit permission because the highest permission always wins.

### Example 2: `user1` has the `Viewer` role and is a member of `team1`

Permissions for a dashboard:

- Everyone with the `Viewer` role can view.
- `user1` has the `Editor` role and can edit.
- `team1` has the `Admin` role.

Result: `user1` has Admin permission because the highest permission always wins.

### Example 3: `user1` has multiple permissions at different levels

Permissions for a dashboard:

- `user1` has the `Admin` role (inherited from parent folder).
- `user1` has the `Editor` role and can edit.

Result: You cannot override to a lower permission. `user1` has `Admin` permission because the highest permission always wins.

### Summary

- **View:** Can only view existing dashboards or folders.
- A more specific permission with a lower permission level will not have any effect if a more general rule exists with higher permission level.

## Data source permissions

You can use data source permissions to restrict access for users to query a data source. For each data source, there is a permission page where you can enable or restrict query permissions to specific **Users** and **Teams**.

### Enabling data source permissions

By default, data sources can be queried by any user. For example, a user with the `Viewer` role can issue any possible query to a data source, not just queries that exist on dashboards they have access to.

When permissions are enabled for a data source, you restrict admin and query access for that data source to Admin users.

#### To enable permissions for a data source

1. Navigate to **Configuration, Data Sources**.
2. Select the data source that you want to enable permissions for.
3. On the **Permissions** tab, choose **Enable**.

#### Warning

If you enable permissions for the default data source, users who are not listed in the permissions are unable to invoke queries. Panels that use the default data source will return the `Access denied to data source` error for those users.

### Allowing users and teams to query a data source

After you enable permissions for a data source, you can assign query permissions to users and teams. The query permissions will allow access to query the data source.

#### To assign query permissions to users and teams

1. Choose **Configuration, Data Sources**.
2. Select the data source that you want to assign query permissions for.
3. On the **Permissions** tab, choose **Add Permission**.
4. Select **Team** or **User**.
5. Select the team or user that you want to grant query access to, and then choose **Save**.

### Disabling data source permissions

If you have enabled permissions for a data source and want to return data source permissions to the default, follow these steps.

**Note**

*All existing permissions created for the data source will be deleted.*

**To disable permissions for a data source**

1. Choose **Configuration, Data Sources**.
2. Select the data source that you want to disable permissions for.
3. On the **Permissions** tab, choose **Disable Permissions**.

## Getting started in your Grafana workspace console

This section provides a high-level look at the Grafana console inside an Amazon Managed Grafana workspace. It's a good place to learn how to use the Grafana console.

### What is Grafana?

Grafana is open source visualization and analytics software. You can use it to query, visualize, alert on, and explore your metrics no matter where they are stored.

For example, if you want to view the metric, log, and trace data for your application, you might create a dashboard. If you are the administrator for a corporation and you manage Grafana for multiple teams, you might need to set up provisioning and authentication.

The following sections provide an overview of things you can do with your Grafana database and links so that you can learn more.

### Explore metrics and logs

Explore your data through one-time, or ad hoc, queries and dynamically drilling down. You can split the view and compare different time ranges, queries, and data sources side by side.

For more information, see [Explore \(p. 254\)](#).

### Alerts

If you're using Grafana alerting, alerts can be sent through different alert notifiers, including the following:

- Amazon SNS
- PagerDuty
- VictorOps
- OpsGenie
- Slack

For more information, see [Alerts \(p. 282\)](#).

### Annotations

Annotate graphs with rich events from different data sources. Pause on events to see the full event metadata and tags.

This feature, which shows up as a graph marker in Grafana, is useful for correlating data in case something goes wrong. You can create the annotations manually by pressing **Ctrl** while you choose a graph and then entering some text. Or you can fetch data from any data source.

For more information, see [Annotations \(p. 238\)](#).

## Dashboard variables

Use template variables to create dashboards that can be reused for many different use cases. With these templates, values aren't hardcoded. This means that you can use a dashboard for multiple servers. For example, if you have a production server and a test server, you can use the same dashboard for both.

Templating helps you drill down into your data. For example, you can drill down from all data to North America data, down to Texas data, and beyond. You can also share these dashboards across teams within your organization. If you create a great dashboard template for a popular data source, you can also contribute it to the whole community to customize and use.

For more information, see [Templates and variables \(p. 265\)](#).

## Creating a dashboard

Follow these steps to create a dashboard in the Grafana console.

### To create your first dashboard

1. Choose the + icon on the left panel, choose **Create Dashboard**, and then choose **Add new panel**.
2. In the **New Dashboard/Edit Panel** view, choose the **Query** tab.
3. Configure your query by selecting the data source that you'd like to query. For example, if you have **TestDB** added as a data source, this generates a sample dashboard called the Random Walk dashboard.

## Introduction to time series

Imagine that you wanted to know how the temperature outside changes throughout the day. Once every hour, you'd check the thermometer and write down the time along with the current temperature. After a while, you'd have something like the following data.

Time	Value
09:00	24°C
10:00	26°C
11:00	27°C

Temperature data such as this are one example of a *time series*—a sequence of measurements, ordered in time. Every row in the table represents one individual measurement at a specific time.

Tables are useful when you want to identify individual measurements, but they can make it difficult to see the big picture. A more common visualization for time series is the *graph*, which instead places each measurement along a time axis. Visual representations such as the graph make it easier to discover patterns and features of the data that otherwise would be difficult to see.

Other examples of time series are:



- CPU and memory usage
- Sensor data
- Stock market index

While each of these examples is a sequence of chronologically ordered measurements, they also share other attributes:

- New data are appended at the end, at regular intervals—for example, hourly at 09:00, 10:00, 11:00, and so on.
- Measurements are seldom updated after they are added. For example, yesterday's temperature doesn't change.

Time series are powerful. They help you understand the past by letting you analyze the state of the system at any point in time. Time series could tell you that the server crashed moments after the free disk space went down to zero.

Time series can also help you predict the future by uncovering trends in your data. For example, if the number of registered users has been increasing monthly by 4 percent for the past few months, you can predict how large your user base will be at the end of the year.

Some time series have patterns that repeat themselves over a known period. For example, the temperature is typically higher during the day, before it dips down at night. By identifying these periodic, or *seasonal*, time series, you can make confident predictions about the next period. If you know that the system load peaks every day around 18:00, you can add more machines right before.

## Aggregating time series

Depending on what you're measuring, the data can vary greatly. What if you wanted to compare periods longer than the interval between measurements? If you'd measure the temperature once every hour, you'd end up with 24 data points per day. To compare the temperature in August over the years, you'd have to combine the 31 times 24 data points into one.

Combining a collection of measurements is called *aggregation*. There are several ways to aggregate time series data. Here are some common ones:

- **Average** returns the sum of all values divided by the total number of values.
- **Min** and **Max** return the smallest, and largest value in the collection.
- **Sum** returns the sum of all values in the collection.
- **Count** returns the number of values in the collection.

For example, by aggregating the data in a month, you can determine that August 2017 was, on average, warmer than the year before. If you wanted to see which month had the highest temperature, you'd compare the maximum temperature for each month.

How you aggregate your time series data is an important decision, and it depends on the story that you want to tell with your data. It's common to use different aggregations to visualize the same time series data in different ways.

## Time series and monitoring

In the IT industry, time series data are often collected to monitor things such as infrastructure, hardware, or application events. Machine-generated time series data are typically collected with short intervals, so that you can react to any unexpected changes, moments after they occur. The data accumulate at a rapid pace, making it vital to have a way to store and query data efficiently. As a result, databases that are optimized for time series data have seen a rise in popularity in recent years.

## Time series databases

A time series database (TSDB) is a database explicitly designed for time series data. While it's possible to use any regular database to store measurements, a TSDB comes with some useful optimizations.

Modern TSDBs take advantage of the fact that measurements are only ever appended, and rarely updated or removed. For example, the timestamps for each measurement change little over time, which results in redundant data being stored.

The following example shows a sequence of Unix timestamps.

```
1572524345, 1572524375, 1572524404, 1572524434, 1572524464
```

Looking at these timestamps, they all start with 1572524, leading to poor use of disk space. Instead, you could store each subsequent timestamp as the difference, or *delta*, from the first one, as shown in the following example.

```
1572524345, +30, +29, +30, +30
```

You could even take it a step further by calculating the deltas of these deltas, as shown in the following example.

```
1572524345, +30, -1, +1, +0
```

If measurements are taken at regular intervals, most of these delta-of-deltas will be 0. Because of optimizations like these, TSDBs use drastically less space than other databases.

Another feature of a TSDB is the ability to filter measurements by using *tags*. Each data point is labeled with a tag that adds context information, such as where the measurement was taken.

The following TSDBs are supported by Grafana:

- [Graphite](#)
- [InfluxDB](#)
- [Prometheus](#)

```
weather,location=us-midwest temperature=82 1465839830100400200
|-----|-----|-----|
|         |         |         |
+-----+-----+-----+-----+
|measurement|,tag_set| |field_set| |timestamp|
+-----+-----+-----+-----+
```

## Collecting time series data

Now that you have a place to store your time series, how do you actually gather the measurements? To collect time series data, you'd typically install a *collector* on the device, machine, or instance that you want to monitor. Some collectors are made with a specific database in mind, and some support different output destinations.

Here are some examples of collectors:

- [collectd](#)
- [statsd](#)
- [Prometheus exporters](#)
- [Telegraf](#)

A collector either *pushes* data to a database or lets the database *pull* the data from the collector. Each approach comes with its own set of pros and cons.

	Pros	Cons
Push	Easier to replicate data to multiple destinations.	The TSDB has no control over how much data gets sent.
Pull	More control over how the amount of data ingested and data authenticity.	Firewalls, VPNs, or load balancers can make it hard to access the agents.

Because it's inefficient to write every measurement to the database, collectors pre-aggregate the data and write to the TSDB at regular intervals.

## Time series dimensions

With time series data, the data is often a set of multiple time series. Many Grafana data sources support this type of data.

The common case is issuing a single query for a measurement with one or more additional properties as dimensions. For example, you might query a temperature measurement along with a location property. In this case, multiple series are returned back from that single query, and each series has unique location as a dimension.

To identify unique series within a set of time series, Grafana stores dimensions in *labels*.

### Labels

Each time series in Grafana optionally has labels. Labels are a set of key-value pairs for identifying dimensions. Example labels are `{location=us}` or `{country=us,state=ma,city=boston}`. Within a set of time series, the combination of its name and labels identifies each series. For example, `temperature {country=us,state=ma,city=boston}`.

Different sources of time series data have dimensions stored natively, or common storage patterns that enable the data to be extracted into dimensions.

Usually, TSDBs natively support dimensionality. Prometheus stores dimensions in *labels*. In TSDBs such as Graphite or OpenTSDB, the term *tags* is used instead.

In table databases such SQL, these dimensions are generally the `GROUP BY` parameters of a query.

### Multiple dimensions in table format

In SQL or SQL-like databases that return table responses, additional dimensions usually are columns in the query response table.

#### Single dimension

For example, consider a query like the following example.

```
SELECT BUCKET(StartTime, 1h), AVG(Temperature) AS Temp, Location FROM T
GROUP BY BUCKET(StartTime, 1h), Location
ORDER BY time asc
```

The query might return a table with three columns.

StartTime	Temp	Location
09:00	24	LGA
09:00	20	BOS
10:00	26	LGA
10:00	22	BOS

The table format is *long* formatted time series, also called *tall*. It has repeated timestamps, and repeated values in Location. In this case, two time series in the set would be identified as Temp {Location=LGA} and Temp {Location=BOS}.

Individual time series from the set are extracted by using the following dimensions:

- The time typed column `StartTime` as the time index of the time series
- The numeric typed column `Temp` as the series name
- The name and values of the string typed `Location` column to build the labels, such as Location=LGA

### Multiple dimensions

If the query is updated to select and group by more than one string column (for example, `GROUP BY BUCKET(StartTime, 1h), Location, Sensor`), an additional dimension is added.

StartTime	Temp	Location	Sensor
09:00	24	LGA	A
09:00	24.1	LGA	B
09:00	20	BOS	A
09:00	20.2	BOS	B
10:00	26	LGA	A
10:00	26.1	LGA	B
10:00	22	BOS	A
10:00	22.2	BOS	B

In this case, the labels that represent the dimensions have two keys based on the two string typed columns, Location and Sensor. The data result in four series:

- Temp {Location=LGA,Sensor=A}

- Temp {Location=LGA,Sensor=B}
- Temp {Location=BOS,Sensor=A}
- Temp {Location=BOS,Sensor=B}

### Note

**Note:** Multiple dimensions are not supported in a way that maps to multiple alerts in Grafana. Instead, they are treated as multiple conditions to a single alert.

## Multiple values

In the case of SQL-like data sources, more than one numeric column can be selected, with or without additional string columns to be used as dimensions; for example, `AVG(Temperature) AS AvgTemp, MAX(Temperature) AS MaxTemp`. This, if combined with multiple dimensions, can result in numerous series. Selecting multiple values is currently designed to be used only with visualization.

## Introduction to histograms and heatmaps

A histogram is a graphical representation of the distribution of numerical data. It groups values into buckets (sometimes also called bins). Then it counts how many values fall into each bucket.

Instead of graphing the actual values, histograms graph the buckets. Each bar represents a bucket, and the bar height represents the frequency (such as count) of values that fell into the interval of that bucket.

Histograms look only at *value distributions* over a specific time range. The problem with histograms is that you cannot see any trends or changes in the distribution over time. This is where heatmaps become useful.

## Heatmaps

A *heatmap* is like a histogram over time, where each time slice represents its own histogram. Instead of using bar height as a representation of frequency, it uses cells, coloring a cell proportional to the number of values in the bucket.

## Pre-bucketed data

A number of data sources support histogram over time, including the following:

- Amazon OpenSearch Service (by using a histogram bucket aggregation)
- Prometheus (with the [histogram](#) metric type and the *Format as* option set to **Heatmap**)

Generally, you can use any data source that returns series with names representing bucket bound or returns series sorted by the bound in ascending order.

## Raw data vs. aggregated data

If you use the heatmap with regular time series data (not pre-bucketed), it's important to remember that your data are often already aggregated by your time series backend. Most time series queries don't return raw sample data. Instead, they include a group by time interval or `maxDataPoints` limit coupled with an aggregation function (usually average).

It depends on the time range of your query. The important point is to know that the histogram bucketing that Grafana performs might be done on already aggregated and averaged data. For more accurate heatmaps, it's better to do the bucketing during metric collection or to store the data in OpenSearch, or in the other data source that supports doing histogram bucketing on the raw data.

If you remove or lower the group by time (or raise `maxDataPoints`) in your query to return more data points, your heatmap is more accurate. But this can also put a heavy load on your CPU and memory. If the number of data points becomes unreasonably large, it might cause stalls and crashes.

## Data sources

Amazon Managed Grafana supports many different *data sources*, which are storage backends that you can query in Grafana, such as to build dashboards. Each data source has a specific query editor that is customized for the features and capabilities that the particular data source exposes.

The query language and capabilities of each data source are different. You can combine data from multiple data sources onto a single dashboard.

Every AWS account that uses Amazon Managed Grafana has access to many data sources. Additional data sources are available if you upgrade your workspace to Grafana Enterprise. Each of those data sources is listed in the following sections. Amazon Managed Grafana also includes three special data sources:

- **TestDB** – Use this built-in data source to generate random walk data. This is useful for testing visualizations and running experiments.
- **Mixed** – Use this to query multiple data sources in the same panel. When you use this data source, you can specify a data source for every new query that you add. The first query uses the data source that you specified before you selected Mixed.

You cannot change an existing query to use Mixed Data Source.

- **Dashboard** – Use this to use a result set from another panel in the same dashboard.

### Topics

- [How Amazon Managed Grafana works with AWS Organizations for AWS data source access \(p. 39\)](#)
- [Built-in data sources \(p. 41\)](#)
- [Data sources available in Grafana Enterprise \(p. 138\)](#)

## How Amazon Managed Grafana works with AWS Organizations for AWS data source access

With Organizations, you can centrally manage data source configuration and permission settings for multiple AWS accounts. In an account that has an Amazon Managed Grafana workspace, you can specify other organizational units that will make their AWS data sources available for viewing in the account with the Amazon Managed Grafana workspace.

For example, you can use one account in the organization as an Amazon Managed Grafana *management account*, and give this account access to data sources in other accounts in the organization. In the management account, list all the organizational units that have AWS data sources that you want to access with the management account. Then, when you go into the Grafana console in the Amazon Managed Grafana workspace, setting up these data sources will be much simpler, with the necessary roles and permission policies automatically created.

For more information about Organizations, see [What is AWS Organizations](#).

Amazon Managed Grafana uses AWS CloudFormation StackSets to automatically create the IAM roles necessary for Amazon Managed Grafana to automatically connect to data sources across your AWS organization. To allow Amazon Managed Grafana to manage your IAM policies for accessing data sources

across your organization, you must enable AWS CloudFormation StackSets in the management account of your organization. Amazon Managed Grafana can automatically enable this when it is needed the first time.

## Deployment scenarios for integration with AWS SSO and Organizations

If you are using Amazon Managed Grafana with both AWS SSO and Organizations, we recommend that you create an Amazon Managed Grafana workspace in your organization using one of the following three scenarios. For each scenario, you need to be signed in to an account with sufficient permissions. For more information, see [Sample policies for Amazon Managed Grafana \(p. 384\)](#).

### Standalone account

A standalone account is an AWS account that is not a member of an organization in Organizations. This is a likely scenario if you are first trying out AWS.

In this scenario, Amazon Managed Grafana will automatically enable AWS SSO and Organizations if they are not already enabled, as long as you are signed in to an account that has the **AWSGrafanaAccountAdministrator**, **AWSSSOMemberAccountAdministrator**, and **AWSSSODirectoryAdministrator** policies. For more information, see [Create and manage Amazon Managed Grafana workspaces and users in a single standalone account using AWS SSO \(p. 385\)](#).

### Member account of an existing organization where AWS SSO is already configured

To create a workspace in a member account, you must be signed in to an account that has the **AWSGrafanaAccountAdministrator**, **AWSSSOMemberAccountAdministrator**, and **AWSSSODirectoryAdministrator** policies. For more information, see [Grafana administrator in a member account using AWS SSO \(p. 385\)](#).

If you create a workspace in a member account and you want that workspace to access resources from other AWS accounts in your organization, you must use customer-managed permissions in the workspace. For more information, see [Customer-managed permissions \(p. 396\)](#).

To use service-managed permissions to allow a workspace to access resources from other AWS accounts in the organization, you would have to create the workspace in the management account of the organization. However, it is not a best practice to create Amazon Managed Grafana workspaces or other resources in the management account of an organization. For more information about Organizations best practices, see [Best practices for the management account](#).

#### Note

If you enabled AWS SSO in the management account before November 25th, 2019, you must also enable AWS SSO-integrated applications in the management account and optionally in the member accounts. If you enable them in the management account only, you can enable them in member accounts later. To enable these applications, choose **Enable access** in the AWS SSO **Settings** page in the AWS SSO-integrated applications section. For more information, see [AWS SSO-integrated application enablement](#).

### Member account of an existing organization where AWS SSO is not yet deployed

In this scenario, you should first sign in as the organization administrator and enable AWS SSO in the organization. You can then proceed with creating the Amazon Managed Grafana workspace in a member account in the organization.

If you are not an organization administrator, you must contact an administrator for Organizations and request that they enable AWS SSO. After AWS SSO is enabled, you can then create the workspace in a member account.

If you create a workspace in a member account and you want that workspace to access resources from other AWS accounts in your organization, you must use customer-managed permissions in the workspace. For more information, see [Customer-managed permissions \(p. 396\)](#).

To create a workspace in a member account, you must be signed in to an account that has the **AWSGrafanaAccountAdministrator**, **AWSSSOMemberAccountAdministrator**, and **AWSSSODirectoryAdministrator** policies. For more information, see [Grafana administrator in a member account using AWS SSO \(p. 385\)](#).

## Built-in data sources

The following data sources are supported in every Amazon Managed Grafana workspace.

### Topics

- [Amazon CloudWatch \(p. 41\)](#)
- [Amazon OpenSearch Service \(p. 50\)](#)
- [AWS IoT SiteWise \(p. 56\)](#)
- [Amazon Managed Service for Prometheus \(AMP\) and open-source Prometheus \(p. 57\)](#)
- [Amazon Timestream \(p. 62\)](#)
- [AWS X-Ray \(p. 65\)](#)
- [Azure Monitor \(p. 68\)](#)
- [Graphite \(p. 76\)](#)
- [Google BigQuery \(p. 80\)](#)
- [Google Cloud Monitoring \(p. 82\)](#)
- [Google Sheets \(p. 89\)](#)
- [InfluxDB \(p. 89\)](#)
- [Jaeger \(p. 89\)](#)
- [JSON \(p. 90\)](#)
- [Loki \(p. 94\)](#)
- [Microsoft SQL Server \(p. 98\)](#)
- [MySQL \(p. 109\)](#)
- [OpenSearch \(p. 120\)](#)
- [OpenTSDB \(p. 124\)](#)
- [PostgreSQL \(p. 126\)](#)
- [Redis \(p. 136\)](#)
- [Tempo \(p. 136\)](#)
- [Zipkin \(p. 137\)](#)

## Amazon CloudWatch

With Amazon Managed Grafana, you can add Amazon CloudWatch as a data source by using the AWS data source configuration option in the Grafana workspace console. This feature simplifies adding CloudWatch as a data source by discovering your existing CloudWatch accounts and manages the configuration of the authentication credentials that are required to access CloudWatch. You can use this method to set up authentication and add CloudWatch as a data source, or you can manually set up the data source and the necessary authentication credentials using the same method that you would on a self-managed Grafana server.

### Topics

- [Use AWS data source configuration to add CloudWatch as a data source \(p. 42\)](#)



- [Manually add CloudWatch as a data source \(p. 42\)](#)
- [Using the CloudWatch data source \(p. 44\)](#)

## Use AWS data source configuration to add CloudWatch as a data source

To use AWS data source configuration, first you use the Amazon Managed Grafana console to enable service-managed IAM roles that grant the workspace the IAM policies necessary to read the CloudWatch resources in your account or in your entire organizational units. Then you use the Amazon Managed Grafana workspace console to add CloudWatch as a data source.

### To use AWS data source configuration to add CloudWatch as a data source

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the upper left corner of the page, choose the menu icon and then choose **All workspaces**.
3. Choose the name of the workspace.
4. If you didn't choose to use service-managed permissions for this workspace when you created it, then change from using customer-managed permissions to use service-managed permissions to ensure that the proper IAM roles and policies are enabled for using the AWS data source configuration option in the Grafana workspace console. To do so, choose the edit icon by **IAM role** and then choose **Service managed, Save changes**. For more information, see [Amazon Managed Grafana permissions and policies for AWS data sources and notification channels \(p. 390\)](#).
5. Choose the **Data sources** tab. Then select the check box for **Amazon CloudWatch**, and choose **Actions, Enable service-managed policy**.
6. Choose the **Data sources** tab again, and then choose **Configure in Grafana** in the **Amazon CloudWatch** row.
7. Sign into the Grafana workspace console using AWS SSO if necessary.
8. In the left navigation bar in the Grafana workspace console, choose the AWS icon and then choose **AWS services, CloudWatch**.
9. Select the default Region that you want the CloudWatch data source to query from, and then select the accounts that you want, and then choose **Add data source**.

## Manually add CloudWatch as a data source

### To manually add the CloudWatch data source

1. In the Grafana console side menu, pause on the **Configuration** (gear) icon, then choose **Data Sources**.
2. Choose **Add data source**.
3. Choose the **CloudWatch** data source. If necessary, you can start typing **CloudWatch** in the search box to help you find it.

### CloudWatch settings

Name	Description
Name	The data source name. This is how you see the data source in panels and queries.
Default	Default data source means that it will be pre-selected for new panels.
Default Region	Used in query editor to set region (can be changed on per query basis).

Name	Description
Custom Metrics namespace	Specify the CloudWatch namespace of custom metrics.
Auth Provider	Specify the provider to get credentials.
Assume Role Arn	Specify the Amazon Resource Name (ARN) of the role to assume.
External ID	If you are assuming a role in another account, that has been created with an external ID, specify the external ID here.

## Authentication

To enable authentication between Amazon Managed Grafana and CloudWatch, you can use the Amazon Managed Grafana console to quickly create the policies and permissions that are needed, or you can manually set up authentication using some of the same methods that you would on a self-managed Grafana server.

To use Amazon Managed Grafana data source configuration to quickly set up the policies, follow the steps in [Use AWS data source configuration to add CloudWatch as a data source \(p. 42\)](#).

To set up the permissions manually, use one of the methods in the following section.

## AWS credentials

There are three different authentication methods available. **AWS SDK Default**— uses the permissions defined in the role that is attached to your workspace. For more information, see [Customer-managed permissions \(p. 396\)](#).

**Credentials file**— can't be used in Amazon Managed Grafana.

**Access & secret key**— corresponds to the SDK for Go `StaticProvider` and uses the given access key ID and secret key to authenticate. This method doesn't have any fallbacks, and will fail if the provided key pair doesn't work.

## IAM roles

Currently all access to CloudWatch is done server side by the Grafana backend using the official AWS SDK. Providing you have chosen the *AWS SDK Default* authentication method, and your Grafana server is running on AWS, you can use IAM roles to handle authentication automatically.

For more information, see [IAM roles](#).

## IAM policies

Grafana needs permissions granted via IAM to be able to read CloudWatch metrics and EC2 tags, instances, and regions. You can attach these permissions to IAM roles and use the built-in Grafana support for assuming roles.

The following code example shows a minimal policy.

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

```
"Sid": "AllowReadingMetricsFromCloudWatch",
"Effect": "Allow",
"Action": [
    "cloudwatch:DescribeAlarmsForMetric",
    "cloudwatch:DescribeAlarmHistory",
    "cloudwatch:DescribeAlarms",
    "cloudwatch:ListMetrics",
    "cloudwatch:GetMetricStatistics",
    "cloudwatch:GetMetricData"
],
"Resource": "*"
},
{
    "Sid": "AllowReadingLogsFromCloudWatch",
    "Effect": "Allow",
    "Action": [
        "logs:DescribeLogGroups",
        "logs:GetLogGroupFields",
        "logs:StartQuery",
        "logs:StopQuery",
        "logs:GetQueryResults",
        "logs:GetLogEvents"
    ],
    "Resource": "*"
},
{
    "Sid": "AllowReadingTagsInstancesRegionsFromEC2",
    "Effect": "Allow",
    "Action": ["ec2:DescribeTags", "ec2:DescribeInstances", "ec2:DescribeRegions"],
    "Resource": "*"
},
{
    "Sid": "AllowReadingResourcesForTags",
    "Effect": "Allow",
    "Action": "tag:GetResources",
    "Resource": "*"
}
]
```

### Assuming a role

The `Assume Role ARN` field allows you to specify which IAM role to assume, if any. If you keep this blank, the provided credentials are used directly and the associated role or user should have the required permissions. If this field is not blank, the provided credentials are used to perform an `sts:AssumeRole` call.

## Using the CloudWatch data source

### Using the query editor

The CloudWatch data source can query data from both CloudWatch and CloudWatch Logs APIs, each with its own specialized query editor. You select which API you want to query with by using the query mode switch at the top of the editor.

### Using the metric query editor

To create a valid query, you must specify the namespace, metric name and at least one statistic. If **Match Exact** is turned on, you also must specify all the dimensions of the metric you're querying, so that the metrics schema matches exactly. For more information, see [CloudWatch search expression syntax](#).

If **Match Exact** is turned off, you can specify any number of dimensions by which you want to filter. Up to 100 metrics matching your filter criteria will be returned.

## Dynamic queries using dimension wildcard characters

You can monitor a dynamic list of metrics by using the asterisk (\*) wildcard character for one or more dimension values.

This helps you monitor metrics for AWS resources, such as EC2 instances or containers. For example, when new instances get created as part of an automatic scaling event, they will automatically appear in the graph without you having to track the new instance IDs. This capability is currently limited to retrieving up to 100 metrics. You can choose **Show Query Preview** to see the search expression that is automatically built to support wildcard characters.

By default, the search expression is defined in such a way that the queried metrics must match the defined dimension names exactly. This means that in the example only metrics with exactly one dimension with name "InstanceId" will be returned.

To include metrics that have other dimensions defined, you can turn off **Match Exact**. Turning off **Match Exact** also creates a search expression even if you don't use wildcard characters. Grafana searches for any metric that matches at least the namespace, metric name, and all defined dimensions.

## Multi-value template variables

When defining dimension values based on multi-valued template variables, a search expression is used to query for the matching metrics. This enables the use of multiple template variables in one query and also allows you to use template variables for queries that have the **Match Exact** option disabled.

Search expressions are currently limited to 1024 characters, so your query may fail if you have a long list of values. If you want to query all metrics that have any value for a certain dimension name, we recommend using the asterisk (\*) wildcard character instead of the `All` option.

The use of multi-valued template variables is only supported for dimension values. Using multi-valued template variables for `Region`, `Namespace`, or `Metric Name` is not supported.

## Metric math expressions

You can create new time series metrics by operating on top of CloudWatch metrics using mathematical functions. Arithmetic operators, unary subtraction, and other functions are supported and can be applied to CloudWatch metrics. For more information about CloudWatch metric math functions, see [Using metric math](#).

As an example, to apply arithmetic operations on a metric, give an ID (a unique string) to the raw metric. You can then use this id and apply arithmetic operations to it in the Expression field of the new metric.

Note that if you use the Expression field to reference another query, such as `queryA * 2`, you cannot create an alert rule based on that query.

## Period

A period is the length of time associated with a specific Amazon CloudWatch statistic. Periods are defined in numbers of seconds, and valid values for period are 1, 5, 10, 30, or any multiple of 60.

If the period field is kept blank or set to **auto**, then it calculates automatically based on the time range. The formula used is `time range in seconds / 2000`, and then it snaps to the next higher value in an array of predefined periods `[60, 300, 900, 3600, 21600, 86400]`. By choosing **Show Query Preview** in the query editor, you can see what period Grafana used.

## Deep linking from Grafana panels to the CloudWatch console

Choosing a time series in the panel shows a context menu with a link to **View in CloudWatch console**. Choosing that link will open a new tab that will take you to the CloudWatch console and display all the

metrics for that query. If you're not currently logged in to the CloudWatch console, the link will forward you to the login page. The provided link is valid for any account but will only display the right metrics if you're logged in to the account that corresponds to the selected data source in Grafana.

This feature is not available for metrics that are based on metric math expressions.

### Using the logs query editor

To query CloudWatch Logs, select the region and up to 20 log groups that you want to query. Use the main input area to write your query. For more information, see [CloudWatch Logs Insights query syntax](#).

You can also write queries returning time series data by using the `stats` command in CloudWatch Logs Insights. When making `stats` queries in Explore, you have to make sure you are in Metrics Explore mode.

To the right of the query input field is a CloudWatch Logs Insights link that opens the CloudWatch Logs Insights console with your query. You can continue exploration there if necessary.

### Using template variables

As with several other data sources, the CloudWatch data source supports the use of template variables in queries. For more information, see [Templates and variables \(p. 265\)](#).

### Deep linking from Grafana panels to the CloudWatch console

If you want to view your query in the CloudWatch Logs Insights console, choose the **CloudWatch Logs Insights** button next to the query editor. If you're not currently logged in to the CloudWatch console, the link will forward you to the login page. The provided link is valid for any account but will only display the right metrics if you're logged in to the account that corresponds to the selected data source in Grafana.

### Alerting

Because CloudWatch Logs queries can return numeric data, for example through the use of the `stats` command, alerts are supported. For more information, see [Alerts \(p. 282\)](#).

### Curated dashboards

The updated CloudWatch data source ships with pre-configured dashboards for five of the most popular AWS services:

- Amazon EC2
- Amazon Elastic Block Store
- AWS Lambda
- Amazon CloudWatch Logs
- Amazon Relational Database Service

To import the pre-configured dashboards, go to the configuration page of your CloudWatch data source and choose the **Dashboards** tab. Choose **Import** for the dashboard you want to use. To customize the dashboard, we recommend saving the dashboard under a different name, because otherwise the dashboard will be overwritten when a new version of the dashboard is released.

### Templated queries

Instead of hardcoding things such as server, application and sensor name in your metric queries you can use variables in their place. Variables are shown as dropdown select boxes at the top of the dashboard. You can use these dropdown boxes to change the data being displayed in your dashboard.

See [Templates \(p. 266\)](#) for an introduction to the templating feature and the different types of template variables.

## Query variable

The CloudWatch data source provides the following queries that you can specify in the **Query** field in the Variable edit view. They allow you to fill a variable's options list with things such as `region`, `namespaces`, `metric names`, and `dimension keys/values`.

In place of `region`, you can specify `default` to use the default region configured in the data source for the query.

Name	Description
<code>regions()</code>	Returns a list of all AWS regions.
<code>namespaces()</code>	Returns a list of namespaces CloudWatch support.
<code>metrics(namespace, [region])</code>	Returns a list of metrics in the namespace. (Specify the region or use "default" for custom metrics.)
<code>dimension_\\__keys(namespace)</code>	Returns a list of dimension keys in the namespace.
<code>dimension_\\__values(region, namespace, metric, dimension_\\__key, [filters])</code>	Returns a list of dimension values matching the specified <code>region</code> , <code>namespace</code> , <code>metric</code> , or <code>dimension_key</code> . Or you can use <code>dimension filters</code> to get a more specific result.
<code>ebs_\\__volume_\\__ids(region, instance_\\__id)</code>	Returns a list of volume IDs matching the specified <code>region</code> , <code>instance_id</code> .
<code>ec2_\\__instance_\\__attribute(region, attribute_\\__name, filters)</code>	Returns a list of attributes matching the specified <code>region</code> , <code>attribute_name</code> , <code>filters</code> .
<code>resource_\\__arns(region, resource_\\__type, tags)</code>	Returns a list of ARNs matching the specified <code>region</code> , <code>resource_type</code> , and <code>tags</code> .
<code>statistics()</code>	Returns a list of all the standard statistics

For details about the metrics that CloudWatch provides, see [AWS services that publish CloudWatch metrics](#).

## Examples of templated queries

The following table shows example dimension queries that will return a list of resources for individual AWS services.

Query	Service
<code>dimension_\\__values(us-east-1,AWS/ELB,RequestCount,LoadBalancerName)</code>	Elastic Load Balancing

Query	Service
<code>dimension_.__values(us-east-1,AWS/ElastiCache,CPUUtilization,CacheClusterId)</code>	Amazon ElastiCache
<code>dimension_.__values(us-east-1,AWS/Redshift,CPUUtilization,ClusterIdentifier)</code>	Amazon Redshift
<code>dimension_.__values(us-east-1,AWS/RDS,CPUUtilization,DBInstanceIdentifier)</code>	Amazon RDS
<code>dimension_.__values(us-east-1,AWS/S3,BucketSizeBytes,BucketName)</code>	Amazon Simple Storage Service (Amazon S3)
<code>dimension_.__values(us-east-1,CWAgent,disk_.__used_.__percent,device,{ "InstanceId": "\$instance_.__id" })</code>	CloudWatch Agent
<code>resource_.__arns(eu-west-1,elasticloadbalancing:loadbalancer,{ "elasticbeanstalk:environment-name": [ "myApp-dev", "myApp-prod" ] })</code>	Elastic Load Balancing
<code>resource_.__arns(eu-west-1,ec2:instance,{ "elasticbeanstalk:environment-name": [ "myApp-dev", "myApp-prod" ] })</code>	Amazon EC2

## Using `ec2_instance_attribute` examples

### JSON filters

The `ec2_instance_attribute` query takes filters in JSON format. You can specify pre-defined filters of `ec2:DescribeInstances`. Note that the actual filtering takes place in AWS, not in Grafana.

The following code example shows the filters syntax.

```
{ filter_name1: [ filter_value1 ], filter_name2: [ filter_value2 ] }
```

The following example shows the `ec2_instance_attribute()` query.

```
ec2_instance_attribute(us - east - 1, InstanceId, { 'tag:Environment': ['production'] });
```

### Selecting attributes

Only one attribute per instance can be returned. Any flat attribute can be selected (that is, if the attribute has a single value and isn't an object or array). The following flat attributes are available.

- `AmiLaunchIndex`
- `Architecture`
- `ClientToken`
- `EbsOptimized`
- `EnaSupport`

- Hypervisor
- IamInstanceProfile
- ImageId
- InstanceId
- InstanceLifecycle
- InstanceType
- KernelId
- KeyName
- LaunchTime
- Platform
- PrivateDnsName
- PrivateIpAddress
- PublicDnsName
- PublicIpAddress
- RamdiskId
- RootDeviceName
- RootDeviceType
- SourceDestCheck
- SpotInstanceRequestId
- SrioVNetSupport
- SubnetId
- VirtualizationType
- VpcId

Tags can be selected by prefixing the tag name with `Tags`.

The following example shows the `ec2_instance_attribute()` query.

```
ec2_instance_attribute(us - east - 1, Tags.Name, { 'tag:Team': ['sysops'] });
```

### Using JSON format template variables

Some queries accept filters in JSON format and Grafana supports the conversion of template variables to JSON.

If `env = 'production', 'staging'`, the following query will return ARNs of EC2 instances for which the `Environment` tag is `production` or `staging`.

```
resource_arns(us-east-1, ec2:instance, {"Environment":${env:json}})
```

### Pricing

The Amazon CloudWatch data source for Grafana uses the `ListMetrics` and `GetMetricData` CloudWatch API calls to list and retrieve metrics. Pricing for CloudWatch Logs is based on the amount of



data ingested, archived, and analyzed via CloudWatch Logs Insights queries. For more information, see [Amazon CloudWatch Pricing](#).

Every time you pick a dimension in the query editor, Grafana issues a `ListMetrics` request. Whenever you change the queries in the query editor, one new request to `GetMetricData` will be issued.

API requests to retrieve data samples use the `GetMetricData` operation. This operation provides better support for CloudWatch metric math. It also supports the automatic generation of search expressions when using wildcard characters or turning off the **Match Exact** option. The `GetMetricData` operation incurs charges. For more information, see [Amazon CloudWatch Pricing](#).

### Service quotas

AWS defines quotas, or limits, for resources, operations, and items in your AWS account. Depending on the number of queries in your dashboard and the number of users accessing the dashboard, you may reach the usage limits for various CloudWatch and CloudWatch Logs resources. Note that quotas are defined per account and per AWS Region. If you're using multiple Regions or you have set up more than one CloudWatch data source to query against multiple accounts, you must request a quota increase for each account and each Region in which you reach the limit.

For more information, see [CloudWatch service quotas](#).

## Amazon OpenSearch Service

With Amazon Managed Grafana, you can add the Amazon OpenSearch Service as a data source by using the AWS data source configuration option in the Grafana workspace console. This data source supports Amazon OpenSearch clusters and other OpenSearch clusters.

The AWS data source configuration option simplifies adding the Amazon OpenSearch Service as a data source by discovering your existing OpenSearch accounts and manages the configuration of the authentication credentials that are required to access OpenSearch. You can use this method to set up authentication and add the Amazon OpenSearch Service as a data source, or you can manually set up the data source and the necessary authentication credentials using the same method that you would on a self-managed Grafana server.

The Amazon OpenSearch Service data source supports piped processing language (PPL). For more information about PPL, see [Querying Amazon OpenSearch Service data using Piped Processing Language](#).

You can use the Amazon OpenSearch Service data source to do many types of simple or complex OpenSearch Service queries to visualize logs or metrics stored in OpenSearch Service. You can also annotate your graphs with log events stored in OpenSearch Service.

### Topics

- [Use AWS data source configuration to add Amazon OpenSearch Service as a data source \(p. 50\)](#)
- [Manually add Amazon OpenSearch Service as a data source \(p. 51\)](#)
- [OpenSearch settings \(p. 51\)](#)
- [Using the OpenSearch data source \(p. 53\)](#)

## Use AWS data source configuration to add Amazon OpenSearch Service as a data source

To use AWS data source configuration, first you use the Amazon Managed Grafana console to enable service-managed IAM roles that grant the workspace the IAM policies necessary to read the OpenSearch

Service resources in your account or in your entire organizational units. Then you use the Amazon Managed Grafana workspace console to add OpenSearch Service as a data source.

### To use AWS data source configuration to add Amazon OpenSearch Service as a data source

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the upper left corner of the page, choose the menu icon and then choose **All workspaces**.
3. Choose the name of the workspace.
4. If you didn't choose to use service-managed permissions for this workspace when you created it, then change from using customer-managed permissions to use service-managed permissions to ensure that the proper IAM roles and policies are enabled for using the AWS data source configuration option in the Grafana workspace console. To do so, choose the edit icon by **IAM role** and then choose **Service managed**, **Save changes**. For more information, see [Amazon Managed Grafana permissions and policies for AWS data sources and notification channels \(p. 390\)](#).
5. Choose the **Data sources** tab. Then select the check box for **Amazon OpenSearch Service**, and choose **Actions**, **Enable service-managed policy**.
6. Choose the **Data sources** tab again, and then choose **Configure in Grafana** in the **Amazon OpenSearch Service** row.
7. Sign into the Grafana workspace console using AWS SSO if necessary.
8. In the left navigation bar in the Grafana workspace console, choose the AWS icon and then choose **AWS services**, **Amazon OpenSearch Service**.
9. Select the Region that you want Amazon Managed Grafana to search to discover OpenSearch resources, and then select the accounts and the OpenSearch clusters you want to add, configure the index settings, and then choose **Add data sources**.

## Manually add Amazon OpenSearch Service as a data source

### To manually add the Amazon OpenSearch Service data source

1. In the Grafana console side menu, pause on the **Configuration** (gear) icon, then choose **Data Sources**.
2. Choose **Add data source**.
3. Choose the **Amazon OpenSearch Service** data source. If necessary, you can start typing **OpenSearch** in the search box to help you find it.

#### Note

If you're not seeing the **Data Sources** link in your side menu, it means that your current user does not have the `Admin` role.

## OpenSearch settings

Name	Description
Name	The data source name. This is how you see the data source in panels and queries.
Default	Default data source means that it will be pre-selected for new panels.
Url	The HTTP protocol, IP, and port of your OpenSearch Service server.
Access	Server (default) = URL must be accessible from the Grafana backend/server. Browser = URL must be accessible from the browser.

Access mode controls how requests to the data source will be handled. Server should be the preferred way if nothing else is stated.

#### Server access mode (default)

All requests are made from the browser to Grafana backend or server, which forwards the requests to the data source, circumventing possible Cross-Origin Resource Sharing (CORS) requirements. If you select this access mode, the URL must be accessible from the Grafana backend or server.

#### Browser (direct) access

All requests are made from the browser directly to the data source and may be subject to CORS requirements. If you select this access mode, the URL must be accessible from the browser.

If you select browser access, you must update your OpenSearch Service configuration to allow other domains to access OpenSearch Service from the browser. You do this by specifying these two options in your **OpenSearch.yml** config file.

```
http.cors.enabled: true
http.cors.allow-origin: "*"

```

#### Index settings

Here you can specify a default for the `time` field and specify the name of your OpenSearch Service index. You can use a time pattern for the index name or a wildcard character.

#### OpenSearch Service version

Be sure to specify your OpenSearch version in the version selection dropdown list. This is important because there are differences in how queries are composed. Currently the versions available are 2.x, 5.x, 5.6+, 6.0+, or 7.0+. The value 5.6+ means version 5.6 or higher, but lower than 6.0. The value 6.0+ means version 6.0 or higher, but lower than 7.0. Finally, 7.0+ means version 7.0 or higher, but lower than 8.0.

#### Min time interval

A lower limit for the auto group by time interval. Recommended to be set to write frequency; for example, 1m if your data is written every minute. This option can also be overridden/configured in a dashboard panel under data source options. This value **must** be formatted as a number followed by a valid time identifier; for example, 1m (1 minute) or 30s (30 seconds). The following time identifiers are supported.

Identifier	Description
y	Year
M	Month
w	Week
d	Day
h	Hour
m	Minute
s	Second

Identifier	Description
ms	Millisecond

## Logs

Two parameters, `Message field name` and `Level field name`, can optionally be configured from the data source settings page that determine which fields will be used for log messages and log levels when visualizing logs in [Explore \(p. 254\)](#).

For example, if you use a default setup of Filebeat for shipping logs to OpenSearch Service, the following configuration should work.

- **Message field name:** message
- **Level field name:** fields.level

## Data links

Data links create a link from a specified field that can be accessed in logs view in Explore.

Each data link configuration consists of the following:

- **Field** – Name of the field used by the data link.
- **URL/query** – If the link is external, then enter the full link URL. If the link is internal link, then this input serves as query for the target data source. In both cases, you can interpolate the value from the field with `${__value.raw }` macro.
- **Internal link** – Select this if the link is internal or external. If the link is internal, a data source selector allows you to select the target data source. Only tracing data sources are supported.

## Using the OpenSearch data source

### Metric query editor

The OpenSearch query editor allows you to select multiple metrics and group by multiple terms or filters. Use the plus and minus icons to the right to add/remove metrics or group by clauses. Some metrics and group by clauses have options. Choose the option text to expand the row to view and edit metric or group by options.

### Using Piped Processing Language (PPL)

The OpenSearch data source supports Piped Processing Language (PPL), which enables simpler yet powerful querying and visualization capabilities for OpenSearch. PPL enables customers to explore and find data without having to compose lengthy OpenSearch Domain Specific Language (DSL) statements or write queries using JSON objects. With PPL, you can write queries as a set of commands delimited by pipes similar to UNIX pipes.

Take the following sample DSL query as an example:

```
GET kibana_sample_data_logs/_search{"from":0,"size":0,"timeout":"1m","query":
{"bool":{"should":[{"term":{"response.keyword":{"value":"404","boost":1}}},{ "term":
{"response.keyword":
{"value":"503","boost":1}}}], "adjust_pure_negative":true,"boost":1}}, {"sort":
[{"_doc":{"order":"asc"}], "aggregations":{"composite_buckets":
{"composite":{"size":1000,"sources":[{"host":{"terms":
{"field":"host.keyword","missing_bucket":true,"order":"asc"}}, {"response":{"terms":
{"field":"response.keyword","missing_bucket":true,"order":"asc"}}]}], "aggregations":
```

```
{ "request_count": { "value_count": { "field": "request.keyword" } }, "sales_bucket_sort": { "bucket_sort": { "sort": [ { "request_count": { "order": "desc" } } ], "size": 10 } } } } >
```

The preceding DSL query can be replaced with the following PPL command that is concise and human readable.

```
source = kibana_sample_data_logs | where response='404' or response='503' | stats count(request) as request_count by host, response | sort -request_count
```

For more information about PPL, see [Querying Amazon OpenSearch Service data using Piped Processing Language](#).

### Series naming and alias patterns

You can control the name for time series via the `Alias` input field.

Pattern	Description
{{term fieldname}}	Replaced with value of a term Group By.
{{metric}}	Replaced with metric name (ex. Average, Min, Max).
{{field}}	Replaced with the metric field name.

### Pipeline metrics

Some metric aggregations are called pipeline aggregations; for example, *Moving Average* and *Derivative*. OpenSearch pipeline metrics require another metric to be based on. Use the eye icon next to the metric to hide metrics from appearing in the graph. This is useful for metrics you only have in the query for use in a pipeline metric.

### Templating

Instead of hardcoding things such as server, application, and sensor name in your metric queries you can use variables in their place. Variables are shown as dropdown select boxes at the top of the dashboard. You can use these dropdown boxes to change the data being displayed in your dashboard.

For more information about templating and template variables, see [Templates and variables \(p. 265\)](#).

### Query variable

The OpenSearch data source supports two types of queries you can use in the *Query* field of *Query* variables. The query is written using a custom JSON string.

Query	Description
{ "find": "fields", "type": "keyword" }	Returns a list of field names with the index type keyword.
{ "find": "terms", "field": "@hostname", "size": 1000 }	Returns a list of values for a field using term aggregation. Query will use current dashboard time range as time range for query.
{ "find": "terms", "field": "@hostname", "query": '<.lucene query>' }	Returns a list of values for a field using term aggregation and a specified Lucene query filter. Query will use current dashboard time range as time range for query.

There is a default size limit of 500 on terms queries. To set a custom limit, set the size property in your query. You can use other variables inside the query. The following code example shows the query definition for a variable named `$host`.

```
{ "find": "terms", "field": "@hostname", "query": "@source:$source" }
```

In the previous example, we use another variable named `$source` inside the query definition. Whenever you change, via the dropdown list, the current value of the `$source` variable, it initiates an update of the `$host` variable. After the update, the `$host` variable contains only hostnames filtered by in this case the `@source` document property.

These queries by default return results in term order (which can then be sorted alphabetically or numerically as for any variable). To produce a list of terms sorted by doc count (a top-N values list), add an `orderBy` property of `doc_count`. This automatically selects a descending sort. Using `asc` with `doc_count` (a bottom-N list) can be done by setting `order: "asc"`, but it is discouraged because it increases the error on document counts. To keep terms in the doc count order, set the variable's **Sort** dropdown list to **Disabled**. Alternatively, you might alternatively still want to use **Alphabetical** to re-sort them.

```
{ "find": "terms", "field": "@hostname", "orderBy": "doc_count" }
```

## Using variables in queries

There are two syntaxes:

- `<varname>` Example: `@hostname:$hostname`
- `[ [varname] ]` Example: `@hostname:[hostname]`

Why two ways? The first syntax is easier to read and write, but it does not allow you to use a variable in the middle of a word. When the *Multi-value* or *Include all value* options are enabled, Grafana converts the labels from plaintext to a Lucene-compatible condition.

In the previous example, we have a lucene query that filters documents based on the `@hostname` property using a variable named `$hostname`. It is also using a variable in the *Terms* group by field input box. This allows you to use a variable to quickly change how the data is grouped.

## Annotations

Annotations allow you to overlay rich event information on top of graphs. You add annotation queries via the Dashboard menu or Annotations view. Grafana can query any OpenSearch index for annotation events. For more information, see [Annotations \(p. 238\)](#).

Name	Description
Query	You can keep the search query blank or specify a Lucene query.
Time	The name of the time field; must be date field.
Time End	Optional name of the time end field must be date field. If set, annotations will be marked as a region between time and time-end.
Text	Event description field.
Tags	Optional field name to use for event tags (can be an array or a CSV string).

## Querying logs

Querying and displaying log data from OpenSearch is available in Explore. To display your logs, select the OpenSearch data source, and then optionally enter a Lucene query. For more information, see [Explore](#) (p. 254).

## Log queries

After the result is returned, the log panel shows a list of log rows and a bar chart where the x-axis shows the time and the y-axis shows the frequency or count.

## Filtering log messages

Optionally, enter a Lucene query into the query field to filter the log messages. For example, using a default Filebeat setup, you should be able to use `fields.level:error` to show only error log messages.

# AWS IoT SiteWise

With Amazon Managed Grafana, you can add AWS IoT SiteWise as a data source by using the AWS data source configuration option in the Grafana workspace console. This feature simplifies adding AWS IoT SiteWise as a data source by discovering your existing AWS IoT SiteWise accounts and manages the configuration of the authentication credentials that are required to access AWS IoT SiteWise. You can use this method to set up authentication and add AWS IoT SiteWise as a data source, or you can manually set up the data source and the necessary authentication credentials using the same method that you would on a self-managed Grafana server.

## Topics

- [Use AWS data source configuration to add AWS IoT SiteWise as a data source](#) (p. 56)
- [Manually adding the AWS IoT SiteWise data source](#) (p. 57)
- [AWS IoT SiteWise settings](#) (p. 57)
- [Using the AWS IoT SiteWise data source](#) (p. 57)

## Use AWS data source configuration to add AWS IoT SiteWise as a data source

To use AWS data source configuration, first you use the Amazon Managed Grafana console to enable service-managed IAM roles that grant the workspace the IAM policies necessary to read the AWS IoT SiteWise resources in your account or in your entire organizational units. Then you use the Amazon Managed Grafana workspace console to add AWS IoT SiteWise as a data source.

### To use AWS data source configuration to add AWS IoT SiteWise as a data source

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the upper left corner of the page, choose the menu icon and then choose **All workspaces**.
3. Choose the name of the workspace.
4. If you didn't choose to use service-managed permissions for this workspace when you created it, then change from using customer-managed permissions to use service-managed permissions to ensure that the proper IAM roles and policies are enabled for using the AWS data source configuration option in the Grafana workspace console. To do so, choose the edit icon by **IAM role** and then choose **Service managed**, **Save changes**. For more information, see [Amazon Managed Grafana permissions and policies for AWS data sources and notification channels](#) (p. 390).
5. Choose the **Data sources** tab. Then select the check box for **AWS IoT SiteWise**, and choose **Actions**, **Enable service-managed policy**.
6. Choose the **Data sources** tab again, and then choose **Configure in Grafana** in the **AWS IoT SiteWise** row.

7. Sign into the Grafana workspace console using AWS SSO if necessary.
8. In the left navigation bar in the Grafana workspace console, choose the AWS icon and then choose **AWS services, IoT SiteWise**.
9. Select the default Region that you want the AWS IoT SiteWise data source to query from, select the accounts, and then choose **Add data source**.

## Manually adding the AWS IoT SiteWise data source

### To manually add the AWS IoT SiteWise data source

1. In the Grafana console side menu, pause on the **Configuration** (gear) icon, then choose **Data Sources**.
2. Choose **Add data source**.
3. Choose the **AWS IoT SiteWise** data source. If necessary, you can start typing **SiteWise** in the search box to help you find it.

## AWS IoT SiteWise settings

Name	Description
Name	The data source name. This is how you see the data source in panels and queries.
Auth Provider	Specify the provider to get credentials.
Default Region	Used in query editor to set the region (can be changed on per query basis).
Credentials profile name	Specify the name of the profile to use (if you use <code>~/.aws/credentials</code> file); keep blank for default.
Assume Role Arn	Specify the ARN of the role to assume.
Endpoint (optional)	If you must specify an alternate service endpoint.

## Using the AWS IoT SiteWise data source

For information about how to use the AWS IoT SiteWise data source, see [AWS IoT SiteWise Datasource](#) on Github.

## Amazon Managed Service for Prometheus (AMP) and open-source Prometheus

In Amazon Managed Grafana, the Prometheus data source supports using both self-managed Prometheus servers and AMP workspaces as data sources. For more information about AMP, see [What is Amazon Managed Service for Prometheus?](#)

With Amazon Managed Grafana, you can add an AMP workspace as a data source by using the AWS data source configuration option in the Grafana workspace console. This feature simplifies adding AMP as a data source by discovering your existing AMP accounts and manages the configuration of the authentication credentials that are required to access AMP.

### Topics



- [Use AWS data source configuration to add AMP as a data source \(p. 58\)](#)
- [Manually adding the Prometheus data source \(p. 58\)](#)
- [Using the Prometheus data source \(p. 58\)](#)

## Use AWS data source configuration to add AMP as a data source

To use AWS data source configuration, first you use the Amazon Managed Grafana console to enable service-managed IAM roles that grant the workspace the IAM policies necessary to read the AMP resources in your account or in your entire organizational units. Then you use the Amazon Managed Grafana workspace console to add AMP as a data source.

### To use AWS data source configuration to add AMP as a data source

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the upper left corner of the page, choose the menu icon and then choose **All workspaces**.
3. Choose the name of the workspace.
4. If you didn't choose to use service-managed permissions for this workspace when you created it, then change from using customer-managed permissions to use service-managed permissions to ensure that the proper IAM roles and policies are enabled for using the AWS data source configuration option in the Grafana workspace console. To do so, choose the edit icon by **IAM role** and then choose **Service managed, Save changes**. For more information, see [Amazon Managed Grafana permissions and policies for AWS data sources and notification channels \(p. 390\)](#).
5. Choose the **Data sources** tab. Then select the check box for **Amazon Managed Service for Prometheus**, and choose **Actions, Enable service-managed policy**.
6. Choose the **Data sources** tab again, and then choose **Configure in Grafana** in the **Amazon Managed Service for Prometheus** row.
7. Sign into the Grafana workspace console using AWS SSO if necessary.
8. In the left navigation bar in the Grafana workspace console, choose the AWS icon and then choose **AWS services, Prometheus**.
9. Select the Region that you want Amazon Managed Grafana to search to discover AMP workspaces, and then select the accounts and AMP workspaces that you want to add, and then choose **Add data source**.

## Manually adding the Prometheus data source

### To manually add the Prometheus data source

1. In the Grafana console side menu, pause on the **Configuration** (gear) icon, then choose **Data Sources**.
2. Choose **Add data source**.
3. Choose the **Prometheus** data source. If necessary, you can start typing **Prometheus** in the search box to help you find it.

## Using the Prometheus data source

### Prometheus settings

Name	Description
Name	The data source name. This is how you see the data source in panels and queries.

Name	Description
Default	Default data source means that it will be pre-selected for new panels.
Url	The URL of your Prometheus server; for example, <code>https://prometheus.example.org:9090</code> .
Access	Server (default) = URL must be accessible from the Grafana backend/server. Browser = URL must be accessible from the browser.
Basic Auth	Enable basic authentication to the Prometheus data source.
User	User name for basic authentication.
Password	Password for basic authentication.
Scrape interval	Set this to the typical scrape and evaluation interval configured in Prometheus. Defaults to 15s.
Disable metrics lookup	Checking this option will disable the metrics chooser and metric/label support in the query field's autocomplete. This helps if you have performance issues with bigger Prometheus instances.
Custom Query Parameters	Add custom parameters to the Prometheus query URL. For example <code>timeout</code> , <code>partial_response</code> , <code>dedup</code> , or <code>max_source_resolution</code> . Multiple parameters should be concatenated together with an "&".

## Prometheus query editor

The following sections provide information and options for Prometheus query editor in the dashboard and in Explore.

### Query editor in dashboards

Open a graph in edit mode by choosing the title and then choosing **Edit** (or by pressing **e** key while pausing on the panel).

Name	Description
Query expression	For more information about Prometheus query expressions, see the <a href="#">Prometheus documentation</a> .
Legend format	Controls the name of the time series, using name or pattern. For example <code>{ {hostname} }</code> is replaced with the label value for the label <code>hostname</code> .
Min step	An additional lower limit for the <a href="#">step parameter of Prometheus range queries</a> and for the <code>step__interval</code> and <code>step__rate_interval</code> variables. The limit is absolute and not modified by the <i>Resolution</i> setting.
Resolution	Controls both the <code>step__interval</code> variable and the <a href="#">step parameter of Prometheus range queries</a> such that each pixel corresponds to one data point. For better performance, use lower resolutions. <code>1/2</code> only retrieves a data point for every other pixel, and <code>1/10</code> retrieves one data point per 10 pixels. Note that both <i>Min time interval</i> and <i>Min step</i> limit the final value of <code>step__interval</code> and <code>step</code> .
Metrics lookup	Search for metric names in this input field.

Name	Description
Format	Switch between Table, Time series, or Heatmap. Table works only in the table panel. Heatmap is suitable for displaying metrics of the histogram type on a heatmap panel. It converts cumulative histograms to regular ones and sorts series by the bucket bound.
Instant	Perform an "instant" query, to return only the latest value that Prometheus has scraped for the requested time series. Instant queries return results much faster than normal range queries. Use them to look up label sets.
Min interval	This value multiplied by the denominator from the <i>Resolution</i> setting sets a lower limit to both the <i>time_interval</i> variable and the <a href="#">step parameter of Prometheus range queries</a> . Defaults to <i>Scrape interval</i> as set in the data source options.

### Note

Amazon Managed Grafana modifies the request dates for queries to align them with the dynamically calculated step. This ensures consistent display of metrics data, but it can result in a small gap of data at the right edge of a graph.

## Instant queries in dashboards

The Prometheus data source allows you to run instant queries, which query only the latest value. You can visualize the results in a table panel to see all available labels of a time series.

Instant query results are made up of only one data point per series. They can be shown in the graph panel with the help of series overrides. To show them in the graph as a latest value point, add a series override and select `Points > true`. To show a horizontal line across the whole graph, add a series override and select `Transform > constant` For more information about series overrides, see [Series overrides](#) (p. 216).

## Query editor in Explore

Name	Description
Query expression	For more information about Prometheus query expression, see the <a href="#">Prometheus documentation</a> .
Step	<a href="#">Step parameter of Prometheus range queries</a> . Time units can be used here, for example: 5s, 1m, 3h, 1d, 1y. Default unit if no unit specified is s (seconds).
Query type	Range, Instant, or Both. When running <b>Range query</b> , the result of the query is displayed in graph and table. Instant query returns only the latest value that Prometheus has scraped for the requested time series and it is displayed in the table. When <b>Both</b> is selected, both instant query and range query is run. Result of range query is displayed in graph and the result of instant query is displayed in the table.

## Metrics browser

The metrics browser allows you to quickly find metrics and select relevant labels to build basic queries. When you open the browser you will see all available metrics and labels. If supported by your Prometheus instance, each metric will show its HELP and TYPE as a tooltip.

When you select a metric, the browser narrows down the available labels to show only the ones applicable to the metric. You can then select one or more labels for which the available label values are shown in lists in the bottom section. Select one or more values for each label to tighten your query scope.

### Note

If you do not remember a metric name to start with, you can also select a few labels first, to narrow down the list and then find relevant label values.

All lists in the metrics browser have a search field above them to quickly filter for metrics or labels that match a certain string. The values section only has one search field. Its filtering applies to all labels to help you find values across labels once they have been selected, for example, among your labels app, job, job\_name only one might with the value you are looking for.

Once you are satisfied with your query, click “Use query” to run the query. The **Use as rate query** button adds a `rate(...)[$__interval]` around your query to help write queries for counter metrics. The “Validate selector” button will check with Prometheus how many time series are available for that selector.

### Limitations

The metrics browser has a hard limit of 10,000 labels (keys) and 50,000 label values (including metric names). If your Prometheus instance returns more results, the browser will continue functioning. However, the result sets will be cut off above those maximum limits.

### Templating

Instead of hardcoding things such as server, application and sensor name in your metric queries, you can use variables in their place. Variables are shown as dropdown select boxes at the top of the dashboard. You can use these dropdown boxes to change the data being displayed in your dashboard.

For more information about templating and template variables, see [Templates and variables \(p. 265\)](#).

### Query variable

Variable of the type *Query* allows you to query Prometheus for a list of metrics, labels, or label values. The Prometheus data source plugin provides the following functions you can use in the **Query** input field.

Name	Description
<code>label_names()</code>	Returns a list of label names.
<code>label_values(label)</code>	Returns a list of label values for the <code>label</code> in every metric.
<code>label_values(metric, label)</code>	Returns a list of label values for the <code>label</code> in the specified metric.
<code>metrics(metric)</code>	Returns a list of metrics matching the specified <code>metric</code> regex.
<code>query_result(query)</code>	Returns a list of Prometheus query result for the <code>query</code> .

For information about what *metric names*, *label names* and *label values* are, see the [Prometheus documentation](#).

### Using interval and range variables

#### Note

Support for `$__range`, `$__range_s`, and `$__range_ms` are available only from Grafana v5.3.

You can use some global built-in variables in query variables: `$__interval`, `$__interval_ms`, `$__range`, `$__range_s`, and `$__range_ms`. For more information, see [Global variables \(p. 274\)](#). These can be convenient to use with the `query_result` function when you must filter variable queries because the `label_values` function doesn't support queries.

To get the correct instances when changing the time range on the dashboard, make sure to set the variable's `refresh` trigger to be `On Time Range Change`.

The following code example shows how to populate a variable with the busiest five request instances based on average QPS over the time range shown in the dashboard.

```
Query: query_result(topk(5, sum(rate(http_requests_total[$__range])) by (instance)))
Regex: /"([^\"]+)"/
```

The following code example shows how to populate a variable with the instances having a certain state over the time range shown in the dashboard, using `$__range_s`.

```
Query: query_result(max_over_time(<metric>[${__range_s}s]) != <state>)
Regex:
```

### Using `$__rate_interval` variable

The `$__rate_interval` variable is meant to be used in the `rate` function. It is defined as `max($__interval + Scrape interval, 4 * Scrape interval)`. *Scrape interval* is the `Min step` setting (AKA `query_interval`, a setting per PromQL query), if any is set, and otherwise the *Scrape interval* as set in the Prometheus data source (but ignoring any `Min interval` setting in the panel, because the latter is modified by the resolution setting).

### Using variables in queries

There are two syntaxes:

- `$<varname>` Example: `rate(http_requests_total{job=~"$job"}[5m])`
- `[ [varname] ]` Example: `rate(http_requests_total{job=~"[[job]]"}[5m])`

Why two ways? The first syntax is easier to read and write but does not allow you to use a variable in the middle of a word. When the *Multi-value* or *Include all value* options are enabled, Grafana converts the labels from plaintext to a regex compatible string. Which means you have to use `=~` instead of `=`.

### Annotations

You can use annotations to overlay rich event information on top of graphs. You add annotation queries via the Dashboard menu or Annotations view. For more information, see [Annotations \(p. 238\)](#).

Prometheus supports two ways to query annotations.

- A regular metric query
- A Prometheus query for pending and firing alerts. For more information, see [Inspecting alerts during runtime](#).

The `step` option is useful to limit the number of events returned from your query.

## Amazon Timestream

With Amazon Managed Grafana, you can add Amazon Timestream as a data source by using the AWS data source configuration option in the Grafana workspace console. This feature simplifies adding Timestream as a data source by discovering your existing Timestream accounts and manages the

configuration of the authentication credentials that are required to access Timestream. You can use this method to set up authentication and add Timestream as a data source, or you can manually set up the data source and the necessary authentication credentials using the same method that you would on a self-managed Grafana server.

## Use AWS data source configuration to add Timestream as a data source

To use AWS data source configuration, first you use the Amazon Managed Grafana console to enable service-managed IAM roles that grant the workspace the IAM policies necessary to read the Timestream resources in your account or in your entire organizational units. Then you use the Amazon Managed Grafana workspace console to add Timestream as a data source.

### To use AWS data source configuration to add Timestream as a data source

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the upper left corner of the page, choose the menu icon and then choose **All workspaces**.
3. Choose the name of the workspace.
4. If you didn't choose to use service-managed permissions for this workspace when you created it, then change from using customer-managed permissions to use service-managed permissions to ensure that the proper IAM roles and policies are enabled for using the AWS data source configuration option in the Grafana workspace console. To do so, choose the edit icon by **IAM role** and then choose **Service managed**, **Save changes**. For more information, see [Amazon Managed Grafana permissions and policies for AWS data sources and notification channels \(p. 390\)](#).
5. Choose the **Data sources** tab. Then select the check box for **Amazon Timestream**, and choose **Actions, Enable service-managed policy**.
6. Choose the **Data sources** tab again, and then choose **Configure in Grafana** in the **Amazon Timestream** row.
7. Sign into the Grafana workspace console using AWS SSO if necessary.
8. In the left navigation bar in the Grafana workspace console, choose the AWS icon and then choose **AWS services, Timestream**.
9. Select the default Region that you want the AWS IoT SiteWise data source to query from, select the accounts, and then choose **Add data source**.

## Manually adding the Timestream data source

### To manually add the Timestream data source

1. In the Grafana console side menu, pause on the **Configuration** (gear) icon, then choose **Data Sources**.
2. Choose **Add data source**.
3. Choose the **Amazon Timestream** data source. If necessary, you can start typing **Timestream** in the search box to help you find it.

## Timestream settings

Name	Description
Name	The data source name. This is how you see the data source in panels and queries.
Auth Provider	Specify the provider to get credentials.

Name	Description
Default Region	Used in query editor to set region (can be changed on per query basis).
Credentials profile name	Specify the name of the profile to use (if you use <code>~/.aws/credentials</code> file), keep blank for default.
Assume Role Arn	Specify the ARN of the role to assume.
Endpoint (optional)	If you must specify an alternate service endpoint.

## Authentication

This section covers the different types of authentication that you can use for the Amazon Timestream data source.

### Example AWS credentials

You can't use the credentials file method of authentication in Amazon Managed Grafana.

## Using the Timestream data source

### Query editor

The query editor accepts Timestream syntax in addition to the macros listed previously and any dashboard template variables.

Press **Ctrl+Space** to open the IntelliSense suggestions.

### Macros

To simplify syntax and to allow for dynamic parts, such as date range filters, the query can contain macros.

Macro example	Description
<code>\$_database</code>	Will specify the selected database. This may use the default from the data source configuration, or the explicit value from the query editor.
<code>\$_table</code>	Will specify the selected database. This may use the default from the datasource config, or the explicit value from the query editor.
<code>\$_measure</code>	Will specify the selected measure. This may use the default from the datasource config, or the explicit value from the query editor.
<code>\$_timeFilter</code>	Will be replaced by an expression that limits the time to the dashboard range
<code>\$_interval_ms</code>	Will be replaced by a number that represents the amount of time a single pixel in the graph should cover.

## AWS X-Ray

Add AWS X-Ray as a data source, and then build dashboards or use Explore with X-Ray to look at traces, analytics, or insights.

With Amazon Managed Grafana, you can add X-Ray as a data source by using the AWS data source configuration option in the Grafana workspace console. This feature simplifies adding X-Ray as a data source by discovering your existing X-Ray accounts and manages the configuration of the authentication credentials that are required to access X-Ray. You can use this method to set up authentication and add X-Ray as a data source, or you can manually set up the data source and the necessary authentication credentials using the same method that you would on a self-managed Grafana server.

### Topics

- [Use AWS data source configuration to add X-Ray as a data source \(p. 65\)](#)
- [Manually adding the X-Ray data source \(p. 65\)](#)
- [X-Ray settings \(p. 66\)](#)
- [Using the X-Ray data source \(p. 67\)](#)

## Use AWS data source configuration to add X-Ray as a data source

To use AWS data source configuration, first you use the Amazon Managed Grafana console to enable service-managed IAM roles that grant the workspace the IAM policies necessary to read the X-Ray resources in your account or in your entire organizational units. Then you use the Amazon Managed Grafana workspace console to add X-Ray as a data source.

### To use AWS data source configuration to add X-Ray as a data source

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the upper left corner of the page, choose the menu icon and then choose **All workspaces**.
3. Choose the name of the workspace.
4. If you didn't choose to use service-managed permissions for this workspace when you created it, then change from using customer-managed permissions to use service-managed permissions to ensure that the proper IAM roles and policies are enabled for using the AWS data source configuration option in the Grafana workspace console. To do so, choose the edit icon by **IAM role** and then choose **Service managed**, **Save changes**. For more information, see [Amazon Managed Grafana permissions and policies for AWS data sources and notification channels \(p. 390\)](#).
5. Choose the **Data sources** tab. Then select the check box for **AWS X-Ray**, and choose **Actions**, **Enable service-managed policy**.
6. Choose the **Data sources** tab again, and then choose **Configure in Grafana** in the **AWS X-Ray** row.
7. Sign into the Grafana workspace console using AWS SSO if necessary.
8. In the left navigation bar in the Grafana workspace console, choose the AWS icon and then choose **AWS services, X-Ray**.
9. Select the default Region that you want the X-Ray data source to query from, select the accounts, and then choose **Add data source**.

## Manually adding the X-Ray data source

### To manually add the X-Ray data source

1. In the Grafana console side menu, pause on the **Configuration** (gear) icon, then choose **Data Sources**.
2. Choose **Add data source**.



- Choose the **X-Ray** data source. If necessary, you can start typing **X-Ray** in the search box to help you find it.

## X-Ray settings

Name	Description
Name	The data source name. This is how you see the data source in panels and queries.
Default	Default data source means that it will be pre-selected for new panels.
Default Region	Used in query editor to set region (can be changed on per query basis).
Auth Provider	Specify the provider to get credentials.
Credentials profile name	Specify the name of the profile to use (if you use <code>~/.aws/credentials</code> file), keep blank for default.
Assume Role Arn	Specify the ARN of the role to assume.
External ID	If you are assuming a role in another account, that has been created with an external ID, specify the external ID here.

## Authentication

This section covers the different types of authentication that you can use for X-Ray data source.

### IAM roles

Currently, all access to X-Ray is done server side by the Grafana workspace backend using the official AWS SDK. If your Grafana server is running on AWS, you can use IAM roles and authentication will be handled automatically.

For more information, see [IAM roles](#).

### IAM policies

Grafana needs permissions granted via IAM to be able to read X-Ray data and EC2 tags/instances/regions. You can attach these permissions to IAM roles and use the built-in Grafana support for assuming roles.

The following code example shows a minimal policy.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "xray:BatchGetTraces",
        "xray:GetTraceSummaries",
        "xray:GetTraceGraph",
        "xray:GetGroups",
        "xray:GetTimeSeriesServiceStatistics",
        "xray:GetInsightSummaries",
        "xray:GetInsight",

```

```
        "ec2:DescribeRegions"
      ],
      "Resource": "*"
    }
  ]
}
```

### Example AWS credentials

You can't use the credentials file method in Amazon Managed Grafana.

## Using the X-Ray data source

### Query editor

The most important field in the editor is the query type. There are four query types:

- Trace List (Traces in AWS)
- Trace Statistics
- Trace Analytics (Analytics in AWS)
- Insights

### Trace List

The Trace List type allows you to search for traces, which are shown in a table. Choosing the trace ID in the first column opens the trace on the right side. Notice the query field in the editor. You can write queries, filter expressions, or you can insert a single trace ID that will be shown in a trace view. For more details about filter expressions, see the [AWS X-Ray documentation](#).

#### Note

The trace list will show only the first 1000 traces.

### Trace Statistics

In Trace Statistics, you can see a graph and a table showing information about error, fault, throttle, success, and total count. You can use the columns field in the query editor to see only specified columns.

### Trace Analytics

In Trace Analytics, you can visualize the following tables.

- Root Cause
  - Response Time
    - Root Cause Service (Last service in path)
    - Path (multiple paths)
  - Error
    - Root Cause Service (Last service in path)
    - Path
    - Error Message
  - Fault
    - Root Cause Service (Last service in path)
    - Path
    - Error Message
- End-user Impact

- URL
- HTTP Status Code

## Insights

In Insights, you can see the summary table for Insights. Choosing the InsightId will take you to AWS Management console.

## Alerting

Because X-Ray queries can return numeric data, alerts are supported. For more information, see [Alerts \(p. 282\)](#).

# Azure Monitor

The Azure Monitor data source supports multiple services in the Azure cloud:

- **Azure Monitor service** is the platform service that provides a single source for monitoring Azure resources. For more information, see [Querying the Azure Monitor service \(p. 69\)](#).
- **Application Insights server** is an extensible Application Performance Management (APM) service for web developers on multiple platforms and can be used to monitor your live web application - it will automatically detect performance anomalies. For more information, see [Querying the Application Insights Analytics service \(p. 75\)](#).
- **Azure Log Analytics** (or Azure Logs) gives you access to log data collected by Azure Monitor. For more information, see [Querying the Azure Log Analytics service \(p. 72\)](#).
- Use the **Application Insights Analytics service** to query [Application Insights data](#) using the same query language used for Azure Log Analytics. For more information, see [Querying the Application Insights Analytics service \(p. 75\)](#).

## Adding the data source

The data source can access metrics from four different services. You can configure access to the services that you use. It is also possible to use the same credentials for multiple services if that is how you have set it up in Azure AD.

- [Guide to setting up an Azure Active Directory Application for Azure Monitor](#)
- [Guide to setting up an Azure Active Directory Application for Azure Log Analytics](#).
- [Quickstart Guide for Application Insights](#).

1. Accessed from the Grafana main menu, newly installed data sources can be added immediately within the Data Sources section. Next, choose the **Add data source** button in the upper right. The Azure Monitor data source will be available for selection in the Cloud section in the list of data sources.
2. In the name field, Grafana will automatically fill in a name for the data source: `Azure Monitor` or something such as `Azure Monitor - 3`. If you are configuring multiple data sources, change the name to something more informative.
3. If you are using Azure Monitor, you need four pieces of information from the Azure portal (for detailed instructions, see the link provided earlier):
  - **Tenant Id** (Azure Active Directory, Properties, Directory ID)
  - **Client Id** (Azure Active Directory, App Registrations, Choose your app, Application ID)
  - **Client Secret** (Azure Active Directory, App Registrations, Choose your app, Keys)
  - **Default Subscription Id** (Subscriptions, Choose subscription, Overview, Subscription ID)

4. Paste these four items into the fields in the Azure Monitor API Details section.
  - The Subscription Id can be changed per query. Save the data source and refresh the page to see the list of subscriptions available for the specified Client Id.
5. If you are also using the Azure Log Analytics service, you must specify these two configuration values or reuse the Client Id and Secret from the previous step.
  - Client Id (Azure Active Directory, App Registrations, Choose your app, Application ID)
  - Client Secret (Azure Active Directory, App Registrations, Choose your app, Keys, Create a key, Use client secret)
6. If you are using Application Insights, you need two pieces of information from the Azure Portal (for detailed instructions, see the link provided earlier):
  - Application ID
  - API Key
7. Paste these two items into the appropriate fields in the Application Insights API Details section.
8. Test that the configuration details are correct by choosing the **Save & Test** button.

Alternatively on step 4, if you are creating a new Azure Active Directory App, use the [Azure CLI](#):

```
az ad sp create-for-rbac -n "http://localhost:3000"
```

## Choosing a service

In the query editor for a panel, after you choose your Azure Monitor data source, the first step is to select a service. There are four options:

- Azure Monitor
- Application Insights
- Azure Log Analytics
- Insights Analytics

The query editor changes depending on which option you select. Azure Monitor is the default.

## Querying the Azure Monitor service

The Azure Monitor service provides metrics for all the Azure services that you have running. It helps you understand how your applications on Azure are performing, and it proactively finds issues affecting your applications.

If your Azure Monitor credentials give you access to multiple subscriptions, choose the appropriate subscription first.

Examples of metrics that you can get from the service are:

- `Microsoft.Compute/virtualMachines` - Percentage CPU
- `Microsoft.Network/networkInterfaces` - Bytes sent
- `Microsoft.Storage/storageAccounts` - Used Capacity

The query editor allows you to query multiple dimensions for metrics that support them. Metrics that support multiple dimensions are those listed in the [Azure Monitor supported Metrics List](#) that have one or more values listed in the **Dimension** column for the metric.

## Formatting legend keys with aliases for Azure Monitor

The default legend formatting for the Azure Monitor API is:

```
metricName{dimensionName=dimensionValue,dimensionTwoName=DimensionTwoValue}
```

These can be long, but you can change this formatting by using aliases. In the **Legend Format** field, you can combine the following aliases any way that you want.

Azure Monitor examples:

- Blob Type: {{ blobtype }}
- {{ resourcegroup }} - {{ resourcename }}

## Alias patterns for Azure Monitor

- {{ resourcegroup }} = replaced with the value of the Resource Group
- {{ namespace }} = replaced with the value of the Namespace (for example, Microsoft.Compute/virtualMachines)
- {{ resourcename }} = replaced with the value of the Resource Name
- {{ metric }} = replaced with metric name (for example, Percentage CPU)
- {{ dimensionname }} = *Legacy as of 7.1+ (for backwards compatibility)* replaced with the first dimension's key/label (as sorted by the key/label) (for example, blobtype)
- {{ dimensionvalue }} = *Legacy as of 7.1+ (for backwards compatibility)* replaced with first dimension's value (as sorted by the key/label) (for example, BlockBlob)
- {{ arbitraryDim }} = *Available in 7.1+* replaced with the value of the corresponding dimension. (for example, {{ blobtype }} becomes BlockBlob)

## Creating template variables for Azure Monitor

Instead of hardcoding things such as server, application and sensor name in your metric queries you can use variables in their place. Variables are shown as dropdown select boxes at the top of the dashboard. You can use these dropdown boxes to change the data being displayed in your dashboard.

Note that the Azure Monitor service does not support multiple values yet. To visualize multiple time series (for example, metrics for server1 and server2), add multiple queries so that you can view them on the same graph or in the same table.

The Azure Monitor data source plugin provides the following queries that you can specify in the **Query** field in the Variable edit view. You can use them to fill a variable's options list.

Name	Description
Subscriptions()	Returns a list of subscriptions.
ResourceGroups()	Returns a list of resource groups.
ResourceGroups(12345678-aaaa-bbbb-cccc-123456789aaa)	Returns a list of resource groups for a specified subscription.
Namespaces(aResourceGroup)	Returns a list of namespaces for the specified resource group.
Namespaces(12345678-aaaa-bbbb-cccc-123456789aaa, aResourceGroup)	Returns a list of namespaces for the specified resource group and subscription.

Name	Description
<code>ResourceNames(aResourceGroup, aNamespace)</code>	Returns a list of resource names.
<code>ResourceNames(12345678-aaaa-bbbb-cccc-123456789aaa, aResourceGroup, aNamespace)</code>	Returns a list of resource names for a specified subscription.
<code>MetricNamespace(aResourceGroup, aNamespace, aResourceName)</code>	Returns a list of metric namespaces.
<code>MetricNamespace(12345678-aaaa-bbbb-cccc-123456789aaa, aResourceGroup, aNamespace, aResourceName)</code>	Returns a list of metric namespaces for a specified subscription.
<code>MetricNames(aResourceGroup, aNamespace, aResourceName)</code>	Returns a list of metric names.
<code>MetricNames(12345678-aaaa-bbbb-cccc-123456789aaa, aResourceGroup, aNamespace, aResourceName)</code>	Returns a list of metric names for a specified subscription.

Examples:

- Resource Groups query: `ResourceGroups()`
- Passing in metric name variable: `Namespaces(cosmo)`
- Chaining template variables: `ResourceNames($rg, $ns)`
- Do not quote parameters: `MetricNames(hg, Microsoft.Network/publicIPAddresses, grafanaIP)`

For more information about templating and template variables, see [Templates \(p. 266\)](#).

### List of supported Azure Monitor metrics

Not all metrics returned by the Azure Monitor API have values. To make building a query easier, the Grafana data source has a list of supported Azure Monitor metrics, and it ignores metrics that will never have values. This list is updated regularly as new services and metrics are added to the Azure cloud. For more information about the list of Azure Monitor metrics, see [current supported namespaces](#).

### Azure Monitor alerting

Grafana alerting is supported for the Azure Monitor service. This is not Azure Alerts support. For more information about Grafana alerting, see [Alerts \(p. 282\)](#).

## Querying the Application Insights service

### Formatting legend keys with aliases for Application Insights

The default legend formatting is:

```
metricName{dimensionName=dimensionValue,dimensionTwoName=DimensionTwoValue}
```

In the Legend Format field, the following aliases can be combined any way you want.

Application Insights examples:

- `city: {{ client/city }}`

- `{{ metric }}` [Location: `{{ client/countryOrRegion }}`, `{{ client/city }}`]

### Alias patterns for Application Insights

- `{{ groupbyvalue }}` = *Legacy as of Grafana 7.1+ (for backwards compatibility)* replaced with the first dimension's key/label (as sorted by the key/label)
- `{{ groupbyname }}` = *Legacy as of Grafana 7.1+ (for backwards compatibility)* replaced with first dimension's value (as sorted by the key/label) (for example, BlockBlob)
- `{{ metric }}` = replaced with metric name (for example, requests/count)
- `{{ arbitraryDim }}` = *Available in 7.1+* replaced with the value of the corresponding dimension. (for example, `{{ client/city }}` becomes Chicago)

### Filter expressions for Application Insights

The filter field takes an OData filter expression.

Examples:

- `client/city eq 'Boydton'`
- `client/city ne 'Boydton'`
- `client/city ne 'Boydton' and client/city ne 'Dublin'`
- `client/city eq 'Boydton' or client/city eq 'Dublin'`

### Templating with variables for Application Insights

Use the one of the following queries in the **Query** field in the Variable edit view.

For more information about templating and template variables, see [Templates \(p. 266\)](#).

Name	Description
<code>AppInsightsMetricNames()</code>	Returns a list of metric names.
<code>AppInsightsGroupBys(aMetricName)</code>	Returns a list of group by clauses for the specified metric name.

Examples:

- Metric Names query: `AppInsightsMetricNames()`
- Passing in metric name variable: `AppInsightsGroupBys(requests/count)`
- Chaining template variables: `AppInsightsGroupBys($metricnames)`

### Application Insights alerting

Grafana alerting is supported for Application Insights. This is not Azure Alerts support. For more information about Grafana alerting, see [Alerts \(p. 282\)](#).

### Querying the Azure Log Analytics service

Queries are written in the new [Azure Log Analytics \(or KustoDB\) Query Language](#). A Log Analytics query can be formatted as time series data or as table data.

If your credentials give you access to multiple subscriptions, then choose the appropriate subscription before entering queries.

### Time series queries

Time series queries are for the graph panel and other panels such as the SingleStat panel. Each query must contain at least a datetime column and a numeric value column. The result must be sorted in ascending order by the datetime column.

The following code example shows a query that returns the aggregated count grouped by hour.

```
Perf
| where $__timeFilter(TimeGenerated)
| summarize count() by bin(TimeGenerated, 1h)
| order by TimeGenerated asc
```

A query can also have one or more non-numeric/non-datetime columns, and those columns are considered dimensions and become labels in the response. For example, a query that returns the aggregated count grouped by hour, Computer, and the CounterName.

```
Perf
| where $__timeFilter(TimeGenerated)
| summarize count() by bin(TimeGenerated, 1h), Computer, CounterName
| order by TimeGenerated asc
```

You can also select additional number value columns (with, or without multiple dimensions). For example, getting a count and average value by hour, Computer, CounterName, and InstanceName:

```
Perf
| where $__timeFilter(TimeGenerated)
| summarize Samples=count(), ["Avg Value"]=avg(CounterValue)
  by bin(TimeGenerated, $__interval), Computer, CounterName, InstanceName
| order by TimeGenerated asc
```

#### Note

**Tip:** In the previous query, the Kusto syntax and `Samples=count()["Avg Value"]=...` are used to rename those columns — the second syntax allowing for the space. This changes the name of the metric that Grafana uses. As a result, things such as series legends and table columns will match what you specify. In this example, `Samples` is displayed instead of `_count`.

### Table queries

Table queries are mainly used in the table panel, and they show a list of columns and rows. This example query returns rows with the six specified columns.

```
AzureActivity
| where $__timeFilter()
| project TimeGenerated, ResourceGroup, Category, OperationName, ActivityStatus, Caller
| order by TimeGenerated desc
```

### Formatting the display name for Log Analytics

The default display name format is:



```
metricName{dimensionName=dimensionValue,dimensionTwoName=DimensionTwoValue}
```

This can be customized by using the display name field option.

## Azure Log Analytics macros

To make writing queries easier, Grafana provides several macros that you can use in the where clause of a query:

- `$__timeFilter()` – Expands to `TimeGenerated ≥ datetime(2018-06-05T18:09:58.907Z)` and `TimeGenerated ≤ datetime(2018-06-05T20:09:58.907Z)` where the from and to datetimes are from the Grafana time picker.
- `$__timeFilter(datetimeColumn)` – Expands to `datetimeColumn ≥ datetime(2018-06-05T18:09:58.907Z)` and `datetimeColumn ≤ datetime(2018-06-05T20:09:58.907Z)` where the from and to datetimes are from the Grafana time picker.
- `$__timeFrom()` – Returns the From datetime from the Grafana picker. Example: `datetime(2018-06-05T18:09:58.907Z)`.
- `$__timeTo()` – Returns the From datetime from the Grafana picker. Example: `datetime(2018-06-05T20:09:58.907Z)`.
- `$__escapeMulti($myVar)` – is to be used with multi-value template variables that contain illegal characters. If `$myVar` has the following two values as a string `'\\grafana-vm\\Network(eth0)\\Total','\\hello!'`, then it expands to: `@'\\grafana-vm\\Network(eth0)\\Total', '@'\\hello!'`. If using single value variables there is no need for this macro, escape the variable inline instead: `@'$myVar'`.
- `$__contains(colName, $myVar)` – is to be used with multi-value template variables. If `$myVar` has the value `'value1','value2'`, it expands to: `colName in ('value1','value2')`.

If using the **All** option, check the **Include All Option** check box and in the **Custom all value** field, enter the following value: **a11**. If `$myVar` has the value **a11**, the macro will instead expand to `1 == 1`. For template variables with numerous options, this increases the query performance by not building a large "where..in" clause.

## Azure Log Analytics built-in variables

There are also some Grafana variables that can be used in Azure Log Analytics queries:

- `$__interval` - Grafana calculates the minimum time grain that can be used to group by time in queries. It returns a time grain such as `5m` or `1h` that can be used in the bin function; for example, `summarize count() by bin(TimeGenerated, $__interval)`. For more information about interval variables, see [Adding an interval variable \(p. 271\)](#).

## Templating with variables for Azure Log Analytics

Any Log Analytics query that returns a list of values can be used in the **Query** field in the Variable edit view. There is also one Grafana function for Log Analytics that returns a list of workspaces.

For information about templates and template variables, see [Templates and variables \(p. 265\)](#).

Name	Description
<code>workspaces()</code>	Returns a list of workspaces for the default subscription.
<code>workspaces(12345678-aaaa-bbbb-cccc-123456789aaa)</code>	Returns a list of workspaces for the specified subscription (the parameter can be quoted or unquoted).

The following table shows example variable queries.

Query	Description
<code>subscriptions()</code>	Returns a list of Azure subscriptions.
<code>workspaces()</code>	Returns a list of workspaces for default subscription.
<code>workspaces("12345678-aaaa-bbbb-cccc-123456789aaa")</code>	Returns a list of workspaces for a specified subscription.
<code>workspaces("\${subscription}")</code>	With template variable for the subscription parameter.
<code>workspace("myWorkspace").Heartbeat \   distinct Computer</code>	Returns a list of virtual machines.
<code>workspace("\${workspace").Heartbeat \   distinct Computer</code>	Returns a list of virtual machines with template variable.
<code>workspace("\${workspace").Perf \   distinct ObjectName</code>	Returns a list of objects from the Perf table.
<code>workspace("\${workspace").Perf \   where ObjectName == "\${object}" \   distinct CounterName</code>	Returns a list of metric names from the Perf table.

The following code xample shows a time series query using variables.

```
Perf
| where ObjectName == "${object}" and CounterName == "${metric}"
| where TimeGenerated >= $__timeFrom() and TimeGenerated <= $__timeTo()
| where $__contains(Computer, $computer)
| summarize avg(CounterValue) by bin(TimeGenerated, $__interval), Computer
| order by TimeGenerated asc
```

### Deep linking from Grafana panels to the Log Analytics query editor in Azure Portal

Choose a time series in the panel to see a context menu with a link to **View in Azure Portal**. Choosing that link opens the Azure Log Analytics query editor in the Azure Portal and runs the query from the Grafana panel there.

If you're not currently logged in to the Azure Portal, then the link opens the login page. The provided link is valid for any account, but it only displays the query if your account has access to the Azure Log Analytics workspace specified in the query.

### Azure Log Analytics alerting

Grafana alerting is supported for Application Insights. This is not Azure Alerts support. For more information about alerting in Grafana workspaces, see [Alerts \(p. 282\)](#).

### Querying the Application Insights Analytics service

If you change the service type to **Insights Analytics**, then a similar editor to the Log Analytics service is available. This service also uses the Kusto language, so the instructions for querying data are identical to [Querying the Azure Log Analytics service \(p. 72\)](#), except that you query Application Insights Analytics data instead.

## Graphite

Grafana has an advanced Graphite query editor that lets you quickly navigate the metric space, add functions, change function parameters and much more. The editor can handle all types of graphite queries. It can even handle complex nested queries through the use of query references.

### Graphite settings

To access Graphite settings, pause on the **Configuration** (gear) icon, then choose **Data Sources**, and then choose the Graphite data source.

Name	Description
Name	The data source name. This is how you see the data source in panels and queries.
Default	Default data source means that it will be pre-selected for new panels.
URL	The HTTP protocol, IP, and port of your graphite-web or graphite-api install.
Access	Server (default) = URL must be accessible from the Grafana backend/server. Browser = URL must be accessible from the browser.
Auth	
Basic Auth	Enable basic authentication to the data source.
User	User name for basic authentication.
Password	Password for basic authentication.
Custom HTTP Headers	Choose <b>Add header</b> to add a custom HTTP header.
Header	Enter the custom header name.
Value	Enter the custom header value.
Graphite details	
Version	Select your version of Graphite.
Type	Select your type of Graphite.

Access mode controls how requests to the data source will be handled. Server should be the preferred way if nothing else is stated.

#### Server access mode (default)

All requests are made from the browser to Amazon Managed Grafana, which forwards the requests to the data source, circumventing possible Cross-Origin Resource Sharing (CORS) requirements. If you select this access mode, the URL must be accessible from Amazon Managed Grafana.

### Browser access mode

All requests will be made from the browser directly to the data source and may be subject to CORS requirements. If you select this access mode, the URL must be accessible from the browser.

### Graphite query editor

Grafana includes a Graphite-specific query editor to help you build your queries.

To see the raw text of the query that is sent to Graphite, choose the **Toggle text edit mode** (pencil) icon.

### Choosing metrics to query

Choose **Select metric** to navigate the metric space. After you start, you can continue using the pointer or keyboard arrow keys. You can select a wildcard character and still continue.

### Functions

To add a function, choose the plus icon next to **Function**. You can search for the function or select it from the menu. After a function is selected, it will be added and your focus will be in the text box of the first parameter. To edit or change a parameter, choose it and it will turn into a text box. - To delete a function, choose the function name followed by the x icon.

Some functions, such as `aliasByNode`, support an optional second argument. To add an argument, pause on the first argument, and then choose the + symbol that appears. To remove the second optional parameter, choose it and keep it blank. The editor will remove it.

### Sort labels

If you want consistent ordering, use `sortByName`. This can be annoying when you have the same labels on multiple graphs, and they are both sorted differently and using different colors. To fix this, use `sortByName()`.

### Nested queries

You can reference queries by the row *letter* that they're on (similar to Microsoft Excel). If you add a second query to a graph, you can reference the first query by typing in #A. This provides a convenient way to build compounded queries.

### Avoiding many queries by using wildcard characters

Occasionally, you might want to see multiple time series plotted on the same graph. For example, you might want to see how the CPU is being used on a machine. You might initially create the graph by adding a query for each time series, such as `cpu.percent.user.g`, `cpu.percent.system.g`, and so on. This results in *n* queries made to the data source, which is inefficient.

To be more efficient one can use wildcard characters in your search, returning all the time series in one query. For example, `cpu.percent.*.g`.

### Modifying the metric name in tables or charts

Use `alias` functions to change metric names on Grafana tables or graphs; for example, `aliasByNode()` or `aliasSub()`.

### Point consolidation

All Graphite metrics are consolidated so that Graphite doesn't return more data points than there are pixels in the graph. By default, this consolidation is done using `avg` function. You can control how Graphite consolidates metrics by adding the `Graphite consolidateBy` function.

### Note

This means that legend summary values (max, min, total) cannot all be correct at the same time. They are calculated client-side by Grafana. And depending on your consolidation function, only one or two can be correct at the same time.

## Combining time series

To combine time series, choose **Combine** in the **Functions** list.

## Data exploration and tags

In Graphite, everything is a tag.

When exploring data, previously selected tags are used to filter the remaining result set. To select data, you use the `seriesByTag` function, which takes tag expressions (`=`, `!=`, `=~`, `!~=`) to filter time series.

The Grafana query builder does this for you automatically when you select a tag.

### Note

**Tip:** The regular expression search can be slow on high-cardinality tags, so try to use other tags to reduce the scope first. Starting off with a particular name or namespace helps reduce the results.

## Template variables

Instead of hardcoding things such as server, application, and sensor name in your metric queries, you can use variables in their place. Variables are shown as dropdown select boxes at the top of the dashboard. You can use these dropdown boxes to change the data being displayed in your dashboard.

For more information about templating and template variables, see [Templates and variables \(p. 265\)](#).

To create a variable using tag values, use the Grafana functions `tags` and `tag_values`.

Query	Description
<code>tags()</code>	Returns all tags.
<code>tags(server=~backend\*)</code>	Returns only tags that occur in series matching the filter expression.
<code>tag_values(server)</code>	Return tag values for the specified tag.
<code>tag_values(server, server=~backend\*)</code>	Returns filtered tag values that occur for the specified tag in series matching those expressions.
<code>tag_values(server, server=~backend\*, app=~\${apps:regex})</code>	Multiple filter expressions and expressions can

Query	Description
	contain other variables.

For more details, see [Graphite docs on the autocomplete API for tags](#).

### Query variable

The query you specify in the query field should be a metric find type of query. For example, a query such as `prod.servers.*` will fill the variable with all possible values that exist in the wildcard position.

You can also create nested variables that use other variables in their definition. For example `apps.$app.servers.*` uses the variable `$app` in its query definition.

### Using `__searchFilter` to filter query variable results

Using `__searchFilter` in the query field will filter the query result based on what you enter in the dropdown select box. When you enter nothing, the default value for `__searchFilter` is `*` and ``` when used as part of a regular expression.

The following example shows how to use `__searchFilter` as part of the query field to enable searching for `server` while the user enters text in the dropdown select box.

Query

```
apps.$app.servers.$__searchFilter
```

TagValues

```
tag_values(server, server=~${__searchFilter:regex})
```

### Variable usage

You can use a variable in a metric node path or as a parameter to a function.

There are two syntaxes:

- `$<varname>` Example: `apps.frontend.$server.requests.count`
- `${varname}` Example: `apps.frontend.${server}.requests.count`

Why two ways? The first syntax is easier to read and write but does not allow you to use a variable in the middle of a word. Use the second syntax in expressions such as `my.server${serverNumber}.count`.

### Variable usage in tag queries

Multi-value variables in tag queries use the advanced formatting syntax introduced in Grafana 5.0 for variables: `{var:regex}`. Non-tag queries will use the default glob formatting for multi-value variables.

The following code example shows a tag expression with regex formatting and using the Equal Tilde operator, `=~`.

```
server=~${servers:regex}
```

For more information, see [Advanced variable format options](#) (p. 276).

## Annotations

Annotations enable you to overlay rich event information on top of graphs. You add annotation queries via the Dashboard menu / Annotations view. For more information, see [Annotations](#) (p. 238).

Graphite supports two ways to query annotations:

- A regular metric query. For this, you use the **Graphite query** text box.
- A Graphite events query. For this, you use the `Graphite event tags` text box, and specify a tag or wildcard character (keeping it empty should also work).

## Google BigQuery

Provides support for [BigQuery](#) as a backend database. Features:

- Query setup
- Raw SQL editor
- Query builder
- Macros support
- Additional functions
- Table view
- Annotations
- BQ queries in variables
- Sharded tables
- Partitioned tables
- Granular slot allocation (Running queries in a project with flat-rate pricing)

## Adding the data source

1. Open the Grafana console in the Amazon Managed Grafana workspace and make sure you are logged in.
2. In the side menu under **Configuration** (the gear icon), choose **Data Sources**.
3. Choose **Add data source**.

### Note

If you don't see the **Data Sources** link in your side menu, it means that your current user does not have the `Admin` role.

4. Select **BigQuery** from the list of data sources.
5. Enter the following information:
  - For **Name**, enter the data source name.
  - **Default** means that it will be pre-selected for new panels.
  - For **Service Account Key**, enter the Service Account Key File for a GCP Project. Instructions for how to create this are later in this document.

You can set query priority `INTERACTIVE` or `BATCH` per data source.

## Authentication

To authenticate the BigQuery plugin, upload a Google JWT file. You need to create a Google Cloud Platform (GCP) Service Account for the Project you want to show data for. An Amazon Managed Grafana datasource integrates with one GCP Project. If you want to visualize data from multiple GCP Projects, then you can give the service account permissions in each project or create one datasource per GCP Project.

### Enable APIs

Go to [BigQuery API](#) and **Enable** the API.

### Create a GCP Service Account for a Project

#### To create a GCP service account for a project

1. Navigate to the [API & Service Credentials Page](#).
2. Choose **Create credentials** and choose **Service account key**.
3. On the **Create service account key** page, choose key type **JSON**. Then in the **Service Account** dropdown, choose the **New service account** option.
4. Enter a name for the service account in the **Service account name** field and then choose the **BigQuery Data Viewer** and **BigQuery Job User** roles from the **Role** dropdown.
5. Choose **Create**. A JSON key file is created and downloaded to your computer. Store this file in a secure place, as it allows access to your BigQuery data.
6. Upload it to Amazon Managed Grafana on the data source configuration page. You can either upload the file or paste in its contents.
7. The file contents are encrypted and saved in the Grafana database. Don't forget to save after uploading the file.

## Using the query builder

The query builder provides a simple yet a user-friendly interface to help you quickly compose a query. The builder enables you to define the basic parts of your query, The common ones are:

- The table that you want to query from
- The time field and metric field
- WHERE clause: either use one of the pre-defined macros, to speed your writing time, or set up your own expression. The existing supported macros are the following:
  - Macro `$__timeFiler` with last 7 days example

```
WHERE `createDate` BETWEEN TIMESTAMP_MILLIS (1592147699012) AND TIMESTAMP_MILLIS  
(1592752499012) AND _PARTITIONTIME >= '2020-06-14 18:14:59' AND _PARTITIONTIME <  
'2020-06-21 18:14:59'
```

- Macro `$__timeFrom` with last 7 days example

```
WHERE `createDate` > TIMESTAMP_MILLIS (1592223758609) AND _PARTITIONTIME >=  
'2020-06-15 15:22:38' AND _PARTITIONTIME < '2020-06-22 15:22:38'
```

- Macro `$__timeTo` with last 7 days

```
WHERE `createDate` < TIMESTAMP_MILLIS (1592828659681) AND _PARTITIONTIME >=  
'2020-06-15 15:24:19' AND _PARTITIONTIME < '2020-06-22 15:24:19'
```



- GROUPBY option: you can use a pre-defined macro or use one of the fields from your query a. time (\$\_interval,none)
- ORDER BY option

#### Note

Note: If your processing location is not the Default US one, set your location from the processing Location drop-down at the top right bottom of the query builder

## Troubleshooting

To troubleshoot a query, use the Query Inspector at the top of the query builder. This helps you see the clean query and troubleshoot SQL errors.

## Google Cloud Monitoring

#### Note

In earlier versions of Grafana, this data source was named Google Stackdriver.

Add the Google Cloud Monitoring data source to be able to build dashboards for your Google Cloud Monitoring metrics.

### Adding the data source

1. Open the side menu by choosing the Grafana icon in the top header.
2. In the side menu, under the **Dashboards** link, you should find the **Data Sources** link.
3. Choose the **+ Add data source** button in the top header.
4. Select **Google Cloud Monitoring** from the **Type** dropdown list.
5. Upload or paste in the Service Account Key file. See later in this document for steps to create a Service Account Key file.

#### Note

If you don't see the **Data Sources** link in your side menu, your current user account does not have the `Admin` role.

Name	Description
Name	The data source name. This is how you refer to the data source in panels and queries.
Default	Default data source means that it will be pre-selected for new panels.
Service Account Key	Service account key file for a GCP Project. See the instructions later in this document about how to create it.

## Authentication

There are two ways to authenticate the Google Cloud Monitoring plugin

- Upload a Google JWT file
- Automatically retrieve credentials from Google metadata server

The latter option is only available when running Grafana on GCE virtual machine.

## Using a Google service account key file

To authenticate with the Google Cloud Monitoring API, you must create a Google Cloud Platform (GCP) Service Account for the Project that you want to show data for. A Grafana data source integrates with one GCP Project. To visualize data from multiple GCP Projects, you must create one data source per GCP Project.

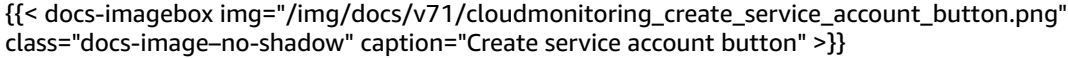
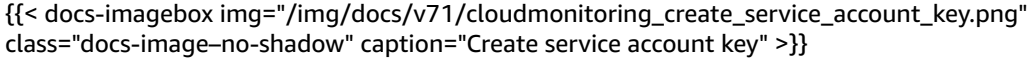
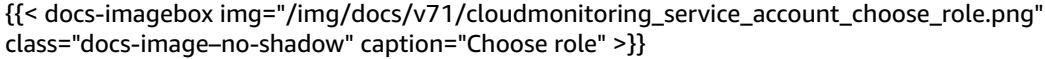
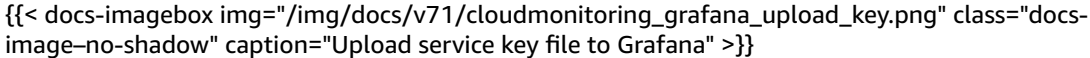
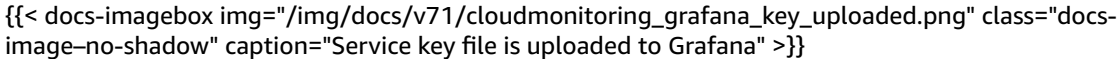
### Enabling APIs

The following APIs must be enabled first:

- [Monitoring API](#)
- [Cloud Resource Manager API](#)

Choose the links listed, and then choose the **Enable** button.

### Creating a GCP service account for a Project

1. Navigate to the [APIs and Services Credentials](#) page.
2. Choose the **Create credentials** dropdown/button and choose the **Service account key** option.  

3. On the **Create service account key** page, choose key type **JSON**. Then, in the **Service Account** dropdown list, choose the **New service account** option.  

4. Some new fields will appear. Fill in a name for the service account in the **Service account name** field and then choose the **Monitoring Viewer** role from the **Role** dropdown list.  

5. Choose the **Create** button. A JSON key file will be created and downloaded to your computer. Store this file in a secure place as it allows access to your Google Cloud Monitoring data.
6. Upload it to Grafana on the data source **Configuration** page. You can either upload the file or paste in the contents of the file.  

7. The file contents will be encrypted and saved in the Grafana database. Don't forget to save after uploading the file!  


## Using the query editor

The Google Cloud Monitoring query editor allows you to build two types of queries - **Metric** and **Service Level Objective (SLO)**. Both types return time series data.

### Metric queries

The metric query editor allows you to select metrics, group/aggregate by labels and by time, and use filters to specify which time series you want in the results.

To create a metric query, follow these steps:

1. Choose the option **Metrics** in the **Query Type** dropdown list.
2. Choose a project from the **Project** dropdown list.
3. Choose a Google Cloud Platform service from the **Service** dropdown list.
4. Choose a metric from the **Metric** dropdown list.
5. To add or remove filters or group by clauses, use the plus and minus icons in the filter and group by sections. This step is optional.

Google Cloud Monitoring metrics can be of different kinds (GAUGE, DELTA, CUMULATIVE) and these kinds have support for different aggregation options (reducers and aligners). The Grafana query editor shows the list of available aggregation methods for a selected metric and sets a default reducer and aligner when you select the metric. Units for the Y-axis are also automatically selected by the query editor.

## Filters

To add a filter, choose the plus icon, choose a field to filter by, and enter a filter value. For example, enter `instance_name = grafana-1`. You can remove the filter by choosing the filter name and selecting `--remove filter--`.

## Simple wildcard characters

When the operator is set to or `,=!=` it is possible to add wildcard characters to the filter value field. For example, `us-*` captures all values that start with "us-", and `*central-a` captures all values that end with "central-a". `*-central-*` captures all values that have the substring of `central-`. Simple wildcard characters are less expensive than regular expressions.

## Regular expressions

When the operator is set to or `,=!=~` it is possible to add regular expressions to the filter value field. For example, `us-central[1-3]-[af]` matches all values that start with "us-central", followed by a number in the range of 1 to 3, a dash and then either an "a" or an "f". Leading and trailing slashes are not needed when creating regular expressions.

## Aggregation

The aggregation field lets you combine time series based on common statistics. For more information about aggregation, refer to [aggregation options](#).

The `Aligner` field allows you to align multiple time series after the same group by time interval. For more information about aligner, refer to [alignment metric selector](#).

## Alignment Period and grouping by time

The `Alignment Period` groups a metric by time if an aggregation is chosen. The default is to use the GCP Google Cloud Monitoring default groupings (which allows you to compare graphs in Grafana with graphs in the Google Cloud Monitoring UI). The option is called `cloud_monitoring_auto` and the defaults are:

- 1m for time ranges < 23 hours
- 5m for time ranges >= 23 hours and < 6 days
- 1h for time ranges >= 6 days

The other automatic option is `grafana_auto`. This will automatically set the group by time depending on the time range chosen and the width of the graph panel. For more information, see [Adding an interval variable \(p. 271\)](#).

It is also possible to choose fixed time intervals to group by, such as 1h or 1d.

## Group By

Group by resource or metric labels to reduce the number of time series and to aggregate the results by a group by. For example, group by instance\_name to see an aggregated metric for a compute instance.

## Metadata labels

Resource metadata labels contain information to uniquely identify a resource in Google Cloud. Metadata labels are only returned in the time series response if they're part of the **Group By** segment in the time series request. There's no API for retrieving metadata labels, so it's not possible to populate the group by dropdown list with the metadata labels that are available for the selected service and metric. However, the **Group By** field dropdown list comes with a pre-defined list of common system labels.

User labels cannot be pre-defined, but it's possible to enter them manually in the **Group By** field. If a metadata label, user label, or system label is included in the **Group By** segment, you can create filters based on it and expand its value in the **Alias** field.

## Alias patterns

The Alias By field allows you to control the format of the legend keys. The default is to show the metric name and labels. This can be long and hard to read. Using the following patterns in the alias field, you can format the legend key the way you want it.

## Metric Type patterns

Alias pattern	Description	Example result
{{metric.type}}	Returns the full Metric Type.	compute.googleapis.com/instance/cpu/utilization
{{metric.name}}	Returns the metric name part.	instance/cpu/utilization
{{metric.service}}	Returns the service part.	compute

## Label patterns

In the Group By dropdown list, you can see a list of metric and resource labels for a metric. These can be included in the legend key using alias patterns.

Alias pattern format	Description	Alias pattern example	Example result
{{metric.label.xxx}}	Returns the metric label value.	{{metric.label.instance_name}}-prod	prod
{{resource.label.xxx}}	Returns the resource label value.	{{resource.label.zone}}us-east1-b	us-east1-b
{{metadata.system_label.xxx}}	Returns the metadata system label value.	{{metadata.system_label.instance_name}}	instance_name
{{metadata.user_label.xxx}}	Returns the metadata user label value.	{{metadata.user_label.project_id}}	project_id

Example Alias By: `{{metric.type}} - {{metric.label.instance_name}}`

Example Result: `compute.googleapis.com/instance/cpu/usage_time - server1-prod`

It is also possible to resolve the name of the Monitored Resource Type.

Alias pattern format	Description	Example result
<code>{{resource.type}}</code>	Returns the name of the monitored resource type.	<code>gce_instance</code>

Example Alias By: `{{resource.type}} - {{metric.type}}`

Example Result: `gce_instance - compute.googleapis.com/instance/cpu/usage_time`

## SLO queries

### Note

SLO queries are available only in Grafana v7.0+

The SLO query builder in the Google Cloud Monitoring data source allows you to display SLO data in time series format. To get an understanding of the basic concepts in service monitoring, refer to the Google Cloud Monitoring [official documentation](#).

## Creating an SLO query

To create an SLO query, follow these steps:

1. Choose the option **Service Level Objectives (SLO)** in the **Query Type** dropdown list.
2. Choose a project from the **Project** dropdown list.
3. Choose an [SLO service](#) from the **Service** dropdown list.
4. Choose an [SLO](#) from the **SLO** dropdown list.
5. Choose a [time series selector](#) from the **Selector** dropdown list.

The friendly names for the time series selectors are shown in Grafana. The following table shows the mapping from the friendly name to the system name that is used in the Service Monitoring documentation.

Selector dropdown list value	Corresponding time series selector used
SLI Value	<code>select_slo_health</code>
SLO Compliance	<code>select_slo_compliance</code>
SLO Error Budget Remaining	<code>select_slo_budget_fraction</code>

## Alias patterns for SLO queries

You can use the Alias By field to control the format of the legend keys for SLO queries.

Alias pattern	Description	Example result
<code>{{project}}</code>	Returns the GCP project name.	<code>myProject</code>

Alias pattern	Description	Example result
{{service}}	Returns the service name.	myService
{{slo}}	Returns the SLO.	latency-slo
{{selector}}	Returns the selector.	select_slo_health

### Alignment Period and grouping by time for SLO queries

SLO queries use the same alignment period functionality as metric queries. For more information, see [Metric queries \(p. 83\)](#).

### Templating

Instead of hardcoding things such as server, application and sensor name in your metric queries you can use variables in their place. Variables are shown as dropdown select boxes at the top of the dashboard. You can use these dropdown boxes to change the data being displayed in your dashboard.

For more information about templating and template variables, see [Templates and variables \(p. 265\)](#).

### Query variable

Variable of the type *Query* allows you to query Google Cloud Monitoring for various types of data. The Google Cloud Monitoring data source plugin provides the following *Query* Types.

Name	Description
Metric Types	Returns a list of metric type names that are available for the specified service.
Labels Keys	Returns a list of keys for <code>metric_label</code> and <code>resource_label</code> in the specified metric.
Labels Values	Returns a list of values for the label in the specified metric.
Resource Types	Returns a list of resource types for the specified metric.
Aggregations	Returns a list of aggregations (cross series reducers) for the specified metric.
Aligners	Returns a list of aligners (per series aligners) for the specified metric.
Alignment periods	Returns a list of all alignment periods that are available in Google Cloud Monitoring query editor in Grafana.
Selectors	Returns a list of selectors that can be used in SLO (Service Level Objectives) queries.
SLO Services	Returns a list of Service Monitoring services that can be used in SLO queries.
Service Level Objectives (SLO)	Returns a list of SLO's for the specified SLO service.

### Using variables in queries

There are two syntaxes:

- `<varname>` Example: `metric.label.$metric_label`
- `[[varname]]` Example: `metric.label.[[metric_label]]`

Why two ways? The first syntax is easier to read and write but does not allow you to use a variable in the middle of a word. When the *Multi-value* or *Include all value* options are enabled, Grafana converts the labels from plaintext to a regex compatible string, which means you have to use `=~` instead of `=`.

## Annotations

You can use annotations to overlay rich event information on top of graphs. You add annotation queries via the Dashboard menu / Annotations view. Annotation rendering is expensive so it is important to limit the number of rows returned. There is no support for showing Google Cloud Monitoring annotations and events yet but it works well with [custom metrics](#) in Google Cloud Monitoring.

For more information about annotations, see [Annotations \(p. 238\)](#).

With the query editor for annotations, you can select a metric and filters. The **Title** and **Text** fields support templating and can use data returned from the query. For example, the Title field could have the following text:

```
{{metric.type}} has value: {{metric.value}}
```

Example Result: `monitoring.googleapis.com/uptime_check/http_status` has this value: 502

## Patterns for the annotation query editor

Alias pattern format	Description	Alias pattern example	Example result
<code>{{metric.value}}</code>	Value of the metric/point.	<code>{{metric.value}}</code>	555
<code>{{metric.type}}</code>	Returns the full Metric Type.	<code>{{metric.type}}</code>	<code>compute.googleapis.com/instance/cpu/utilization</code>
<code>{{metric.name}}</code>	Returns the metric name part.	<code>{{metric.name}}</code>	<code>instance/cpu/utilization</code>
<code>{{metric.service}}</code>	Returns the service part.	<code>{{metric.service}}</code>	<code>compute</code>
<code>{{metric.label.instance_name}}</code>	Returns the metric label value.	<code>{{metric.label.instance_name}}</code>	<code>us-east1-b</code>
<code>{{resource.label.zone}}</code>	Returns the resource label value.	<code>{{resource.label.zone}}</code>	<code>us-east1-b</code>

## Deep linking from Grafana panels to the Metrics Explorer in Google Cloud Console

### Note

This feature is available only for Metric queries.

Choose a time series in the panel to see a context menu with a link to View in Metrics Explorer in Google Cloud Console. Choosing that link opens the Metrics Explorer in the Google Cloud Console and runs the query from the Grafana panel there. The link navigates the user first to the Google Account Chooser. After successfully selecting an account, the user is redirected to the Metrics Explorer. The provided link is valid for any account, but it only displays the query if your account has access to the GCP project specified in the query.

## Google Sheets

Use this data source to visualize your Google spreadsheets.

For configuration information, see [Configuring the Google Sheets data source](#).

For provisioning information, see [Provisioning](#).

For information about using the editor, see [Using the editor](#).

For quota information, see [Usage limits](#).

## InfluxDB

Grafana ships with a feature-rich data source plugin for InfluxDB. The plugin includes a custom query editor and supports annotations and query templates.

### Adding the data source

1. Open the side menu by choosing the Grafana icon in the top header.
2. In the side menu under the link, **Dashboards** you should find a link named **Data Sources**.
3. Choose the **+ Add data source** button in the top header.
4. Select **InfluxDB** from the **Type** dropdown list.
5. Select **InfluxQL** or **Flux** from the **Query Language** list.

#### Note

If you don't see the **Data Sources** link in your side menu, it means that your current user does not have the `Admin` role.

## Jaeger

The Jaeger data source provides open-source, end-to-end distributed tracing.

### Adding the data source

To access Jaeger settings, choose the **Configuration** (gear) icon, then choose **Data Sources**, and then choose **Jaeger**.

Name	Description
Name	The data source name. This is how you see the data source in panels, queries, and Explore.
Default	Default data source means that it will be pre-selected for new panels.
URL	The URL of the Jaeger instance; e.g., <code>http://localhost:16686</code> .
Access	Server (default) = URL must be accessible from the Grafana backend/server. Browser = URL must be accessible from the browser.
Basic Auth	Enable basic authentication to the Jaeger data source.
User	User name for basic authentication.
Password	Password for basic authentication.



## Query traces

You can query and display traces from Jaeger via Explore. For more information, see [Explore \(p. 254\)](#).

The Jaeger query editor allows you to query by trace ID directly or selecting a trace from trace selector. To query by trace ID, insert the ID into the text input.

Use the trace selector to pick particular trace from all traces logged in the time range you have selected in Explore. The trace selector has three levels of nesting: 1. The service you are interested in. 1. Particular operation is part of the selected service. 1. Specific trace in which the selected operation occurred, represented by the root operation name and trace duration.

## Linking to the trace ID from logs

You can link to Jaeger trace from logs in Loki by configuring a derived field with internal link. For more information, see [Derived fields \(p. 95\)](#).

## JSON

The JSON data source executes requests against arbitrary backends and parses JSON response into Grafana dataframes.

### Adding the data source

1. Open the Grafana console in the Amazon Managed Grafana workspace and make sure you are logged in.
2. In the side menu under **Configuration** (the gear icon), choose **Data Sources**.
3. Choose **Add data source**.

#### Note

If you don't see the **Data Sources** link in your side menu, it means that your current user does not have the `Admin` role.

4. Select **JSON** from the list of data sources.
5. In the **URL** field, enter your API endpoint. That's where the data source will make requests to.

## JSON API

There is an OpenAPI definition for this data source. For more information, see [simPod / GrafanaJsonDatasource /](#)

To work with this datasource, the backend must implement four endpoints:

- `GET /` with 200 status code response. Used for "Test connection" on the datasource configuration page.
- `POST /search` returning available metrics when invoked.
- `POST /query` returning metrics based on input.
- `POST /annotations` returning annotations.

The following two URLs are optional:

- `POST /tag-keys` returning tag keys for ad hoc filters.
- `POST /tag-values` returning tag values for ad hoc filters.

### [/search](#)

`POST /search`

Amazon Managed Grafana issues this request on:

- **Variables, New/edit** page. The `Query` field is passed in a body as the following:

```
{ "target": "query field value" }
```

- **Panel, Queries** page. `Format As` and `Metric` values are passed in a body as the following:

```
{ "type": "timeseries", "target": "upper_50" }
```

The way you handle those values is up to you. The response body can either contain an array or a map.  
Example array response:

```
[ "upper_25", "upper_50", "upper_75", "upper_90", "upper_95" ]
```

Example map response:

```
[ { "text": "upper_25", "value": 1 }, { "text": "upper_75", "value": 2 } ]
```

## [/query](#)

POST `/query`

Example timeseries request:

```
{
  "panelId": 1,
  "range": {
    "from": "2016-10-31T06:33:44.866Z",
    "to": "2016-10-31T12:33:44.866Z",
    "raw": {
      "from": "now-6h",
      "to": "now"
    }
  },
  "rangeRaw": {
    "from": "now-6h",
    "to": "now"
  },
  "interval": "30s",
  "intervalMs": 30000,
  "maxDataPoints": 550,
  "targets": [
    { "target": "Packets", "refId": "A", "type": "timeseries", "data": { "additional":
"optional json" } },
    { "target": "Errors", "refId": "B", "type": "timeseries" }
  ],
  "adhocFilters": [{
    "key": "City",
    "operator": "=",
    "value": "Berlin"
  }]
}
```

Example timeseries response (metric value as a float, unixtimestamp in milliseconds):

```
[
  {
```

```

    "target": "pps in",
    "datapoints": [
      [622, 1450754160000],
      [365, 1450754220000]
    ]
  },
  {
    "target": "pps out",
    "datapoints": [
      [861, 1450754160000],
      [767, 1450754220000]
    ]
  },
  {
    "target": "errors out",
    "datapoints": [
      [861, 1450754160000],
      [767, 1450754220000]
    ]
  },
  {
    "target": "errors in",
    "datapoints": [
      [861, 1450754160000],
      [767, 1450754220000]
    ]
  }
}
]

```

The relation between `target` in request and response is 1:n. You can return multiple targets in response for one requested target.

Example table response to be returned if the metric selected is `"type": "table"`:

```

[
  {
    "columns": [
      { "text": "Time", "type": "time" },
      { "text": "Country", "type": "string" },
      { "text": "Number", "type": "number" }
    ],
    "rows": [
      [1234567, "SE", 123],
      [1234567, "DE", 231],
      [1234567, "US", 321]
    ],
    "type": "table"
  }
]

```

### Additional data

Sending additional data for each metric is supported via the `Additional JSON Data` input field that allows you to enter JSON.

For example, when `{ "additional": "optional json" }` is entered into `Additional JSON Data`, it is attached to the target data under the `"data"` key:

```

{ "target": "upper_50", "refId": "A", "type": "timeseries", "data": { "additional":
  "optional json" } }

```

You can also enter variables:

```
{"key": $variableValue}
```

## [/annotations](#)

POST /annotations

The JSON request body looks like this:

```
{
  "range": {
    "from": "2016-04-15T13:44:39.070Z",
    "to": "2016-04-15T14:44:39.070Z"
  },
  "rangeRaw": {
    "from": "now-1h",
    "to": "now"
  },
  "annotation": {
    "name": "deploy",
    "datasource": "JSON Datasource",
    "iconColor": "rgba(255, 96, 96, 1)",
    "enable": true,
    "query": "#deploy"
  },
  "variables": []
}
```

Amazon Managed Grafana expects a response containing an array of annotation objects.

Field explanation:

- **text**— (Required) Text for the annotation.
- **title**— (Optional) The title for the annotation tooltip.
- **isRegion**— (Optional) Whether this is a region.
- **time**— (Required) Time since UNIX epoch in milliseconds.
- **timeEnd**— (Required if **isRegion** is true) Time since UNIX epoch in milliseconds.
- **tags**— (Optional) Tags for the annotation.

```
[
  {
    "text": "text shown in body",
    "title": "Annotation Title",
    "isRegion": true,
    "time": "timestamp",
    "timeEnd": "timestamp",
    "tags": ["tag1"]
  }
]
```

### Note

Note: If the datasource is configured to connect directly to the backend, you also need to implement `OPTIONS /annotations` that responds with the correct CORS headers:

- `Access-Control-Allow-Headers:accept, content-type`
- `Access-Control-Allow-Methods:POST`
- `Access-Control-Allow-Origin:*`

### [/tag-keys](#)

POST /tag-keys

The JSON request body looks like this:

```
{ }
```

The tag keys API returns the following:

```
[  
  {"type": "string", "text": "City"},  
  {"type": "string", "text": "Country"}  
]
```

### [/tag-values](#)

POST /tag-values

The JSON request body looks like this:

```
{ "key": "City" }
```

The tag keys API returns the following:

```
[  
  {"text": "Eins!"},  
  {"text": "Zwei!"},  
  {"text": "Drei!"}  
]
```

## Loki

The Loki data source provides access to Loki, Grafana's log aggregation system.

### Adding the data source

1. Open the Grafana workspace and make sure you are logged in.
2. In the side menu under the **Configuration** link you should find a **Data Sources** link.
3. choose the **Add data source** button at the top.
4. Select **Loki** from the list of data sources.

#### Note

If you don't see the **Data Sources** link in your side menu, it means that your current user does not have the Admin role.

Name	Description
Name	The data source name. This is how you see the data source in panels, queries, and Explore.
Default	Default data source means that it will be pre-selected for new panels.
URL	The URL of the Loki instance; e.g., <code>http://localhost:3100</code> .

Name	Description
Maximum lines	Upper limit for number of log lines returned by Loki (default is 1000). Decrease if your browser is sluggish when displaying logs in Explore.

## Derived fields

You can use the *derived fields* configuration to do the following:

- Add fields parsed from the log message.
- Add a link that uses the value of the field.

You can use this functionality to link to your tracing backend directly from your logs, or link to a user profile page if a `userId` is present in the log line. These links appear in the log details. For more information, see [Labels and detected fields \(p. 259\)](#).

Each derived field consists of the following:

- **Name** – Shown in the log details as a label.
- **Regex** – A Regex pattern that runs on the log message and captures part of it as the value of the new field. Can only contain a single capture group.
- **URL/query** – If the link is external, then enter the full link URL. If the link is internal link, then this input serves as query for the target data source. In both cases, you can interpolate the value from the field with `${__value.raw }` macro.
- **Internal link** – Select if the link is internal or external. In case of internal link, a data source selector allows you to select the target data source. Only tracing data sources are supported.

You can use a debug section to see what your fields extract and how the URL is interpolated. choose **Show example log message** to show the text area where you can enter a log message.

The new field with the link shown in log details.

## Querying logs

Querying and displaying log data from Loki is available via Explore and with the logs panel in visualizations. Select the Loki data source, and then enter a LogQL query to display your logs. For more information about LogQL, see [LogQL](#).

### Log queries

A log query consists of two parts: **log stream selector**, and a **search expression**. For performance reasons, you must start by choosing a log label for a log stream.

The Logs Explorer (the **Log labels** button) next to the query field shows a list of labels of available log streams. An alternative way to write a query is to use the query field's automatic completion. You start by typing a left curly brace `{` and the autocomplete menu will suggest a list of labels. Press the **Enter** key to run the query.

After the result is returned, the log panel shows a list of log rows and a bar chart where the x-axis shows the time and the y-axis shows the frequency/count.

### Log Stream Selector

For the label part of the query expression, wrap it in curly braces `{ }` and then use the key value syntax for selecting labels. Multiple label expressions are separated by a comma:

```
{app="mysql",name="mysql-backup"}
```

The following label matching operators are currently supported:

- `=` exactly equal.
- `!=` not equal.
- `=~` regex-match.
- `!~` do not regex-match.

Examples:

- `{name=~"mysql.+"}`
- `{name!~"mysql.+"}`

Another way to add a label selector is in the table section. choose **Filter** beside a label to add the label to the query expression. This even works for multiple queries and will add the label selector to each query.

### Search expressions

After writing the Log Stream Selector, you can filter the results further by writing a search expression. The search expression can be just text or a regex expression.

Example queries:

- `{job="mysql"} |= "error"`
- `{name="kafka"} |~ "tsdb-ops.*io:2003"`
- `{instance=~"kafka-[23]",name="kafka"} != "kafka.server:type=ReplicaManager"`

Filter operators can be chained and will sequentially filter down the expression. The resulting log lines will satisfy every filter.

Example

```
{job="mysql"} |= "error" != "timeout"
```

The following filter types are currently supported:

- `|=` line contains string.
- `!=` line doesn't contain string.
- `|~` line matches regular expression.
- `!~` line does not match regular expression.

#### Note

For more information about LogQL, Loki's query language, see [Loki LogQL](#).

### Live tailing

Loki supports Live tailing which displays logs in real-time. This feature is supported in Explore.

Note that Live Tailing relies on two WebSocket connections: one between the browser and the Amazon Managed Grafana workspace, and another between the workspace and the Loki server. If you run any reverse proxies, configure them accordingly. The following example code for Apache2 can be used for proxying between the browser and the Grafana server.

```
ProxyPassMatch "^/(api/datasources/proxy/\d+/loki/api/v1/tail)" "ws://127.0.0.1:3000/$1"
```

The following example shows basic NGINX proxy configuration. It assumes that the Grafana server is available at `http://localhost:3000/`, Loki server is running locally without proxy, and your external site uses HTTPS. If you also host Loki behind an NGINX proxy, then you might want to repeat the following configuration for Loki as well.

In the `http` section of NGINX configuration, add the following map definition.

```
map $http_upgrade $connection_upgrade {
    default upgrade;
    '' close;
}
```

In your `server` section, add the following configuration.

```
location ~ /(api/datasources/proxy/\d+/loki/api/v1/tail) {
    proxy_pass          http://localhost:3000$request_uri;
    proxy_set_header    Host                $host;
    proxy_set_header    X-Real-IP           $remote_addr;
    proxy_set_header    X-Forwarded-for     $proxy_add_x_forwarded_for;
    proxy_set_header    X-Forwarded-Proto  "https";
    proxy_set_header    Connection         $connection_upgrade;
    proxy_set_header    Upgrade            $http_upgrade;
}

location / {
    proxy_pass          http://localhost:3000/;
    proxy_set_header    Host                $host;
    proxy_set_header    X-Real-IP           $remote_addr;
    proxy_set_header    X-Forwarded-for     $proxy_add_x_forwarded_for;
    proxy_set_header    X-Forwarded-Proto  "https";
}
```

#### Note

This feature is available only in Grafana v6.3+.

## Log context

When using a search expression as detailed above, you now have the ability to retrieve the context surrounding your filtered results. By choosing the `Show Context` link on the filtered rows, you'll be able to investigate the log messages that came before and after the log message you're interested in.

## Templating

Instead of hardcoding things such as server, application and sensor name in your metric queries, you can use variables in their place. Variables are shown as dropdown select boxes at the top of the dashboard. You can use these dropdown boxes to change the data being displayed in your dashboard.

For more information about templating and template variables, see [Templates and variables \(p. 265\)](#).

## Annotations

You can use any non-metric Loki query as a source for annotations. Log content will be used as annotation text and your log stream labels as tags, so there is no need for additional mapping.



## Microsoft SQL Server

Use the Microsoft SQL Server (MSSQL) data source to query and visualize data from any Microsoft SQL Server 2005 or newer, including Microsoft Azure SQL Database.

### Important

Grafana version 8.0 changes the underlying data structure for data frames for the Microsoft SQL Server, Postgres, and MySQL. As a result, a time series query result is returned in a wide format. For more information, see [Wide format](#) in the Grafana data frames documentation. To make your visualizations work as they did before, you might have to do some manual migrations. One solution is documented on Github at [Postgres/MySQL/MSSQL: Breaking change in v8.0 related to time series queries and ordering of data column](#).

## Adding the data source

1. Open the side menu by choosing the Grafana icon in the top header.
2. In the side menu under the link, **Configuration** you should find a **Data Sources** link.
3. Choose the **+ Add data source** button in the top header.
4. Select **Microsoft SQL Server** from the **Type** dropdown list.

### Data source options

Name	Description
Name	The data source name. This is how you see the data source in panels and queries.
Default	Default data source means that it will be pre-selected for new panels.
Host	The IP address/hostname and optional port of your MSSQL instance. If port is omitted, default 1433 will be used.
Database	Name of your MSSQL database.
User	Database user's login/username.
Password	Database user's password.
Encrypt	This option determines whether or to which extent a secure SSL TCP/IP connection will be negotiated with the server, default <code>false</code> (Grafana v5.4+).
Max open	The maximum number of open connections to the database, default <code>unlimited</code> (Grafana v5.4+).
Max idle	The maximum number of connections in the idle connection pool, default <code>2</code> (Grafana v5.4+).
Max lifetime	The maximum amount of time in seconds a connection may be reused, default <code>14400/4 hours</code> .

### Min time interval

A lower limit for the `$_interval` `$_interval_ms` variables. Recommended to be set to write frequency, for example `1m` if your data is written every minute. This option can also be overridden/

configured in a dashboard panel under data source options. This value **must** be formatted as a number followed by a valid time identifier; for example, 1m (1 minute) or 30s (30 seconds). The following time identifiers are supported.

Identifier	Description
y	Year
M	Month
w	Week
d	Day
h	Hour
m	Minute
s	Second
ms	Millisecond

### Database user permissions

#### Important

The database user that you specify when you add the data source should only be granted SELECT permissions on the specified database and tables you want to query. Grafana does not validate that the query is safe. The query could include any SQL statement. For example, statements such as `DELETE FROM user;` and `DROP TABLE user;` would be run. To protect against this, we highly recommend that you create a specific MSSQL user with restricted permissions.

The following example code shows creating a specific MSSQL user with restricted permissions.

```
CREATE USER grafanareader WITH PASSWORD 'password'
GRANT SELECT ON dbo.YourTable3 TO grafanareader
```

Make sure that the user does not get any unwanted permissions from the public role.

### Known issues

If you're using an older version of Microsoft SQL Server such as 2008 and 2008R2, you might need to disable encryption to be able to connect. If possible, we recommend you to use the latest service pack available for optimal compatibility.

### Query editor

You will find the MSSQL query editor in the metrics tab in the graph, Singlestat, or table panel's edit mode. You enter edit mode by choosing the panel title and then choosing Edit. The editor allows you to define a SQL query to select data to be visualized.

1. Select *Format as Time series* (for use in Graph or Singlestat panel's among others) or *Table* (for use in Table panel among others).
2. This is the actual editor where you write your SQL queries.
3. Show help section for MSSQL below the query editor.
4. Show the SQL query that was run. Will be available first after a successful query has been run.

5. Add an additional query where an additional query editor will be displayed.

## Macros

To simplify syntax and to allow for dynamic parts, such as date range filters, the query can contain macros.

Macro example	Description
<code>\$__time(dateColumn)</code>	Will be replaced by an expression to rename the column to <i>time</i> . For example, <i>dateColumn as time</i> .
<code>\$__timeEpoch(dateColumn)</code>	Will be replaced by an expression to convert a DATETIME column type to Unix timestamp and rename it to <i>time</i> . For example, <i>DATEDIFF(second, "1970-01-01", dateColumn) AS time</i> .
<code>\$__timeFilter(dateColumn)</code>	Will be replaced by a time range filter using the specified column name. For example, <i>dateColumn BETWEEN "2017-04-21T05:01:17Z" AND "2017-04-21T05:06:17Z"</i> .
<code>\$__timeFrom()</code>	Will be replaced by the start of the currently active time selection. For example, <i>"2017-04-21T05:01:17Z"</i> .
<code>\$__timeTo()</code>	Will be replaced by the end of the currently active time selection. For example, <i>"2017-04-21T05:06:17Z"</i> .

Macro example	Description
<code>\$__timeGroup(dateColumn,'5m',[, fillvalue])</code>	Will be replaced by an expression usable in GROUP BY clause. Providing a <i>fillValue</i> of <i>NULL</i> or <i>floating value</i> will automatically fill empty series in time range with that value. For example, <code>CAST(ROUND(DATEDIFF(second, "1970-01-01", time_column)/300.0, 0) as bigint)*300.</code>
<code>\$__timeGroup(dateColumn,'5m', 0)</code>	Same as preceding but with a fill parameter so missing points in that series will be added by grafana and 0 will be used as value.
<code>\$__timeGroup(dateColumn,'5m', NULL)</code>	Same as above but NULL will be used as value for missing points.
<code>\$__timeGroup(dateColumn,'5m', previous)</code>	Same as above but the previous value in that series will be used as fill value if no value has been seen yet NULL will be used (only available in Grafana 5.3+).

The query editor has a **Generated SQL** link that shows up after a query has been run, while in panel edit mode. Choose it and it will expand and show the raw interpolated SQL string that was run.

## Table queries

If the query option is set to **Format as Table** then you can basically do any type of SQL query. The table panel will automatically show the results of whatever columns and rows your query returns.

The following example code shows a database table.

```
CREATE TABLE [event] (
```

```
time_sec bigint,
description nvarchar(100),
tags nvarchar(100),
)
```

```
CREATE TABLE [mssql_types] (
  c_bit bit, c_tinyint tinyint, c_smallint smallint, c_int int, c_bigint bigint, c_money
money, c_smallmoney smallmoney, c_numeric numeric(10,5),
  c_real real, c_decimal decimal(10,2), c_float float,
  c_char char(10), c_varchar varchar(10), c_text text,
  c_nchar nchar(12), c_nvarchar nvarchar(12), c_ntext ntext,
  c_datetime datetime, c_datetime2 datetime2, c_smalldatetime smalldatetime, c_date date,
  c_time time, c_datetimeoffset datetimeoffset
)

INSERT INTO [mssql_types]
SELECT
  1, 5, 20020, 980300, 1420070400, '$20000.15', '£2.15', 12345.12,
  1.11, 2.22, 3.33,
  'char10', 'varchar10', 'text',
  N'#nchar12#', N'#nvarchar12#', N'#text#',
  GETDATE(), CAST(GETDATE() AS DATETIME2), CAST(GETDATE() AS SMALLDATETIME), CAST(GETDATE()
AS DATE), CAST(GETDATE() AS TIME), SWITCHOFFSET(CAST(GETDATE() AS DATETIMEOFFSET),
'-07:00')
```

The following example code shows a query.

```
SELECT * FROM [mssql_types]
```

You can control the name of the Table panel columns by using regular AS SQL column selection syntax, as shown in the following example code.

```
SELECT
  c_bit as [column1], c_tinyint as [column2]
FROM
  [mssql_types]
```

The resulting table panel:

## Time series queries

If you set **Format as** to **Time series**, for use in Graph panel for example, the query must have a column named `time` that returns either a SQL datetime or any numeric data type representing Unix epoch in seconds. You may return a column named `metric` that is used as metric name for the value column. Any column except `time` and `metric` is treated as a value column. If you omit the `metric` column, the name of the value column will be the metric name. You may select multiple value columns, each will have its name as `metric`. If you return multiple value columns and a column named `metric` then this column is used as prefix for the series name.

Result sets of time series queries must be sorted by time.

The following example code shows a database table.

```
CREATE TABLE [event] (
  time_sec bigint,
  description nvarchar(100),
  tags nvarchar(100),
)
```

```
)
```

```
CREATE TABLE metric_values (  
    time datetime,  
    measurement nvarchar(100),  
    valueOne int,  
    valueTwo int,  
)  
  
INSERT metric_values (time, measurement, valueOne, valueTwo) VALUES('2018-03-15 12:30:00',  
    'Metric A', 62, 6)  
INSERT metric_values (time, measurement, valueOne, valueTwo) VALUES('2018-03-15 12:30:00',  
    'Metric B', 49, 11)  
...  
INSERT metric_values (time, measurement, valueOne, valueTwo) VALUES('2018-03-15 13:55:00',  
    'Metric A', 14, 25)  
INSERT metric_values (time, measurement, valueOne, valueTwo) VALUES('2018-03-15 13:55:00',  
    'Metric B', 48, 10)
```

The following example code shows one value and one metric column.

```
SELECT  
    time,  
    valueOne,  
    measurement as metric  
FROM  
    metric_values  
WHERE  
    $__timeFilter(time)  
ORDER BY 1
```

When the preceding query is used in a graph panel, it will produce two series named `Metric A` and `Metric B` with the values `valueOne` and `valueTwo` plotted over time.

The following example code shows multiple value columns.

```
SELECT  
    time,  
    valueOne,  
    valueTwo  
FROM  
    metric_values  
WHERE  
    $__timeFilter(time)  
ORDER BY 1
```

When the preceding query is used in a graph panel, it will produce two series named `Metric A` and `Metric B` with the values `valueOne` and `valueTwo` plotted over time.

The following example code shows using the `$__timeGroup` macro.

```
SELECT  
    $__timeGroup(time, '3m') as time,  
    measurement as metric,  
    avg(valueOne)  
FROM  
    metric_values
```

```
WHERE
  $__timeFilter(time)
GROUP BY
  $__timeGroup(time, '3m'),
  measurement
ORDER BY 1
```

When the previous query is used in a graph panel, it will produce two series named `Metric A` and `Metric B` with the values `valueOne` and `valueTwo` plotted over `time`. Any two series lacking a value in a three-minute window will render a line between those two lines. You'll notice that the graph to the right never goes down to zero.

The following example code shows using the `$__timeGroup` macro with `fill` parameter set to zero.

```
SELECT
  $__timeGroup(time, '3m', 0) as time,
  measurement as metric,
  sum(valueTwo)
FROM
  metric_values
WHERE
  $__timeFilter(time)
GROUP BY
  $__timeGroup(time, '3m'),
  measurement
ORDER BY 1
```

When this query is used in a graph panel, the result is two series named `Metric A` and `Metric B` with a sum of `valueTwo` plotted over `time`. Any series lacking a value in a 3 minute window will have a value of zero which you'll see rendered in the graph to the right.

## Templating

Instead of hardcoding things such as server, application and sensor name in your metric queries you can use variables in their place. Variables are shown as dropdown select boxes at the top of the dashboard. You can use these dropdown boxes to change the data being displayed in your dashboard.

For more information about templating and template variables, see [Templates and variables \(p. 265\)](#).

### Query variable

If you add a template variable of the type `query`, you can write a MSSQL query that can return things such as measurement names, key names or key values that are shown as a dropdown select box.

For example, you can have a variable that contains all values for the `hostname` column in a table if you specify a query such as this in the templating variable `Query` setting.

```
SELECT hostname FROM host
```

A query can return multiple columns and Grafana will automatically create a list from them. For example, the following query will return a list with values from `hostname` and `hostname2`.

```
SELECT [host].[hostname], [other_host].[hostname2] FROM host JOIN other_host ON [host].[city] = [other_host].[city]
```

Another option is a query that can create a key/value variable. The query should return two columns that are named `__text` and `__value`. The `__text` column value should be unique (if it is not unique then

the first value is used). The options in the dropdown list will have a text and value that allow you to have a friendly name as text and an id as the value. An example query with `hostname` as the text and `id` as the value:

```
SELECT hostname __text, id __value FROM host
```

You can also create nested variables. For example, if you had another variable named `region`. Then you could have the hosts variable only show hosts from the current selected region with a query such as this (if `region` is a multi-value variable, then use the `IN` comparison operator rather than `=` to match against multiple values).

```
SELECT hostname FROM host WHERE region IN ($region)
```

### Using variables in queries

#### Note

Template variable values are only quoted when the template variable is a multi-value.

If the variable is a multi-value variable then use the `IN` comparison operator rather than `=` to match against multiple values.

There are two syntaxes:

`$<varname>` Example with a template variable named `hostname`:

```
SELECT
  atimestamp time,
  aint value
FROM table
WHERE $__timeFilter(atimestamp) and hostname in($hostname)
ORDER BY atimestamp
```

`[[varname]]` Example with a template variable named `hostname`:

```
SELECT
  atimestamp as time,
  aint as value
FROM table
WHERE $__timeFilter(atimestamp) and hostname in([[hostname]])
ORDER BY atimestamp
```

### Turning off quoting for multi-value variables

Grafana automatically creates a quoted, comma-separated string for multi-value variables. For example, if `server01` and `server02` are selected then it will be formatted as: `'server01', 'server02'`. To turn off quoting, use the `csv` formatting option for variables.

```
${servers:csv}
```

For more information about variable formatting options, see [Templates and variables \(p. 265\)](#).

## Annotations

You can use annotations to overlay rich event information on top of graphs. You add annotation queries via the Dashboard menu / Annotations view. For more information, see [Annotations \(p. 238\)](#).



## Columns:

Name	Description
time	The name of the date/time field. Could be a column with a native SQL date/time data type or epoch value.
timeend	Optional name of the end date/time field. Could be a column with a native SQL date/time data type or epoch value.
text	Event description field.
tags	Optional field name to use for event tags as a comma separated string.

The following example code shows database tables.

```
CREATE TABLE [events] (
  time_sec bigint,
  description nvarchar(100),
  tags nvarchar(100),
)
```

We also use the database table defined in [Time series queries \(p. 102\)](#).

The following example code shows a query using a time column with epoch values.

```
SELECT
  time_sec as time,
  description as [text],
  tags
FROM
  [events]
WHERE
  $__unixEpochFilter(time_sec)
ORDER BY 1
```

The following example code shows a region query using time and timeend columns with epoch values.

```
SELECT
  time_sec as time,
  time_end_sec as timeend,
  description as [text],
  tags
FROM
  [events]
WHERE
  $__unixEpochFilter(time_sec)
ORDER BY 1
```

The following example code shows a query using a time column of native SQL date/time data type.

```
SELECT
```

```
time,
measurement as text,
convert(varchar, valueOne) + ',' + convert(varchar, valueTwo) as tags
FROM
metric_values
WHERE
$__timeFilter(time_column)
ORDER BY 1
```

## Stored procedure support

Stored procedures have been verified to work. However, there might be edge cases where it won't work as you would expect. Stored procedures should be supported in table, time series, and annotation queries as long as you use the same naming of columns and return data in the same format as described previously in the respective sections.

Macro functions will not work inside a stored procedure.

## Examples

For the following examples, the database table is defined in Time series queries. Let's say that you want to visualize four series in a graph panel, such as all combinations of columns `valueOne`, `valueTwo` and `measurement`. The graph panel to the right visualizes what we want to achieve. To solve this, you must use two queries:

The following example code shows the first query.

```
SELECT
  $__timeGroup(time, '5m') as time,
  measurement + ' - value one' as metric,
  avg(valueOne) as valueOne
FROM
  metric_values
WHERE
  $__timeFilter(time)
GROUP BY
  $__timeGroup(time, '5m'),
  measurement
ORDER BY 1
```

The following example code shows the second query.

```
SELECT
  $__timeGroup(time, '5m') as time,
  measurement + ' - value two' as metric,
  avg(valueTwo) as valueTwo
FROM
  metric_values
GROUP BY
  $__timeGroup(time, '5m'),
  measurement
ORDER BY 1
```

## Stored procedure using time in epoch format

You can define a stored procedure that will return all data that you need to render four series in a graph panel such as above. In this case, the stored procedure accepts two parameters, `@from` and `@to`, of `int` data types, which should be a time range (from-to) in epoch format which will be used to filter the data to return from the stored procedure.

This mimics the `$__timeGroup(time, '5m')` in the select and group by expressions, and that's why numerous lengthy expressions are needed. These could be extracted to MSSQL functions, if wanted.

```
CREATE PROCEDURE sp_test_epoch(
    @from int,
    @to int
) AS
BEGIN
    SELECT
        cast(cast(DATEDIFF(second, {d '1970-01-01'}, DATEADD(second,
DATEDIFF(second,GETDATE(),GETUTCDATE()), time))/600 as int)*600 as int) as time,
        measurement + ' - value one' as metric,
        avg(valueOne) as value
    FROM
        metric_values
    WHERE
        time >= DATEADD(s, @from, '1970-01-01') AND time <= DATEADD(s, @to, '1970-01-01')
    GROUP BY
        cast(cast(DATEDIFF(second, {d '1970-01-01'}, DATEADD(second,
DATEDIFF(second,GETDATE(),GETUTCDATE()), time))/600 as int)*600 as int),
        measurement
    UNION ALL
    SELECT
        cast(cast(DATEDIFF(second, {d '1970-01-01'}, DATEADD(second,
DATEDIFF(second,GETDATE(),GETUTCDATE()), time))/600 as int)*600 as int) as time,
        measurement + ' - value two' as metric,
        avg(valueTwo) as value
    FROM
        metric_values
    WHERE
        time >= DATEADD(s, @from, '1970-01-01') AND time <= DATEADD(s, @to, '1970-01-01')
    GROUP BY
        cast(cast(DATEDIFF(second, {d '1970-01-01'}, DATEADD(second,
DATEDIFF(second,GETDATE(),GETUTCDATE()), time))/600 as int)*600 as int),
        measurement
    ORDER BY 1
END
```

Then you can use the following query for your graph panel.

```
DECLARE
    @from int = $__unixEpochFrom(),
    @to int = $__unixEpochTo()

EXEC dbo.sp_test_epoch @from, @to
```

### Stored procedure using time in datetime format

You can define a stored procedure that will return all data that you need to render four series in a graph panel such as above. In this case, the stored procedure accepts two parameters, `@from` and `@to`, of datetime data types, which should be a time range (from-to) that will be used to filter the data to return from the stored procedure.

This mimics the `$__timeGroup(time, '5m')` in the select and group by expressions, and that's why numerous lengthy expressions are needed. These could be extracted to MSSQL functions, if wanted.

```
CREATE PROCEDURE sp_test_datetime(
    @from datetime,
    @to datetime
```

```
) AS
BEGIN
  SELECT
    cast(cast(DATEDIFF(second, {d '1970-01-01'}, time)/600 as int)*600 as int) as time,
    measurement + ' - value one' as metric,
    avg(valueOne) as value
  FROM
    metric_values
  WHERE
    time >= @from AND time <= @to
  GROUP BY
    cast(cast(DATEDIFF(second, {d '1970-01-01'}, time)/600 as int)*600 as int),
    measurement
  UNION ALL
  SELECT
    cast(cast(DATEDIFF(second, {d '1970-01-01'}, time)/600 as int)*600 as int) as time,
    measurement + ' - value two' as metric,
    avg(valueTwo) as value
  FROM
    metric_values
  WHERE
    time >= @from AND time <= @to
  GROUP BY
    cast(cast(DATEDIFF(second, {d '1970-01-01'}, time)/600 as int)*600 as int),
    measurement
  ORDER BY 1
END
```

Then you can use the following query for your graph panel.

```
DECLARE
  @from datetime = $__timeFrom(),
  @to datetime = $__timeTo()

EXEC dbo.sp_test_datetime @from, @to
```

## Alerting

Time series queries should work in alerting conditions. Table formatted queries are not yet supported in alert rule conditions.

## MySQL

Add the MySQL data source to be able to query and visualize data from a MySQL compatible database.

### Important

Grafana version 8.0 changes the underlying data structure for data frames for the MySQL, Postgres, and Microsoft SQL Server data sources. As a result, a time series query result is returned in a wide format. For more information, see [Wide format](#) in the Grafana data frames documentation.

To make your visualizations work as they did before, you might have to do some manual migrations. One solution is documented on Github at [Postgres/MySQL/MSSQL: Breaking change in v8.0 related to time series queries and ordering of data column](#).

## Adding the data source

1. Open the side menu by choosing the Grafana icon in the top header.
2. In the side menu under the **Dashboards** link, you should find a link named **Data Sources**.
3. Choose the **+ Add data source** button in the top header.

4. Select **MySQL** from the **Type** dropdown list.

### Data source options

Name	Description
Name	The data source name. This is how you see the data source in panels and queries.
Default	Default data source means that it will be pre-selected for new panels.
Host	The IP address/hostname and optional port of your MySQL instance.
Database	Name of your MySQL database.
User	Database user's login/username.
Password	Database user's password.
Max open	The maximum number of open connections to the database, default unlimited (Grafana v5.4+).
Max idle	The maximum number of connections in the idle connection pool, default 2 (Grafana v5.4+).
Max lifetime	The maximum amount of time in seconds a connection may be reused, default 14400/4 hours. This should always be lower than configured <a href="#">wait_timeout</a> in MySQL (Grafana v5.4+).

### Min time interval

A lower limit for the `$_interval` `$_interval_ms` variables. Recommended to be set to write frequency, for example `1m` if your data is written every minute. This option can also be overridden/configured in a dashboard panel under data source options. This value **must** be formatted as a number followed by a valid time identifier; for example, `1m` (1 minute) or `30s` (30 seconds). The following time identifiers are supported.

Identifier	Description
y	Year
M	Month
w	Week
d	Day
h	Hour
m	Minute
s	Second

Identifier	Description
ms	Millisecond

## Database user permissions

### Important

The database user that you specify when you add the data source should be granted only `SELECT` permissions on the specified database and tables that you want to query. Grafana does not validate that the query is safe. The query could include any SQL statement. For example, statements such as `USE otherdb;` and `DROP TABLE user;` would be run. To protect against this, we strongly recommend that you create a specific MySQL user with restricted permissions.

The following code example shows creating a specific MySQL user with restricted permissions.

```
CREATE USER 'grafanaReader' IDENTIFIED BY 'password';  
GRANT SELECT ON mydatabase.mytable TO 'grafanaReader';
```

To grant access to more databases and tables, you can use wildcard characters (\*) in place of database or table if you want.

## Query editor

You find the MySQL query editor in the metrics tab in a panel's edit mode. You enter edit mode by choosing the panel title, then **Edit**.

The query editor has a **Generated SQL** link that shows up after a query has been run, while in panel edit mode. Choose it, and it will expand and show the raw interpolated SQL string that was run.

### Select table, time column, and metric column (FROM)

When you enter edit mode for the first time or add a new query, Grafana will try to prefill the query builder with the first table that has a timestamp column and a numeric column.

In the FROM field, Grafana will suggest tables that are in the configured database. To select a table or view in another database that your database user has access to, you can manually enter a fully qualified name (database.table) such as `otherDb.metrics`.

The Time column field refers to the name of the column holding your time values. Selecting a value for the Metric column field is optional. If a value is selected, the Metric column field will be used as the series name.

The metric column suggestions will only contain columns with a text data type (text, tinytext, mediumtext, longtext, varchar, char). If you want to use a column with a different data type as metric column, you may enter the column name with a cast: `CAST(numericColumn as CHAR)`. You may also enter arbitrary SQL expressions in the metric column field that evaluate to a text data type such as `CONCAT(column1, " ", CAST(numericColumn as CHAR))`.

### Columns and aggregation functions (SELECT)

In the `SELECT` row, you can specify what columns and functions you want to use. In the column field, you may write arbitrary expressions instead of a column name such as `column1 * column2 / column3`.

If you use aggregate functions, you must group your result set. The editor will automatically add a `GROUP BY time` if you add an aggregate function.

You may add further value columns by choosing the plus button and selecting `Column` from the menu. Multiple value columns will be plotted as separate series in the graph panel.

### Filtering data (WHERE)

To add a filter, choose the plus icon to the right of the `WHERE` condition. You can remove filters by choosing on the filter and selecting **Remove**. A filter for the current selected time range is automatically added to new queries.

### Group By

To group by time or any other columns, choose the plus icon at the end of the `GROUP BY` row. The suggestion dropdown list will only show text columns of your currently selected table but you may manually enter any column. You can remove the group by choosing on the item and then selecting **Remove**.

If you add any grouping, all selected columns must have an aggregate function applied. The query builder will automatically add aggregate functions to all columns without aggregate functions when you add groupings.

### Gap filling

Grafana can fill in missing values when you group by time. The time function accepts two arguments. The first argument is the time window that you want to group by, and the second argument is the value you want Grafana to fill missing items with.

### Text editor mode (raw)

You can switch to the raw query editor mode by choosing the hamburger icon and selecting **Switch editor mode** or by choosing **Edit SQL** below the query.

#### Note

If you use the raw query editor, be sure that your query at minimum has `ORDER BY time` and a filter on the returned time range.

### Macros

To simplify syntax and to allow for dynamic parts, such as date range filters, the query can contain macros.

Macro example	Description
<code>\$__time(dateColumn)</code>	Will be replaced by an expression to convert to a UNIX timestamp and rename the column to <code>time_sec</code> ; for example, <code>UNIX_TIMESTAMP(dateColumn) as time_sec</code> .
<code>\$__timeEpoch(dateColumn)</code>	Will be replaced by an expression to convert to a UNIX timestamp and rename the column to <code>time_sec</code> ; for example, <code>UNIX_TIMESTAMP(dateColumn) as time_sec</code> .

Macro example	Description
<code>\$__timeFilter(dateColumn)</code>	Will be replaced by a time range filter using the specified column name. For example, <i>dateColumn BETWEEN FROM_UNIXTIME(1494410783) AND FROM_UNIXTIME(1494410983).</i>
<code>\$__timeFrom()</code>	Will be replaced by the start of the currently active time selection. For example, <i>FROM_UNIXTIME(1494410783).</i>
<code>\$__timeTo()</code>	Will be replaced by the end of the currently active time selection. For example, <i>FROM_UNIXTIME(1494410983).</i>
<code>\$__timeGroup(dateColumn, '5m')</code>	Will be replaced by an expression usable in GROUP BY clause. For example, <i>cast(cast(UNIX_TIMESTAMP(dateColumn) as signed)300 as signed),*</i>
<code>\$__timeGroup(dateColumn, '5m', 0)</code>	Same as the previous row, but with a fill parameter so missing points in that series will be added by grafana and 0 will be used as value.
<code>\$__timeGroup(dateColumn, '5m', NULL)</code>	Same as above but NULL will be used as value for missing points.



Macro example	Description
<code>\$__timeGroup(dateColumn, '5m', previous)</code>	Same as above but the previous value in that series will be used as fill value if no value has been seen yet NULL will be used (only available in Grafana 5.3+).
<code>\$__timeGroupAlias(dateColumn, '5m')</code>	Will be replaced identical to <code>\$__timeGroup</code> but with an added column alias (available only in Grafana 5.3+).
<code>\$__unixEpochFilter(dateColumn)</code>	Will be replaced by a time range filter using the specified column name with times represented as Unix timestamp. For example, <code>dateColumn &gt; 1494410783 AND dateColumn &lt; 1494497183</code> .
<code>\$__unixEpochFrom()</code>	Will be replaced by the start of the currently active time selection as Unix timestamp. For example, <code>1494410783</code> .
<code>\$__unixEpochTo()</code>	Will be replaced by the end of the currently active time selection as Unix timestamp. For example, <code>1494497183</code> .

Macro example	Description
<code>\$__unixEpochNanoFilter(dateColumn)</code>	Will be replaced by a time range filter using the specified column name with times represented as nanosecond timestamp. For example, dateColumn > 1494410783152415214 AND dateColumn < 1494497183142514872.
<code>\$__unixEpochNanoFrom()</code>	Will be replaced by the start of the currently active time selection as nanosecond timestamp. For example, 1494410783152415214.
<code>\$__unixEpochNanoTo()</code>	Will be replaced by the end of the currently active time selection as nanosecond timestamp. For example, 1494497183142514872.
<code>\$__unixEpochGroup(dateColumn, "5m", [fillmode])</code>	Same as <code>\$__timeGroup</code> but for times stored as Unix timestamp (available only in Grafana 5.3+).
<code>\$__unixEpochGroupAlias(dateColumn, "5m", [fillmode])`</code>	Same as above but also adds a column alias (available only in Grafana 5.3+).

The query editor has a **Generated SQL** link that shows up after a query has run, while in panel edit mode. Choose it, and it will expand and show the raw interpolated SQL string that was run.

## Table queries

If the **Format as** query option is set to **Table**, you can basically do any type of SQL query. The table panel will automatically show the results of whatever columns and rows your query returns.

The following code shows an example query.

```
SELECT
  title as 'Title',
  user.login as 'Created By' ,
  dashboard.created as 'Created On'
FROM dashboard
INNER JOIN user on user.id = dashboard.created_by
WHERE $__timeFilter(dashboard.created)
```

You can control the name of the Table panel columns by using regular as SQL column selection syntax.

## Time series queries

If you set **Format as** to **Time series**, for use in a graph panel for example, the query must return a column named `time` that returns either a SQL datetime or any numeric data type representing Unix epoch. Any column except `time` and `metric` is treated as a value column. You may return a column named `metric` that is used as metric name for the value column. If you return multiple value columns and a column named `metric`, this column is used as prefix for the series name (available only in Grafana 5.3+).

Result sets of time series queries must be sorted by time.

The following code example shows the `metric` column.

```
SELECT
  $__timeGroup(time_date_time, '5m'),
  min(value_double),
  'min' as metric
FROM test_data
WHERE $__timeFilter(time_date_time)
GROUP BY time
ORDER BY time
```

The following code example shows using the `fill` parameter in the `$__timeGroup` macro to convert null values to be zero instead.

```
SELECT
  $__timeGroup(createdAt, '5m', 0),
  sum(value_double) as value,
  measurement
FROM test_data
WHERE
  $__timeFilter(createdAt)
GROUP BY time, measurement
ORDER BY time
```

The following code example shows multiple columns.

```
SELECT
  $__timeGroup(time_date_time, '5m'),
  min(value_double) as min_value,
  max(value_double) as max_value
FROM test_data
WHERE $__timeFilter(time_date_time)
GROUP BY time
ORDER BY time
```

Currently, there is no support for a dynamic group by time based on time range and panel width. This may be supported in the future.

## Templating

Instead of hardcoding things such as server, application and sensor name in your metric queries you can use variables in their place. Variables are shown as dropdown select boxes at the top of the dashboard. You can use these dropdown boxes to change the data being displayed in your dashboard.

For more information about templating and template variables, see [Templates \(p. 266\)](#).

### Query variable

If you add a template variable of the type `Query`, you can write a MySQL query that can return things such as measurement names, key names, or key values that are shown as a dropdown select box.

For example, you can have a variable that contains all values for the `hostname` column in a table if you specify a query such as this in the templating variable `Query` setting.

```
SELECT hostname FROM my_host
```

A query can return multiple columns and Grafana will automatically create a list from them. For example, the following query will return a list with values from `hostname` and `hostname2`.

```
SELECT my_host.hostname, my_other_host.hostname2 FROM my_host JOIN my_other_host ON  
my_host.city = my_other_host.city
```

To use time range dependent macros such as `$__timeFilter(column)` in your query, the refresh mode of the template variable must be set to *On Time Range Change*.

```
SELECT event_name FROM event_log WHERE $__timeFilter(time_column)
```

Another option is a query that can create a key/value variable. The query should return two columns that are named `__text` and `__value`. The `__text` column value should be unique (if it is not unique, the first value is used). The options in the dropdown list will have a text and value so that you can have a friendly name as text and an ID as the value.

The following code example shows a query with `hostname` as the text and `id` as the value.

```
SELECT hostname AS __text, id AS __value FROM my_host
```

You can also create nested variables. For example, if you had another variable named `region`. Then you could have the hosts variable show only hosts from the current selected region with a query such as this (if `region` is a multi-value variable then use the `IN` comparison operator rather than `=` to match against multiple values).

```
SELECT hostname FROM my_host WHERE region IN($region)
```

### Using `__searchFilter` to filter results in Query Variable

Using `__searchFilter` in the query field will filter the query result based on what the user types in the dropdown select box. When nothing has been entered by the user, the default value for `__searchFilter` is `%`.

### Note

Important that you surround the `__searchFilter` expression with quotes as Grafana does not do this for you.

The following example shows how to use `__searchFilter` as part of the query field to enable searching for hostname while the user types in the dropdown select box.

```
SELECT hostname FROM my_host WHERE hostname LIKE '$__searchFilter'
```

### Using variables in queries

From Grafana 4.3.0 to 4.6.0, template variables are always quoted automatically so if it is a string value do not wrap them in quotes in where clauses.

From Grafana 4.7.0, template variable values are only quoted when the template variable is a multi-value.

If the variable is a multi-value variable, use the `IN` comparison operator rather than `=` to match against multiple values.

There are two syntaxes:

`$<varname>` Example with a template variable named hostname:

```
SELECT
  UNIX_TIMESTAMP(atimestamp) as time,
  aint as value,
  avarchar as metric
FROM my_table
WHERE $__timeFilter(atimestamp) and hostname in($hostname)
ORDER BY atimestamp ASC
```

`[ [varname] ]` Example with a template variable named hostname:

```
SELECT
  UNIX_TIMESTAMP(atimestamp) as time,
  aint as value,
  avarchar as metric
FROM my_table
WHERE $__timeFilter(atimestamp) and hostname in([[hostname]])
ORDER BY atimestamp ASC
```

### Turning off quoting for multi-value variables

Grafana automatically creates a quoted, comma-separated string for multi-value variables. For example: if `server01` and `server02` are selected then it will be formatted as: `'server01', 'server02'`. To turn off quoting, use the `csv` formatting option for variables.

`${servers:csv}`

For more information about variable formatting options, see [Advanced variable format options \(p. 276\)](#).

### Annotations

You can use annotations to overlay rich event information on top of graphs. You add annotation queries via the Dashboard menu / Annotations view. For more information, see .

The following example code shows a query using a time column with epoch values.

```
SELECT
    epoch_time as time,
    metric1 as text,
    CONCAT(tag1, ',', tag2) as tags
FROM
    public.test_data
WHERE
    $__unixEpochFilter(epoch_time)
```

The following example code shows a region query using time and timeend columns with epoch values.

**Note**

Available only in Grafana v6.6+.

```
SELECT
    epoch_time as time,
    epoch_timeend as timeend,
    metric1 as text,
    CONCAT(tag1, ',', tag2) as tags
FROM
    public.test_data
WHERE
    $__unixEpochFilter(epoch_time)
```

The following example code shows a query using a time column of native SQL date/time data type.

```
SELECT
    native_date_time as time,
    metric1 as text,
    CONCAT(tag1, ',', tag2) as tags
FROM
    public.test_data
WHERE
    $__timeFilter(native_date_time)
```

Name	Description
time	The name of the date/time field. Could be a column with a native SQL date/time data type or epoch value.
timeend	Optional name of the end date/time field. Could be a column with a native SQL date/time data type or epoch value.
text	Event description field.
tags	Optional field name to use for event tags as a comma-separated string.

## Alerting

Time series queries should work in alerting conditions. Table formatted queries are not yet supported in alert rule conditions.

## OpenSearch

With Amazon Managed Grafana, you can add OpenSearch as a data source. You can do many types of simple or complex OpenSearch queries to visualize logs or metrics stored in OpenSearch. You can also annotate your graphs with log events stored in OpenSearch.

### Add OpenSearch as a data source

1. Open the side menu by choosing the Grafana icon in the top header.
2. In the side menu under the **Dashboards** link, you should find the named **Data Sources** link.
3. Choose the **+ Add data source** button in the top header.
4. Select **Opensearch** from the **Type** dropdown list.

#### Note

If you're not seeing the **Data Sources** link in your side menu, it means that your current user does not have the **Admin** role.

Name	Description
Name	The data source name. This is how you see the data source in panels and queries.
Default	Default data source means that it will be pre-selected for new panels.
Url	The HTTP protocol, IP, and port of your OpenSearch server.
Access	Server (default) = URL must be accessible from the Grafana backend/server. Browser = URL must be accessible from the browser.

Access mode controls how requests to the data source will be handled. Server should be the preferred way if nothing else is stated.

#### Server access mode (default)

All requests are made from the browser to Grafana backend or server, which forwards the requests to the data source, circumventing possible Cross-Origin Resource Sharing (CORS) requirements. If you select this access mode, the URL must be accessible from the Grafana backend or server.

#### Browser (direct) access

All requests are made from the browser directly to the data source and may be subject to CORS requirements. If you select this access mode, the URL must be accessible from the browser.

If you select browser access, you must update your OpenSearch configuration to allow other domains to access OpenSearch from the browser. You do this by specifying these two options in your **OpenSearch.yml** config file.

```
http.cors.enabled: true
http.cors.allow-origin: "*"

```

#### Index settings

Here you can specify a default for the `time` field and specify the name of your OpenSearch index. You can use a time pattern for the index name or a wildcard character.

## OpenSearch version

Be sure to specify your OpenSearch version in the version selection dropdown list. This is important because there are differences in how queries are composed. Currently the versions available are 2.x, 5.x, 5.6+, 6.0+, or 7.0+. The value 5.6+ means version 5.6 or higher, but lower than 6.0. The value 6.0+ means version 6.0 or higher, but lower than 7.0. Finally, 7.0+ means version 7.0 or higher, but lower than 8.0.

## Min time interval

A lower limit for the auto group by time interval. Recommended to be set to write frequency; for example, 1m if your data is written every minute. This option can also be overridden/configured in a dashboard panel under data source options. This value **must** be formatted as a number followed by a valid time identifier; for example, 1m (1 minute) or 30s (30 seconds). The following time identifiers are supported.

Identifier	Description
y	Year
M	Month
w	Week
d	Day
h	Hour
m	Minute
s	Second
ms	Millisecond

## Logs

Two parameters, `Message field name` and `Level field name`, can optionally be configured from the data source settings page that determine which fields will be used for log messages and log levels when visualizing logs in [Explore \(p. 254\)](#).

For example, if you use a default setup of Filebeat for shipping logs to OpenSearch, the following configuration should work.

- **Message field name:** message
- **Level field name:** fields.level

## Data links

Data links create a link from a specified field that can be accessed in logs view in Explore.

Each data link configuration consists of the following:

- **Field** – Name of the field used by the data link.
- **URL/query** – If the link is external, then enter the full link URL. If the link is internal link, then this input serves as query for the target data source. In both cases, you can interpolate the value from the field with `${__value.raw }` macro.
- **Internal link** – Select this if the link is internal or external. If the link is internal, a data source selector allows you to select the target data source. Only tracing data sources are supported.



## Using the Opensearch data source

### Metric query editor

The OpenSearch query editor allows you to select multiple metrics and group by multiple terms or filters. Use the plus and minus icons to the right to add/remove metrics or group by clauses. Some metrics and group by clauses have options. Choose the option text to expand the row to view and edit metric or group by options.

### Series naming and alias patterns

You can control the name for time series via the `Alias` input field.

Pattern	Description
<code>{{term fieldname}}</code>	Replaced with value of a term Group By.
<code>{{metric}}</code>	Replaced with metric name (ex. Average, Min, Max).
<code>{{field}}</code>	Replaced with the metric field name.

### Pipeline metrics

Some metric aggregations are called pipeline aggregations; for example, *Moving Average* and *Derivative*. OpenSearch pipeline metrics require another metric to be based on. Use the eye icon next to the metric to hide metrics from appearing in the graph. This is useful for metrics you only have in the query for use in a pipeline metric.

### Templating

Instead of hardcoding things such as server, application, and sensor name in your metric queries you can use variables in their place. Variables are shown as dropdown select boxes at the top of the dashboard. You can use these dropdown boxes to change the data being displayed in your dashboard.

For more information about templating and template variables, see [Templates and variables \(p. 265\)](#).

### Query variable

The Opensearch data source supports two types of queries you can use in the *Query* field of *Query* variables. The query is written using a custom JSON string.

Query	Description
<code>{"find": "fields", "type": "keyword"}</code>	Returns a list of field names with the index type keyword.
<code>{"find": "terms", "field": "@hostname", "size": 1000}</code>	Returns a list of values for a field using term aggregation. Query will use current dashboard time range as time range for query.
<code>{"find": "terms", "field": "@hostname", "query": '&lt;.lucene query&gt;'}</code>	Returns a list of values for a field using term aggregation and a specified Lucene query filter. Query will use current dashboard time range as time range for query.

There is a default size limit of 500 on terms queries. To set a custom limit, set the `size` property in your query. You can use other variables inside the query. The following code example shows the query definition for a variable named `$host`.

```
{ "find": "terms", "field": "@hostname", "query": "@source:$source" }
```

In the previous example, we use another variable named `$source` inside the query definition. Whenever you change, via the dropdown list, the current value of the `$source` variable, it initiates an update of the `$host` variable. After the update, the `$host` variable contains only hostnames filtered by in this case the `@source` document property.

These queries by default return results in term order (which can then be sorted alphabetically or numerically as for any variable). To produce a list of terms sorted by doc count (a top-N values list), add an `orderBy` property of `doc_count`. This automatically selects a descending sort. Using `asc` with `doc_count` (a bottom-N list) can be done by setting `order: "asc"`, but it is discouraged because it increases the error on document counts. To keep terms in the doc count order, set the variable's **Sort** dropdown list to **Disabled**. Alternatively, you might alternatively still want to use **Alphabetical** to re-sort them.

```
{ "find": "terms", "field": "@hostname", "orderBy": "doc_count" }
```

## Using variables in queries

There are two syntaxes:

- `$<varname>` Example: `@hostname:$hostname`
- `[ [varname] ]` Example: `@hostname:[hostname]`

Why two ways? The first syntax is easier to read and write, but it does not allow you to use a variable in the middle of a word. When the *Multi-value* or *Include all value* options are enabled, Grafana converts the labels from plaintext to a Lucene-compatible condition.

In the previous example, we have a lucene query that filters documents based on the `@hostname` property using a variable named `$hostname`. It is also using a variable in the *Terms* group by field input box. This allows you to use a variable to quickly change how the data is grouped.

## Annotations

Annotations allow you to overlay rich event information on top of graphs. You add annotation queries via the Dashboard menu or Annotations view. Grafana can query any OpenSearch index for annotation events. For more information, see [Annotations \(p. 238\)](#).

Name	Description
Query	You can keep the search query blank or specify a Lucene query.
Time	The name of the time field; must be date field.
Time End	Optional name of the time end field must be date field. If set, annotations will be marked as a region between time and time-end.
Text	Event description field.
Tags	Optional field name to use for event tags (can be an array or a CSV string).

## Querying logs

Querying and displaying log data from OpenSearch is available in Explore. To display your logs, select the OpenSearch data source, and then optionally enter a Lucene query. For more information, see [Explore \(p. 254\)](#).

## Log queries

After the result is returned, the log panel shows a list of log rows and a bar chart where the x-axis shows the time and the y-axis shows the frequency or count.

## Filtering log messages

Optionally, enter a Lucene query into the query field to filter the log messages. For example, using a default Filebeat setup, you should be able to use `fields.level:error` to show only error log messages.

# OpenTSDB

Amazon Managed Grafana ships with advanced support for OpenTSDB.

## Adding the data source

1. Open the side menu by choosing the Grafana icon in the top header.
2. In the side menu under the **Dashboards** link, you should find a **Data Sources** link.
3. Choose the **+ Add data source** button in the top header.
4. Select **OpenTSDB** from the **Type** dropdown list.

### Note

If you don't see the **Data Sources** link in your side menu, it means that your current user does not have the `Admin` role.

Name	Description
Name	The data source name. This is how you see the data source in panels and queries.
Default	Default data source means that it will be pre-selected for new panels.
Url	The HTTP protocol, ip and port of your opentsdb server (default port is usually 4242).
Access	Server (default) = URL must be accessible from the Grafana backend/server. Browser = URL must be accessible from the browser.
Version	Version = opentsdb version, either <=2.1 or 2.2.
Resolution	Metrics from opentsdb may have data points with either second or millisecond resolution.

## Query editor

Open a graph in edit mode by choose the title. Query editor will differ if the data source has version <=2.1 or = 2.2. In the former version, only tags can be used to query OpenTSDB. But in the latter version, filters as well as tags can be used to query opentsdb. Fill Policy is also introduced in OpenTSDB 2.2.

### Note

While using the OpenTSDB 2.2 data source, make sure you use either Filters or Tags as they are mutually exclusive. If used together, might give you weird results.

## Using autocomplete suggestions

As soon as you start typing metric names, tag names and tag values , you should see highlighted auto complete suggestions for them. The autocomplete only works if the OpenTSDB suggest API is enabled.

## Templating queries

Instead of hardcoding things such as server, application and sensor name in your metric queries you can use variables in their place. Variables are shown as dropdown select boxes at the top of the dashboard. You can use these dropdown boxes to change the data being displayed in your dashboard.

For more information about templating and template variables, see [Templates and variables \(p. 265\)](#).

### Query variable

The OpenTSDB data source supports template variable queries. This means you can create template variables that fetch the values from OpenTSDB. For example, metric names, tag names, or tag values.

When using OpenTSDB with a template variable of query type you can use following syntax for lookup.

Query	Description
<code>metrics(prefix)</code>	Returns metric names with specific prefix (can be empty).
<code>tag_names(cpu)</code>	Returns tag names (i.e., keys) for a specific cpu metric.
<code>tag_values(cpu, hostname)</code>	Returns tag values for metric cpu and tag key hostname.
<code>suggest_tagk(prefix)</code>	Returns tag names (i.e., keys) for all metrics with specific prefix (can be empty).
<code>suggest_tagv(prefix)</code>	Returns tag values for all metrics with specific prefix (can be empty).

If you do not see template variables being populated in `Preview of values` section, you must enable `tsd.core.meta.enable_realtime_ts` in the OpenTSDB server settings. Also, to populate metadata of the existing time series data in OpenTSDB, you must run `tsdb uid metasync` on the OpenTSDB server.

### Nested templating

One template variable can be used to filter tag values for another template variable. First parameter is the metric name, second parameter is the tag key for which you need to find tag values, and after that all other dependent template variables. Some examples are mentioned below to make nested template queries work successfully.

Query	Description
<code>tag_values(cpu, hostname, env=\$env)</code>	Returns tag values for cpu metric, selected env tag value, and tag key hostname.
<code>tag_values(cpu, hostname, env=\$env, region=\$region)</code>	Returns tag values for cpu metric, selected env tag value, selected region tag value, and tag key hostname.

For more information about OpenTSDB metric queries, see [OpenTSDB documentation](#)

## PostgreSQL

You can use the PostgreSQL data source to query and visualize data from your Amazon Aurora PostgreSQL databases.

### Important

Grafana version 8.0 changes the underlying data structure for data frames for the Postgres, MySQL, and Microsoft SQL Server data sources. As a result, a time series query result is returned in a wide format. For more information, see [Wide format](#) in the Grafana data frames documentation.

To make your visualizations work as they did before, you might have to do some manual migrations. One solution is documented on Github at [Postgres/MySQL/MSSQL: Breaking change in v8.0 related to time series queries and ordering of data column](#).

## Adding the data source

1. Open the side menu by choosing the Grafana icon in the top header.
2. In the side menu under the **Configuration** icon, you should find a **Data Sources** link.
3. Choose the **+ Add data source** button in the top header.
4. Select **PostgreSQL** from the **Type** dropdown list.

### Data source options

Name	Description
Name	The data source name. This is how you see the data source in panels and queries.
Default	Default data source means that it will be pre-selected for new panels.
Host	The IP address/hostname and optional port of your PostgreSQL instance. <i>Do not</i> include the database name. The connection string for connecting to Postgres will not be correct and will cause errors.
Database	Name of your PostgreSQL database.
User	Database user's login/username.
Password	Database user's password
SSL Mode	This option determines whether or with what priority a secure SSL TCP/IP connection will be negotiated with the server.
Max open	The maximum number of open connections to the database, default unlimited (Grafana v5.4+).
Max idle	The maximum number of connections in the idle connection pool, default 2 (Grafana v5.4+).
Max lifetime	The maximum amount of time in seconds a connection may be reused, default 14400/4 hours (Grafana v5.4+).

Name	Description
Version	This option determines which functions are available in the query builder (only available in Grafana 5.3+).
TimescaleDB	TimescaleDB is a time-series database built as a PostgreSQL extension. If enabled, Grafana will use <code>time_bucket</code> in the <code>\$__timeGroup</code> macro and display TimescaleDB specific aggregate functions in the query builder (only available in Grafana 5.3+).

### Min time interval

A lower limit for the `$_interval` `$_interval_ms` variables. Recommended to be set to write frequency, for example `1m` if your data is written every minute. This option can also be overridden/configured in a dashboard panel under data source options. This value **must** be formatted as a number followed by a valid time identifier; for example, `1m` (1 minute) or `30s` (30 seconds). The following time identifiers are supported.

Identifier	Description
y	Year
M	Month
w	Week
d	Day
h	Hour
m	Minute
s	Second
ms	Millisecond

### Database user permissions

#### Important

The database user that you specify when you add the data source should only be granted `SELECT` permissions on the specified database and tables you want to query. Grafana does not validate that the query is safe. The query could include any SQL statement. For example, statements such as `DELETE FROM user;` and `DROP TABLE user;` would be run. To protect against this, we highly recommend that you create a specific PostgreSQL user with restricted permissions.

The following example code shows creating a specific PostgreSQL user with restricted permissions.

```
CREATE USER grafanareader WITH PASSWORD 'password';
GRANT USAGE ON SCHEMA schema TO grafanareader;
GRANT SELECT ON schema.table TO grafanareader;
```

Make sure that the user does not get any unwanted permissions from the public role.

## Query editor

You find the PostgreSQL query editor in the metrics tab in Graph or Singlestat panel's edit mode. You enter edit mode by choosing the panel title, then edit.

The query editor has a **Generated SQL** link that shows up after a query has been run, while in panel edit mode. Choose it, and it will expand and show the raw interpolated SQL string that was run.

### Select table, time column, and metric column (FROM)

When you enter edit mode for the first time or add a new query, Grafana will try to prefill the query builder with the first table that has a timestamp column and a numeric column.

In the FROM field, Grafana will suggest tables that are in the `search_path` of the database user. To select a table or view not in your `search_path` you can manually enter a fully qualified name (schema.table) such as `public.metrics`.

The Time column field refers to the name of the column holding your time values. Selecting a value for the Metric column field is optional. If a value is selected, the Metric column field will be used as the series name.

The metric column suggestions will only contain columns with a text data type (char,varchar,text). To use a column with a different data type as metric column, you can enter the column name with a cast: `ip::text`. You can also enter arbitrary SQL expressions in the metric column field that evaluate to a text data type such as `hostname || ' ' || container_name`.

### Columns, window, and aggregation functions (SELECT)

In the SELECT row, you can specify what columns and functions you want to use. In the column field, you can write arbitrary expressions instead of a column name such as `column1 * column2 / column3`.

The available functions in the query editor depend on the PostgreSQL version you selected when configuring the data source. If you use aggregate functions, you must group your result set. If you add an aggregate function, the editor will automatically add a `GROUP BY time`.

The editor tries to simplify and unify this part of the query.

You may add further value columns by choosing the plus button and selecting **Column** from the menu. Multiple value columns will be plotted as separate series in the graph panel.

### Filtering data (WHERE)

To add a filter, choose the plus icon to the right of the WHERE condition. You can remove filters by choosing the filter and selecting **Remove**. A filter for the current selected time range is automatically added to new queries.

### Group By

To group by time or any other columns choose the plus icon at the end of the GROUP BY row. The suggestion dropdown list will only show text columns of your currently selected table but you may manually enter any column. You can remove the group by choosing the item and then selecting **Remove**.

If you add any grouping, all selected columns must have an aggregate function applied. The query builder will automatically add aggregate functions to all columns without aggregate functions when you add groupings.

### Gap filling

Amazon Managed Grafana can fill in missing values when you group by time. The time function accepts two arguments. The first argument is the time window that you want to group by, and the second argument is the value you want Grafana to fill missing items with.

## Text editor mode (RAW)

You can switch to the raw query editor mode by choosing the hamburger icon and selecting **Switch editor mode** or by choosing **Edit SQL** below the query.

### Note

If you use the raw query editor, be sure that your query at minimum has `ORDER BY time` and a filter on the returned time range.

## Macros

Macros can be used within a query to simplify syntax and allow for dynamic parts.

Macro example	Description
<code>\$__time(dateColumn)</code>	Will be replaced by an expression to convert to a UNIX timestamp and rename the column to <code>time_sec</code> . For example, <code>UNIX_TIMESTAMP(dateColumn) as time_sec</code> .
<code>\$__timeEpoch(dateColumn)</code>	Will be replaced by an expression to convert to a UNIX timestamp and rename the column to <code>time_sec</code> . For example, <code>UNIX_TIMESTAMP(dateColumn) as time_sec</code> .
<code>\$__timeFilter(dateColumn)</code>	Will be replaced by a time range filter using the specified column name. For example, <code>dateColumn BETWEEN FROM_UNIXTIME(1494410783) AND FROM_UNIXTIME(1494410983)</code> .
<code>\$__timeFrom()</code>	Will be replaced by the start of the currently active time selection. For example, <code>FROM_UNIXTIME(1494410783)</code> .
<code>\$__timeTo()</code>	Will be replaced by the end of the



Macro example	Description
	currently active time selection. For example, <i>FROM_UNIXTIME(1494410983).</i>
<code>\$__timeGroup(dateColumn, '5m')</code>	Will be replaced by an expression usable in GROUP BY clause. For example, <i>cast(cast(UNIX_TIMESTAMP(dateColumn) as signed)300 as signed),*</i>
<code>\$__timeGroup(dateColumn, '5m', 0)</code>	Same as the previous row, but with a fill parameter so missing points in that series will be added by grafana and 0 will be used as value.
<code>\$__timeGroup(dateColumn, '5m', NULL)</code>	Same as above but NULL will be used as value for missing points.
<code>\$__timeGroup(dateColumn, '5m', previous)</code>	Same as above but the previous value in that series will be used as fill value if no value has been seen yet NULL will be used (only available in Grafana 5.3+).
<code>\$__timeGroupAlias(dateColumn, '5m')</code>	Will be replaced identical to <code>\$__timeGroup</code> but with an added column alias

Macro example	Description
<code>\$__unixEpochFilter(dateColumn)</code>	Will be replaced by a time range filter using the specified column name with times represented as Unix timestamp. For example, <code>*dateColumn &gt; 1494410783 AND dateColumn &lt; 1494497183*</code>
<code>\$__unixEpochFrom()</code>	Will be replaced by the start of the currently active time selection as Unix timestamp. For example, <code>*1494410783*</code>
<code>\$__unixEpochTo()</code>	Will be replaced by the end of the currently active time selection as Unix timestamp. For example, <code>*1494497183*</code>
<code>\$__unixEpochNanoFilter(dateColumn)</code>	Will be replaced by a time range filter using the specified column name with times represented as nanosecond timestamp. For example, <code>*dateColumn &gt; 1494410783152415214 AND dateColumn &lt; 1494497183142514872*</code>
<code>\$__unixEpochNanoFrom()</code>	Will be replaced by the start of the currently active time selection as nanosecond timestamp. For example, <code>*1494410783152415214*</code>

Macro example	Description
<code>\$__unixEpochNanoTo()</code>	Will be replaced by the end of the currently active time selection as nanosecond timestamp. For example, <code>*1494497183142514872*</code>
<code>\$__unixEpochGroup(dateColumn, "5m", [fillmode])</code>	Same as <code>\$__timeGroup</code> but for times stored as Unix timestamp.

## Table queries

If the query option is set to **Format as Table**, you can basically do any type of SQL query. The table panel will automatically show the results of whatever columns and rows your query returns.

You can control the name of the Table panel columns by using regular as SQL column selection syntax.

## Time series queries

If you set **Format as** to **Time series**, for use in a graph panel for example, the query must return a column named `time` that returns either a SQL datetime or any numeric data type representing Unix epoch. Any column except `time` and `metric` is treated as a value column. You may return a column named `metric` that is used as metric name for the value column. If you return multiple value columns and a column named `metric`, this column is used as prefix for the series name.

Result sets of time series queries must be sorted by time.

The following example code shows a `metric` column.

```
SELECT
  $__timeGroup("time_date_time", '5m'),
  min("value_double"),
  'min' as metric
FROM test_data
WHERE $__timeFilter("time_date_time")
GROUP BY time
ORDER BY time
```

The following code example shows using the `fill` parameter in the `$__timeGroup` macro to convert null values to be zero instead.

```
SELECT
  $__timeGroup("createdAt", '5m', 0),
  sum(value) as value,
  measurement
FROM test_data
WHERE
  $__timeFilter("createdAt")
GROUP BY time, measurement
```

```
ORDER BY time
```

The following example code shows multiple columns.

```
SELECT
  $__timeGroup("time_date_time",'5m'),
  min("value_double") as "min_value",
  max("value_double") as "max_value"
FROM test_data
WHERE $__timeFilter("time_date_time")
GROUP BY time
ORDER BY time
```

## Templating

Instead of hardcoding things such as server, application and sensor name in your metric queries you can use variables in their place. Variables are shown as dropdown select boxes at the top of the dashboard. You can use these dropdown boxes to change the data being displayed in your dashboard.

For more information about templating and template variables, see [Templates \(p. 266\)](#).

### Query variable

If you add a template variable of the type `Query`, you can write a PostgreSQL query that can return things such as measurement names, key names, or key values that are shown as a dropdown select box.

For example, you can have a variable that contains all values for the `hostname` column in a table if you specify a query such as this in the templating variable `Query` setting.

```
SELECT hostname FROM host
```

A query can return multiple columns and Grafana will automatically create a list from them. For example, the following query will return a list with values from `hostname` and `hostname2`.

```
SELECT host.hostname, other_host.hostname2 FROM host JOIN other_host ON host.city =
other_host.city
```

To use time range dependent macros such as `$__timeFilter(column)` in your query, the refresh mode of the template variable must be set to *On Time Range Change*.

```
SELECT event_name FROM event_log WHERE $__timeFilter(time_column)
```

Another option is a query that can create a key/value variable. The query should return two columns that are named `__text` and `__value`. The `__text` column value should be unique (if it is not unique, the first value is used). The options in the dropdown list will have a text and value that allows you to have a friendly name as text and an id as the value. An example query with `hostname` as the text and `id` as the value:

```
SELECT hostname AS __text, id AS __value FROM host
```

You can also create nested variables. Using a variable named `region`, you could have the hosts variable show only hosts from the current selected region. The following code example shows a query such as

this (if `region` is a multi-value variable, use the `IN` comparison operator rather than `=` to match against multiple values).

```
SELECT hostname FROM host WHERE region IN($region)
```

### Using `__searchFilter` to filter results in Query Variable

Using `__searchFilter` in the query field will filter the query result based on what the user types in the dropdown select box. When nothing has been entered by the user, the default value for `__searchFilter` is `%`.

#### Note

Important that you surround the `__searchFilter` expression with quotes as Grafana does not do this for you.

The following example shows how to use `__searchFilter` as part of the query field to enable searching for `hostname` while the user types in the dropdown select box.

```
SELECT hostname FROM my_host WHERE hostname LIKE '$__searchFilter'
```

### Using variables in queries

Template variable values are only quoted when the template variable is a multi-value.

If the variable is a multi-value variable, use the `IN` comparison operator rather than `=` to match against multiple values.

There are two syntaxes:

`$<varname>` Example with a template variable named `hostname`:

```
SELECT
  atimestamp as time,
  aint as value
FROM table
WHERE $__timeFilter(atimestamp) and hostname in($hostname)
ORDER BY atimestamp ASC
```

`[[varname]]` Example with a template variable named `hostname`:

```
SELECT
  atimestamp as time,
  aint as value
FROM table
WHERE $__timeFilter(atimestamp) and hostname in([[hostname]])
ORDER BY atimestamp ASC
```

### Turning off quoting for multi-value variables

Amazon Managed Grafana automatically creates a quoted, comma-separated string for multi-value variables. For example: if `server01` and `server02` are selected then it will be formatted as: `'server01', 'server02'`. To turn off quoting, use the `csv` formatting option for variables.

`${servers:csv}`

For more information about variable formatting options, see [Templates and variables \(p. 265\)](#).

## Annotations

Use annotations to overlay rich event information on top of graphs. You add annotation queries via the Dashboard menu / Annotations view. For more information, see [Annotations \(p. 238\)](#).

The following example code shows a query using a time column with epoch values.

```
SELECT
  epoch_time as time,
  metric1 as text,
  concat_ws(' ', metric1::text, metric2::text) as tags
FROM
  public.test_data
WHERE
  $__unixEpochFilter(epoch_time)
```

The following example code shows a region query using time and timeend columns with epoch values.

### Note

This is available only in Grafana v6.6+.

```
SELECT
  epoch_time as time,
  epoch_time_end as timeend,
  metric1 as text,
  concat_ws(' ', metric1::text, metric2::text) as tags
FROM
  public.test_data
WHERE
  $__unixEpochFilter(epoch_time)
```

The following example code shows a query using a time column of native SQL date/time data type.

```
SELECT
  native_date_time as time,
  metric1 as text,
  concat_ws(' ', metric1::text, metric2::text) as tags
FROM
  public.test_data
WHERE
  $__timeFilter(native_date_time)
```

Name	Description
time	The name of the date/time field. Could be a column with a native SQL date/time data type or epoch value.
timeend	Optional name of the end date/time field. Could be a column with a native SQL date/time data type or epoch value (Grafana v6.6+).
text	Event description field.
tags	Optional field name to use for event tags as a comma-separated string.

## Alerting

Time series queries should work in alerting conditions. Table formatted queries are not yet supported in alert rule conditions.

## Redis

The Redis Data Source for Grafana is a plugin that allows users to connect to any Redis database on-premises or in the cloud. It provides out-of-the-box predefined dashboards and lets you build customized dashboards to monitor Redis and application data.

### Adding the data source

1. Open the Grafana console in the Amazon Managed Grafana workspace and make sure you are logged in.
2. In the side menu under **Configuration** (the gear icon), choose **Data Sources**.
3. Choose **Add data source**.

#### Note

If you don't see the **Data Sources** link in your side menu, it means that your current user does not have the `Admin` role.

4. Select **Redis** from the list of data sources.
5. Enter the following information:
  - For **Address**, specify the `host:port` address or a URI to connect to Redis. Use `/db=number` or `?db=db=number` to specify the logical database number as defined in the [Schema](#).
  - **Cluster** tab. For redundancy, provide multiple `host:port` addresses or URIs comma separated.
  - In the **Sentinel** tab, the **Address** can contain multiple values (`host:port` address or a URI) with commas. The **Master Name** is required to connect to the Sentinel and open Redis connections.
  - In Unix socket mode, the **Address** should contain the path to the socket file.
  - For **Pool Size**, the recommendation is 5. The data source will keep open at least the given number of connections to the Redis instance. You can increase the pool size if dashboards have a lot of panels and multiple users.

### Query editor

1. Choose **Type** to select core Redis, Custom, or Redis Module.
2. Select one of the supported commands.
3. Provide all required parameters.
4. Enable streaming to visualize data on Graph or Time-Series panels.

Template variables can query any command which returns a list of values and use other variables as parameters.

## Tempo

Tempo is a high-volume, minimal dependency trace storage, OSS tracing solution from Grafana Labs.

### Adding the data source

To access Tempo settings, choose the **Configuration** (gear) icon, then choose **Data Sources**, and then choose **Tempo**.

Name	Description
Name	The name using which you will refer to the data source in panels, queries, and Explore.
Default	Default data source means that it will be pre-selected for new panels.
URL	The URL of the Tempo instance; e.g., <code>http://tempo</code> .
Basic Auth	Enable basic authentication to the Tempo data source.
User	User name for basic authentication.
Password	Password for basic authentication.

## Trace to logs

This is a configuration for the trace to logs feature. The target data source currently must be Loki. For more information, see [Tracing integration \(p. 259\)](#).

- **Data source** – Target data source.
- **Tags** – The tags that will be used in the Loki query. The default is 'cluster', 'hostname', 'namespace', 'pod'
- **Span start time shift** – Shift in the start time for the Loki query based on the span start time. In order to extend to the past, you need to use a negative value. Time units can be used here, for example, 5s, 1m, 3h. The default is 0.
- **Span end time shift** – Shift in the end time for the Loki query based on the span end time. Time units can be used here, for example, 5s, 1m, 3h. The default is 0.

## Query traces

You can query and display traces from Tempo via Explore. You can search for traces if you set up the trace to logs setting in the data source configuration page. To find traces to visualize, use the Loki query editor. To get search results, you must have derived fields configured, which point to this data source.

To query a particular trace, select the TraceID query type, and then put the ID into the Trace ID field.

## Linking to the trace ID from logs

You can link to Tempo trace from logs in Loki or Elastic by configuring an internal link. For more information, see [Derived fields \(p. 95\)](#).

## Zipkin

Zipkin is an open source, distributed tracing system. Add the Zipkin data source to be able to query your traces in Explore in Amazon Managed Grafana

## Adding the data source

To access Zipkin settings, choose the **Configuration** (gear) icon, then choose **Data Sources**, and then choose **Zipkin**.

Name	Description
Name	The data source name. This is how you see the data source in panels, queries, and Explore.



Name	Description
Default	Default data source means that it will be pre-selected for new panels.
URL	The URL of the Zipkin instance; e.g., <code>http://localhost:9411</code> .
Access	Server (default) = URL needs to be accessible from the Grafana backend/server, Browser = URL needs to be accessible from the browser.
Basic Auth	Enable basic authentication to the Zipkin data source.
User	User name for basic authentication.
Password	Password for basic authentication.

## Query traces

Querying and displaying traces from Zipkin is available via Explore.

The Zipkin query editor allows you to query by trace ID directly or selecting a trace from trace selector. To query by trace ID, insert the ID into the text input.

Use the trace selector to pick particular trace from all traces logged in the time range you have selected in Explore. The trace selector has three levels of nesting: 1. The service you are interested in. 1. Particular operation is part of the selected service 1. Specific trace in which the selected operation occurred, represented by the root operation name and trace duration.

## Data mapping in the trace UI

Zipkin annotations are shown in the trace view as logs with annotation value shown under annotation key.

## Linking to the trace ID from logs

You can link to Zipkin trace from logs in Loki by configuring a derived field with internal link. For more information, see [Derived fields \(p. 95\)](#).

# Data sources available in Grafana Enterprise

The following data sources are supported in workspaces that have been upgraded to Grafana Enterprise. For more information, see [Upgrade a workspace to Grafana Enterprise \(p. 370\)](#).

### Topics

- [AppDynamics \(p. 139\)](#)
- [Datadog \(p. 140\)](#)
- [Dynatrace \(p. 142\)](#)
- [GitLab \(p. 145\)](#)
- [Honeycomb \(p. 147\)](#)
- [Jira \(p. 148\)](#)
- [MongoDB \(p. 152\)](#)
- [New Relic \(p. 154\)](#)
- [Oracle Database \(p. 156\)](#)
- [Salesforce \(p. 160\)](#)

- [SAP HANA \(p. 163\)](#)
- [ServiceNow \(p. 169\)](#)
- [Snowflake \(p. 175\)](#)
- [Splunk \(p. 179\)](#)
- [Wavefront \(VMware Tanzu Observability by Wavefront\) \(p. 184\)](#)

## AppDynamics

The AppDynamics data source for Amazon Managed Grafana enables you to query metrics from AppDynamics using its Metrics API and visualize them in Grafana dashboards.

### Note on the Data source configuration

Use Server (proxy) access (to avoid CORS and users looking up your password) and basic authentication. Remember that the username should be "user@account", (that is, your.name@customer1 or my\_user@saas\_account\_name).

Configure the password using the following steps:

1. Navigate to <https://accounts.appdynamics.com/subscriptions>
2. Choose the link in the **Name** column on the row for your subscription.
3. Navigate to the **License details** by choosing the tab at top of the page.
4. The Access Key field has a **Show** button. Choose the **Show** button to show the Access Key.
5. Copy the Access Key into the Password field in the Basic Auth Details on config page in Grafana.

Set up a user and role for Amazon Managed Grafana using the following steps.

1. In AppDynamics, navigate to Settings, Administration.
2. Select the **Roles** tab, and choose the "+" button to create a new role; for example, grafana\_readonly.
3. In the **Account** tab of the Create Role section, add the permission View Business Flow.
4. In the **Applications** tab, check the **View** box to allow Grafana to view application data.
5. In the **Databases** tab, check the **View** box to allow Grafana to view database data.
6. In the **Analytics** tab, check the **Can view data from all Applications** box to allow Grafana to view application analytics data.
7. In the **Users** tab of the Administration page, create a new user; for example, grafana. Assign the new user (or a Group to which the user belongs) to the role you just created; for example, grafana\_readonly.

## Templating

The supported template queries for now are:

1. Applications (All Applications)
2. AppName.BusinessTransactions (All BTs for the Application Name)
3. AppName.Tiers (All Tiers for the Application Name)
4. AppName.Nodes (All Nodes for the Application Name)
5. AppName.TierName.BusinessTransactions (All BTs for a specific Tier)
6. AppName.TierName.Nodes (All Nodes for a specific Tier)
7. AppName.Path.<Any Metric Path> (Any metric Path can be specified)

## Legend keys

The default for the legend key can be quite long but this formatting can be customized.

The legend key can be prefixed with the application name by choosing the `App on legend` option. For example: `MyApp - Overall Application Performance|Average Response Time (ms)`.

If the query is for a `singlestat` or other panel where you cannot see the legend key, then choose the `Show Metadata` option to see what the legend key (also called an alias) for the query is.

The Legend dropdown list has three options: `Full Path`, `Segments` and `Custom`.

### Legend option – full path

The legend key is the full metric path; for example, `Overall Application Performance|Average Response Time (ms)`.

### Legend option – segments

The metric name is made up of segments. You can choose which segments to show.

For example, with a metric name:

```
Errors|mywebsite|Error|Errors per Minute
```

entering the following `2,4` in the Segments field returns `mywebsite|Errors per minute`.

The indexing starts with `1` so `1` returns `Errors`.

### Legend option – custom

Create a custom legend by combining text with the following aliasing patterns to be able to mix in metric metadata.

- `{{app}}` returns the Application name
- `{{1}}` returns a segment from the metric path.

For example, the metric: `Overall Application Performance|Average Response Time (ms)` has two segments. `{{1}}` returns the first segment, `{{2}}` returns the second segment.

Examples of legend key patterns and the legend keys that are generated:

- `custom legend key => custom legend key`
- `App: {{app}} MetricPart2: {{2}} => App: MyApp MetricPart2: Average Response Time (ms)`

## Datadog

The Datadog data source enables you to visualize metrics from the Datadog monitoring service in Amazon Managed Grafana.

### Usage

#### Caching

For large dashboards, that make lots of queries it is possible to be rate limited by the Datadog API (reach the maximum number of API calls per hour that the Datadog API allows). The caching feature caches unique queries for 60 seconds. This interval can be changed to be longer or shorter on the config page.

## Query editor

It's easy - select aggregation and metric. If you want to filter result, select one or more tags.

The Datadog data source supports all of the advanced functions that the Datadog query editor supports. Select it from the dropdown list and arrange by choosing a function name.

### Alias by field usage possibilities:

- Enter the alias into the "Alias by" field.
- Use scoped variables:
  - `$__metric` = replaced with metric name
  - `$__display_name` = replaced with metric name
  - `$__expression` = replaced with full metric expression
  - `$__aggr` = replaced with metric aggregation function (for example, avg, max, min, sum)
  - `$__scope` = replaced with metric scope (for example, region, site, env, host)
- Use regular expressions:
  - Enter your regular expression into "Alias RegExp" field in `/you regexp here/flags` format.
  - If "Alias by" field is empty, RegExp results will be joined using. Example with metric expression = `avg:system.load.5{*}` : "Alias by" field input: `""` "Alias RegExp" field input: `avg:(.+)\.(\d)` Result: `system.load, 5`
  - Use `$_<group_number>` variables in "Alias by" field. Example with metric expression = `avg:system.load.5{*}` : "Alias by" field input: `$1: $2 seconds` "Alias RegExp" field input: `avg:(.+)\.(\d)` Result: `system.load: 5 seconds`
  - Use `$0` to get the whole expression. Example with metric expression = `datadog.dogstatsd.packet.count{*}` : "Alias by" field input: `Expression: $0` "Alias RegExp" field input: `DOGstatsd\.(.*)\.(.*){\\*}/i` Result: `Expression: datadog.dogstatsd.packet.count{*}`

Note: you'll get an error using nonexistent group number.

## Metric arithmetic

To use metric arithmetic set *Query type* to *Arithmetic*. Link to the metric that you want by using # sign. For example, `#A * 2` will double the result of query `A`. Arithmetic between two metrics works in the same way - add queries which results you want to use for the calculation and then link to these metrics in the third query, such as `#A / #B`.

## Annotations

An annotation is an event that is overlaid on top of graphs - an example of an event is a deployment or an outage. With this data source, you can fetch events from Datadog and overlay them on graphs in Amazon Managed Grafana. Annotations events can be filtered by source, tag or priority.

## Templating

There are a few options for getting values of template variable - metrics and tags. To fetch the list of available metrics specify `*` in the *Query* field.

To return all tags use the value: `tag` or `scope`.

To return tags for a specified tag group then use one of the following default category values:

- `host`
- `device`

- `env`
- `region`
- `site`
- `status`
- `version`

For custom tag groups, then just enter the tag group name. For example, if your custom tag group name is `subscription_name`, enter that in the *Query* field.

Filter results by using the *Regex* field. Multi-value variables are supported when using tags - multiple selected tag values will be converted into a comma separated list of tags.

### Ad-hoc filters

There is a new special type of template variable in Grafana called *Ad-hoc filters*. This variable will apply to *all* the Datadog queries in a dashboard. This allows using it like a quick filter. An ad-hoc variable for Datadog fetches all the key-value pairs from tags, for example, `region: east`, `region: west`, and uses them as query tags. To create this variable, select the *Ad-hoc filters* type and choose your Datadog data source. You can set any name for this variable.

## Dynatrace

Data source for <https://www.dynatrace.com/>. To use this data source, you must have a Dynatrace account.

### Known limitations

Template variables can't be multi-select. Only single selection is supported.

Only v2 metric APIs are supported.

## Features

### Core features

- Template Variables
  - Metric Names
  - Single selection only (**no multi-select**)
  - Ad-Hoc Filters
- Annotations
  - Not currently supported
- Aliasing
  - Metric Names
  - Aggregation
  - Display Name
  - Host
  - Description
- Alerting
  - Full alerting support

### Dynatrace specific features

Supports both built-in and custom metrics using the Dynatrace metrics v2 API. For more information, see the Dynatrace documentation: [Metrics API v2](#) and [Metric ingestion](#).

Depending on the metric, the API might support additional transformation options.

## Dynatrace permissions

You will need the following permissions in Dynatrace - Read metrics using API V2 (metrics.read) permission - Read entities using API V2 (entities.read) permission

## Get an API key from Dynatrace

To set up an API token, see [Dynatrace API - Tokens and authentication](#)

Set the `metrics.read` and `entities.read` permissions for your API token.

## Configuration

1. Choose **Settings/Data Sources** within the logical Grafana server UI and choose **Add data source**.
2. On the **Add data source** page, filter for **Dynatrace**, and select the Dynatrace plugin.
3. Configuring a Dynatrace data source requires the following parameters:
  - **Name** - The name you want to apply to the Dynatrace data source (default: Dynatrace).
  - **Dynatrace API Type** - The type of Dynatrace instance that you're connecting to. This is either `SaaS` or `Managed Cluster`.
  - **Dynatrace API Token** - This is the API token you generated in the previous step..

The next two settings depend on whether you are Dynatrace SaaS or managed

- In a SaaS example of `yfc55578.live.dynatrace.com`, your **Environment ID** would be `yfc55578`.
  - In the Managed example of `yd8888.managed-sprint.dynalabs.io/e/abc99984-3af2-55tt-72kl-0672983gc45`, your **Environment ID** would be `abc99984-3af2-55tt-72kl-0672983gc45` and your **Domain** would be `yd8888.managed-sprint.dynalabs.io`
4. After all of the configuration values have been set, choose **Save & Test** to validate the configuration and save your changes.

## Query the data source

Use the query editor to query Dynatrace metrics and problems. The query type can be `metric` or `problem`.

### Metric query type

- **Metric**— Select the metric that you want to see. To get the metric list from Dynatrace again, choose **Refresh** button.
- **Aggregations**— Select the aggregation you want to use for a specific metric. Choose the aggregations value to change the aggregation type or choose **+** to add another aggregation.
- **Transformations**— You can select transformations in the query editor. Afterwards, enter a number of parameters into the selected transformation. Currently, only the merge transformation is supported. For more information about the merge transforms, see [Merge transformation](#).
- **Filters**— The Dynatrace data source dynamically queries the appropriate filters for each metric. To add a filter, choose the **+** symbol next to the **Filters** label on the Dynatrace query editor, select which field to filter on, select the operator to use, and then select a value to filter by. The Dynatrace data source allows you to create Filter Groups that you can join together to create complex logical comparisons. For most use cases, Filter Groups are not required. When creating filters with Tags, regardless of the conjunction selected, Dynatrace will always use AND. Dynatrace does not support OR filters with Tags.

- **Alias**— There are two different types of aliases you will encounter while using the Dynatrace data source. The first is a static alias. An alias of this type is available on every query that you build, and the name of the alias starts with a lowercase letter. The second is a dynamic alias, which changes based on the metric that you are using in your query, and the name of the alias starts with an uppercase letter. The Dynatrace plugin supports several different aliases: `Metric Names`, `Aggregation`, `Display Name`, `Host`, and `Description`.

Name	Value
<code>\$name</code>	<code>builtin:apps.other.keyUserActions.reportedErrorCount.os</code>
<code>\$aggregation</code>	<code>auto,value</code>
<code>\$displayName</code>	Reported error count (by key user action, OS) [mobile, custom]

### Problems query type

- **Problem Query Type**— Select a problem query type. Currently, only the feed problem query type is supported. For information about the feed problem query type, see [Merge transformation](#)
- **Status Filter**— Filter the result problems by the status.
- **Impact Filter**— Filter the result problems by the impact level.
- **Severity Filter**— Filter the result problems by the severity level.
- **Expand Details**— Include related events to the response, if set..

### Using template variables

To add a new Dynatrace query variable, see [add a new template variable \(p. 267\)](#). Use your Dynatrace data source as your data source for the following available queries:

- **Query type**— Select a query type. The query type associates some data with some key or descriptor.

Query type	Description
<code>Metric names</code>	Returns a list of all metric names
<code>Filter keys</code>	Returns a list of all the possible dimensions (e.g. Hostname) that can be used to filter
<code>Filter values for key</code>	Returns a list of all filtered values by a key name or a key name template variable
<code>Problem status options</code>	Returns a list of all problem statuses
<code>Problem impact options</code>	Returns a list of all problem impacted areas
<code>Problem severity options</code>	Returns a list of all problem severity types

- **Regex**— (Optional) Filter out any of the returned values from your query with a regular expression.

#### Note

`Multi-value` and `Include All` option are currently not supported by the Dynatrace data source.

After creating a variable, you can find it in the **Metric** drop-down menu.

## Import a dashboard for Dynatrace

To import a dashboard, see [Importing a dashboard \(p. 247\)](#). Imported dashboards can be found in **Configuration > Data Sources** > select your Dynatrace data source > select the **Dashboards** tab to see available pre-made dashboards.

## GitLab

The GitLab data source allows you to keep track of detailed GitLab statistics, such as top contributors, commits per day, or deployments per day. You can also use template variables, such as projects, to set up filters for your dashboards. You can combine data from the GitLab API with data from other sources.

### Known limitations

Alerting is not supported yet on this plugin because transformations are not supported in alert queries and transformations is the only way to obtain meaningful aggregate metrics from GitLab API raw data.

### Adding the data source

1. Open the Grafana console in the Amazon Managed Grafana workspace and make sure you are logged in.
2. In the side menu under **Configuration** (the gear icon), choose **Data Sources**.
3. Choose **Add data source**.

#### Note

If you don't see the **Data Sources** link in your side menu, it means that your current user does not have the `Admin` role.

4. Select **GitLab** from the list of data sources.
5. Enter the following information:
  - For **Name**, enter a name for this GitLab data source.
  - For **URL**, enter the root URL for your GitLab instance, such as `https://gitlab.com/api/v4`.
  - For **Access token**, enter your GitLab personal access token.

### Query the GitLab data source

From the GitLab Query Editor you can select different resource types, such as commits, issues, or releases.

#### Filter and view projects

1. From the dropdown menu, choose **Projects**.
2. (Optional) Filter by the projects that you own.
3. Use the dropdown and select **Yes** or **No** to filter the results.

#### Note

Fetching all the projects **Owned = No** may take long time.

#### Filter and view commits

1. From the dropdown menu, choose **Commits**.
2. Use the input field to add the project ID.
3. (Optional) To filter by branch/tag use the input field to add a branch/tag reference.



### Filter and view issues

1. From the dropdown menu, choose **Issues**.
2. Use the input field to add the project ID.
3. (Optional) To filter by title/description, use the input field to search issues based on their **title** and **description**.

### View releases

1. From the dropdown menu, choose **Deployments**.
2. Use the input field to add the project ID.
3. (Optional) To filter by environment/status, use the input fields. The **status** attribute can be one of the following values: `created`, `running`, `success`, `failed`, or `canceled`.

### View labels

1. From the dropdown menu, choose **Labels**.
2. Use the input field to add the project ID.

## Templates and variables

To add a new GitLab query variable, see [Adding a query variable \(p. 267\)](#). Use your GitLab data source as the data source. Choose a resource type: **Releases**, **Projects**, or **Labels**.

To get a dynamic list of projects, labels, and so on to choose from, create a Query type variable. Query type variables use the GitLab Query Editor to query and return Projects, Labels, and so on. The following example creates a Project variable to parameterize your queries

### Create a Project variable to parameterize your queries

1. Add a variable of type **Query** named **project**.
2. Select your GitLab data source and refresh **On Dashboard Load**.
3. Select the **Projects** resource type, **Yes** for **Owned**, **name** for **display field** and **id** for **value field**.
4. Choose **Update** to add the variable to the dashboard.
5. Add a new panel to the dashboard and use **\$project** as the project ID.

Now, when choosing from the dropdown, you get the results that belong to that project.

## Using transformations from Grafana to answer common questions

Now that you can perform basic GitLab queries to find commits, issues, etc, you can use Transformations to visualize, aggregate, group, and join datasets, along with many other types of transformations to transform simple results into answers for complex questions. Below are a few common questions and how to use transformations to answer them.

### How many commits/issues/deployments per day in my project?

1. Add a query. Select **Commits** for the resource type and add the project ID.
2. Add a new **Group by** transformation: for **Group by**, select **created\_at\_date** and then calculate **(Count)=id**
3. Choose the **Graph** visualization.

### What is the average time to close issues in my project?

1. Add a query. Select **Issues** for the resource type and add the project ID.
2. Add a new **Add field from calculation** transformation: for **Mode**, select **Binary Operation**, for **Operation**, select **closed\_at = created\_at** and for **Alias** choose **resolution\_time**.
3. Add a new **Add field from calculation** transformation: for **Mode**, select **Binary Operation**, for **Operation**, select **resolution\_time / 86400000** and for **Alias** choose **resolution\_time**.

For **Replace all fields**, choose **True**.

4. Choose the **Stat** visualization.
  - Show = Calculate
  - Calculation = Mean
  - Fields = **resolution\_time**

## Honeycomb

The Honeycomb data source allows you to query and visualize Honeycomb metrics and link to Honeycomb traces from within Amazon Managed Grafana.

### Known limitations

- The Honeycomb API that this data source is developed against is not yet generally available. It may cause unexpected behavior in the data source.
- This data source does not yet support or ad-hoc queries.
- Because of API limitations, the variable editor can only return the first 1000 unique values for a selected column.
- Because of API limitations, the data source can query only the last 7 days of data.

### Adding the data source

1. Open the Grafana console in the Amazon Managed Grafana workspace and make sure you are logged in.
2. In the side menu under **Configuration** (the gear icon), choose **Data Sources**.
3. Choose **Add data source**.
4. Select **Honeycomb** from the list of data sources.

#### Note

If you don't see the **Data Sources** link in your side menu, it means that your current user does not have the `Admin` role.

### Honeycomb settings

Name	Description
Name	The data source name. This is how you see the data source in panels, queries, and Explore.
Honeycomb API key	The API key that you obtained from Honeycomb.
URL	The URL of the Honeycomb API. For example, <code>https://api.honeycomb.io</code> .

Name	Description
Team	The Honeycomb team associated with the API key.

## Query the Honeycomb data source

To query metrics, enter values into the editor fields:

- Select a dataset.
- The default query is a `COUNT` over the selected dataset.
- To refine the query, select values for any of the remaining fields, such as **Visualization**, **Visualization**, **Where**, **Constraint**, **Group by**, **Order by**, or **Limit**.

## Templates and variables

To add a new Honeycomb query variable, see [Adding a query variable \(p. 267\)](#).

YOu can create variables containing Datasets, Columns, or Column Values.

- If no dataset is selected, the variable will contain datasets.
- If only a dataset is selected, the variable will contain column names.
- If both a dataset and a column are selected, the variable will contain column values. Column values can be further constrained using the **Where** fields in the editor. .

## View query in Honeycomb UI

To see the query you have created in the Honeycomb UI from the dashboard panel, choose any point in the graph and choose **Open in Honeycomb**.

To see the query you have created in the Honeycomb UI from the Query Editor, choose **Open in Honeycomb**.

## Import a dashboard for Honeycomb

To import a dashboard, see [Importing a dashboard \(p. 247\)](#).

To find your imported dashboards, choose **Configuration**, **Data sources**.

To see the avaialble pre-made dashboards, choose the Honeycomb data source and choose the **Dashboards** tab.

## Jira

Get the whole picture of your development process by combining issue data from Jira with application performance data from other sources.

- Create annotations based on issue creation or resolution, to see the relationship between issues and metrics.
- Track detailed Jira stats, such as mean time to resolution and issue throughput.

In order to use the Jira data source, you need an Atlassian account with access to a Jira project.

## Known limitations

Custom field types from Jira addons might not be supported.

## Adding the data source

1. Open the Grafana console in the Amazon Managed Grafana workspace and make sure you are logged in.
2. In the side menu under **Configuration** (the gear icon), choose **Data Sources**.
3. Choose **Add data source**.

### Note

If you don't see the **Data Sources** link in your side menu, it means that your current user does not have the `Admin` role.

4. Select **Jira** from the list of data sources.
5. Enter the following information:
  - For **Name**, enter a name for this Jira data source.
  - For **URL**, enter the root URL for your Atlassian instance, such as `https://bletchleypark.atlassian.net`.
  - For **User**, enter an email address for the user/service account.
  - For **API token**, enter an API token generated for the user.

## Query the Jira data source

From the Jira Query Editor you can select fields and query issues.

The Jira data source queries Jira for issues, which can represent bugs, user stories, support tickets, or other tasks in Jira

### Filter and view issues

1. Choose **Fields** choose the dropdown and use type-ahead to select from any of the fields in your Jira instance, including custom fields. Some fields to try:
  - **Summary**— The name of the issue
  - **Epic Name**— The epis that an issue belongs to
  - **Story Point Estimate**— The number of story points that the team has estimated for an issue
2. Filter or sort the issues. To do so, enter any valid JQL expression to filter or sort the issues based on any of their fields such as **Project**, **Assignee**, or **Sprint** with the Atlassian query language JQL.

From here, you can display your data in a table or use Grafana transformations to manipulate that issue data, run calculations, or turn the data into a time series graph. For more information, see [Applying a transformation \(p. 192\)](#).

## Time series query

To show time series data, choose a **Date** field along with a numeric field, then switch to graph visualization. For example: **Sprint Start Date**, **Story point estimate**.

The preceding example, on its own, is not very useful. The numeric field can be (and will most likely be) calculated from Transformations. Using the **Group By** Transformation would allow grouping by **Sprint Start Date** and summarizing the **Story point estimate** allowing a visualization of Story Points over time per Sprint. For more information about transformations, see [Applying a transformation \(p. 192\)](#).

## Templates and variables

To add a new Jira query variable, see [Adding a query variable \(p. 267\)](#). Use your Jira data source as the data source.

You can define variables on your dashboards and reference them in JQL expressions. For example, you can create a project status dashboard and choose between projects, or an epic status dashboard and choose different epics, or a task status dashboard and choose different assignees.

To get a dynamic list of projects, epics, assignees, and so on to choose from, create a Query type variable. Query type variables use JQL to query issues and return Projects, Epics, Assignees, or anything related to issues. The following is an example:

### Create an Assignee variable to get the status of issues by Assignee

1. Add a variable of type **Query** named **assignee**.
2. Select **Field: Assignee**.
3. )Optional) Add a JQL filter **project = 'your project'**.
4. Choose **Run** to see a list of Assignees.
5. Choose **Update** to add the variable to the dashboard.
6. Add a new panel to the dashboard and edit the JQL to filter using your new variable **assignee = \$assignee**.

Now, when choosing from the dropdown, you see only the issues assigned to that user.

Multi-value variables allow selecting multiple options and can be used as part of the IN clause. For example, **assignee IN (\$assignee)**.

## Using transformations from Grafana to answer common questions

Macros are variables that reference the Dashboard time window so you can filter issues only within the range of the Dashboard window. There are 2 macros:

- **\$\_\_timeFrom**
- **\$\_\_timeTo**.

The following example JQL query filters issues created within the dashboard time window:

```
createdDate >= $__timeFrom AND createdDate <= $__timeTo
```

## Get the most out of the data source

Using Grafana's transformations and other built-in features can help you meaningfully view your Jira data.

### Using transformations to augment JQL

While there are many Transformations in Grafana to choose from, the following provide a powerful augmentation to give JQL some of the features/power of SQL.

**Group By** This transformation provides a key feature that is not part of the standard Jira JQL syntax: Grouping. Using the **Group By** transformation, you can group by Sprints or other Issue fields, and aggregate by group to get metrics like velocity and story point estimates vs actual completed in a Sprint.

**Outer Join** Similar to SQL joins, you can join 2 or more queries together by common fields. This provides a way to combine datasets from queries and use other transformations to calculate values from multiple queries/datasets.

**Add Field from Calculation** Similar to SQL expressions, this transformation allows adding new fields to your dataset based on calculations of other fields. The fields used in the calculation can be from a single query or from queries you've joined together. You can also chain together calculations and perform calculations from calculated fields.

## Using transformations from Grafana to answer common questions

You can use Transformations to visualize, aggregate, group, and join datasets, along with many other types of transformations to transform simple results into answers for complex questions.

### How do I show Velocity per Sprint?

1. Select Fields: **Sprint Name, Story point estimate.**
2. Add a JQL filter: `project = "Your Project" AND type != epic AND status = done order by created ASC`
3. Add a **Group By** transformation:
  - Sprint Name | Group By
  - Story Point Estimate | Calculate | Total
4. Choose the **Bar Gauge** visualization.

### How do I show what was Completed vs Estimated in a Sprint?

1. Add a query. First, select Fields: **Sprint Name, Sprint Start Date,, Story point estimate.**  
Then add a JQL filter: `project = 'Your Project' AND type != epic`
2. Add a second query. First, select Fields: **Sprint Name, Sprint Start Date,, Story point estimate.**  
Then add a JQL filter: `project = 'Your Project' AND type != epic AND status = done`
3. Add a **Group By** transformation:
  - Sprint Name | Group By
  - Sprint Start Date | Group By
  - Story Point Estimate | Calculate | Total
4. Choose the **Graph** visualization.

### What is the Average time to complete issues in my project?

1. Add a query. First, select Fields: **Created, Status Category Changed.**  
Then add a JQL filter: `project = 'Your Project' AND type != epic AND status = done`
2. Add a transformation: **Add field from calculation**
  - Mode = Reduce Row
  - Calculation = Difference
3. Add a transformation: **Add field from calculation**
  - Mode = Binary Operation
  - Operation = Difference / 86000000
  - Alias = Days
4. Add a transformation: **Organize fields**
  - Hide Different field
5. Add a transformation: **Filter data by values**
  - Filter Type = Include
  - conditions = Match any
    - Field = Days | Match = Is Greater | Value = 1

6. Add a transformation: **Reduce**
  - Mode = Series to Rows
  - Calculations = mean
7. Choose the **Stat** visualization.

## MongoDB

The MongoDB Data source enables you to visualize data from MongoDB in Amazon Managed Grafana.

### Usage

#### Query editor

The query editor supports the same syntax as the MongoDB Shell, with some limitations: \* You can only run one command/query. \* Only read commands are supported: **find** and **aggregate** \* Most Object constructors are not supported (with the exception of **ISODate**, which is supported)

The editor expands upon the MongoDB Shell syntax in the following ways:

- **Database selection** – You can supply the name of the database in place of the normal "db":

**Note**

You can still use "db". It will refer to the default database in your connection string.

```
sample_mflix.movies.find()
```

- **Aggregate sorting** – Normally sorting happens with a step within the aggregate pipeline, however the MongoDB Atlas free tier doesn't allow sorting. We have expanded the syntax to allow it for those using the free tier.

**Note**

MongoDB doesn't perform the sort with this syntax. The sort happens after the results are queried from the collection.

```
sample_mflix.movies.aggregate({}).sort({"time": 1})
```

- With a blank editor, **Ctrl + Space** will show a selection of all the available databases.
- Entering a dot after the database will show a selection of all the available collections for that database.
- Entering a dot after the collection will show the available query methods.
- Entering a dot after the query method will show additional functions: sort/limit.

### Running the query

Press **Cmd + S** to run the query

#### Time series

When visualizing time series data, the plugin needs to know which field to use as the time. Simply project the field with a name alias of "time". The field data type must be a date.

You can coerce non-date data types to date. Doing so will allow using non-date fields as the time series time. The following example shows how to convert the int field "year" to a date that is projected as "time" using the MongoDB \$dateFromParts pipeline operator.

```
sample_mflix.movies.aggregate([
{"$match": { "year": {"$gt" : 2000} }},
{"$group": { "_id": "$year", "count": { "$sum": 1 } }},
{"$project": { "_id": 0, "count": 1, "time": { "$dateFromParts": {"year": "$_id", "month": 2} } } }
]
).sort({"time": 1})
```

## Diagnostics

### Diagnostic Commands

The following diagnostic commands are currently supported: "stats", "serverStatus", "replSetGetStatus", "getLog", "connPoolStats", "connectionStatus", "buildInfo", "dbStats", "hostInfo", "lockInfo"

Examples:

```
admin.connectionStatus() // run the connectionStatus command
admin.connectionStatus({"authInfo.authenticatedUserRoles": 1}) // run and only return the
"authInfo.authenticatedUserRoles" field
admin.connPoolStats({arg: "pool"}) // run the connPoolStats command and pass 1 argument
admin.serverStatus({args: {repl: 0, metrics: 0}}) // run the serverStatus command and pass
multiple args
```

## Macros

You can reference the dashboard time range in your queries.

- `$__timeFrom` — a macro that references the dashboard start time
- `$__timeTo` — a macro that references the dashboard end time

```
$__timeTo - `` sample_mflix.movies.find({released: {$gt:
"$__timeFrom"}}).sort({year: 1})
```

## Template variables

MongoDB supports the idea of "Compound Variables", which enable you to use one variable as multiple variables to perform complex multi-key filters.

To create a Compound Variable, use the naming convention of breaking the variables up by using underscores (must start with underscore): `_var1_var2` When querying, the response must be in the format: `val1-val2`

**Example: I want to filter results on both movie name and year.**

1. Create a variable of type Query: `_movie_year`
2. Set the variable query to a query that will return an array of items with one movie-year property, as shown in the following example.

```
// Example sample_mflix.movies.aggregate([
  {"$match": {year: {"$gt": 2011}}},
  {"$project": {_id: 0, movie_year: {"$concat":
    ["$title", " - ", {"$toString": "$year"}]}}}]
```



```
  ] )
```

```
// [{"movie-year": "Ted - 2016"},  
  {"movie-year": "The Terminator -  
  1985"}]
```

3. Now in your query, you can reference "Movie" and "Year" as separate template variables using the syntax "\$\_variable".

### Using ad-hoc filters

In addition to the standard "ad-hoc filter" type variable of any name, a second helper variable must be created. It should be a "constant" type with the name `mongodb\_adhoc\_query` and a value compatible with the query editor. The query result will be used to populate the selectable filters. You may choose to hide this variable from view as it serves no further purpose.

```
sample_mflix.movies.aggregate([  
  {"$group": { "_id": "$year" }},  
  {"$project": { "year": "$_id", "_id":  
    0 }} ] )
```

## New Relic

This section covers New Relic [APM](#) and [Insights](#) for Grafana.

### Features

- Template variables
  - Metric names
  - Metric values
- Annotations
- Aliasing
  - Metric names
  - Metric values
- Ad-hoc filters
  - Not currently supported
- Alerting

### Configuration

Add the data source, filling out the fields for your [admin API key](#), [personal API key](#) and [account ID](#).

### Usage

#### Service types

- **Metrics**; for querying New Relic APM via New Relic's [REST API](#).
- **Insights**; for querying New Relic Insights via [NRQL](#).

#### Aliases

You can combine plaintext with the following variables to produce custom output.

Variable	Description	Example value
<code>\$_nr_metric</code>	Metric name	CPU/User time
<code>\$_nr_metric_value</code>	Metric values	average_value

For example:

```
<para>
  Server: $_nr_server Metric: $_nr_metric
</para>
<programlisting>
```

### Templates and variables

1. Create a template variable for your dashboard. For more information, see [Templates and variables \(p. 265\)](#).
2. Select the "Query" type.
3. Select the "New Relic" data source.
4. Formulate a query using relative [REST API](#) endpoints (excluding file extensions).

List of available applications:

```
<para>
  applications
</para>
<programlisting>
```

List of available metrics for an application:

```
<para>
  applications/{application_id}/metrics
</para>
<programlisting>
```

### NRQL macros

To improve the writing experience when creating New Relic Query Language (NRQL) queries, the editor supports predefined macros:

- `$_timeFilter` (or `[[timeFilter]]`) will interpolate to `SINCE <from> UNTIL <to>`; based on your dashboard's time range.

Example:

```
<para>
  SELECT average(value) FROM $event_template_variable
  $_timeFilter TIMESERIES
</para>
<programlisting>
```

For further hints on how to use macros and template variables, refer to the editor's help section.

### Alert events

Select your New Relic data source and set additional filters. Without any filters set, all events will be returned.

If you want to filter events by *Entity ID*, use template variables because you will be able to select the entity name instead of ID. For example, to filter events for a particular application, create a variable `__app__` which retrieves a list of apps and uses it as an *Entity ID* filter.

### Deployment events

*Application ID* is a required field.

## Oracle Database

### Adding the data source

Select **Data sources** on the left panel of Grafana.

Select Add Datasource:

Enter **oracle** to find the data source.

Enter Oracle server details.

Enter a hostname (or IP address) along with the port number, and the username and password to connect.

With the tnsnames option toggle, any valid entry found in your tnsnames.ora configuration file can be used, along with basic authentication.

Similar to the previous example, but using Kerberos for authentication. See the kerberos specific setup guide for details on how to configure the OS or docker container to use kerberos.

Optionally change the time zone used to connect to the Oracle server and to be used by timezone aware macros. The default setting is UTC.

Save and Test the data source, you should see a green message with "Database Connection OK"

### Usage

#### Macros

To simplify syntax and to allow for dynamic parts, such as date range filters, the query can contain macros. The column name must be contained within double-quotes (").

Macro example	Description
*\$__time(dateColumn)*   Will be replaced by an expression to rename the column to `time`. For example, `dateColumn as time` * \$__timeEpoch(dateColumn)*	Will be replaced by an expression to rename the column to <code>time</code> and converting the value to unix timestamp (in milliseconds).
*\$__timeFilter(dateColumn)*   Will be replaced by a time range filter using the specified column name. For example, `dateColumn BETWEEN TO_DATE('19700101','yyyymmdd') + (1/24/60/60/1000) * 1500376552001`	Will be replaced by the start of the currently active time selection converted to <code>DATE</code> data type. For example, <code>TO_DATE('19700101','yyyymmdd') + (1/24/60/60/1000) * 1500376552001</code> .

Macro example	Description
AND TO_DATE('19700101','yyyymmdd') + (1/24/60/60/1000) * 1500376552002 ` * \$__timeFrom()*	
*\$__timeTo()*   Will be replaced by the end of the currently active time selection converted to `DATE` data type. * \$__timeGroup(dateColumn,"5m")*	Will be replaced by an expression usable in GROUP BY clause.
*\$__timeGroup(dateColumn,"5m",[ fillvalue])*	Will be replaced by an expression usable in GROUP BY clause. Providing a fillValue of NULL or floating value will automatically fill empty series in time range with that value. For example, \$__timeGroup(createdAt, '1m', 0.*\$__timeGroup(dateColumn,"5m", 0)*.
*\$__timeGroup(dateColumn, '5m', NULL) *   Same as above but NULL will be used as the previous series previous)*	Will be replaced by an expression usable in GROUP BY clause. Providing a fillValue of NULL or floating value will automatically fill empty series in time range with that value. For example, \$__timeGroup(createdAt, '1m', 0.*\$__timeGroup(dateColumn,"5m", 0)*.
*\$__unixEpochFilter(dateColumn)*   Will be replaced by a time range filter using the specified column name with times represented as unix timestamp (in milliseconds). For example, `dateColumn >= 1500376552001 AND dateColumn <= 1500376552002` * \$__unixEpochFrom()*	Will be replaced by the start of the currently active time selection as unix timestamp. For example, 1500376552001.
*\$__unixEpochTo()*	Will be replaced by the end of the currently active time selection as unix timestamp. For example, 1500376552002.

The plugin also supports notation using braces { }. Use this notation when queries are needed inside parameters.

#### Note

Use one notation type per query. If the query needs braces, all macros in the query must use braces.

```

$__timeGroup{"dateColumn", '5m'}
$__timeGroup{SYS_DATE_UTC("SDATE"), '5m'}
$__timeGroup{FROM_TZ(CAST("SDATE" as timestamp), 'UTC'), '1h'}

```

The query editor has a **Generated SQL** link that shows up after a query has run, while in panel edit mode. When you choose the link, it expands and shows the raw interpolated SQL string that was run.

## Table queries

If the **Format as** query option is set to **Table** then you can basically do any type of SQL query. The table panel will automatically show the results of whatever columns & rows your query returns. You can control the name of the Table panel columns by using regular as SQL column selection syntax.

## Time series queries

If you set **Format as** to **Time series**, for use in Graph panel for example, the query must return a column named `time` that returns either a SQL datetime or any numeric data type representing unix epoch in

seconds. Grafana interprets DATE and TIMESTAMP columns without explicit time zone as UTC. Any column except time and metric is treated as a value column. You may return a column named metric that is used as metric name for the value column.

The following code example shows the metric column.

```
SELECT
  $__timeGroup("time_date_time", '5m') AS time,
  MIN("value_double"),
  'MIN' as metric
FROM test_data
WHERE $__timeFilter("time_date_time")
GROUP BY $__timeGroup("time_date_time", '5m')
ORDER BY time
```

### More queries – using oracle-fake-data-gen

```
SELECT
  $__timeGroup("createdAt", '5m') AS time,
  MIN("value"),
  'MIN' as metric
FROM "grafana_metric"
WHERE $__timeFilter("createdAt")
GROUP BY $__timeGroup("createdAt", '5m')
ORDER BY time
```

The following code example shows a Fake Data time series.

```
SELECT
  "createdAt",
  "value"
FROM "grafana_metric"
WHERE $__timeFilter("createdAt")
ORDER BY "createdAt" ASC
```

```
SELECT
  "createdAt" as time,
  "value" as value
FROM "grafana_metric"
WHERE $__timeFilter("createdAt")
ORDER BY time ASC
```

The following example shows a useful table result.

```
select tc.table_name Table_name
,tc.column_id Column_id
,lower(tc.column_name) Column_name
,lower(tc.data_type) Data_type
,nvl(tc.data_precision,tc.data_length) Length
,lower(tc.data_scale) Data_scale
,tc.nullable nullable
FROM all_tab_columns tc
,all_tables t
WHERE tc.table_name = t.table_name
```

## Templating

Instead of hardcoding things such as server, application and sensor name in your metric queries you can use variables in their place. Variables are shown as dropdown select boxes at the top of the dashboard. These dropdown boxes make it easy to change the data being displayed in your dashboard.

### Query variable

If you add a template variable of the type `Query`, you can write a Oracle query that can return things such as measurement names, key names or key values that are shown as a dropdown select box.

For example, you can have a variable that contains all values for the `hostname` column in a table if you specify a query like this in the templating variable `Query` setting.

```
SELECT "hostname" FROM host
```

A query can return multiple columns and Grafana will automatically create a list from them. For example, the following query will return a list with values from `hostname` and `hostname2`.

```
SELECT "host.hostname", "other_host.hostname2" FROM host JOIN other_host ON host.city = other_host.city
```

To use time range dependent macros such as `$__timeFilter("time_column")` in your query the refresh mode of the template variable needs to be set to *On Time Range Change*.

```
SELECT "event_name" FROM event_log WHERE $__timeFilter("time_column")
```

Another option is a query that can create a key/value variable. The query should return two columns that are named `__text` and `__value`. The `__text` column value should be unique (if it is not unique then the first value is used). The options in the dropdown list will have a text and value that allows you to have a friendly name as text and an id as the value. The following example code shows a query with `hostname` as the text and `id` as the value.

```
SELECT "hostname" AS __text, "id" AS __value FROM host
```

You can also create nested variables. For example, if you had another variable named `region`. Then you could have the hosts variable only show hosts from the current selected region with a query like this (if `region` is a multi-value variable then use the `IN` comparison operator rather than `=` to match against multiple values).

```
SELECT "hostname" FROM host WHERE region IN('$region')
```

## Using variables in queries

Template variable values are only quoted when the template variable is a `multi-value`.

If the variable is a multi-value variable then use the `IN` comparison operator rather than `=` to match against multiple values.

There are two syntaxes:

`<varname>` Example with a template variable named `hostname`:

```
SELECT
  "atimestamp" as time,
  "aint" as value
FROM table
WHERE $__timeFilter("atimestamp") AND "hostname" IN('$hostname')
ORDER BY "atimestamp" ASC
```

`[ [varname] ]` Example with a template variable named `hostname`:

```
SELECT
  "atimestamp" as time,
  "aint" as value
FROM table
WHERE $__timeFilter("atimestamp") AND "hostname" IN('[[hostname]]')
ORDER BY atimestamp ASC
```

## Salesforce

The Salesforce data source allows you to visualize data from Salesforce within Amazon Managed Grafana.

To use this data source, you must have a [Salesforce](#) account and a [Salesforce Connected App](#).

### Known limitations

- Ad-hoc filters are not supported yet.
- Only SOQL queries, and data that is accessible via SOQL are currently supported. SOSL and SAQL query formats are not yet supported.

### Required settings

The following settings are required.

#### Note

The plugin currently uses the OAuth 2.0 Username-Password Flow. The required callback URL in the Connected App is not used. Thus, you can set it to any valid URL.

Name	Description
Enable OAuth settings	You must check this to enable OAuth.
Callback URL	Not used in this plugin, so you can specify any valid URL.
Selected OAuth Scopes (minimum requirements)	Access and Manage your data (api).

Name	Description
Require Secret for Refresh Token Flow	You can either enable or disable this.

## Adding the data source

1. Open the Grafana console in the Amazon Managed Grafana workspace and make sure you are logged in.
2. In the side menu under **Configuration** (the gear icon), choose **Data Sources**.
3. Choose **Add data source**.

### Note

If you don't see the **Data Sources** link in your side menu, it means that your current user does not have the `Admin` role.

4. Select **Salesforce** from the list of data sources.
5. Enter the following information:
  - For **User Name**, enter the username for the Salesforce account that you want to use to connect and query Salesforce.
  - For **Password**, enter the password for that user.
  - For **Security Token**, enter the security token for that user.
  - For **Consumer Key**, enter A Consumer key to connect to Salesforce. You can obtain this from your Salesforce Connected App.
  - For **Consumer Secret**, enter A Consumer secret to connect to Salesforce. You can obtain this from your Salesforce Connected App.
  - For **Use Sandbox**, select this if you want to use a Salesforce sandbox.

## Query the Salesforce data source

The query editor supports the modes Query Builder and SOQL Editor. SOQL stands for [Salesforce Object Query Language](#).

### Query Builder (SOQL Builder)

Query Builder is a user friendly interface for building SOQL queries. If you are not familiar with writing SOQL queries, you can use this mode to build the SOQL to query Salesforce objects. The **FROM** field in the query builder refers to the entity or entities in Salesforce. You need to select the **FROM** field before any other operation in the query builder. After you choose the **FROM** field, you need to choose the builder mode. SOQL Builder currently supports the following modes.

- **List**— List the items with their fields from the selected table/salesforce. Use this mode to get results such as, "Show me list of opportunities created in this fiscal quarter along with their name, value, and stage."
- **Aggregate**— Aggregate the items in an entity. Use this mode to get results such as, "Count the opportunities created in last month." or "What is the total value of the opportunities grouped by their stage name?"
- **Trend**— Display the aggregated results over time. Use this mode to get results such as, "Count the number of opportunities by CreatedDate." or "What is the total sum of value grouped by opportunities' closing dates."



After you choose the **Entity**/**FROM** and the **mode** in the query editor, build your query using the following options.

Fields	Applicable to	Descriptions
SELECT	ALL	Select the list of fields that you want to see. For the aggregate or trend view, also select how you want to aggregate the values.
WHERE	ALL	(Optional) Specify the filter conditions. The results are filtered based on the conditions that you select.
ORDER BY	LIST, AGGREGATE	(Optional) Select the field name and the sort order that you want for the results.
LIMIT	LIST, AGGREGATE	(Optional) Limit the number of results returned. The default is 100.
GROUP BY	AGGREGATE	(Optional) Select the field if you want to split the aggregated value by any specific field.
TIME FIELD	TREND	Specify the date field by which you want to group your results. Results are filtered based on Grafana's time picker range.

As you configure the preceding fields in the query editor, you will also see a preview of generated SOQL below the query editor. If you are blocked with any limitations in the query builder, you can safely switch to SOQL Editor, where you can customize the generated SOQL query.

### SOQL editor

The raw SOQL editor provides the option to query Salesforce objects via raw a SOQL query. The SOQL editor provides autocomplete suggestions, such as available entities per tables and corresponding fields. Use Ctrl+Space after SELECT or WHERE to see the available entities per tables. You can see the available fields, if you enter a dot after the entity name.

### Shortcuts

Use CTRL + SPACE to show code completion, which shows available contextual options.

CMD + S runs the query.

### Query as time series

Make a time series query by aliasing a date field to time, and a metric field to metric, then grouping by the metric and date. The following is an example:

```
SELECT sum(Amount) amount, CloseDate time, Type metric from Opportunity
group by Type, CloseDate
```

### Macros

To filter by the dashboard time range, you can use Macros in your SOQL queries:

- `$__timeFrom`— Will be replaced by the start of the currently active time selection converted to the time data type.
- `$__timeTo`— Will be replaced by the end of the currently active time selection converted to the time data type.

- `$__quarterStart`— The start of the fiscal quarter (derived from Salesforce Fiscal Year Settings).
- `$__quarterEnd`— The end of the fiscal quarter (derived from Salesforce Fiscal Year Settings).

```
SELECT UserId, LoginTime from LoginHistory where LoginTime > $__timeFrom
```

## Templates and variables

To add a new Salesforce query variable, see [Adding a query variable \(p. 267\)](#). Use your Salesforce data source as your data source. You can use any SOQL query here.

If you want to use name/value pairs, for example a user id and user name, return two fields from your SOQL query. The first field will be used as the ID. You may want to do this when you want to filter by key (ID, etc) in your query editor SOQL.

Use the variable in your SOQL queries by using Variable syntax. For more information, see [Variable syntax \(p. 266\)](#).

## SAP HANA

[SAP HANA](#) is a high-performance, in-memory database that speeds up data-driven, real-time decisions and \ actions. It is developed and marketed by SAP. The SAP HANA data source plugin helps you to connect your SAP HANA instance with Grafana.

With the SAP HANA Grafana Enterprise plugin, you can visualize your SAP HANA data alongside all of your other data sources in Grafana as well as log and metric data in context. This plugin includes a built-in query editor, supports annotations, and it allows you to set alerting thresholds, control access, set permissions, and more.

## Features

- **Query editor**— The plugin comes with an built-in SQL query editor with syntax highlighting that allows you to visualize time series or table data and auto completes basic Grafana macros.
- **Data source permissions**— Control who can view or query SAP HANA data in Grafana.
- **Annotations**— Overlay SAP HANA events or data on any Grafana graph to correlate events with other graph data.
- **Alerting**— Set alerts-based metrics stores in SAP HANA.
- **Variables for queries**— Create template variables in Grafana, which are based on SAP HANA data, and include variables in SAP HANA queries to make dashboards interactive.

## Adding the data source

1. Open the Grafana console in the Amazon Managed Grafana workspace and make sure you are logged in.
2. In the side menu under **Configuration** (the gear icon), choose **Data Sources**.
3. Choose **Add data source**.

### Note

If you don't see the **Data Sources** link in your side menu, it means that your current user does not have the `Admin` role.

4. Select **SAP HANA** from the list of data sources.
5. In the Config editor, enter the following information:
  - For **Server address**, Provide the address of the SAP HANA instance. Example : `xxxxxxx-xxxx-xxxx-xxxx-xxxxx.hana.trial-us10.hanacloud.ondemand.com`.

- For **Server port**, provide the port of the SAP HANA instance.
- For **Username**, enter the username to use to connect to the SAP HANA instance.
- For **Password**, enter the password for this user.
- (Optional) Enable **Skip TLS verify** if you want to skip TLS verification.
- (Optional) Enable **TLS Client Auth** if you need to provide a client cert and key.
- (Optional) Enable **With CA cert** if you want to enable verifying self-signed TLS Certs.
- (Optional) For **Default schema**, enter a default schema to be used. If you omit this, you will need to specify the schema in every query.

### Access and permissions

To connect Grafana to SAP HANA, use dedicated credentials. Only provide required permissions to the user. First, create a restricted user with username and password. The following query is an example to create a restricted user. This query also disables the force password change.

```
CREATE RESTRICTED USER <USER> PASSWORD <PASSWORD> NO FORCE_FIRST_PASSWORD_CHANGE;
```

Next, allow the the user to connect the system through clients such as Grafana with the following:

```
ALTER USER <USER> ENABLE CLIENT CONNECT;
```

Finally, give the user access to the necessary views, tables, and schemas.

```
ALTER USER <USER> GRANT ROLE PUBLIC;  
GRANT SELECT ON SCHEMA <SCHEMA> TO <USER>;
```

### User level permissions

Limit access to SAP HANA by clicking on the Permissions tab in the datasource configuration page to enable data source permissions. On the permission page, Admins can enable permissions and restrict query permissions to specific Users and Teams.

## Query editor

The SAP HANA Grafana plugin comes with an SQL query editor where you can enter any HANA queries. If your query return timeseries data, you can format it as timeseries for visualizing them in a graph panel. The query editor provides auto completion for supported Grafana macros and syntax highlighting of your SQL query.

## Annotations

You can use SAP HANA queries as the sources of Grafana annotations. Your annotation query should return at least one time column and one text column. For more information about annotations, see [Annotations \(p. 238\)](#).

### To create annotations from SAP HANA

1. Choose the **Dashboard settings** gear icon.
2. From the left menu, choose **Annotations, New**.
3. From the **Data source** drop-down menu, select your SAP HANA data source instance.
4. In the **Query** field, enter a SAP HANA query that returns at least one time field and one text field.
5. In the **Format as** drop-down menu, select **Time Series**.

6. For each annotation, configure the **From** fields.

## Templates and variables

To add a new SAP HANA query variable, see [Adding a query variable \(p. 267\)](#). Use your SAP HANA data source as your data source.

The following example query returns the distinct list of username from the users table.

```
select distinct("username") from "users"
```

### Note

Be sure to only select one column in your variable query. If your query returns two columns, the first column will be used as display value and the second column will be used as the actual value of the variable. If your query returns more than two columns, they will be rejected.

## Templates and variables

You can use any Grafana variable in your query. The following examples shows how to use the single / multi variable in your query.

```
-- For example, following query
select * from "users" where "city" = ${city}
-- will be translated into
select * from "users" where "city" = 'london'
--- where you can see ${city} variable translated into actual value in the variable
```

Similar to text, variables also works for numeric fields. In the below example, `${age}` is a text box variable where it accepts numbers and then compares against the numeric field in the table.

```
select * from "users" where "age" > ${age}
--- will be translated into
select * from "users" where "age" > '36'
```

If your variable returns multiple values, then you can use it in SAP HANA query's in condition like below. Note the brackets surrounding the variable to make the where in condition valid in SAP HANA.

```
select * from "users" where "city" in (${cities})
--- will be translated into
select * from "users" where "city" in ('london','perth','delhi')
--- where you can see ${cities} turned into a list of grafana variables selected.
--- You can also write the same query using shorthand notation as shown below
select * from "users" where "city" in (${cities})
```

## Macros

- `$__timeFilter(<time_column>)`— Applies Grafana's time range to the specified column when used in the raw query. Applicable to date/timestamp/long time columns.
- `$__timeFilter(<time_column>, <format>)`— Same as above. But gives the ability to specify the format of the time\_column stored in the database.
- `$__timeFilter(<time_column>, "epoch", <format>)`— Same as above but can be used when your time column is in epoch. format can be one of 's','ms' and 'ns'.
- `$__fromTimeFilter(<time_column>)`— Same as above but can be used when your time column is in epoch. format can be one of 's','ms' and 'ns'.
- `$__fromTimeFilter(<time_column>, <comparison_predicate>)`— Same as above but able to specify comparison\_predicate.

- `$__fromTimeFilter(<time_column>, <format>)`— Same as above but able to specify format of the time column.
- `$__fromTimeFilter(<time_column>, <format>, <comparison_predicate>)`— Same as above but able to specify comparison\_predicate.
- `$__toTimeFilter(<time_column>)`— Returns time condition based on grafana's to time over a time field.
- `$__toTimeFilter(<time_column>, <comparison_predicate>)`— Same as above but able to specify comparison\_predicate.
- `$__toTimeFilter(<time_column>, <format>)`— Same as above but able to specify format of the time column.
- `$__toTimeFilter(<time_column>, <comparison_predicate>)`— Same as above but able to specify comparison\_predicate.
- `$__timeGroup(<time_column>, <interval>)`— Expands the time column into interval groups. Applicable to date/timestamp/long time columns..

### `$__timeFilter(<time_column>)` macro

The following example explains the `$__timeFilter(<time_column>)` macro:

```
- In the following example, the query
select ts, temperature from weather where $__timeFilter(ts)
--- will be translated into
select ts, temperature from weather where ts > '2021-02-24T12:52:48Z' AND ts <
'2021-03-24T12:52:48Z'
--- where you can see the grafana dashboard's time range is applied to the column ts in the
query.
```

### `$__timeFilter(<time_column>, <format>)` macro

In some cases, time columns in the database may stored in custom formats. The following example explains the `$__timeFilter(<time_column>, <format>)` macro, which helps to filter custom timestamps based on grafana's time picker:

```
SELECT TO_TIMESTAMP("TS",'YYYYMMDDHH24MISS') AS METRIC_TIME , "VALUE" FROM "SCH"."TBL"
WHERE $__timeFilter("TS",'YYYYMMDDHH24MISS') -- TS is in 20210421162012 format
SELECT TO_TIMESTAMP("TS",'YYYY-MON-DD') AS METRIC_TIME , "VALUE" FROM "SCH"."TBL" WHERE
$__timeFilter("TS",'YYYY-MON-DD') -- TS is in 2021-JAN-15 format
```

In the macro, the format can be one of the valid HANA formats matching your timestamp column. For example, `YYYYMMDDHH24MISS` is a valid format when your data is stored in `20210421162012` format.

### `$__timeFilter(<time_column>, "epoch" <format>)` macro

In some cases, you may have timestamp stored as epoch timestamps in your DB. The following example explains the `$__timeFilter(<time_column>, "epoch" <format>)` macro which helps to filter epoch timestamps based on grafana's time picker. In the macro, format can be one of `ms`, `s` or `ns`. If not specified, `s` will be treated as default format.

```
SELECT ADD_SECONDS('1970-01-01', "TIMESTAMP") AS "METRIC_TIME", "VALUE" FROM "SCH"."TBL"
WHERE $__timeFilter("TIMESTAMP","epoch") -- Example : TIMESTAMP field stored in
epoch_second format 1257894000
SELECT ADD_SECONDS('1970-01-01', "TIMESTAMP") AS "METRIC_TIME", "VALUE" FROM "SCH"."TBL"
WHERE $__timeFilter("TIMESTAMP","epoch","s") -- Example : TIMESTAMP field stored in
epoch_second format 1257894000
SELECT ADD_SECONDS('1970-01-01', "TIMESTAMP"/1000) AS "METRIC_TIME", "VALUE" FROM
"SCH"."TBL" WHERE $__timeFilter("TIMESTAMP","epoch","ms") -- Example : TIMESTAMP field
stored in epoch_ms format 1257894000000
```

```
SELECT ADD_SECONDS('1970-01-01', "TIMESTAMP"/1000000000) AS "METRIC_TIME", "VALUE" FROM
"SCH"."TBL" WHERE $__timeFilter("TIMESTAMP","epoch","ns") -- Example : TIMESTAMP field
stored in epoch_nanoseconds format 1257894000000000000
```

Instead of using third argument to the `$__timeFilter`, you can use one of `epoch_s`, `epoch_ms` or `epoch_ns` as your second argument..

```
SELECT ADD_SECONDS('1970-01-01', "TIMESTAMP"/1000) AS "METRIC_TIME", "VALUE" FROM
"SCH"."TBL" WHERE $__timeFilter("TIMESTAMP","epoch","ms")
-- is same as
SELECT ADD_SECONDS('1970-01-01', "TIMESTAMP"/1000) AS "METRIC_TIME", "VALUE" FROM
"SCH"."TBL" WHERE $__timeFilter("TIMESTAMP","epoch_ms")
```

### **`$__fromTimeFilter()` and `$__toTimeFilter()` macros**

The `$__fromTimeFilter()` macro expands to a condition over a time field based on Grafana time picker's from time.

This accepts three parameters. First parameter is time field name. You can pass `comparison_predicate` or format of the time column as second argument. If you want to pass both, then format is second parameter and use `comparison_predicate` as your third parameter.

**<format>** If the format is not specified, plugin will assume that the time column is of timestamp/date type. If your time column is stored in any other format than timestamp/date, then pass the format as second argument. **<format>** can be one of `epoch_s`, `epoch_ms`, `epoch_ns` or any other custom format like `YYYY-MM-DD`.

**<comparison\_predicate>** optional parameter. If not passed, plugin will use `>` as comparison predicate. **<comparison\_predicate>** can be one of `=`, `!=`, `<>`, `<`, `<=`, `>`, `>=`

`$__toTimeFilter()` works the same as `$__fromTimeFilter()`. Instead of using Grafana's from time, it will use to time. Also the default comparison predicate will be `<`.

### **`$__timeGroup(<time_column>, <interval>)`**

For example, the macro `$__timeGroup(timecol,1h)` is expanded to `SERIES_ROUND("timecol", 'INTERVAL 1 HOUR')` in the query.

The following example explains the `$__timeGroup(<time_column>, <interval>)` macro.

```
SELECT $__timeGroup(timestamp,1h), "user", sum("value") as "value"
FROM "salesdata"
WHERE $__timeFilter("timestamp")
GROUP BY $__timeGroup(timestamp,1h), "user"
ORDER BY $__timeGroup(timestamp,1h) ASC
```

This is translated into the following query where `$__timeGroup(timestamp,1h)` is expanded into `SERIES_ROUND("timestamp", 'INTERVAL 1 HOUR')`.

```
SELECT SERIES_ROUND("timestamp", 'INTERVAL 1 HOUR') as "timestamp", "user", sum("value")
as "value"
FROM "salesdata"
WHERE "timestamp" > '2020-01-01T00:00:00Z' AND "timestamp" < '2020-01-01T23:00:00Z'
GROUP BY SERIES_ROUND("timestamp", 'INTERVAL 1 HOUR'), "user"
ORDER BY "timestamp" ASC
```

### **Note**

When using group by with `$__timeGroup` macro, make sure that your select, sort by fields follows the same name as your group by field. Otherwise, HANA may not recognize the query.

If you don't want to hard code the interval in \$\_\_timeGroup() function, then you can leave that to Grafana by specifying \$\_\_interval as your interval. Grafana will calculate that interval from dashboard time range. Example query:

```
SELECT $__timeGroup(timestamp, $__interval), sum("value") as "value"
FROM "salesdata"
WHERE $__timeFilter("timestamp")
GROUP BY $__timeGroup(timestamp, $__interval)
ORDER BY $__timeGroup(timestamp, $__interval) ASC
```

That query is translated into the followin query based on the dashboard time range.

```
SELECT SERIES_ROUND("timestamp", 'INTERVAL 1 MINUTE'), sum("value") as "value"
FROM "salesdata"
WHERE "timestamp" > '2019-12-31T23:09:14Z' AND "timestamp" < '2020-01-01T23:17:54Z'
GROUP BY SERIES_ROUND("timestamp", 'INTERVAL 1 MINUTE')
ORDER BY SERIES_ROUND("timestamp", 'INTERVAL 1 MINUTE') ASC
```

## Alerting

### To set up a SAP HANA alert in Grafana

1. Create a graph panel in your dashboard.
2. Create a SAP HANA query in time series format.
3. Choose the **Alert** tab and specify the alerting criteria.
4. Choose **Test Rule** to test the alert query.
5. Specify the alert recipients, message, and error handling.
6. Save the dashboard.

### Alerting on non-timeseries data

To alert on non-timeseries data, use the TO\_TIMESTAMP('\${\_\_to:date}') macro to make non-timeseries metrics into timeseries. This will convert your metric into single point time series query. The format of the query is given below

```
SELECT TO_TIMESTAMP('${__to:date}'), <METRIC> FROM <TABLE# WHERE <YOUR CONDITIONS>
```

In the following example, a table has four fields called username, age, city and role. This table doesn't have any time field. We want to notify when the number of users with dev role is less than three.

```
SELECT TO_TIMESTAMP('${__to:date}'), count(*) as "count" FROM (
  SELECT 'John' AS "username", 32 AS "age", 'Chennai' as "city", 'dev' as "role" FROM
  dummy
  UNION ALL SELECT 'Jacob' AS "username", 32 AS "age", 'London' as "city", 'accountant' as
  "role" FROM dummy
  UNION ALL SELECT 'Ali' AS "username", 42 AS "age", 'Delhi' as "city", 'admin' as "role"
  FROM dummy
  UNION ALL SELECT 'Raja' AS "username", 12 AS "age", 'New York' as "city", 'ceo' as
  "role" FROM dummy
  UNION ALL SELECT 'Sara' AS "username", 35 AS "age", 'Cape Town' as "city", 'dev' as
  "role" FROM dummy
  UNION ALL SELECT 'Ricky' AS "username", 25 AS "age", 'London' as "city", 'accountant' as
  "role" FROM dummy
  UNION ALL SELECT 'Angelina' AS "username", 31 AS "age", 'London' as "city", 'cxo' as
  "role" FROM dummy
) WHERE "role" = 'dev'
```

## ServiceNow

This is the ServiceNow data source that is used to connect to ServiceNow instances.

### Features

- Queries
  - Stat API Queries
  - Table API Queries
    - Incidents, Changes, and any other table
- Alerts
- Annotations (beta feature)
- Template Variables

### Configuration

Select data sources on the left panel of Grafana.

Select Add Datasource:

Enter **servicenow** to find the data source plugin:

Enter ServiceNow URL:

Choose **Save & Test**. You should see a green message with "ServiceNow Connection OK".

### Example dashboards

Pre-made dashboards are included with the plugin and can be imported through the data source configuration page, under the dashboards tab.

### Usage

There are two ways to return data in the query editor.

- TableAPI
- AggregateAPI

Users can currently choose between querying pre-defined tables, such as the following:

- Changes
- Incidents

Or, as of v1.4.0, an API-driven list of tables and fields using the **Other (Custom Table)** option. This option will allow you to query data that is in any table available to the user used to set up the ServiceNow data source.

The **Custom Table** option should support all of the same features as the pre-defined table lists.

### TableAPI queries

The TableAPI returns data suitable for displaying in a table panel. It allows for an ordered selection of fields to display plus filtering options. The query editor also provides a field to limit the number of rows returned by a query.

Example table panel showing results from the previous query.



## Show

The *Show* row provides a selector for a field to be displayed. Multiple fields can be also be specified. The fields will be returned in the exact order specified.

## Display Values

The *Display Values* flag will cause the query to return human-friendly values, or display vaules, instead of numeric values.

For example, a severity of 1 without this flag would only display 1. If the flag is enabled, the value displayed will be 1 – High.

According to the [ServiceNow API documentation](#), this can have a negative performance impact.

### Note

[...] specifying the display value may cause performance issues since it is not reading directly from the database and may include referencing other fields and records.

## Filters (general)

The *Filters* row provides the ability to narrow down the displayed rows based on multiple field and value criteria.

All filters are combined with an *AND* or an *OR* operation.

The following fields are available when not using a custom table (this list will expand in the future).

```
Active
Asset
Group
Assigned To
Escalation
Issue Number
Description
Priority
State
Type
Change Risk
Change State
Start Date
End Date
On Hold
```

When selecting a custom table, fields are automatically populated from the Service Now API.

## Date filters

Time Field	Operators	Value
Opened At	At or Before Today Not Today Before At or Before After At or After	timestamp javascript:gs.daysAgo(30)
Activity Due		
Closed At		

Time Field	Operators	Value
Due Date		
Expected Start		
Reopened Time		
Resolved At		
Work End		
Work Start		
Ignore Time		

For additional date values, see: [https://developer.servicenow.com/app.do#!/api\\_doc?v=newyork&id=r\\_SGSYS-dateGenerate\\_S\\_S](https://developer.servicenow.com/app.do#!/api_doc?v=newyork&id=r_SGSYS-dateGenerate_S_S)

#### Operators (general, string-based)

- Starts With
- Ends With
- Like
- Not Like
- Equals
- Not Equals
- Is Empty

#### Operators (time-based)

- Today
- Not Today
- Before
- At or Before
- After
- At or After

#### Values

Value selection depends on the type of filter selected.

- Boolean filters have True/False options
- Text filters will allow typing any value
- Escalation, Priority has a fixed set of numerical values

## Sort By

The *Sort By* row provides the ability to narrow down the displayed rows based on multiple field and value criteria.

All filters are combined with an *AND* operation. Support for additional operators will be added.

## Limit

A row limit can be specified to prevent returning too much data. The default value is 25.

## Time Field

The *Time Field* is what turns your queried data into a time series. Your data being handled as a time series means that values in your selected "time field" that do not fall within your dashboard / panel's time range will not be displayed.

The default time field used is "Opened At", but can be changed to any available field that holds a time value.

A special value "Ignore Time" is provided to allow results "up until now" and also to enable the filters to control what data is displayed.

## AggregateAPI queries (Stats)

The AggregateAPI will always return metrics, with the following aggregations: avg, min, max, sum. Filtering is also available to narrow queries.

## Show

The *Show* row provides a selector for a metric to be displayed. Multiple metrics can also be specified.

## Filters (general)

Aggregate *Filters* provide the ability to narrow down the displayed metrics based on field and value criteria, similar to the table option.

All filters are combined with an *AND* operation. Support for additional operators will be added.

Stat filter options are the same as the TableAPI.

## Aggregation

There are four types of metric aggregations, plus a "count":

- Average
- Minimum
- Maximum
- Sum
- Count - this returns the "number" of metrics returned by a query

## Group By

This selector provides the ability to split metrics into lesser aggregates. Grouping by "priority" would return the metrics with a "tag" of priority and the unique values separated.

## Templating

Instead of hardcoding names in your queries, you can use variables in their place. Variables are shown as dropdown select boxes at the top of the dashboard. You can use these dropdown boxes to change the data being displayed on your dashboard.

See the example in the **Query Variable** section on how to add a query variable and reference that with a Template value.

### Query variable

If you add a template variable of the type `Query`, you can write a query that can return items such as category names, key names, or key values that are shown as a dropdown select box.

For example, you can have a variable that contains all values for `categories` by specifying a query such as this in the templating variable `Query` setting.

When choosing the **Query** setting, a **Filter** section is displayed, allowing you to choose a **Type** and **Field**. Currently, **Type** is limited to Incidents and Changes. When selecting a type, you are provided with a list of fields applicable to that Type. Once a **Type** and **Field** are selected, a preview of values will be displayed at the bottom showing the options available for that Type/Field. Those values will be displayed in a dropdown list on the Dashboard, which you can use along with Templating to filter data on your Dashboard Panels.

For example, if you add a Variable named `category` then select Type = Incidents and Field = Category, you will see a list of options for Category. If you then add a Filter to a panel, and select Category Equals `${category}`, the panel data will show only data for that Category that is selected from the Dashboard dropdown list.

Import the **Incidents By Category** dashboard to see an example.

### Using variables in queries

There are two syntaxes:

`${varname}` Example with a template variable named `hostname`:

`[ [varname] ]` Example with a template variable named `hostname`:

## Alerting

Standard Grafana alerting is supported. Any queries defined in a graph panel can be used to generate alerts.

The following is an example query and an alert. This query will return a graph of all open critical high priority incidents:

This alert will be initiated when there are more than five open critical high priority incidents:

Testing the alert rule will display output from the alert rule, and selecting the state history will show the alert transitioning from ok to pending to alerting.

The graph view will show a vertical line and the heart icon at the top will turn orange while the alert is pending.

Once the criteria for alerting has been met, the rule transitions to red.

In the graph view, the red vertical line will appear and the heart icon at the top will turn red.

### Writing incidents for alerts

#### Beta feature

- Configure a Notification Channel for your ServiceNow data source.

This will configure a [Grafana Notification Channel](#) which uses your configured user to create incidents on the ServiceNow instance for this data source.

This action requires that the ServiceNow data source user has permissions for writing incidents.

### Using an HTTP proxy

When using an HTTP proxy, Amazon Managed Grafana will need the following environment variable(s) set to the location of the proxy:

- `HTTP_PROXY` (or `http_proxy`)
  - Full path - `http://host:port`
  - or just: `host:port`
- `HTTPS_PROXY` (or `https_proxy`):
  - Full path - `https://host:port`
  - or just: `host:port`

## Annotations

Grafana Annotations are a **beta feature** as of v1.4.0 of this data source. Annotations give you the ability to overlay events on graphs.

The Annotations query supports the same options as the standard query editor with a few minor differences:

- Only one "Show" column is selectable. This is likely going to be fixed in a future improvement.
- The time field is required.

## FAQ

### What if we don't have the ITSM Roles Plugin?

#### Administrator access is required to perform the following actions

Option 1: Grant Grafana user admin permissions to allow access to all tables.

Option 2: Create a role and apply ACLs to all tables that must be accessed by Grafana.

Administrator access is required to perform the following actions.

1. The logged in administrator needs to elevate access to `security_admin`.
  - a. In the top right navigation pane, choose the profile icon. The profile icon has dropdown caret indicator.
  - b. From the dropdown list, choose **Elevate Roles**.
  - c. From the modal that is shown, select the **security\_admin** check box.
  - d. Choose OK.
2. Create a new role with whatever naming convention you want.
  - a. Navigate to the roles section in the left hand navigation System Security => Users and Groups => Roles
  - b. Choose **New** at the top.
  - c. Enter a name for the role and a relevant description.
  - d. Choose **Submit**.
3. Create a new user or modify an existing user with the needed roles.
  - a. The role you create in Step 2

- b. `personalize_dictionary`
  - c. `personalize_choices`
  - d. `cmdb_read` (this will grant read access to all cmdb tables)
4. Create Table ACLs for the required tables and fields.
  - Create an ACL for the `sys_db_object` table.
    - i. In the second search header column **Name**, enter `sys_db_object`, and press **Enter**.
    - ii. The filtered result should show **Table**. Choose **Table** to navigate into the record.
    - iii. On the tab section, choose **Controls**.
    - iv. On the lower portion of the page, make sure that **Access Controls** is the selected tab.
    - v. Choose **New** to create a new ACL.
    - vi. Change the **Operation** selection to read.
    - vii. In the **Requires Role** section in the lower part of the screen, choose (double-click) **Insert New Row**, and search for the role that you created.
    - viii. After you select the role you created, choose the green check mark.
    - ix. Choose **Submit** in the lower part of the screen to create the ACL, and then choose **Continue** when the modal appears.
5. Create ACLs for specific `sys_db_object` fields. The following steps must be repeated for each of the following fields: Name, Label, Display Name, and Extends table.
  - a. While still on the table record view for `sys_db_object`, select the **Columns** tab in the tab group closest to the top of the screen.
  - b. Locate the field name and select it.
  - c. In the lower tab section, choose **New** on the **Access Controls** tab.
  - d. Change the operation to read
  - e. Choose (double-click) the insert a row text in the bottom "Requires role" table.
  - f. Search for the role that you created, and choose the green check mark.
  - g. Choose **Submit**.
  - h. Make sure that you've repeated these steps for all required fields: Name, Label, Display Name, and Extends table.
6. Repeat the steps from 4.1 on Change, Incident, and any other non-CMDB tables that you want to query from Grafana. Do not repeat the steps from 4.2; that step is only required for `sys_db_object`.

## Snowflake

With the Snowflake Enterprise data source, you can visualize your Snowflake data alongside all of your other data sources in Grafana as well as log and metric data in context. This data source includes a powerful type-ahead query editor, supports complex annotations, set alerting thresholds, control access and permissions and more.

### Overview

#### What is Snowflake?

Snowflake offers a cloud-based data storage and analytics service, generally termed "data warehouse-as-a-service" that offers a solution for data warehousing, data lakes, data engineering, data science, data application development, and data sharing. Over the last few years, Snowflake has gained massive popularity because of its ability to affordably store and analyze data using cloud-based hardware and software; recently culminating in the largest software IPO ever. Today, many companies use Snowflake as their primary database to store application and business data such as transaction counts, active user sessions, and even time series and metric data.

## Making the most of Snowflake and Amazon Managed Grafana

**Visualize Snowflake data without moving it:** Grafana's unique architecture queries data directly where it lives rather than moving it and paying for redundant storage and ingestion.

**Compose panels from varied sources:** With pre-built and custom dashboards, bring data together from many different data sources into a single pane of glass.

**Transform and compute at the user level:** Users can transform data and run various computations on data they see, requiring less data preparation.

**Combine, compute, and visualize within panels:** Create mixed-data source panels that display related data from Snowflake and other sources.

### Features

**Query editor:** The query editor is a Smart SQL autocompletion editor that allows you to visualize time series or table data, handles SQL syntax errors, and autocompletes basic SQL keywords.

**Data source permissions:** Control who can view or query Snowflake data in Grafana

**Annotations:** Overlay Snowflake events on any Grafana graph, to correlate events with other graph data

**Alerting:** Set alerts based metrics stores in Snowflake

**Variables for queries:** Create template variables in Grafana based on Snowflake data, and include variables in Snowflake queries to make interactive dashboards.

**Multi-metric queries:** Write a single query that returns multiple metrics, each in its own column

## Get started with the Snowflake plugin

Here are five quick steps to get started with the Snowflake plugin in Grafana:

### Step 1: Set up the Snowflake Data Source

To configure the data source, choose **Configuration, Data Sources, Add data source, Snowflake**.

Add your authentication details, and the data source is ready to query!

The following configuration fields are available.

Name	Description
Account	Account for Snowflake.
Username	Username for the service account.
Password	Password for the service account.
Schema (optional)	Sets a default schema for queries.
Warehouse (optional)	Sets a default warehouse for queries.
Database (optional)	Sets a default database for queries.
Role (optional)	Assumes a role for queries.

### Step 2: Write queries for your Snowflake data

Create a panel in a dashboard, and select a Snowflake Data Source to start using the query editor.

- Date / time can appear anywhere in the query as long as it is included.
- A numerical column must be included. This can be an aggregation or an int/float column.
- Optionally, you can include string columns to create separate data series, if your time series data is formatted for different metrics.

### Layout of a Snowflake query

```
select
  <time_column>,
  <any_numerical_column>
  <other_column_1>,
  <other_column_2>,
  <...>
from
  <any_table>
where
  $__timeFilter(<time_column>) // predefined where clause for time range
  and $__custom_variable = 1 // custom variables start with dollar sign
```

### SQL query format for time series group by interval

```
select
  $__timeGroup(created_ts, '1h'), // group time by interval of 1h
  <time_column>,
  <any_numerical_column>,
  <metric_column>
from
  <any_table>
where
  $__timeFilter(<time_column>) // predefined where clause for time range
  and $__custom_variable = 1 // custom variables start with dollar sign
group by <time_column>
```

### SQL query format for tables

```
select
  <time_column>, // optional if result format option is table
  <any_column_1>
  <any_column_2>
  <any_column_3>
from
  <any_table>
where
  $__timeFilter(<time_column>) // macro for time range, optional if format as option is table
  and $__custom_variable = 1 // custom variables start with dollar sign
```

## Step 3: Create and use Template Variables

### Using template variables

You can include template variables in queries, as shown in the following example.

```
select
  <column>
from
```



```
<table>
WHERE column >= '$variable'
```

The following example shows using multi-value variables in a query.

```
select
  <column>
from
  <table>
WHERE <column> regexp '${variable:regex}'
```

### Using the Snowflake data source to create variables

In the dashboard settings, choose **Variables**, and choose **New**.

Using the "Query" variable type, select the Snowflake data source as the "Datasource".

#### **Important**

Be sure to select only one column in your variable query.

Example:

```
SELECT DISTINCT query_type from account_usage.query_history;
```

will give you these variables:

```
ALL DESCRIBE USE UNKNOWN GRANT SELECT CREATE DROP SHOW
```

### Step 4: Set up an alert

You can set alerts on specific Snowflake metrics or on queries you've created.

Choose the alert tab button within the query editor, and choose **Create Alert**.

### Step 5. Create an annotation

Annotations allow you to overlay events on a graph.

To create an annotation, in the dashboard settings, choose **Annotations**, and **New**, and select Snowflake as the data source.

Because annotations are events, they require at least one time column and one column to describe the event.

The following example code shows a query to annotate all failed logins to Snowflake.

```
SELECT
  EVENT_TIMESTAMP as time,
  EVENT_TYPE,
  CLIENT_IP
FROM ACCOUNT_USAGE.LOGIN_HISTORY
WHERE $__timeFilter(time) AND IS_SUCCESS!='YES'
ORDER BY time ASC;
```

And

- time: `TIME`
- title: `EVENT_TYPE`
- text: `CLIENT_IP`

This will overlay annotations of all failed logins to Snowflake on your dashboard panels.

## Additional functionality

### Using the Display Name field

This plugin uses the Display Name field in the Field tab of the Options panel to shorten or alter a legend key based on its name, labels, or values. Other data sources use custom `alias` functionality to modify legend keys, but the Display Name function is a more consistent way to do so.

### Data source permissions

Limit access to Snowflake by choosing the **Permissions** tab in the data source configuration page to enable data source permissions. On the permission page, Admins can enable permissions and restrict query permissions to specific Users and Teams.

### Understand your Snowflake billing and usage data

Within the Snowflake data source, you can import a billing and usage dashboard that shows you useful billing and usage information.

Add the dashboard in the Snowflake Data Source configuration page:

This dashboard uses the `ACCOUNT_USAGE` database, and requires the querier to have the `ACCOUNTADMIN` role. To do this securely, create a new Grafana data source that has a user with the `ACCOUNTADMIN` role. Then select that data source in the variables.

## Splunk

### Configuration

#### Data source configuration

When configuring the Data Source, ensure that the URL field utilizes `https` and points to the your configured Splunk port. The default Splunk API point is 8089, not 8000 (this is default web UI port). Enable *Basic Auth* and specify Splunk username and password.

#### Browser (direct) access mode and CORS

If you are using CORS, you must configure the Splunk server to allow Grafana to communicate with it using a CORS connection. To do this, add your web site's address as a trusted HTTP origin to the `crossOriginSharingPolicy` attribute in the `server.conf` configuration file.

For example, add the stanza shown in the following example to the `$SPLUNK_HOME/etc/system/local/server.conf` configuration file, and then restart Splunk.

```
[httpServer]
crossOriginSharingPolicy = http://localhost:3000
```

For more information, see the article [Communicate with the Splunk server for apps outside of Splunk Web](#).

### Note

We recommend connecting to Splunk via the Grafana backend in a proxy server access mode. Use the browser (direct) access mode if you really need it and you know how it works.

## Advanced options

### Stream mode

Enable stream mode if you want to get search results as they become available. This is experimental feature, don't enable it until you really need it.

### Poll result

Run search and then periodically check for result. Under the hood this option runs `search/jobs` API call with `exec_mode` set to `normal`. In this case API request returns job SID, and then Grafana checks job status time to time, in order to get job result. This option may be helpful for slow queries. By default this option is disabled and Grafana sets `exec_mode` to `oneshot` which allows returning search result in the same API call. See more about `search/jobs` API endpoint in [Splunk docs](#).

### Search polling interval

This option allow to adjust how often Amazon Managed Grafana will poll splunk for search results. Time for next poll choosing randomly from `[min, max)` interval. If you run a lot of heavy searches, it makes sense to increase these values. Tips: increase *Min* if search jobs execution takes a long time, and *Max* if you run a lot of parallel searches (a lot of splunk metrics on Grafana dashboard). Default is `[500, 3000)` milliseconds interval.

### Automatic cancellation

If specified, the job automatically cancels after this many seconds of inactivity (0 means never auto-cancel). Default is 30.

### Status buckets

The most status buckets to generate. 0 indicates to not generate timeline information. Default is 300.

### Fields search mode

When you use visual query editor, data source attempts to get list of available fields for selected source type.

- quick - use first available result from preview
- full - wait for job finish and get full result.

### Default earliest time

Some searches can't use dashboard time range (such as template variable queries). This option helps to prevent search for all time, which can slow down Splunk. The syntax is an integer and a time unit `[+|-]<time_integer><time_unit>`. For example `-1w`. **Time unit** can be `s`, `m`, `h`, `d`, `w`, `mon`, `q`, `y`.

### Variables search mode

Search mode for template variable queries. Possible values:

- fast - Field discovery off for event searches. No event or field data for stats searches.
- smart - Field discovery on for event searches. No event or field data for stats searches.

- verbose - All event & field data.

## Usage

### Query editor

#### Editor modes

Query editor support two modes: raw and visual. To switch between these modes choose hamburger icon at the right side of editor and select *Toggle Editor Mode*.

#### Raw mode

Use `timechart` command for time series data, as shown in the following code example.

```
index=os sourcetype=cpu | timechart span=1m avg(pctSystem) as system, avg(pctUser) as user,
avg(pctIowait) as iowait
index=os sourcetype=ps | timechart span=1m limit=5 useother=false avg(cpu_load_percent) by
process_name
```

Queries support template variables, as shown in the following example.

```
sourcetype=cpu | timechart span=1m avg($cpu)
```

Keep in mind that Grafana is time series-oriented application and your search should return time series data (timestamp and value) or single value. You can read about [timechart](#) command and find more search examples in official [Splunk Search Reference](#)

### Splunk Metrics and mstats

Splunk 7.x provides `mstats` command for analyzing metrics. To get charts working properly with `mstats`, it should be combined with `timeseries` command and `prestats=t` option must be set.

```
Deprecated syntax:
| mstats prestats=t avg(_value) AS Value WHERE index="collectd"
metric_name="disk.disk_ops.read" OR metric_name="disk.disk_ops.write" by metric_name
span=1m
| timechart avg(_value) span=1m by metric_name

Actual:
| mstats prestats=t avg(disk.disk_ops.read) avg(disk.disk_ops.write) WHERE index="collectd"
by metric_name span=1m
| timechart avg(disk.disk_ops.read) avg(disk.disk_ops.write) span=1m
```

Read more about `mstats` command in [Splunk Search Reference](#).

### Format as

There are two supported result format modes - *Time series* (default) and *Table*. Table mode suitable for using with Table panel when you want to display aggregated data. That works with raw events (returns all selected fields) and `stats` search function, which returns table-like data. Examples:

```
index="os" sourcetype="vmstat" | fields host, memUsedMB
```

```
index="os" sourcetype="ps" | stats avg(PercentProcessorTime) as "CPU time",  
latest(process_name) as "Process", avg(UsedBytes) as "Memory" by PID
```

The result is similar to *Statistics* tab in Splunk UI.

Read more about `stats` function usage in [Splunk Search Reference](#).

### Visual mode

This mode provides step-by-step search creating. Note that this mode creates `timechart` splunk search. Just select index, source type, and metrics, and set split by fields if you want.

### Metric

You can add multiple metrics to search by choosing *plus* button at the right side of metric row. Metric editor contains list of frequently used aggregations, but you can specify here any other function. Just choose agg segment (`avg` by default) and type what you need. Select interested field from the dropdown list (or enter it), and set alias if you want.

### Split by and Where

If you set Split by field and use *Time series* mode, Where editor will be available. Choose *plus* and select operator, aggregation and value, for example *Where avg in top 10*. Note, this *Where* clause is a part of *Split by*. See more at [timechart docs](#).

### Options

To change default timechart options, choose **Options** at the last row.

See more about these options in [timechart docs](#).

### Rendered splunk search

Choose the target letter at the left to collapse the editor and show the rendered splunk search.

### Annotations

Use annotations if you want to show Splunk alerts or events on graph. Annotation can be either predefined Splunk alert or regular splunk search.

### Splunk alert

Specify an alert name, or keep the field blank to get all fired alerts. Template variables are supported.

### Splunk search

Use splunk search to get needed events, as shown in the following example.

```
index=os sourcetype=iostat | where total_ops > 400  
index=os sourcetype=iostat | where total_ops > $io_threshold
```

Template variables are supported.

The **Event field as text** option is suitable if you want to use field value as annotation text. The following example shows error message text from logs.

```
Event field as text: _raw
```

```
Regex: WirelessRadioManagerd[\d*]: (.*)
```

Regex allows to extract a part of message.

### Template variables

Template variables feature supports Splunk queries which return list of values, for example with `stats` command.

```
index=os sourcetype="iostat" | stats values(Device)
```

This query returns list of `Device` field values from `iostat` source. Then you can use these device names for time series queries or annotations.

There are two possible types of variable queries can be used in Grafana. The first is a simple query (as presented earlier), which returns a list of values. The second type is a query that can create a key/value variable. The query should return two columns that are named `_text` and `_value`. The `_text` column value should be unique (if it is not unique then the first value is used). The options in the dropdown list will have a text and value so that you can have a friendly name as text and an ID as the value.

For example, this search returns table with columns `Name` (Docker container name) and `Id` (container id).

```
source=docker_inspect | stats count latest(Name) as Name by Id | table Name, Id
```

To use container name as a visible value for variable and id as it's real value, query should be modified, as in the following example.

```
source=docker_inspect | stats count latest(Name) as Name by Id | table Name, Id | rename  
Name as "_text", Id as "_value"
```

### Multi-value variables

It's possible to use multi-value variables in queries. An interpolated search will be depending on variable usage context. There are a number of that contexts which plugin supports. Assume there's a variable `$container` with selected values `foo` and `bar`:

- Basic filter for search command

```
source=docker_stats $container  
=>  
source=docker_stats (foo OR bar)
```

- Field-value filter

```
source=docker_stats container_name=$container  
=>  
source=docker_stats (container_name=foo OR container_name=bar)
```

- Field-value filter with the `IN` operator and `in()` function

```
source=docker_stats container_name IN ($container)
```

```
=>
source=docker_stats container_name IN (foo, bar)

source=docker_stats | where container_name in($container)
=>
source=docker_stats | where container_name in(foo, bar)
```

### Multi-value variables and quotes

If variable wrapped in quotes (both double or single), its values also will be quoted, as in the following example.

```
source=docker_stats container_name="$container"
=>
source=docker_stats (container_name="foo" OR container_name="bar")

source=docker_stats container_name='$container'
=>
source=docker_stats (container_name='foo' OR container_name='bar')
```

## Wavefront (VMware Tanzu Observability by Wavefront)

The Wavefront (VMware Tanzu Observability by Wavefront) data source enables Amazon Managed Grafana users to query and visualize the data they're collecting directly from Wavefront and easily visualize it alongside any other metric, log, tracing, or other data source. This flexible, single-pane view makes it easier to track system health and debug issues.

### What is Wavefront?

[Wavefront](#) is a cloud monitoring and analytics tool developed by VMware. Wavefront is a cloud-hosted service where you send your time-series (metric) data – from CollectD, StatsD, JMX, Ruby's logger, AWS, or other tools. With Wavefront, users can perform mathematical operations on those series, render charts to see anomalies, track KPIs, and create alerts.

### Maximizing your tech stack with Wavefront and Grafana

While on the surface, Grafana and Wavefront may sound similar, many organizations use both Wavefront and Grafana as critical parts of their observability workflows.

**Visualize without Moving Data Sources:** Grafana's unique architecture queries data directly where it lives rather than moving it and paying for redundant storage and ingestion.

**Compose Panels from Varied Sources** With pre-built and custom dashboards, bring data together from many different data sources into a single pane of glass.

**Transform and Compute at the User Level:** Users can transform data and run various computations on data they see, requiring less data preparation.

**Combine, Compute, and Visualize within Panels:** Create mixed-data source panels that display related data from Waveferont and other sources, such as Prometheus and InfluxDB.

## Documentation

### Features

- Timeseries Visualizations
- Table Visualizations

- Heatmap Visualizations
- Single Stat Visualizations
- Guided Query Editor
- Raw WQL Query Editor
- Annotations for event data
- Template Variables
- Ad-Hoc Filters
- Alerting

## Configuration

Configuring the Wavefront data source is relatively straightforward. There are only two fields required to complete the configuration: `API URL` and `Token`.

- `API URL` will be the URL you use to access your wavefront environment. Example: `https://myenvironment.wavefront.com`.
- `Token` must be generated from a user account or service account.
  1. To create a user account based token, log into your Wavefront environment, choose the cog on the top right corner of the page, choose your username (for example, `me@grafana.com`), select the **API Access** tab at the top of the user page, then copy an existing key or choose **generate**.
  2. To create a service account based token, log into your Wavefront environment, choose the cog on the top right corner of the page, choose account management. On the left navigation, select **Accounts, Groups, & Roles**, choose the **Service Accounts** tab at the top, and then choose **Create New Account**. Enter a name for the service account. This can be anything you want. Copy the token that is provided under the **Tokens** section.
  3. The last step is to make sure that the **Accounts, Groups, & Roles** check box selected under **Permissions**.

After you have the token, add that to the `Token` configuration field and you should be set!

The finalized configuration page should look similar to this:

## Usage

### Using the query editor

The Wavefront query editor has two modes: **Query Builder** and **Raw Query**. To toggle between them, use the selector in the top right of the query form:

In **Query Builder** mode, you will be presented with four choices to make:

1. What metric do you want to query?
2. What aggregation do you want to perform on that metric?
3. How do you want to filter the results from that metric query?
4. Do you want to apply any additional functions to the result?

The metric selector is a categorized hierarchy. Select a category, then choose again to drill into the subcategories. Repeat this process until you have reached the metric that you want.

After selecting a metric, the available filters and filter values will be automatically populated for you.

In **Raw Query** mode, you will see a single field labeled **Query**. This allows you to run any [WQL \(p. 188\)](#) query that you want.



## Using filters

The Wavefront plugin will dynamically query the appropriate filters for each metric.

To add a filter, choose the **+** next to the **Filters** label on the Wavefront query editor, select which field you want to filter on, and select a value to filter by.

## Using functions

Functions provide an additional way to aggregate, manipulate, and perform calculations on the metric response data. To view the available functions, choose the dropdown list by the function label on the **Query Builder**. Based on the function you select, you will be able to perform further actions such as setting a group by field or applying thresholds. Users are able to chain multiple functions together to perform advanced calculations or data manipulations.

## Adding a query template variable

1. To create a new Wavefront template variable for a dashboard, choose the settings cog on the top right portion of the dashboard.
2. In the panel at the left, choose **Variables**.
3. At the top right of the Variables page, choose **New**.
4. Enter a **Name** and a **Label** for the template variable you want to create. **Name** is the value you will use inside of queries to reference the template variable. **Label** is a human friendly name to display for the template variable on the dashboard select panel.
5. Select the type **Query** for the type field (it should be selected by default).
6. Under the **Query Options** heading, select **Wavefront** in the **Data source** dropdown list.
7. See [Template Variable Query Structure \(p. 186\)](#) for details on what should be entered into the **Query** field.
8. If you want to filter out any of the returned values from your query, enter a regular expression in the **Regex** input field.
9. Apply any sorting preferences you might have by choosing a sort type in the **Sort** dropdown list.
10. After verifying the configuration, choose **Add** to add the template variable, then choose **Save dashboard** on the left hand navigation panel to save your changes.

## Template variable query structure

metric lists: metrics: ts(...)

source lists: sources: ts(...)

source tag lists: sourceTags: ts(...)

matching source tag lists: matchingSourceTags: ts(...)

tag name lists: tagNames: ts(...)

tag value lists: tagValues(<tag>): ts(...)

### Notes

- The **s** at the end of each query type is optional
- Support for all lowercase. You can use `tagnames` or `tagNames`, but not `TAGNAMES`.
- Using spaces around the `:` is optional

### WARNING

`Multi-value` and `Include All` option are currently not supported by the Wavefront plugin.

## Using template variables

After completing the steps to [add a new template variable \(p. 186\)](#), you're now ready to use the template variable within your dashboard panels to create dynamic visualizations.

1. Add a new dashboard panel using the panel+ icon in the top right corner of your dashboard.
2. Select the aggregate you want to use for your query.
3. Choose the + icon beside **Filters** label and select the key type that matches your template variable. `host=` for a host filter, for example.
4. Enter the name of the template variable you created in the **Value** input field of the filter.
5. Save the dashboard.

You should now be able to cycle through different values of your template variable and have your panel dynamically update!

## Using Ad-Hoc filters

To use ad-hoc filters, we must create two template variables. The first one is a helper variable that will be used to select a metric so that add-hoc filters can be populated for that metric name. The other will be the actual ad-hoc filter variable.

### Important

The helper variable that is required has to be named `metriclink`. This can be an custom variable with the list of metrics that you want to use or a query based variable using the [Template Variable Query Structure \(p. 186\)](#). If you want to populate the ad-hoc filter fields with only the values from a single metric, you can hide the `metriclink` template variable.

After creating the `metriclink` variable, you can now add the ad-hoc filter by following the same steps detailed in [Adding a Query Template Variable \(p. 186\)](#). The difference being that you will select **Ad Hoc Filters** as the **Type** and no inputs are required for a query.

## Adding annotations

1. To create a new Wavefront annotation for a dashboard, choose the settings cog on the top right portion of the dashboard.
2. In the panel at the left, choose **Annotations**.
3. At the top right of the Annotations page, choose **New**.
4. Enter a name for the annotation (this will be used as the name of the toggle on the dashboard).
5. Select the **Data source** of Wavefront.
6. By default, annotations have a limit of 100 alert events that will be returned. To change that, set the **Limit** field to the value that you want.
7. Choose **Add**.

## Using annotations

When annotations are toggled on, you should now see the alert events and issues that correlate with a given time period.

If you pause on the bottom of an annotated section of a visualization, a pop-up window will be displayed that shows the alert name and provides a direct link to the alert in Wavefront.

## Using the Display Name field

This data source uses the Display Name field in the Field tab of the Options panel to shorten or alter a legend key based on its name, labels, or values. Other datasources use custom `alias` functionality to modify legend keys, but the Display Name function is a more consistent way to do so.

## References

- [WQL \(Wavefront Query Language\)](#)

# Panels

The panel is the basic visualization building block in a Grafana server. With the exception of a few special-use panels, a panel is a visual representation of one or more queries. The queries display data over time. This can range from temperature fluctuations to current server status to a list of logs or alerts.

Each panel has a query editor specific to the data source selected in the panel. The query editor allows you to extract a visualization to display on the panel.

To display data, you must have at least one data source added to your workspace. For more information about data sources, see [Data sources \(p. 39\)](#).

There are a wide variety of styling and formatting options for each panel. Panels can be dragged, rearranged, and resized.

## Topics

- [Adding or editing a panel \(p. 188\)](#)
- [Queries \(p. 189\)](#)
- [Transformations \(p. 192\)](#)
- [Field options and overrides \(p. 199\)](#)
- [Panel editor \(p. 205\)](#)
- [Library panels \(p. 206\)](#)
- [Visualizations \(p. 208\)](#)

## Adding or editing a panel

You can use panels to show your data in visual form. This topic walks you through the basic steps to build a panel.

### To add a panel to a dashboard

1. Choose the dashboard that you want to add a panel to.
2. Choose the **Add panel** icon.
3. Choose **Add new panel**.

The Grafana workspace creates an empty graph panel with your default data source selected.

4. While not required, we recommend that you add a helpful title and description to your panel. You can optionally use variables that you have defined in either field, but not global variables. For more information, see [Templates and variables \(p. 265\)](#).
  - **Panel title** – Text entered in this field is displayed at the top of your panel in the panel editor and in the dashboard.
  - **Description** – Text entered in this field is displayed in a tooltip in the upper-left corner of the panel. Write a description of the panel and the data you are displaying.
5. Write a query for the panel. To display a visualization, each panel needs at least one query. You write queries on the **Query** tab of the panel editor. For more information, see [Queries \(p. 189\)](#).

- a. Choose a data source. In the first line of the **Query** tab, choose the dropdown list to see all available data sources. This list includes all data sources that you added. For more information about data sources, see [Data sources \(p. 39\)](#).
  - b. Write or construct a query in the query language of your data source. Options will vary. See your specific data source documentation for specific guidelines.
6. In the **Visualization** section of the **Panel** tab, choose a visualization type. The Grafana workspace displays a preview of your query results with that visualization applied.
7. We recommend that you add a note to describe your changes before you choose **Save**. Notes are very helpful if you need to revert the dashboard to a previous version.
8. To save the dashboard, choose **Save** in the upper-right corner of the screen.

## Queries

Grafana workspace panels use *Queries* to communicate with data sources to get data for the visualization. A query is a question written in the query language that is used by the data source. If the query is properly formed, the data source responds. In the panel data source options, you can adjust how often the query is sent to the data source and how many data points are collected.

Grafana workspaces supports up to 26 queries per panel.

## Query editors

Query editors are forms that help you write queries. Depending on your data source, the query editor might provide automatic completion, metric names, or variable suggestion.

Because of differences between query languages, data sources might have query editors that look different.

## Query syntax

Data sources have different query languages and syntaxes to ask for the data. Here are two query examples.

### PostgreSQL

```
SELECT hostname FROM host WHERE region IN($region)
```

### PromQL

```
query_result(max_over_time(<metric>[#{__range_s}s]) != <state>)
```

For more information about writing a query for your data source, see the documentation for that data sources. Data sources are listed in [Data sources \(p. 39\)](#).

## Query tab UI

The **Query** tab consists of the following elements:

- Data source selector
- Query options

- Query inspector button
- Query editor list

## Data source selector

The data source selector is a dropdown list. Choose it to select a data source that you have added. When you create a panel, Amazon Managed Grafana automatically selects your default data source. For more information about data sources, see [Data sources \(p. 39\)](#).

In addition to the data sources that you have configured in your Grafana workspace, three special data sources are available.

- **TestDataDB** – A built-in data source that generates random walk data. The Grafana data source is useful for testing visualizations and running experiments.
- **Mixed** – A data source for querying multiple data sources in the same panel. When this data source is selected, you can select a data source for every new query that you add.
  - The first query will use the data source that was selected before you selected **Mixed**.
  - You cannot change an existing query to use the Mixed data source.
- **Dashboard** A data source for using a result set from another panel in the same dashboard.

## Query options

To see settings for your selected data source, choose **Query options** next to the data source selector. Changes you make here affect only the queries made in this panel.

Amazon Managed Grafana sets defaults that are shown in dark gray text. Changes are displayed in white text. To return a field to the default setting, delete the white text from the field.

You can use the following panel data source query options:

- **Max data points** – If the data source supports it, sets the maximum numbers of data points for each series returned. If the query returns more data points than the max data points setting, the data source consolidates them (reduces the number of points returned by aggregating them together by average or max or other function).

There are two main reasons for limiting the number of points: performance and smoothing the line. The default value is the width (or number of pixels) of the graph, which avoids having more data points than the graph panel can display.

With streaming data, the max data points value is used for the rolling buffer. (Streaming is a continuous flow of data, and buffering is a way to divide the stream into chunks).

- **Min interval** – Sets a minimum limit for the automatically calculated interval, typically the minimum scrape interval. If a data point is saved every 15 seconds, you don't need to have an interval lower than that. Another use case is to set it to a higher minimum than the scrape interval to get more coarse-grained, well-functioning queries.
- **Interval** – A time span that you can use when aggregating or grouping data points by time.

Amazon Managed Grafana automatically calculates an appropriate interval that can be used as a variable in templated queries. The variable is either in seconds: `$__interval`; or in milliseconds: `$__interval_ms`. It is typically used in aggregation functions like `sum` or `average`. For example, this is a Prometheus query using the interval variable: `rate(http_requests_total[$__interval])`.

This automatic interval is calculated based on the width of the graph. If the user zooms out a lot, the interval becomes greater, resulting in a more coarse-grained aggregation. If the user zooms in, the interval decreases, resulting in a more fine-grained aggregation.

For more information, see [Global variables \(p. 274\)](#).

- **Relative time** – Override of the relative time range for individual panels, causing them to be different from what is selected in the dashboard time picker in the top-right corner of the dashboard. This allows you to show metrics from different time periods or days on the same dashboard.
- **Time shift** – Provides another way to override the time range for individual panels. This function works only with relative time ranges, and you can adjust the time range.

For example, you can shift the time range for the panel to be 2 hours earlier than the dashboard time picker. For more information, see [Time range controls \(p. 244\)](#).

- **Cache timeout** – (This field is visible only if it is available in your data source.) Overrides the default cache timeout if your time series store has a query cache. It is specified as a numeric value in seconds.

## Query inspector button

You can choose **Query inspector** to open the **Query** tab of the panel inspector. On the **Query** tab, you can see the query request sent by the panel and the response.

Choose **Refresh** to see the full text of the request sent by this panel to the server.

### Note

You need to add at least one query before the Query inspector can return results.

For more information about the panel inspector, see [\[Inspect a panel\] {link}](#).

## Query editor list

In the UI, queries are organized in collapsible query rows. Each query row contains a query editor and is identified with a letter (A, B, C, and so on).

## Sharing query results between panels

With Amazon Managed Grafana, you can use the query result from one panel for any other panel in the dashboard. Sharing query results across panels reduces the number of queries made to your data source, which can improve the performance of your dashboard.

The Dashboard data source lets you select a panel in your dashboard that contains the queries that you want to share the results for. Instead of sending a separate query for each panel, Amazon Managed Grafana sends one query, and other panels use the query results to construct visualizations.

This strategy can drastically reduce the number of queries being made when, for example, you have several panels visualizing the same data.

### To share data source queries with another panel

1. Create a dashboard. For more information, see [Creating a dashboard \(p. 33\)](#).
2. Add a panel. For more information, see [Adding or editing a panel \(p. 188\)](#).
3. Change the title to **Source panel1**. You'll use this panel as a source for the other panels. Define the query or queries that will be shared. If you don't have a data source available at the moment, you can use the **Grafana** data source, which returns a random time series that you can use for testing.
4. Add a second panel, and then select the **Dashboard** data source in the query editor.
5. In **Use results from panel list**, select the first panel that you created.

All queries defined in the source panel are now available to the new panel. Queries made in the source panel can be shared with multiple panels.

To go to a panel where a query is defined, choose that query.

## Transformations

Transformations process the result set before it's passed to the visualization. You access transformations in the **Transform** tab of the Amazon Managed Grafana panel editor.

You can use transformations to rename fields, join separate time series together, do math across queries, and more. If you have large dashboards or heavy queries, being able to reuse the query result from one panel in another panel can provide a huge performance gain.

### Note

Transformations sometimes result in data that cannot be graphed. When that happens, Amazon Managed Grafana displays a suggestion on the visualization. Choose the suggestion to switch to table visualization. This often helps you better understand what the transformation is doing to your data.

Amazon Managed Grafana applies transformations in the sequence that they are listed on the screen. Every transformation creates a new result set that is passed to the next transformation in the pipeline.

The order can make a huge difference in how your results look. For example, if you use a Reduce transformation to condense all the results of one column to a single value, you can apply transformations only to that single value.

### Prerequisites

Before you apply transformations, all of the following must be true:

- You have entered a query and returned data from a data source. For more information about queries, see [Queries].
- You have applied a visualization that supports queries, such as one of the following visualizations:
  - Bar gauge
  - Gauge
  - Graph
  - Heatmap
  - Logs
  - Stat
  - Table

## Applying a transformation

Transformations are available from the **Transform** tab in the bottom pane of the panel editor, next to the **Queries** tab.

### To apply a transformation

1. On the panel that you want to add transformations to, choose the panel title, and then choose **Edit**.
2. Choose the **Transform** tab.
3. Select a transformation.

In the transformation row that appears, you can configure the transformation options.

4. To apply another transformation, choose **Add transformation**. Keep in mind that the next transformation acts on the result set returned by the previous transformation.

If you have trouble, choose the bug icon to [debug your transformations \(p. 199\)](#).

To remove a transformation, choose the trash can icon.

## Transformation types and options

Grafana workspaces include the following transformations.

### Topics

- [Reduce \(p. 193\)](#)
- [Merge \(p. 193\)](#)
- [Filter by name \(p. 194\)](#)
- [Filter data by query \(p. 194\)](#)
- [Organize fields \(p. 194\)](#)
- [Join by field \(outer join\) \(p. 195\)](#)
- [Add field from calculation \(p. 195\)](#)
- [Labels to fields \(p. 195\)](#)
- [Group By \(p. 172\)](#)
- [Group By \(p. 172\)](#)
- [Series to rows \(p. 197\)](#)
- [Filter data by value \(p. 198\)](#)
- [Debug transformations \(p. 199\)](#)

### Reduce

Apply a **Reduce** transformation when you want to simplify your results down to one value. Reduce basically removes the time component. If visualized as a table, it reduces a column down to one row (value).

In the **Calculations** field, enter one or more calculation types. Choose to see a list of calculation choices. For information about available calculations, see [Calculations list \(p. 236\)](#).

After you select at least one calculation, Amazon Managed Grafana reduces the results down to one value using the calculation you selected. If you select more than one calculation, more than one value is displayed.

### Merge

Use this transformation to combine the results from multiple queries into one single result. This is helpful when using the table panel visualization. Values that can be merged are combined into the same row. Values can be merged if the shared fields contain the same data.

In the following example, two queries return table data. The data are visualized as two separate tables before applying the transformation.

Query A:

Time	Job	Uptime
2020-07-07 11:34:20	node	25260122



Time	Job	Uptime
2020-07-07 11:24:20	postgre	123001233

Query B:

Time	Job	Errors
2020-07-07 11:34:20	node	15
2020-07-07 11:24:20	postgre	5

Here is the result after applying the **Merge** transformation.

Time	Job	Errors	Uptime
2020-07-07 11:34:20	node	15	25260122
2020-07-07 11:24:20	postgre	5	123001233

## Filter by name

Use this transformation to remove portions of the query results.

Amazon Managed Grafana displays the **Identifier** field, followed by the fields returned by your query.

You can apply filters in one of two ways:

- Enter a regex expression.
- Choose a field to toggle filtering on that field. Filtered fields are displayed with dark gray text, unfiltered fields have white text.

## Filter data by query

Use this transformation in panels that have multiple queries, if you want to hide one or more of the queries.

Amazon Managed Grafana displays the query identification letters in dark gray text. Choose a query identifier to toggle filtering. If the query letter is white, the results are displayed. If the query letter is dark, the results are hidden.

## Organize fields

Use this transformation to rename, reorder, or hide fields returned by the query.

### Note

This transformation works only in panels that have a single query. If your panel has multiple queries, you must either apply a **Join by field (outer join)** transformation or remove the extra queries.

Amazon Managed Grafana displays a list of fields returned by the query. You can make any of the following changes:

- Change the field order by pausing over a field. The cursor turns into a hand, and then you can drag the field to its new place.

- Hide or show a field by choosing the eye icon next to the field name.
- Rename fields by typing a new name in the **Rename** box.

## Join by field (outer join)

Use this transformation to join multiple time series from a result set by field.

This transformation is especially useful if you want to combine queries so that you can calculate results from the fields.

## Add field from calculation

Use this transformation to add a new field calculated from two other fields. Each transformation allows you to add one new field.

- **Mode** – Select a mode:
  - **Reduce row** – Apply selected calculation on each row of selected fields independently.
  - **Binary option** – Apply a basic math operation (*sum*, *multiply*, and so on) on values in a single row from two selected fields.
- **Field name** – Select the names of fields that you want to use in the calculation for the new field.
- **Calculation** – Select a calculation to use when Amazon Managed Grafana creates the new field. Choose the field to see a list of calculation choices. For information about available calculations, see [Calculations list \(p. 236\)](#).
- **Alias** – (Optional) Enter the name of your new field. If you leave this blank, the field will be named to match the calculation.
- **Replace all fields** – (Optional) Use this option if you want to hide all other fields and display only your calculated field in the visualization.

## Labels to fields

### Note

To apply this transformation, your query needs to return labeled fields.

When you select this transformation, Amazon Managed Grafana automatically transforms all labeled data into fields.

For example, consider a query result of two time series.

1: labels Server=Server A, Datacenter=EU 2: labels Server=Server B, Datacenter=EU

This transformation would result in the following table.

Time	Server	Datacenter	Value
2020-07-07 11:34:20	Server A	EU	1
2020-07-07 11:34:20	Server B	EU	2

**Value field name;** If you selected *Server* as the **Value field name**, you would get one field for every value of the *Server* label.

Time	Datacenter	Server A	Server B
2020-07-07 11:34:20	EU	1	2

## Group By

This transformation sorts each frame by the configured field. When `reverse` is checked, the values are returned in the opposite order.

## Group By

This transformation groups the data by a specified field (column) value and processes calculations on each group. The available calculations are the same as for the Reduce transformation.

Here's an example of original data.

Time	Server ID	CPU Temperature	Server Status
2020-07-07 11:34:20	server 1	80	Shutdown
2020-07-07 11:34:20	server 3	62	OK
2020-07-07 10:32:20	server 2	90	Overload
2020-07-07 10:31:22	server 3	55	OK
2020-07-07 09:30:57	server 3	62	Rebooting
2020-07-07 09:30:05	server 2	88	OK
2020-07-07 09:28:06	server 1	80	OK
2020-07-07 09:25:05	server 2	88	OK
2020-07-07 09:23:07	server 1	86	OK

This transformation takes two steps. First, you specify one or multiple fields to group the data by. This will group all the same values of those fields together, as if you sorted them. For instance, if you **Group By** the `Server ID` field, it will group the data this way:

Time	Server ID	CPU Temperature	Server Status
2020-07-07 11:34:20	<b>server 1</b>	80	Shutdown
2020-07-07 09:28:06	<b>server 1</b>	80	OK
2020-07-07 09:23:07	<b>server 1</b>	86	OK

```
2020-07-07 10:32:20 | server 2 | 90 | Overload
2020-07-07 09:30:05 | server 2 | 88 | OK
2020-07-07 09:25:05 | server 2 | 88 | OK

2020-07-07 11:34:20 | server 3 | 62 | OK
2020-07-07 10:31:22 | server 3 | 55 | OK
2020-07-07 09:30:57 | server 3 | 62 | Rebooting
```

All rows with the same value of `Server ID` are grouped together.

After choosing which field you want to group your data by, you can add various calculations on the other fields, and the calculation will be applied on each group of rows. For instance, you might want to

calculate the average CPU temperature for each of those servers. You can add the *mean* calculation applied on the CPU Temperature field to get the following:

Server ID	CPU Temperature (mean)
server 1	82
server 2	88.6
server 3	59.6

And you can add more than one of those calculation. For instance, you can use the following calculations.

- For field `Time`, you can calculate the *Last* value, to know when the last data point was received for each server.
- For field `Server Status`, you can calculate the *Last* value to know what is the last state value for each server.
- For field `Temperature`, you can also calculate the *Last* value to know what is the latest monitored temperature for each server.

The Group By transformation produces the following results.

Server ID	CPU Temperature (mean)	CPU Temperature (last)	Time (last)	Server Status (last)
server 1	82	80	2020-07-07 11:34:20	Shutdown
server 2	88.6	90	2020-07-07 10:32:20	Overload
server 3	59.6	62	2020-07-07 11:34:20	OK

Using this transformation, you can extract some key information out of your time series and display it in a convenient way.

## Series to rows

Use this transformation to combine the results from multiple time series data queries into one single result. This is helpful when using the table panel visualization.

The result from this transformation will contain three columns: `Time`, `Metric`, and `Value`. The `Metric` column is added so that you can see from which query the metric originates. Customize this value by defining `Label` on the source query.

In the example below, two queries return time series data. It is visualized as two separate tables before the transformation is applied.

Query A:

Time	Temperature
2020-07-07 11:34:20	25

Time	Temperature
2020-07-07 10:31:22	22
2020-07-07 09:30:05	19

Query B:

Time	Humidity
2020-07-07 11:34:20	24
2020-07-07 10:32:20	29
2020-07-07 09:30:57	33

Applying the `Series to rows` transformation produces the following results.

Time	Metric	Value
2020-07-07 11:34:20	Temperature	25
2020-07-07 11:34:20	Humidity	22
2020-07-07 10:32:20	Humidity	29
2020-07-07 10:31:22	Temperature	22
2020-07-07 09:30:57	Humidity	33
2020-07-07 09:30:05	Temperature	19

## Filter data by value

This transformation allows you to filter your data directly in the Grafana workspace and remove some data points from your query result. You have the option to include or exclude data that match one or more conditions you define. The conditions are applied on a selected field.

This transformation is useful if your data source does not natively filter by values. You might also use this to narrow values to display if you are using a shared query.

The available conditions for all fields are:

- **Regex** – Match a regex expression.
- **Is Null** – Match if the value is null.
- **Is Not Null** – Match if the value is not null.
- **Equal** – Match if the value is equal to the specified value.
- **Different** – Match if the value is different than the specified value.

The available conditions for number fields are:

- **Greater** – Match if the value is greater than the specified value.
- **Lower** – Match if the value is lower than the specified value.
- **Greater or equal** – Match if the value is greater than or equal to the specified value.

- **Lower or equal** – Match if the value is lower than or equal to the specified value.
- **Range** – Match a range between a specified minimum and maximum. The minimum and maximum are included in the range.

You can add more than one condition to the filter. When you have more than one condition, you can choose if you want the action (include / exclude) to be applied on rows that Match all conditions or Match any of the conditions you added.

Conditions that are not valid or incompletely configured are ignored.

## Debug transformations

To see the input and the output result sets of the transformation, choose the bug icon on the right side of the transformation row.

Amazon Managed Grafana displays the transformation debug view below the transformation row.

## Field options and overrides

This section explains what field options and field overrides in Amazon Managed Grafana are and how to use them.

The data model used in Grafana workspaces, the data frame, is a columnar-oriented table structure that unifies both time series and table query results. Each column within this structure is called a *field*. A field can represent a single time series or table column.

Field options allow you to change how the data is displayed in your visualizations. Options and overrides that you apply do not change the data, they change how Amazon Managed Grafana displays the data.

### Field options

*Field options*, both standard and custom, can be found on the **Field** tab in the panel editor. Changes that are made on this tab apply to all fields (that is, series and columns). For example, if you change the unit to percentage, all fields with numeric values are displayed in percentages. Learn how to apply a field option in [Configure all fields \(p. 199\)](#).

### Field overrides

*Field overrides* can be added on the **Overrides** tab in the panel editor. There you can add the same options as you find on the **Field** tab, but they are applied only to specific fields. Learn how to apply an override in [Configure specific fields \(p. 201\)](#).

## Available field options and overrides

Field option types are common to both field options and field overrides. The only difference is whether the change will apply to all fields (applied on the **Field** tab) or to a subset of fields (applied on the **Overrides** tab).

- [Standard field options \(p. 202\)](#) apply to all panel visualizations that allow transformations.
- [Table field options \(p. 228\)](#) apply only to table panel visualizations.

### Configure all fields

To change how all fields display data, you can change an option on the **Field** tab. On the **Overrides** tab, you can then override the field options for specific fields. For more information, see [Configure specific fields \(p. 201\)](#).

For example, you can change the number of decimal places shown in all fields by changing the **Decimals** option. For more information about options, see [Standard field options \(p. 202\)](#) and [Table field options \(p. 228\)](#).

### Change a field option

You can change as many options as you want to.

#### To change a field option

1. Choose the panel that you want to edit, choose the panel title, and then choose **Edit**.
2. Choose the **Field** tab.
3. Find the option that you want to change. You can define the following:
  - [Standard field options \(p. 202\)](#), which apply to all panel visualizations that allow transformations.
  - [Table field options \(p. 228\)](#), which only apply to table panel visualizations.
4. Add options by adding values in the fields. To return options to default values, delete the white text in the fields.
5. When finished, choose **Save** to save all panel edits to the dashboard.

### Field option example

Let's assume that the result set is a data frame that consists of two fields: time and temperature.

time	temperature
2020-01-02 03:04:00	45.0
2020-01-02 03:05:00	47.0
2020-01-02 03:06:00	48.0

Each field (column) of this structure can have field options applied that alter the way its values are displayed. This means that you can, for example, set the Unit to Temperature > Celsius, resulting in the following table:

time	temperature
2020-01-02 03:04:00	45.0 °C
2020-01-02 03:05:00	47.0 °C
2020-01-02 03:06:00	48.0 °C

The decimal place doesn't add anything to this display. You can change the Decimals from auto to zero (0), resulting in the following table:

time	temperature
2020-01-02 03:04:00	45 °C

time	temperature
2020-01-02 03:05:00	47 °C
2020-01-02 03:06:00	48 °C

## Configure specific fields

You can use overrides to change the settings for one or more fields. Field options for overrides are exactly the same as the field options that are available in a particular visualization. The only difference is that you choose which fields to apply them to.

For example, you can change the number of decimal places shown in all numeric fields or columns by changing the **Decimals** option for **Fields with type** that matches **Numeric**. For more information about options, see [Standard field options \(p. 202\)](#), which apply to all panel visualizations that allow transformations, and [Table field options \(p. 228\)](#), which apply only to table panel visualizations.

### Add a field override

You can override as many field options as you want to.

#### To add a field override

1. Choose the panel that you want to edit, choose the panel title, and then choose **Edit**.
2. Choose the **Overrides** tab.
3. Choose **Add an override for**.
4. Select the fields to which you want to apply an override rule.
  - **Fields with name** – Use this to select a field from the list of all available fields. Properties that you add to a rule with this selector are applied only to this single field.
  - **Fields with name matching regex** – Use this to specify fields to override with a regular expression. Properties that you add to a rule by using this selector are applied to all fields where the field name match the regex.
  - **Fields with type** – Use this to select fields by type, such as string, numeric, and so on. Properties that you add to a rule with this selector are applied to all fields that match the selected type.
5. Choose **Add override property**.
6. Select the field option that you want to apply.
7. Enter options by adding values in the fields. To return options to default values, delete the white text in the fields.
8. Continue to add overrides to this field by choosing **Add override property**, or you can choose **Add override** and select a different field to add overrides to.
9. When finished, choose **Save** to save all panel edits to the dashboard.

### Delete a field override

1. Choose the Overrides tab that contains the override that you want to delete.
2. Choose the trash can icon next to the override.

### Field override example

Let's assume that our result set is a data frame that consists of four fields: time, high temp, low temp, and humidity.



time	high temp	low temp	humidity
2020-01-02 03:04:00	45.0	30.0	67
2020-01-02 03:05:00	47.0	34.0	68
2020-01-02 03:06:00	48.0	31.0	68

Let's apply the field options from the [Field option example \(p. 200\)](#) to apply the Celsius unit and get rid of the decimal place. This results in the following table.

time	high temp	low temp	humidity
2020-01-02 03:04:00	45 °C	30 °C	67 °C
2020-01-02 03:05:00	47 °C	34 °C	68 °C
2020-01-02 03:06:00	48 °C	31 °C	68 °C

The temperature fields look good, but the humidity is nonsensical. You can fix this by applying a field option override to the humidity field and changing the unit to Misc > percent (0-100). This results in a table that makes a lot more sense.

time	high temp	low temp	humidity
2020-01-02 03:04:00	45 °C	30 °C	67%
2020-01-02 03:05:00	47 °C	34 °C	68%
2020-01-02 03:06:00	48 °C	31 °C	68%

## Standard field options

This section explains the available field options. They are listed in alphabetical order.

You can apply standard field options to most built-in Grafana workspace panels. Some older panels and community panels that have not updated to the new panel and data model will be missing either all or some of these field options.

Most field options will not affect the visualization until you choose outside of the field option box you are editing or press Enter.

For more information about applying these options, see the following sections:

- [Configure all fields \(p. 199\)](#).
- [Configure specific fields \(p. 201\)](#).

### Decimals

This option sets the number of decimals to include when rendering a value. Leave empty for Amazon Managed Grafana to use the number of decimals provided by the data source.

To change this setting, enter a number in the field.

## Data links

This option controls the URL to which a value or visualization links. For more information and instructions, see [Data links \(p. 263\)](#).

## Display name

This option sets the display title of all fields. You can use variables in the field title. For more information about templating and template variables, see [Templates and variables \(p. 265\)](#).

When multiple stats, fields, or series are shown, this field controls the title in each stat. You can use expressions such as `${__field.name}` to use only the series name or the field name in the title.

Given a field with a name of Temp, and labels of {"Loc":"PBI", "Sensor":"3"}

Expression syntax	Example	Renders to	Explanation
<code>\${__field.name}</code>	Same as syntax	Temp	Displays the field name, and labels in { } if they are present. If there is only one label key in the response, then for the label portion, Amazon Managed Grafana displays the value of the label without the enclosing braces.
<code>\${__field.name}</code>	Same as syntax	Temp	Displays the name of the field (without labels).
<code>\${__field.labels}</code>	Same as syntax	{"Loc":"PBI", "Sensor":"3"}	Displays the labels without the name.
<code>\${__field.labels, 1}</code>	Same as syntax	PBI	Displays the value of the specified label key.
<code>\${__field.labels, 3}</code>	Same as syntax	3	Displays the values of the labels separated by a comma (without label keys).

If the value is an empty string after rendering the expression for a particular field, the default display method is used.

## Max

This option sets the maximum value used in percentage threshold calculations. For automatic calculation based on all series and fields, leave this setting blank.

## Min

This option sets the minimum value used in percentage threshold calculations. For automatic calculation based on all series and fields, leave this setting blank.

## No value

Enter what Amazon Managed Grafana should display if the field value is empty or null.

## Unit

This option specifies what unit a field should use. Choose the **Unit** field, then drill down until you find the unit that you want. The unit that you select is applied to all fields except time.

## Custom units

You can also use the unit dropdown list to specify custom units, custom prefix or suffix, and date/time formats.

To select a custom unit, enter the unit and select the last `Custom: xxx` option in the dropdown list.

- `suffix:<suffix>` for custom unit that should go after value.
- `time:<format>` for custom date/time formats; for example, `time:YYYY-MM-DD`. For the format syntax and options, see [Display](#).
- `si:<base scale><unit characters>` for custom SI units; for example, `si: mF`. This option is a bit more advanced because you can specify both a unit and the source data scale. For example, if your source data is represented as milli (thousands of) something, prefix the unit with that SI scale character.
- `count:<unit>` for a custom count unit.
- `currency:<unit>` for custom a currency unit.

You can also paste a native emoji in the unit picker and pick it as a custom unit.

## String units

Amazon Managed Grafana can sometimes be too aggressive in parsing strings and displaying them as numbers. To make Amazon Managed Grafana show the original string, create a field override and add a unit property with the `string` unit.

## Color scheme

The field color option defines how Amazon Managed Grafana colors series or fields. There are multiple modes here that work very differently, and their utility depends largely on the currently selected visualization.

Continuous color modes use the percentage of a value relative to min and max to interpolate a color.

- **Single color** – Specific color set by using the color picker. This is most useful from an override rule.
- **From thresholds** – Color derived from the matching threshold. This is useful for gauges, stat, and table visualizations.

## Color by series

Amazon Managed Grafana includes color schemes that define color by series. This is useful for graphs and pie charts, for example.

## Color by value

Amazon Managed Grafana also includes continuous (gradient) color schemes. This is useful for visualizations that color individual values; for example, stat panels and the table panel.

## Thresholds

You can use thresholds to change the color of a field based on the value. For more information and instructions, see [Thresholds \(p. 233\)](#).

## Value mapping

You can use this option to set rules that translate a field value or range of values into explicit text. You can add more than one value mapping.

- **Mapping type** – Choose an option.
  - **Value** – Enter a value. If the field value is greater than or equal to the value, the **Text** is displayed.
  - **From** and **To** – Enter a range. If the field value is between or equal to the values in the range, the **Text** is displayed.
- **Text** – Text that is displayed if the conditions are met in a field. This field accepts variables.

## Panel editor

This topic describes the parts of the Amazon Managed Grafana panel editor, and it includes links to where you can find more information.

### Opening the panel editor

There are several ways to access the panel editor, also called the **Edit Panel** screen, *edit mode*, or *panel edit mode*.

- Choose the **Add panel** icon at the top of the screen, and then choose **Add new panel**. The new panel opens in the panel editor. For more information about how to add a panel, see [Adding or editing a panel \(p. 188\)](#).
- Choose the title of an existing panel, and then choose **Edit**. The panel opens in edit mode.
- Choose anywhere on an existing panel, and then press **e** on your keyboard. The panel opens in edit mode.

### Resizing panel editor sections

Drag to resize sections of the panel editor. If the side pane becomes too narrow, the **Panel**, **Field**, and **Overrides** tabs change to a dropdown list.

### Parts of the panel editor

This section describes the parts of the panel editor screen, with information about fields, options, or tasks associated with each part.

#### Header

The header section lists the name of the dashboard that the panel is in and some dashboard commands. You can also choose the **Go back** arrow to return to the dashboard.

On the right side of the header are the following options:

- **Dashboard settings (gear) icon** – Choose to access the dashboard settings.
- **Discard** Choose to discard all changes that you have made to the panel since you last saved the dashboard.
- **Save** – Choose to save the dashboard, including all changes that you have made in the panel editor.
- **Apply** – Choose to apply changes that you made and then close the panel editor, returning to the dashboard. You must also save the dashboard to persist the applied changes.

#### Visualization preview

The visualization preview section contains viewing options, time range controls, the visualization preview, and (if applicable) the panel title, axes, and legend.

- **Fill** – The visualization preview fills the available space in the preview part. If you change the width of the side pane or height of the bottom pane, the visualization adapts to fill whatever space is available.
- **Fit** – The visualization preview fills in the available space, but it preserves the aspect ratio of the panel.
- **Exact** – The visualization preview has the exact size as the size on the dashboard. If not enough space is available, the visualization scales down, preserving the aspect ratio.
- **Time range controls** – For more information, see [Time range controls \(p. 244\)](#).

## Data section (lowest pane)

The data section contains tabs where you enter queries, transform your data, and create alert rules (if applicable).

- **Query tab** – Select your data source and enter queries here. For more information, see [Queries \(p. 189\)](#).
- **Transform tab** – Apply data transformations. For more information, see [Transformations \(p. 192\)](#).
- **Alert tab** – Write alert rules. For more information, see [Creating alerts \(p. 286\)](#).

## Panel and field options (side pane)

This section contains tabs where you control almost every aspect of how your data is visualized. Not all tabs are available for each visualization.

Features on these tabs are documented in the following topics:

- [Adding or editing a panel \(p. 188\)](#)
- [Visualizations \(p. 208\)](#)
- [Field options and overrides \(p. 199\)](#)
- [Panel links \(p. 262\)](#) and [Data links \(p. 263\)](#), which help you connect your visualization to other resources

# Library panels

Library panels allow users to create reusable panels where any changes made to one instance of the library panel is reflected on every dashboard affecting all other instances where the panel is used. These panels can be saved in folders alongside Dashboards and streamline reuse of panels across multiple dashboards.

## Create a library panel

### Note

When you create library panels, the panel on the source dashboard is converted to a library panel as well. You will need to save the original dashboard once a panel is converted.

### To create a library panel

1. Create a panel as you normally would. You can also use an existing panel.
2. Choose the title of the panel and choose **Edit**.
3. In the panel display options side pane, choose the down arrow option to bring changes to the visualization.
4. Choose **Library panels**, and choose **Create new library panel**.
5. Enter a name for the library panel, and select the folder to save it in.

6. Choose **Create library panel** and then save the dashboard.

You can also create library panels by using the **Share** option for any panel.

Once created, you can modify the library panel using any dashboard on which it appears. Once the library panel changes are saved, all instances of the library panel will reflect these modifications.

## Add a library panel

### To add a library panel to a dashboard

1. Pause on the + option on the left menu, then choose **Create**.
2. Choose **Add a panel from the panel library**.
3. Filter the list of library panels to find the panel that you want.
4. Choose that panel and add it to the dashboard.

## Unlink a library panel

If you have a library panel on your dashboard that you want to modify without affecting all other instances of the library panel, you can unlink the library panel.

### To unlink a library panel from a dashboard

1. Pause on **Dashboard** on the left menu, then choose **Library panels**.
2. Select a library panel. You will see a list of all the dashboards where the library panel is used..
3. Select the panel that you want to unlink and update.
4. Choose the title of the panel and choose **Edit**.
5. Choose **Unlink**.

## Delete a library panel

Before you delete a library panel, verify that it is no longer in use on any dashboard.

### To delete a library panel

1. Pause on **Dashboard** on the left menu, then choose **Library panels**.
2. Select a library panel. You will see a list of all the dashboards where the library panel is used..
3. Select the panel that you want to delete.
4. Choose the delete icon next to the panel name.

## Parts of the panel editor

This section describes the parts of the panel editor screen, with information about fields, options, or tasks associated with each part.

### Header

The header section lists the name of the dashboard that the panel is in and some dashboard commands. You can also choose the **Go back** arrow to return to the dashboard.

On the right side of the header are the following options:

- **Dashboard settings (gear) icon** – Choose to access the dashboard settings.
- **Discard** Choose to discard all changes that you have made to the panel since you last saved the dashboard.
- **Save** – Choose to save the dashboard, including all changes that you have made in the panel editor.
- **Apply** – Choose to apply changes that you made and then close the panel editor, returning to the dashboard. You must also save the dashboard to persist the applied changes.

## Visualization preview

The visualization preview section contains viewing options, time range controls, the visualization preview, and (if applicable) the panel title, axes, and legend.

- **Fill** – The visualization preview fills the available space in the preview part. If you change the width of the side pane or height of the bottom pane, the visualization adapts to fill whatever space is available.
- **Fit** – The visualization preview fills in the available space, but it preserves the aspect ratio of the panel.
- **Exact** – The visualization preview has the exact size as the size on the dashboard. If not enough space is available, the visualization scales down, preserving the aspect ratio.
- **Time range controls** – For more information, see [Time range controls \(p. 244\)](#).

## Data section (lowest pane)

The data section contains tabs where you enter queries, transform your data, and create alert rules (if applicable).

- **Query tab** – Select your data source and enter queries here. For more information, see [Queries \(p. 189\)](#).
- **Transform tab** – Apply data transformations. For more information, see [Transformations \(p. 192\)](#).
- **Alert tab** – Write alert rules. For more information, see [Creating alerts \(p. 286\)](#).

## Panel and field options (side pane)

This section contains tabs where you control almost every aspect of how your data is visualized. Not all tabs are available for each visualization.

Features on these tabs are documented in the following topics:

- [Adding or editing a panel \(p. 188\)](#)
- [Visualizations \(p. 208\)](#)
- [Field options and overrides \(p. 199\)](#)
- [Panel links \(p. 262\)](#) and [Data links \(p. 263\)](#), which help you connect your visualization to other resources

# Visualizations

Amazon Managed Grafana offers a variety of visualizations to suit different use cases. The following sections list the visualizations that are available in Amazon Managed Grafana and their display settings.

### Topics

- [Alert list panel \(p. 209\)](#)
- [Bar chart panel \(p. 210\)](#)

- [Bar gauge panel \(p. 212\)](#)
- [Clock panel \(p. 213\)](#)
- [Dashboard list panel \(p. 213\)](#)
- [Gauge panel \(p. 214\)](#)
- [Graph panel \(p. 214\)](#)
- [Heatmap \(p. 218\)](#)
- [Histogram panel \(p. 220\)](#)
- [Logs panel \(p. 222\)](#)
- [News panel \(p. 222\)](#)
- [Node graph panel \(Beta\) \(p. 222\)](#)
- [Pie chart panel \(p. 223\)](#)
- [Stat panel \(p. 224\)](#)
- [State timeline panel \(p. 225\)](#)
- [Status history panel \(p. 226\)](#)
- [Table panel \(p. 227\)](#)
- [Text panel \(p. 229\)](#)
- [Time series panel \(p. 229\)](#)
- [Thresholds \(p. 233\)](#)
- [Inspect a panel \(p. 234\)](#)
- [Calculations list \(p. 236\)](#)

## Alert list panel

The alert list panel displays your dashboards alerts. You can configure the list to show current state or recent state changes. For more information about alerts, see [Alerts \(p. 282\)](#).

Use these settings to refine your visualization.

### Options

- **Show** – Choose whether the panel should display the current alert state or recent alert state changes.
- **Max Items** – Set the maximum number of alerts to list.
- **Sort order** – Select how to order the alerts displayed.
  - **Alphabetical (asc)** – Alphabetical order
  - **Alphabetical (desc)** – Reverse alphabetical order
  - **Importance** – By importance according to the following values, with 1 being the highest:
    - alerting: 1
    - no\_data: 2
    - pending: 3
    - ok: 4
    - paused: 5
- **Alerts from this dashboard** – Show alerts only from the dashboard that the alert list is in.

### Filter

Use the following options to limit the alerts shown to only those that match the query, folder, or tags that you choose:



- **Alert name** – Enter an alert name query.
- **Dashboard title** – Enter a dashboard title query.
- **Folder** – Select a folder. Only alerts from dashboards in the selected folder will be displayed.
- **Dashboard tags** - Select one or more tags. Only alerts from dashboards with one or more of the tags will be displayed.

## State filter

Choose which alert states to display in this panel.

- Ok
- Paused
- No data
- Execution error
- Alerting
- Pending

## Bar chart panel

This panel visualization allows you to graph categorical data.

### Supported data formats

Only one data frame is supported and it needs to have at least one string field that will be used as the category for an X or Y axis and one or more numerical fields. Example:

Browser	Market share
Chrome	50
Intetnet Explorer	17.5

If you have more than one numerical field, the panel shows grouped bars.

### Visualizing time series or multiple result sets

If you have multiple time series or tables you first need to join them using a join or reduce transform. For example if you have multiple time series and you want to compare their last and max value add the Reduce transform and specify Max and Last as options under Calculations.

## Bar chart options

Use these options to refine your visualizations:

### Orientation

- **Auto** – Grafana decides the bar orientation based on the panel dimensions.
- **Horizontal** – Makes the X axis the category axis.
- **Vertical** – Makes the Y axis the category axis.

### Show values

Control whether values are shown on top of or to the left of bars.

- **Auto** – Values are shown if there is space.
- **Always** – Always show values.
- **Never** – Never show values.

**Group width** controls the width of groups. 1=max and 0=Min width.

**Bar width** controls the width of bars. 1=max and 0=Min width.

**Line width** controls line width of the bars.

**Fill opacity** controls the fill opacity bars.

**Gradient mode** sets the mode of the gradient fill. Fill gradient is based on the line color. To change the color, use the standard color scheme field option. Gradient appearance is influenced by the Fill opacity setting.

- **None** – no gradient fill, this is the default setting.
- **Opacity** – Transparency of the gradient is calculated based on the values on the y-axis. Opacity of the fill is increasing with the values on the Y-axis.
- **Hue** – Gradient color is generated based on the hue of the line color.

**Tooltip mode** When you hover your cursor over the visualization, Grafana can display tooltips. Choose how tooltips behave.

- **Single** – The hover tooltip shows only a single series, the one that you are hovering over on the visualization.
- **All** – The hover tooltip shows all series in the visualization. Grafana highlights the series that you are hovering over in bold in the series list in the tooltip.
- **Hidden** – Do not display the tooltip when you interact with the visualization.

#### **Note**

Use an override to hide individual series from the tooltip.

**Legend mode** Use these settings to refine how the legend appears in your visualization.

- **List** – Displays the legend as a list. This is a default display mode of the legend.
- **Table** – Displays the legend as a table.
- **Hidden** – Hides the legend.

**Legend placement** Choose where to place the legend.

- **Bottom** – Below the graph.
- **Right** – To the right of the graph.

**Legend calculations** Choose which of the standard calculations to show in the legend. You can have more than one.

**Text size** Enter a Value to change the size of the text on your bar chart..

**Axis** se the following field settings to refine how your axes display. Some field options will not affect the visualization until you click outside of the field option box you are editing or press Enter.

- **Placement** – Sets the placement of the Y-axis.
- **Auto** – Grafana automatically assigns Y-axis to the series. When there are two or more series with different units, then Grafana assigns the left axis to the first unit and right to the following units.
- **Left** – Display all Y-axes on the left side.
- **Right** – Display all Y-axes on the right side.
- **Hidden** – Hide all Y-axes.
- **Label** – Set a Y-axis text label. If you have more than one Y-axis, then you can give assign different labels with an override.
- **Width** – Set a fixed width of the axis. By default, Grafana dynamically calculates the width of an axis.

By setting the width of the axis, data whose axes types are different can share the same display proportions. This makes it easier to compare more than one graph's worth of data because the axes are not shifted or stretched within visual proximity of each other.

- **Soft min and soft max** – Set a Soft min or soft max option for better control of Y-axis limits. By default, Grafana sets the range for the Y-axis automatically based on the dataset.

Soft min and soft max settings can prevent blips from turning into mountains when the data is mostly flat, and hard min or max derived from standard min and max field options can prevent intermittent spikes from flattening useful detail by clipping the spikes past a defined point.

You can set standard min/max options to define hard limits of the Y-axis.

## Bar gauge panel

The bar gauge simplifies your data by reducing every field to a single value. You choose how Amazon Managed Grafana calculates the reduction.

This panel can show one or more bar gauges depending on how many series, rows, or columns your query returns.

### Data and field options

Bar gauge visualizations allow you to apply the following options:

- [Transformations \(p. 192\)](#)
- [Field options and overrides \(p. 199\)](#)
- [Thresholds \(p. 233\)](#)

### Display options

Use the following options to refine your visualization:

- **Show** – Choose how Amazon Managed Grafana displays your data.
  - **Calculate** – Show a calculated value based on all rows. For a list of available calculations, see [Calculations list \(p. 236\)](#).
  - **All values** – Show a separate stat for every row. If you select this option, you can also select a **Limit**, or the maximum number of rows to display.
- **Value** – Select a reducer function that Amazon Managed Grafana will use to reduce many fields to a single value. Choose the **Value** list to see functions and brief descriptions.
- **Orientation** – Choose a stacking direction.
  - **Auto** – Amazon Managed Grafana selects what the orientation that it thinks fits best.
  - **Horizontal** – Bars stretch horizontally, left to right.

- **Vertical** – Bars stretch vertically, top to bottom.
- **Display mode** – Choose a display mode.
  - **Gradient** – Choose a threshold level to define a gradient.
  - **Retro LCD** – Display the gauge split into small cells that are lit or unlit.
  - **Basic** – Use a single color based on the matching threshold.
- **Show unfilled area** – Select this option if you want to render the unfilled region of the bars as dark gray. This option is not available for the Retro LCD display mode.

## Clock panel

The clock panel shows the current time or a countdown. It updates every second.

- **Mode** – The default is **time**. If you choose **countdown**, set the **Countdown Deadline** to start the countdown.
- **12 or 24 hour** – The options for showing the time are 12-hour format and 24-hour format.
- **Timezone** – The time zones are supplied by the moment timezone library. The default is the time zone on your computer.
- **Countdown Deadline** – Specify the time and date to count down to, if you have set **Mode** to **countdown**.
- **Countdown End Text** – Specify the text to show when the countdown ends.
- **Date/Time formatting options** – Customize the font size, weight, and date/time formatting. If you are showing a countdown and don't want to see the seconds ticking down, change the time format to `HH:mm` for the 24-hour clock or `h:mm A` for the 12-hour clock. For a complete list of options, see [Display](#).
- **Bg Color** – Select a background color for the clock.

## Dashboard list panel

The dashboard list panel allows you to display dynamic links to other dashboards. The list can be configured to use starred dashboards, recently viewed dashboards, a search query, and dashboard tags.

On each dashboard load, this panel queries the dashboard list, always providing the most up-to-date results.

### Options

Use the following options to refine your visualization:

- **Starred** – Display starred dashboards in alphabetical order.
- **Recently viewed** – Display recently viewed dashboards in alphabetical order.
- **Search** – Display dashboards by search query or tags. This option requires you to enter at least one value in **Query** or **Tags**.
- **Show headings** – Show the chosen list selection (Starred, Recently viewed, Search) as a heading.
- **Max items** – Set the maximum number of items to list per section. For example, if you leave this at the default value of 10 and choose to display Starred and Recently viewed dashboards, the panel displays up to 20 total dashboards, 10 in each section.

### Search

The following options apply only if the **Search** option is selected.

- **Query** – Enter the query you want to search by. Queries are case-insensitive, and partial values are accepted.
- **Folder** – Select the dashboard folders that you want to display.
- **Tags** – Enter your tags that you want to search by. Note that existing tags will not appear as you type, and tags *are* case sensitive.

**Note**

When multiple tags and strings appear, the dashboard list displays those matching *all* conditions.

## Gauge panel

**Gauge** is a single-value panel that can repeat a gauge for every series, column, or row.

### Data and field options

Gauge visualizations allow you to apply the following options:

- [Transformations \(p. 192\)](#)
- [Field options and overrides \(p. 199\)](#)
- [Thresholds \(p. 233\)](#)

### Display options

To refine your visualization, use the following options:

- **Show** – Choose how Amazon Managed Grafana displays your data.
  - **Calculate** – Show a calculated **Value** based on all rows. For a list of available calculations, see [Calculations list \(p. 236\)](#).
  - **All values** – Show a separate stat for every row. If you select this option, you can also select a **Limit**, or the maximum number of rows to display.
- **Orientation** – Choose a stacking direction.
  - **Auto** – Amazon Managed Grafana selects what it thinks is the best orientation.
  - **Horizontal** – Bars stretch horizontally, left to right.
  - **Vertical** – Bars stretch vertically, top to bottom.
- **Show threshold labels** – Choose whether to show threshold values.
- **Show threshold markers** – Choose whether to show a threshold band outside the inner gauge value band.

## Graph panel

A graph panel can render as a line, a path of dots, or a series of bars. This type of graph is versatile enough to display almost any time-series data.

### Data and field options

When using graph visualizations, you can apply the following options:

- [Transformations \(p. 192\)](#)
- **Alerts**. This is the only type of visualization that allows you to set alerts. For more information, see [Alerts \(p. 282\)](#).
- [Thresholds \(p. 233\)](#)

## Display options

To refine your visualization, use the following settings:

- **Bars** – Display values as a bar chart.
- **Lines** – Display values as a line graph.
- **Line width** – Specify the width of the line for a series. The default is 1.
- **Staircase** – Draw adjacent points as staircase.
- **Area fill** – Specify the amount of color fill for a series. The default is 1; 0 is none.
- **Fill gradient** – Specify the degree of gradient on the area fill. The default is 0, which is no gradient; 10 is a steep gradient.
- **Points** – Display points for values.
- **Point radius** – Control how large the points are.
- **Alert thresholds** – Display alert thresholds and regions on the panel.

## Stacking and null value

- **Stack** – Each series is stacked on top of another.
- **Percent** – Each series is drawn as a percentage of the total of all series. This option is available when **Stack** is selected.
- **Null value** – Specify how null values are displayed. *This is a very important setting.* See the note below.
  - **connected** – If there is a gap in the series, meaning a null value or values, the line will skip the gap and connect to the next non-null value.
  - **null** – If there is a gap in the series, meaning a null value, the line in the graph will be broken and show the gap. This is the default setting.
  - **null as zero** – If there is a gap in the series, meaning a null value, it will be displayed as a zero value in the graph panel.

### Important

If you are monitoring a server's CPU load and the load reaches 100 percent, the server will lock up, and the agent sending statistics will not be able to collect the load statistic. This leads to a gap in the metrics, and using the default *null* setting means that Amazon Managed Grafana will show the gaps and indicate that something is wrong. If this is set to *connected*, it will be easy to miss this signal.

## Hover tooltip

Use these settings to change the appearance of the tooltip that appears when you pause over the graph visualization.

- **Mode** – Determines how many series the hover tooltip shows.
  - **All series** – The hover tooltip shows all series in the graph. In the series list in the tooltip, the Grafana workspace highlights the series that you are pausing on in bold.
  - **Single** – The hover tooltip shows only a single series, the one that you are pausing on in the graph.
- **Sort order** – Sorts the order of series in the hover tooltip if you have selected **All series** mode. When you pause on a graph, Amazon Managed Grafana displays the values associated with the lines. Generally, users are most interested in the highest or lowest values. Sorting these values can make it much easier to find the data that you want.
  - **None** – The order of the series in the tooltip is determined by the sort order in your query. For example, series could be sorted alphabetically by series name.
  - **Increasing** – The series in the hover tooltip are sorted by value and in increasing order, with the lowest value at the top of the list.

- **Decreasing** – The series in the hover tooltip are sorted by value and in decreasing order, with the highest value at the top of the list.

## Series overrides

Series overrides allow a series in a graph panel to be rendered differently from the others. You can customize display options on a per-series bases or by using regex rules. For example, one series can have a thicker line width to make it stand out or be moved to the right Y-axis.

You can add multiple series overrides.

### To add a series override

1. Choose **Add series override**.
2. In **Alias or regex**, type or select a series. Choose the field to see a list of available series.  
  
For example, `/Network.* /` would match two series named `Network out` and `Network in`.
3. Choose **+** and then select a style to apply to the series. You can add multiple styles to each entry.

- **Bars** – Show series as a bar graph.
- **Lines** – Show series as a line graph.
- **Line fill** – Show a line graph with area fill.
- **Fill gradient** – Specify the area fill gradient amount.
- **Line width** – Set line width.
- **Null point mode** – Use this option to ignore null values or replace them with zero. This is important if you want to ignore gaps in your data.
- **Fill below to** – Fill the area between two series.
- **Staircase line** – Show series as a staircase line.
- **Dashes** – Show a line with dashes.
- **Hidden Series** Hide the series.
- **Dash Length** – Set the length of dashes in the line.
- **Dash Space** – Set the length of the spaces between the dashes in the line.
- **Points** – Show series as separate points.
- **Point Radius** – Set the radius for point rendering.
- **Stack** – Set the stack group for the series.
- **Color** – Set the series color.
- **Y-axis** – Set the series y-axis.
- **Z-index** – Set the series z-index (rendering order). This option is important when you are overlaying different styles, such as bar charts and area charts.
- **Transform** – Transform value to negative to render below the y-axis.
- **Legend** – Control whether a series is shown in the legend.
- **Hide in tooltip** – Control whether a series is shown in a graph tooltip.

## Axes

Use these options to control the display of axes in the visualization.

### Left Y/Right Y

Options are identical for both y-axes.

- **Show** – Choose to show or hide the axis.
- **Unit** – Choose the display unit for the y value.
- **Scale** – Choose the scale to use for the y value: **linear**, or **logarithmic**. The default is **linear**.
- **Y-Min** – The minimum y value. The default is **auto**.
- **Y-Max** – The maximum Y value. The default is **auto**.
- **Decimals** – Define how many decimals are displayed for the y value. The default is **auto**.
- **Label** – Specify the y axis label. The default is ""

## Y-Axes

- **Align** – Align left and right y-axes by value. The default is unchecked/false.
- **Level** – Enter the value to use for alignment of left and right y-axes, starting from Y=0. The default is 0. This option is available when **Align** is selected.

## X-Axis

- **Show** – Choose to show or hide the axis.
- **Mode** – The display mode completely changes the visualization of the graph panel. It's like three panels in one. The main mode is the time series mode with time on the x-axis. The other two modes are a basic bar chart mode with series on the x-axis instead of time and a histogram mode.
  - **Time** (default) – The x-axis represents time and the data is grouped by time (for example, by hour, or by minute).
  - **Series** – The data is grouped by series, and not by time. The y-axis still represents the value.
    - **Value** – This is the aggregation type to use for the values. The default is **total** (summing the values together).
  - **Histogram** – This option converts the graph into a histogram. A histogram is a kind of bar chart that groups numbers into ranges, often called buckets or bins. Taller bars show that more data falls in that range.

For more information about histograms, see [Introduction to histograms and heatmaps \(p. 38\)](#).

- **Buckets** – Sets the number of buckets to group the values by. If left empty, Amazon Managed Grafana tries to calculate a suitable number of buckets.
- **X-Min** – Filters out values from the histogram that are less than this minimum limit.
- **X-Max** – Filters out values that are greater than this maximum limit.

## Legend

Use these settings to refine how the legend appears in your visualization.

### Options

- **Show** – Clear to hide the legend. The default is selected (true).
- **As Table** – Select to display the legend in the table. The default is checked (true).
- **To the right** – Select to display the legend to the right.
- **Width** – Enter the minimum width for the legend in pixels. This option is available when **To the right** is selected.

### Values

Additional values can be shown alongside the legend names.



- **Min** – The minimum of all values returned from the metric query.
- **Max** – The maximum of all values returned from the metric query.
- **Avg** – The average of all values returned from the metric query.
- **Current** – The last value returned from the metric query.
- **Total** – The sum of all values returned from the metric query.
- **Decimals** – How many decimals are displayed for legend values and graph hover tooltips.

The legend values are calculated on the client side by Amazon Managed Grafana and depend on the type of aggregation or point consolidation that your metric query is using. All the above legend values cannot be correct at the same time.

For example, if you plot a rate such requests/second, which is probably using average as an aggregator, the Total in the legend will not represent the total number of requests. It is just the sum of all data points received by Amazon Managed Grafana.

### Hide series

Hide series when all values of a series from a metric query are of a specific value.

- **With only nulls** – Value=null (default unchecked)
- **With only zeroes** – Value=zero (default unchecked)

### Time regions

Time regions allow you to highlight certain time regions of the graph to make it easier to see, for example, weekends, business hours, and off-work hours. All configured time regions refer to UTC time.

## Heatmap

The heatmap panel visualization allows you to view histograms over time. For more information about histograms, see [Introduction to histograms and heatmaps \(p. 38\)](#).

### Axes options

Use these settings to adjust how axes are displayed in your visualization.

#### Y Axis

- **Unit** – The display unit for the y-axis value
- **Scale** – The scale to use for the y-axis value
  - **linear** – Linear scale
  - **log (base 2)** – Logarithmic scale with base 2
  - **log (base 10)** – Logarithmic scale with base 10
  - **log (base 32)** – Logarithmic scale with base 32
  - **log (base 1024)** – Logarithmic scale with base 1024
- **Y-Min** – The minimum y value (default is auto)
- **Y-Max** – The maximum y value (default is auto)
- **Decimals** – Number of decimals to render y-axis values with (default is auto)

#### Buckets

##### Note

If the data format is **Time series buckets**, this section is not be available.

- **Y Axis Buckets** – The number of buckets that the y-axis will be split into.
- **Size** – The size of each Y axis bucket (visible only if **Scale** is *linear*). This option has priority over **Y Axis Buckets**.
- **Split Factor** – (Only visible if **Scale** is *log (base 2)* or greater). By default, Amazon Managed Grafana splits y values by log base. This option enables you to split each default bucket into the specified number of buckets.
- **X Axis Buckets** – The number of buckets that the x-axis will be split into.
- **Size** – The size of each x-axis bucket. Number or time interval (10s, 5m, 1h, etc). Supported intervals: ms, s, m, h, d, w, M, y. This option has priority over **X Axis Buckets**.

### Bucket bound

When Data format is Time series buckets, the data source returns series with names representing bucket bound. But depending on the data source, a bound may be upper or lower. You can use this option to adjust a bound type. If **Auto** is set, a bound option is chosen based on panels' data source type.

### Bucket size

The Bucket count and size options are used by Amazon Managed Grafana to calculate how big each cell in the heatmap is. You can define the bucket size either by count (the first input box) or by specifying a size interval. For the y-axis, the size interval is just a value. For the X-bucket, you can specify a time interval in the **Size** input. For example, you can set the time range to 1h. This will make the cells 1h wide on the x-axis.

### Data format

Choose an option in the **Format** list.

- **Time series** – Amazon Managed Grafana does the bucketing by going through all time series values. The bucket sizes and intervals are set in the **Buckets** options.
- **Time series buckets** – Each time series already represents a y-axis bucket. The time series name (alias) must be a numeric value representing the upper or lower interval for the bucket. The Grafana workspace does no bucketing, so the bucket size options are hidden.

## Display options

Use these settings to refine your visualization.

### Colors

The color spectrum controls the mapping between value count (in each bucket) and the color assigned to each bucket. The color on the far-left side of the spectrum represents the minimum count, and the color on the far-right side represents the maximum count. Some color schemes are automatically inverted when using the light theme.

You can also change the color mode to **Opacity**. In this case, the color will not change, but the amount of opacity will change with the bucket count.

- **Mode**
  - **Opacity** – Bucket value represented by cell opacity. An opaque cell means the maximum value.
    - **Color** – Cell base color.
    - **Scale** – Scale for mapping bucket values to the opacity.
      - **linear** – Linear scale. Bucket value maps linearly to the opacity.
      - **sqrt** – Power scale. Cell opacity calculated as  $\text{value}^k$ , where  $k$  is a configured **Exponent** value. If the exponent is less than 1, you will get a logarithmic scale. If the exponent is greater than 1, you will get an exponential scale. In the case of 1, the scale will be the same as linear.

- **Exponent** – value of the exponent, greater than 0.
- **spectrum** – Bucket value represented by cell color.
- **Scheme** – If the mode is **spectrum**, select a color scheme.

### Color scale

By default, Amazon Managed Grafana calculates cell colors based on minimum and maximum buckets values. With **Min** and **Max**, you can overwrite those values. Think of a bucket value as a z-axis and Min and Max as Z-Min and Z-Max respectively.

- **Min** – Minimum value used for cell color calculation. If the bucket value is less than Min, it is mapped to the minimum color. The default is `series min value`.
- **Max** – Maximum value used for cell color calculation. If the bucket value is greater than Max, it is mapped to the maximum color. The default is `series max value`.

### Legend

Choose whether to display the heatmap legend on the visualization or not.

### Buckets

- **Hide zero** – Do not draw cells that have zero values.
- **Space** – Set the space between cells in pixels. Default is 1 pixel.
- **Round** – Set the cell roundness in pixels. Default is 0.

### Tooltip

- **Show tooltip** – Show the heatmap tooltip.
- **Histogram** – Show the y-axis histogram on the tooltip. The histogram represents distribution of the bucket values for the specific timestamp.
- **Decimals** – Set the number of decimals to render bucket value with (default is auto).

## Histogram panel

The histogram visualization calculates the distribution of values and presents them as a bar chart. The Y-axis and the height of each bar represent the count of values that fall into each bracket while the X-axis represents the value range.

Histogram visualization supports time series and any table results with one or more numerical fields.

### Display options

Use these options to refine your visualizations:

#### Bucket size

The size of the buckets. Leave this empty for automatic bucket sizing (~10% of the full range).

#### Bucket offset

If the first bucket should not start at zero. A non-zero offset has the effect of shifting the aggregation window. For example, 5-sized buckets that are 0-5, 5-10, 10-15 with a default 0 offset would become 2-7, 7-12, 12-17 with an offset of 2; offsets of 0, 5, or 10, in this case, would effectively do nothing.

Typically, this option would be used with an explicitly defined bucket size rather than automatic. For this setting to affect, the offset amount should be greater than 0 and less than the bucket size; values outside this range will have the same effect as values within this range.

### Combine series

This will merge all series and fields into a combined histogram.

**Line width** controls line width of the bars.

**Fill opacity** controls the fill opacity of the bars.

**Gradient mode** sets the mode of the gradient fill. Fill gradient is based on the line color. To change the color, use the standard color scheme field option. Gradient appearance is influenced by the Fill opacity setting.

- **None** – No gradient fill, this is the default setting.
- **Opacity** – Transparency of the gradient is calculated based on the values on the Y-axis. Opacity of the fill is increasing with the values on the Y-axis.
- **Hue** – Gradient color is generated based on the hue of the line color.

**Tooltip mode** When you hover your cursor over the graph, Grafana can display tooltips. Choose how tooltips behave:

- **Single** – The hover tooltip shows only the series that you are hovering over.
- **All** – The hover tooltip shows all the series in the visualization. Grafana highlights the series that you are hovering over in bold in the series list in the tooltip.
- **Hidden** – Do not display the tooltip.

### Note

Use an override to hide individual series from the tooltip.

## Legend options

When the legend option is enabled it can show either the value mappings or the threshold brackets. To show the value mappings in the legend, it's important that the Color scheme option under Standard options is set to Single color or Classic palette. To see the threshold brackets in the legend set the Color scheme to From thresholds.

**Legend mode** Use these settings to refine how the legend appears in your visualization.

- **List** – Displays the legend as a list. This is a default display mode of the legend.
- **Table** – Displays the legend as a table.
- **Hidden** – Hides the legend.

**Legend placement** Choose where to place the legend.

- **Bottom** – Below the graph.
- **Right** – To the right of the graph.

### Legend calculations

Choose which calculations to show in the legend. For more information, see [Calculations list \(p. 236\)](#).

## Logs panel

The logs panel visualization shows log lines from data sources that support logs, such as Elastic, Influx, and Loki. Typically, you would use this panel next to a graph panel to display the log output of a related process.

The logs panel shows the result of queries that were entered on the **Query** tab. The results of multiple queries are merged and sorted by time. You can scroll inside the panel if the data source returns more lines than can be displayed.

To limit the number of lines rendered, you can use the **Max data points** setting in the **Query options**. If it is not set, the data source will usually enforce a default limit.

## Display options

Use the following settings to refine your visualization:

- **Time** – Show or hide the time column. This is the timestamp associated with the log line as reported from the data source.
- **Unique labels** – Show or hide the unique labels column, which shows only non-common labels.
- **Wrap lines** – Toggle line wrapping.
- **Order** – Display results in descending or ascending time order. The default is **Descending**, showing the newest logs first. Set to **Ascending** to show the oldest log lines first.

## News panel

This panel displays an RSS feed. By default, it displays articles from the Grafana Labs blog.

In the **Display** section, in the **URL** field, enter the URL of an RSS feed. This panel type does not accept any other queries.

## Node graph panel (Beta)

The node graph panel visualizes directed graphs or networks. It uses directed force layout to effectively position of the nodes so it can help with displaying complex infrastructure maps, hierarchies or execution diagrams.

## Data requirements

The node graph panel requires a specific shape of the data to be able to display its nodes and edges. Not every data source or query can be visualized in this panel.

The Node graph visualization consists of *nodes* and *edges*.

- A *node* is displayed as a circle. A node might represent an application, a service, or anything else that is relevant from an application perspective.
- An *edge* is displayed as a line that connects two nodes. The connection might be a request, an execution, or some other relationship between the two nodes.

## Nodes

Usually, nodes show two statistical values inside the node and two identifiers just below the node, usually name and type. Nodes can also show another set of values as a color circle around the node, with

sections of different color represents different values that should add up to 1. For example, you can have the percentage of errors represented by red portion of the circle.

Additional details can be displayed in a context menu when which is displayed when you choose the node. There also can be additional links in the context menu that can target either other parts of the Grafana workspace or any external link.

## Edges

Edges can also show statistics when you hover over the edge. Similar to nodes, you can open a context menu with additional details and links by choosing the edge.

The first data source supporting this visualization is the AWS X-Ray data source for it's service map feature. For more information, see [AWS X-Ray \(p. 65\)](#).

Additional details can be displayed in a context menu when which is displayed when you choose the node. There also can be additional links in the context menu that can target either other parts of the Grafana workspace or any external link.

## Navigating the node graph

You can pan within the node graph by choosing outside of any node or edge and dragging the mouse.

You can zoom by using the buttons on the upper left corner of the node graph.

## Pie chart panel

The pie chart displays reduced series, or values in a series, from one or more queries, as they relate to each other, in the form of slices of a pie. The arc length, area and central angle of a slice are all proportional to the slices value, as it relates to the sum of all values. This type of chart is best used when you want a quick comparison of a small set of values in an aesthetically pleasing form.

Pie chart visualizations allow you to apply the following options:

- [Transformations \(p. 192\)](#).
- [Field options and overrides \(p. 199\)](#).
- [Thresholds \(p. 233\)](#).

## Options

You can use the following options to refine your visualization.

- **Show** – Choose how much information to show. **Calculate** reduces each value to a single value per series. **All values** displays every value from a single series.
- **Calculation** – Select a calculation to reduce each series when **Calculate** has been selected. For information about available calculations, see [Calculations list \(p. 236\)](#).
- **Limit** – When displaying every value from a single series, this limits the number of values displayed.
- **Fields** – Select which fields to display in the visualization.
  - **Numeric fields** – All fields with numeric values.
  - **All fields** – All fields that are not removed by transformations.
  - **Time** – All fields with time values.

## Labels

Select labels to display on the pie chart. You can select more than one.

- **Name** – The series or field name.
- **Percent** – The percentage of the whole.
- **Value** – The raw numerical value.

Labels are displayed in white over the body of the chart. You might need to select darker chart colors to make them more visible. Long names or numbers might be clipped.

## Legend placement and values

Choose where to display the legend.

- **Bottom** – Below the chart.
- **Right** – To the right of the chart.

Select values to display in the legend. You can select more than one. **Percent** is the percentage of the whole, **Value** is the raw numerical value.

## Stat panel

The stat panel shows a one large stat value with an optional graph sparkline. You can control the background or value color by using thresholds.

By default, the stat panel shows one of the following displays:

- Only the value for a single series or field.
- Both the value and name for multiple series or fields.

You can use the **Text mode** option to control whether the text is displayed or not.

## Data and field options

Stat visualizations allow you to apply the following options:

- [Transformations \(p. 192\)](#).
- [Field options and overrides \(p. 199\)](#).
- [Thresholds \(p. 233\)](#).

## Automatic layout adjustment

The panel automatically adjusts the layout depending on available width and height in the dashboard. It automatically hides the graph (sparkline) if the panel becomes too small.

## Display options

Use the following options to refine your visualization:

- **Show** – Choose how Amazon Managed Grafana displays your data.
  - **Calculate** – Show a calculated value based on all rows.
    - **Calculation** – Select a calculation to apply. For information about available calculations, see [Calculations list \(p. 236\)](#).
  - **All values** – Show a separate stat for each row.
    - **Limit** – Specify the maximum number of rows to display.

- **Fields** – Select a field name or a field type (including **All fields** or **Numeric fields**) to include in this panel.
- **Value** – Select a reducer function that Amazon Managed Grafana will use to reduce many fields to a single value. Choose the **Value** list to see functions and brief descriptions.
- **Orientation** – Choose a stacking direction.
  - **Auto** – Amazon Managed Grafana selects what it thinks is the best orientation.
  - **Horizontal** – Bars stretch horizontally, left to right.
  - **Vertical** – Bars stretch vertically, top to bottom.
- **Text mode** – You can use the **Text mode** option to control what text the panel displays. If only the name and color are important, and the value is not, change the **Text mode** to **Name**. The value is still used to determine color and is displayed in a tooltip.
  - **Auto** – If the data contains multiple series or fields, show both the name and the value.
  - **Value** – Show only the value, never the name. The name is displayed in tooltip.
  - **Value and name** – Always show the value and the name.
  - **Name** – Show the name instead of the value. The value is displayed in the tooltip.
  - **None** – Show nothing (empty). The name and the value are displayed in the tooltip.
- **Color mode** – Choose a color mode.
  - **Value** – Colors only the value and graph area.
  - **Background** – Colors the background as well.
- **Graph mode** – Choose a graph mode.
  - **None** – Hides the graph and shows only the value.
  - **Area** – Shows the area graph below the value. This option requires that your query returns a time column.
- **Alignment mode** – Choose an alignment mode.
  - **Auto** – If only a single value is shown (no repeat), the value is centered. If multiple series or rows are shown, the value is left-aligned.
  - **Center** – Stat value is centered.

## State timeline panel

The state timeline panel visualization shows discrete state changes over time. Each field or series is rendered as its unique horizontal band. State regions can either be rendered with or without values. This panel works well with string or boolean states but can also be used with time series. When used with time series, the thresholds are used to turn the numerical values into discrete state regions.

### State timeline options

Use these options to refine your visualizations:

#### Merge equal consecutive values

Controls whether Grafana merges identical values if they are next to each other.

#### Show values

Controls whether values are rendered inside the state regions. Auto will render values if there is sufficient space.

#### Align values

Controls value alignment inside state regions.

#### Row height

Controls how much space between rows there are. 1 = no space = 0.5 = 50% space.



### Line width

Controls the line width of state regions.

### Fill opacity

Controls the opacity of state regions.

## Value mappings

To assign colors to boolean or string values, use [Value mapping \(p. 204\)](#).

## Time series data with thresholds

The panel can be used with time series data as well. In this case, the thresholds are used to turn the time series into discrete colored state regions.

## Legend options

When the legend option is enabled it can show either the value mappings or the threshold brackets. To show the value mappings in the legend, it's important that the Color scheme option under Standard options is set to Single color or Classic palette. To see the threshold brackets in the legend set the Color scheme to From thresholds.

**Legend mode** Use these settings to refine how the legend appears in your visualization.

- **List** – Displays the legend as a list. This is a default display mode of the legend.
- **Table** – Displays the legend as a table.
- **Hidden** – Hides the legend.

**Legend placement** Choose where to place the legend.

- **Bottom** – Below the graph.
- **Right** – To the right of the graph.

## Status history panel

The Status history visualization shows periodic states over time. Each field or series is rendered as a horizontal row. Boxes are rendered and centered around each value.

Status history visualization works with string, boolean and numerical fields or time series. A time field is required. You can use value mappings to color strings or assign text values to numerical ranges.

## Display options

Use these options to refine your visualizations:

### Show values

Controls whether values are rendered inside the value boxes. Auto will render values if there is sufficient space.

**Column width** controls the width of boxes. 1=max and 0=Min width.

**Line width** controls line width of state regions.

**Fill opacity** controls the fill opacity of state regions.

## Value mappings

To assign colors to boolean or string values, use [Value mapping \(p. 204\)](#).

## Time series data with thresholds

The panel can be used with time series data as well. In this case, the thresholds are used to color the boxes. You can also use gradient color schemes to color values.

## Legend options

When the legend option is enabled it can show either the value mappings or the threshold brackets. To show the value mappings in the legend, it's important that the Color scheme option under Standard options is set to Single color or Classic palette. To see the threshold brackets in the legend set the Color scheme to From thresholds.

**Legend mode** Use these settings to refine how the legend appears in your visualization.

- **List** – Displays the legend as a list. This is a default display mode of the legend.
- **Table** – Displays the legend as a table.
- **Hidden** – Hides the legend.

**Legend placement** Choose where to place the legend.

- **Bottom** – Below the graph.
- **Right** – To the right of the graph.

## Table panel

The table panel is supports multiple modes for time series and for tables, annotation, and raw JSON data. This panel also provides date formatting, value formatting, and coloring options.

## Data and field options

Table visualizations allow you to apply the following options:

- [Transformations \(p. 192\)](#).
- [Field options and overrides \(p. 199\)](#).
- [Thresholds \(p. 233\)](#).

## Display options

- **Show header** – Show or hide column names imported from your data source.
- **Sort ascending/descending** – Choose a column title to change the sort order from the default to descending to ascending. Each time you choose, the sort order changes to the next option in the cycle. You can sort by only one column at a time.
- [Table field options \(p. 228\)](#) – Change field options such as column width, alignment, and cell display mode.
- [Filter table columns \(p. 229\)](#) – Temporarily change how column data is displayed. For example, you can order values from highest to lowest or hide specific values.

## Annotation support

Annotations are not currently supported in the new table panel.

## Table field options

This section explains all available table field options. The options are listed in the same order as in Amazon Managed Grafana. Options listed in this topic apply only to table panel visualizations.

Most field options will not affect the visualization until you choose outside of the field option box that you are editing or press Enter.

For more information about applying these options, see [Configure all fields \(p. 199\)](#) and [Configure specific fields \(p. 201\)](#).

### Column alignment

Choose how Amazon Managed Grafana should align cell contents:

- Auto (default)
- Left
- Center
- Right

### Column width

By default, Amazon Managed Grafana automatically calculates the column width based on the cell contents. In this field option, you can override the setting and define the width for all columns in pixels.

For example, if you enter 100 in the field, all the columns will be set to 100 pixels wide when you choose outside the field.

### Cell display mode

By default, Amazon Managed Grafana automatically chooses display settings. You can override the settings by choosing one of the following options to change all fields.

#### Note

If you set these in the **Field** tab, the display modes apply to all fields, including the time field. Many options work best if you set them in the **Override** tab.

### Color text

If thresholds are set, the field text is displayed in the appropriate threshold color.

### Color background

If thresholds are set, the field background is displayed in the appropriate threshold color.

### Gradient gauge

The threshold levels define a gradient.

### LCD gauge

The gauge is split up in small cells that are lit or unlit.

### JSON view

The value is shown formatted as code. If a value is an object, the JSON view that allows you to browse the JSON object appears when you pause on the value.

## Image

If you have a field value that is an image URL or a base64 encoded image, you can configure the table to display it as an image.

## Column filter

### Filter table columns

If you turn on the **Column filter** in table options, you can filter table options. For more information, see [Table field options \(p. 228\)](#).

#### Turn on column filtering

1. In Amazon Managed Grafana, choose the dashboard that shows the table with the columns that you want to filter.
2. On the table panel that you want to filter, [Opening the panel editor \(p. 205\)](#).
3. Choose the **Field** tab.
4. In **Table** options, turn on the **Column filter** option.

A filter icon appears next to each column title.

#### Filter column values

To filter column values, choose the filter (funnel) icon next to a column title. The Grafana workspace displays the filter options for that column.

Select the check boxes next to the values that you want to display. Enter text in the search field at the top to show those values in the display so that you can select them rather than scroll to find them.

#### Clear column filters

Columns with filters applied have a blue funnel displayed next to the title.

To remove the filter, choose the blue funnel icon, and then choose **Clear filter**.

## Text panel

You can use the text panel to make information and description panels for your dashboards.

In **Mode**, select whether you want to use markdown or HTML to style your text, then enter content in the box below. The Grafana workspace includes a title and paragraph to help you get started, or you can paste content from another editor.

## Time series panel

The time series panel can render a time series as a line, a path of dots, or a series of bars. This type of graph is versatile enough to display almost any time-series data.

### Note

You can migrate Graph panel visualizations to Time series visualizations. To migrate, on the **Panel** tab, choose **Time series visualization**. Grafana transfers all applicable settings.

Time series visualizations allow you to apply the following options:

- [Transformations \(p. 192\)](#)
- [Field options and overrides \(p. 199\)](#)

- [Thresholds \(p. 233\)](#)

You can also use field options to create different types of graphs or adjust your axes.

Use these settings to refine your visualization.

## Tooltip mode

When you hover your cursor over the graph, Grafana can display tooltips. Choose how tooltips behave:

- **Single** – The hover tooltip shows only the series that you are hovering over.
- **All** – The hover tooltip shows all the series in the graph. Grafana highlights the series that you are hovering over in bold in the series list in the tooltip.
- **Hidden** – Do not display the tooltip.

## Legend mode and placement

Choose how the legend appears.

- **List** – Displays the legend as a list. This is the default.
- **Table** – Displays the legend as a table.
- **Hidden** – Hides the legend.

Choose where to display the legend.

- **Bottom** – Below the graph.
- **Right** – To the right of the graph.

## Legend calculations

Choose which calculations to show in the legend. For more information, see [Calculations list \(p. 236\)](#).

## Graph time series as lines

This section explains how to use Time series field options to visualize time series data as lines and illustrates what the options do.

### Create the panel

1. Create a panel, selecting the **Time series** visualization. For more information, see [Adding or editing a panel \(p. 188\)](#).
2. In the **Panel editor**, choose **Field**.
3. In **Style**, choose **Lines**.

### Style the lines

There are a variety of options for styling the lines.

- **Line interpolation** – Choose how Grafana interpolates the series line. The choices are **Linear**, **Smooth**, **Step before**, and **Step after**.
- **Line width** – Set the line thickness between 0 and 10 pixels.
- **Fill opacity** – Set the opacity of the series fill, from 0 to 100 percent.

- **Gradient mode** – Set the mode of the gradient fill. Fill gradient is based on the line color. To change the color, use the standard color scheme field option.

Gradient appearance is influenced by the setting for **Fill opacity**.

The choices for gradient fill are **None**, **Opacity**, and **Hue**. With **Opacity**, Transparency of the gradient is calculated based on the values on the y-axis. Opacity of the fill is increasing with the values on the y-axis. With **Hue**, the gradient color is generated based on the hue of the line color.

- **Line style** – Set the style of the line. To change the color, use the standard color scheme field option.

Line style appearance is influenced by the settings for **Line width** and **Fill opacity**.

The choices for line style are **Solid**, **Dash**, and **Dots**.

- **Null values** – Choose how gaps in the data are displayed. Null values can be connected to form a continuous line or, optionally, set a threshold above which gaps in the data should no longer be connected. You can choose to **Never** connect data points with gaps, **Always** connect data points with gaps, or set a **Threshold** at which gaps in the data should no longer be connected.
- **Show points** – Choose when the points should be shown on the graph. The choices are **Auto**, **Always** and **Never**.

### Fill below to

This option is available only in the overrides tab.

#### To fill the area between two series

1. Select the fields to fill below.
2. In **Add override property**, choose **Fill below to**.
3. Select the series that you want the fill to stop at.

### Graph time series as bars

This section explains how to use Time series field options to visualize time series data as bars and illustrates what the options do.

#### Create the panel

1. Create a panel, selecting the **Time series** visualization. For more information, see [Adding or editing a panel \(p. 188\)](#).
2. In the **Panel editor**, choose **Field**.
3. In **Style**, choose **Bars**.

### Style the bars

There are a variety of options for styling the bars.

- **Bar alignment** – Set the position of the bar relative to a data point. The choices are **Before**, **Center**, and **After**.
- **Line width** – Set the thickness of the bar outlines between 0 and 10 pixels.
- **Fill opacity** – Set the opacity of the bar fill, from 0 to 100 percent.
- **Gradient mode** – Set the mode of the gradient fill. Fill gradient is based on the line color. To change the color, use the standard color scheme field option.

Gradient appearance is influenced by the setting for **Fill opacity**.

The choices for gradient fill are **None**, **Opacity**, and **Hue**. With **Opacity**, Transparency of the gradient is calculated based on the values on the y-axis. Opacity of the fill is increasing with the values on the y-axis. With **Hue**, the gradient color is generated based on the hue of the line color.

- **Show points** – Choose when the points should be shown on the graph. The choices are **Auto**, **Always** and **Never**.

## Graph time series as points

This section explains how to use Time series field options to visualize time series data as points and illustrates what the options do.

### Create the panel

1. Create a panel, selecting the **Time series** visualization. For more information, see [Adding or editing a panel \(p. 188\)](#).
2. In the **Panel editor**, choose **Field**.
3. In **Style**, choose **Points**.

### Style the points

When you graph as points, you can choose the point size.

- **Point size** – Choose the point size, between 1 and 40 pixels in diameter..

## Change axis display

This section explains how to use Time series field options to control the display of axes in the visualization and illustrates what the axis options do.

There are a variety of options for the axes.

- **Y-axis placement** – Set the placement of the y-axis. The choices are **Left**, **Right**, and **Hidden**.
- **Y-axis label** – Set a text label for the y-axis. If you have more than one y-axis, you can use the **Override** tab to assign them different labels.
- **Width** – Set the fixed width of the axis. By default, the Grafana workspace dynamically calculates the axis width. By setting the width of the axis, data whose axes types are different can share the same display proportions. This makes it easier to compare more than one graph's worth of data because the axes are not shifted or stretched within visual proximity of each other.
- **Soft min and soft max** – Set a **Soft min** or **Soft max** for better control of y-axis limits. By default, the Grafana workspace sets the range for the y-axis automatically based on the data.

**Soft min** or **Soft max** settings can prevent blips from appearing as mountains when the data is mostly flat, and hard min or max derived from standard min and max field options can prevent intermittent spikes from flattening useful detail by clipping the spikes past a defined point.

- **Scale** – Set the scale to use for y-axis values. The choices are **Linear** and **Logarithmic**.

## Graph stacked time series

This section explains how to use Time series panel field options to control the stacking of the series and illustrates what the stacking options do. Stacking allows Grafana to display series on top of each other. Be cautious when using stacking in the visualization as it can easily create misleading graphs. You can read more on why stacking may be not the best approach here: [Stacked Area Graphs Are Not Your Friend](#).

## Stack series in groups

The stacking group option is only available as an override.

### To stack series in the same group

1. In the Overrides section, create a field override for the **Stack series** option.
2. Choose the **Normal** stacking mode.
3. Name the stacking group that you want the series to appear in. The stacking group name option is available only when creating an override.

## Thresholds

Thresholds set the color of either the value text or the background depending on conditions that you define.

You can define thresholds one of two ways:

- **Absolute** thresholds are defined based on a number; for example, 80 on a scale of 1 to 150.
- **Percentage** thresholds are defined relative to minimum or maximum; for example, 80 percent.

You can apply thresholds to the following visualizations:

- [Bar gauge panel \(p. 212\)](#)
- [Gauge panel \(p. 214\)](#)
- [Graph panel \(p. 214\)](#)
- [Stat panel \(p. 224\)](#)
- [Table panel \(p. 227\)](#)

## Default thresholds

On visualizations that support it, Amazon Managed Grafana sets the following default threshold value: 80 = red; Base = green; Mode = Absolute.

The **Base** value represents minus infinity. It is generally the *good* color.

## Adding a threshold

You can add as many thresholds to a panel as you want. The Grafana workspace automatically sorts thresholds from highest value to lowest.

### Note

These instructions apply only to the stat, gauge, bar gauge, and table visualizations.

1. Choose the panel that you want to add a threshold to.
2. Choose the **Field** tab.
3. Choose **Add threshold**.

Amazon Managed Grafana adds a threshold with suggested numerical and color values.

4. Accept the recommendations or edit the new threshold.
  - **Edit color** – Choose the color dot that you want to change, and then select a new color.
  - **Edit number** – Choose the number that you want to change, and then enter a new number.



- **Thresholds mode** – Choose the mode to change it for all thresholds on this panel.
5. Choose **Save** to save the changes in the dashboard.

## Adding a threshold to a graph panel

In the graph panel visualization, you can use thresholds to add arbitrary lines or sections to the graph to make it easier to see when the graph crosses a particular threshold.

1. Choose the graph panel that you want to add a threshold to.
2. On the **Panel** tab, choose **Thresholds**.
3. Choose **Add threshold**.
4. Fill in as many fields as you want. Only the **T1** fields are required.
  - **T1** – Both values are required to display a threshold.
    - **lt** or **gt** – Select **lt** for less than or **gt** for greater than to indicate what the threshold applies to.
    - **Value** – Enter a threshold value. The Grafana workspace draws a threshold line along the y-axis at that value.
  - **Color** – Choose a condition that corresponds to a color, or define your own color.
    - **custom** – You define the fill color and line color.
    - **critical** – Fill and line color are red.
    - **warning** – Fill and line color are yellow.
    - **ok** – Fill and line color are green.
  - **Fill** – Choose whether the threshold fill is displayed.
  - **Line** – Choose whether the threshold line is displayed.
  - **Y-Axis** – Choose **left** or **right**.
5. Choose **Save** to save the changes in the dashboard.

## Deleting a threshold

1. Choose the panel you want to remove a threshold from.
2. Choose the **Field** tab. (Or, for a graph panel, choose the **Panel** tab.)
3. Choose the trash can icon next to the threshold that you want to remove.
4. Choose **Save** to save the changes in the dashboard.

## Inspect a panel

The panel inspector helps you understand and troubleshoot your panels. You can inspect the raw data for any Grafana workspace panel, export that data to a comma-separated values (CSV) file, view query requests, and export panel and data JSON.

### Panel inspector UI

The panel inspector displays **Inspect: <NameOfPanelBeingInspected>** at the top of the pane. Choose the arrow in the upper right corner to expand or reduce the pane.

The panel inspector consists of four tabs:

- **Data tab** – Shows the raw data returned by the query with transformations applied. Field options, such as overrides and value mappings, are not applied by default.
- **Stats tab** – Shows how long your query takes and how much it returns.

- **JSON tab** – Allows you to view and copy the panel JSON, panel data JSON, and data frame structure JSON. This is useful if you are provisioning or administering Amazon Managed Grafana.
- **Query tab** – Shows you the requests to the server sent when Amazon Managed Grafana queries the data source.

**Note**

Not all panel types include all four tabs. For example, dashboard list panels do not have raw data to inspect, so they do not display the Stats, Data, or Query tabs.

## Panel inspector tasks

In the panel inspector, you can inspect panels, inspect and download raw query results, inspect query performance, view panel JSON models, and view the raw request and response to the data source.

### Open the panel inspector

You can inspect any panel that you can view.

1. In the Grafana workspace console, choose the dashboard that contains the panel that you want to inspect.
2. Choose the title of the panel that you want to inspect, and then choose **Inspect**. Or pause over the panel title, and then press **i**.

The panel inspector pane opens on the right side of the screen.

### Inspect raw query results

View raw query results in a table. These are the data returned by the query with transformations applied and before the panel applies field options or field option overrides.

1. Open the panel inspector, and then choose the **Data** tab. Or in the panel menu, choose **Inspect, Data**.
2. If your panel contains multiple queries or queries multiple nodes, you have additional options.
  - **Select result** – Choose which result set data you want to view.
  - **Transform data**
    - **Join by time** – View raw data from all your queries at the same time, with one result set per column. Choose a column heading to reorder the data.

View raw query results in a table with field options and option overrides applied.

1. Open the **Data** tab in the panel inspector.
2. Above the table, choose on **Data display options**.
3. Choose the **Apply field configuration** toggle button.

### Download raw query results as a CSV file

Amazon Managed Grafana generates a CSV file in your default browser download location. You can open it in the viewer of your choice.

1. Open the panel inspector.
2. Inspect the raw query results as described above. Adjust settings until you see the raw data that you want to export.
3. Choose **Download CSV**.

To download a CSV file specifically formatted for Excel, expand the **Data options** panel and turn on the **Download for Excel** option before you choose **Download CSV**.

### Inspect query performance

The **Stats** tab displays statistics that tell you how long your query takes, how many queries you send, and the number of rows returned. This information can help you troubleshoot your queries, especially if any of the numbers are unexpectedly high or low.

1. Open the panel inspector.
2. Choose the **Stats** tab.

Statistics are displayed in read-only format.

### View panel JSON models

Explore and export panel, panel data, and data frame JSON models.

1. Open the panel inspector, and then choose the **JSON** tab. Or, in the panel menu, choose **Inspect, Panel JSON**.
2. In **Select source**, choose one of the following options:
  - **Panel JSON** – Displays a JSON object representing the panel.
  - **Panel data** – Displays a JSON object representing the data that was passed to the panel.
  - **DataFrame structure** – Displays the raw result set with your transformations, field configuration, and overrides applied.
3. You can expand or collapse portions of the JSON to explore it, or you can choose **Copy to clipboard** and paste the JSON in another application.

### View raw request and response to data source

1. Open the panel inspector, and then choose the **Query** tab. Or, in the panel menu, choose **Inspect, Query**.
2. Choose **Refresh**.

Amazon Managed Grafana sends a query to the server to collect information, and then it displays the result. You can drill down on specific portions of the query, expand or collapse all of it, or copy the data to the clipboard to use in other applications.

## Calculations list

This topic lists and defines the calculations used in Amazon Managed Grafana.

Among other places, these calculations are used in the **Transform** tab and the bar gauge, gauge, and stat visualizations.

Calculation	Description
All nulls	True when all values are null
All zeros	True when all values are 0
Change count	Number of times the field's value changes
Count	Number of values in a field

Calculation	Description
Delta	Cumulative change in value
Difference	Difference between first and last value of a field
Distinct count	Number of unique values in a field
First (not null)	First, not null value in a field
Max	Maximum value of a field
Mean	Mean value of all values in a field
Min	Minimum value of a field
Min (above zero)	Minimum, positive value of a field
Range	Difference between maximum and minimum values of a field
Step	Minimal interval between values of a field
Total	Sum of all values in a field

## Dashboards

A *dashboard* is a set of one or more panels organized and arranged into one or more rows. Amazon Managed Grafana ships with a variety of panels. Amazon Managed Grafana makes it easy to construct the right queries and customize the display properties so that you can create the dashboard you need. Each panel can interact with data from any configured data source.

### Manage dashboards

To control the time period for the dashboard, you can use the [Time range controls \(p. 244\)](#) in the upper right of the dashboard.

Dashboards can use templates and variables to make them more dynamic and interactive. For more information, see [Templates and variables \(p. 265\)](#).

Dashboards can use [Annotations \(p. 238\)](#) to display event data across panels. This can help correlate the time series data in the panel with other events.

Dashboards can be shared easily in a variety of ways. For more information, see [Sharing a dashboard \(p. 244\)](#).

Dashboards can be tagged, and the dashboard picker provides quick, searchable access to all dashboards in a particular organization.

### Rows

A *row* is a logical divider within a dashboard. It is used to group panels together.

Rows are always 12 *units* wide. These units are automatically scaled based on the horizontal resolution of your browser. You can control the relative width of panels within a row by setting their specific width.

Amazon Managed Grafana uses a unit abstraction to optimize appearance on all screen sizes.

### Note

With MaxDataPoint functionality, Amazon Managed Grafana can display the required number of data points, regardless of resolution or time range.

To collapse a row, choose the row title. If you save a dashboard with a row collapsed, the dashboard is saved in that state, and those graphs will not load until you expand the row.

Use the repeating rows functionality to dynamically create or remove entire rows of panels, based on the template variables selected.

## Annotations

Annotations provide a way to mark points on the graph with rich events. When you pause on an annotation, you can see an event description and event tags. The text field can include links to other systems for more detail.

### Native annotations

Amazon Managed Grafana comes with a native annotation store and the ability to add annotation events directly from the graph panel.

### Adding annotations

To add an annotation, press **Ctrl** or **Cmd** and choose where you want to add the annotation. To make the annotation searchable from other dashboards, add tags to it.

#### Adding region annotations

To create an annotation for a region, press **Ctrl** or **Cmd** while you choose the region.

### Built-in query

After you add an annotation, it remains visible. This is because a built-in annotation query exists on all dashboards. This annotation query fetches all annotation events that originate from the current dashboard and displays them on the panel where they were created. This includes alert state history annotations. You can stop annotations from being fetched and displayed by choosing the **Dashboard settings** (gear) icon, choosing **Annotations**, and then modifying the query named `Annotations & Alerts (Built-in)`.

When you copy a dashboard using the **Save As** feature, the new dashboard has a new dashboard ID, so annotations created on the source dashboard are not visible on the copy. If the source dashboard annotations have tags to filter by, you can show the annotations on the copy by adding a new **Annotation Query** and filtering by the tags.

### Query by tag

You can create new annotation queries that fetch annotations from the native annotation store by using the `-- Grafana --` data source and setting **Filter by** to **Tags**. Specify at least one tag. For example, create an annotation query named `outages`, and specify a tag named `outage`. This query will show all annotations that you create (from any dashboard or via API) that have the `outage` tag.

By default, if you add multiple tags in the annotation query, Amazon Managed Grafana will show only annotations that have all the tags you supplied. To show annotations that contain at least one of the tags you supplied, turn on **Match any**.

In Amazon Managed Grafana, it's possible to use template variables in the tag query. For example, if you have a dashboard showing stats for different services and a template variable that controls which services to show, you can use the same template variable in your annotation query to show annotations for only those services.

## Querying other data sources

Annotation events are fetched by using annotation queries. To add a new annotation query to a dashboard, choose the **Dashboard settings** (gear) icon, choose **Annotations**, and then choose **New**.

Specify a name for the annotation query. This name is displayed by the check box for showing or hiding annotation events for this query. For example, you might have two annotation queries named **Deploys** and **Outages**. You can select or clear the check boxes to specify which annotations to show.

### Annotation query details

The annotation query options are different for each data source.

- [Annotations using Graphite query editor](#)
- [Annotations using OpenSearch data source](#)
- [Annotations using Prometheus](#)
- [Annotations using MySQL](#)
- [Annotations using PostgreSQL](#)

## Dashboard folders

Folders are a way to organize and group dashboards. This is useful if you have a lot of dashboards or if multiple teams use the same Grafana workspace.

### Creating a folder

To create a folder, do one of the following:

- On the side menu, under the + icon, choose the **Create Folder** link.
- On the **Manage Dashboards** page, choose the **Create Folder** button.

On the **Create Folder** page, enter a unique name for the folder, and then choose **Create**.

When saving a dashboard, you can either choose an existing folder or create a new folder.

### Manage dashboards

On the **Manage Dashboards** page, you can perform a variety of tasks:

- Create a folder.
- Create a dashboard.
- Move dashboards into folders.
- Delete multiple dashboards.
- Navigate to a folder page (where you can set permissions for a folder or its dashboards).

### Dashboard Folder page

To open a Dashboard Folder page, choose the cog icon that appears when you pause on a folder in the dashboard list in the search results or on the **Manage Dashboards** page.

The Dashboard Folder page is similar to the **Manage Dashboards** page. On the Dashboard Folder page, you can perform the following tasks:

- Move or delete dashboards in a folder.
- Rename a folder (on the **Settings** tab).
- Set permissions for the folder (inherited by dashboards in the folder).

## Permissions

Permissions can be assigned to a folder and inherited by the dashboards that it contains. An Access Control List (ACL) is used where **Organization Role**, **Team** and Individual **User** can be assigned permissions. For more information, see [Dashboard and folder permissions \(p. 29\)](#).

## Playlist

A playlist is a list of dashboards that are displayed in a sequence. You can use a playlist to build situational awareness or to present your metrics to your team or visitors.

Amazon Managed Grafana automatically scales dashboards to any resolution, including big screens.

You can access the **Playlist** feature from the side menu, in the **Dashboards** submenu.

## Creating a playlist

A playlist presents dashboards in a sequence, with a set order and a time interval between dashboards.

1. To access the **Playlist** feature, pause on the side menu.
2. Choose **Playlists**.
3. Choose **New playlist**.
4. In the **Name** text box, enter a name for your playlist.
5. In the **Interval** text box, enter a time interval.

The time interval is the amount of time for Amazon Managed Grafana to stay on a particular dashboard before advancing to the next one on the playlist.

6. Next to each dashboard that you want to add to your playlist, choose **Add to playlist**.
7. Choose **Create**.

## Editing a playlist

You can edit playlists while creating them or after saving them.

1. To access the Playlist feature, pause on the side menu.
2. Choose **Playlists**.
3. Choose the playlist that you want to edit.

### Editing the name of a playlist

1. Choose the **Name** text box.
2. Edit the name.
3. Choose **Save** to save your changes.

### Editing the interval of a playlist

1. Choose the **Interval** text box.

2. Edit the interval.
3. Choose **Save** to save your changes.

## Adding a dashboard to a playlist

1. Next to the dashboard that you want to add, choose **Add to playlist**.
2. Choose **Save** to save your changes.

## Searching for a dashboard to add

1. Under **Add dashboards**, choose the **Search dashboards by name** text box.
2. Enter a name or regular expression.
3. If needed, filter your results by starred status or tags. By default, your starred dashboards appear as options to add to the playlist.
4. Choose **Save** to save your changes.

## Rearranging dashboard order

1. Next to the dashboard that you want to move, choose the up or down arrow.
2. Choose **Save** to save your changes.

## Removing a dashboard

1. Choose the x icon to remove a dashboard from the playlist.
2. Choose **Save** to save your changes.

## Deleting a playlist

1. Choose **Playlists**.
2. Next to the playlist that you want to delete, choose the x icon.

## Saving a playlist

You can save a playlist to add it to your **Playlists** page, where you can start it. Be sure to add all the dashboards that you want to appear in your playlist before you save it.

1. To access the **Playlist** feature, pause on the side menu.
2. Choose **Playlists**.
3. Choose the playlist.
4. Edit the playlist.

Ensure that your playlist has a **Name**, **Interval**, and at least one **Dashboard** added to it.

5. Choose **Save**.

## Starting a playlist

You can start a playlist in five different view modes. The mode determines how the menus and navigation bar are displayed on the dashboards.



By default, each dashboard is displayed for the amount of time entered in the **Interval** field, which can be set while creating or editing a playlist. After you start a playlist, you can control it by using the navbar at the top of your screen.

1. On the **Dashboards** menu, choose **Playlists**.
2. Next to the playlist that you want to start, choose **Start playlist**.
3. In the dropdown list, choose one of the following display modes:
  - **Normal mode**
    - The side menu remains visible.
    - The navbar, row, and panel controls appear at the top of the screen.
  - **TV mode**
    - The side menu is hidden or removed.
    - The navbar, row, and panel controls appear at the top of the screen.
    - TV mode is turned on automatically after 1 minute of user inactivity.
    - You can turn TV mode on manually by using the **d v** sequence shortcut, or by appending the parameter `?inactive` to the dashboard URL.
    - You can disable TV mode with any mouse movement or keyboard action.
  - **TV mode (with auto fit panels)**
    - The side menu is hidden or removed.
    - The navbar, row, and panel controls appear at the top of the screen.
    - Dashboard panels automatically adjust to optimize space on screen.
  - **Kiosk mode**
    - The side menu, navbar, row, and panel controls are completely hidden or removed from view.
    - You can turn Kiosk mode on manually by using the **d v** sequence shortcut after the playlist has started.
    - You can turn off Kiosk mode manually by using the same shortcut.
  - **Kiosk mode (with auto fit panels):**
    - The side menu, navbar, row, and panel controls are completely hidden or removed from view.
    - Dashboard panels automatically adjust to optimize space on screen.

## Controlling a playlist

You can control a playlist in **Normal** or **TV** mode after it has started by using the navigation bar at the top of your screen.

Button	Result
Next (double right arrow)	Advances to the next dashboard.
Back (left arrow)	Returns to the previous dashboard.
Stop (square)	Ends the playlist, and exits to the current dashboard.
Cycle view mode (monitor icon)	Changes the display of the dashboards to different view modes.
Time range	Displays data within a time range. It can be set to display the last 5 minutes up to 5 years ago, or a custom time range, using the dropdown arrow.

Button	Result
Refresh (circle arrow)	Reloads the dashboard to display the current data. It can be set to reload automatically, from every 5 seconds to 1 day, by using the dropdown arrow.

To stop the playlist from your keyboard, press **Esc**.

## Sharing a playlist in a view mode

You can share a playlist by copying the URL in the view mode that you want and pasting the URL to your destination.

1. From the **Dashboards** menu, choose **Playlists**.
2. Next to the playlist that you want to share, choose **Start playlist**, and then choose the view mode that you want.
3. To copy the URL to your clipboard, choose **Copy Link Address**.

For example, the URL for a playlist on the Grafana Play site in Kiosk mode could be `https://play.grafana.org/d/000000010/annotations?orgId=1&kiosk`

4. Paste the URL to your destination.

## Dashboard search

Dashboards can be searched by the dashboard name, filtered by one (or many) tags, or filtered by starred status. The dashboard search is accessed through the dashboard picker, available in the dashboard top navigation bar. The dashboard search can also be opened by using the shortcut **F**.

When using only a keyboard, you can use the keyboard arrow keys to navigate the results, and press **Enter** to open the dashboard that you want.

## Finding by dashboard name

Type any part of the dashboard name in the search bar. As you type, search returns results for any partial string match in real time.

Dashboard search is the following:

- Real time
- Not case sensitive
- Functional across stored and file-based dashboards

## Filtering by tags

Tags are a helpful way to organize your dashboards, especially as the number of dashboards grows. Tags can be added and managed in the dashboard **Settings**.

To filter the dashboard list by tag, choose any tag that appears in the right column. You can further filter the list by choosing additional tags.

To see a list of all available tags, choose the **Filter by tags** dropdown menu. When you select a tag, the dashboard search is instantly filtered.

When using only a keyboard, press **Tab** to focus on the tags link, press the down arrow key to find a tag, and press **Enter** to select the tag.

**Note**

When multiple tags are selected, Amazon Managed Grafana shows dashboards that include all the tags.

## Sharing a dashboard

To share a dashboard, choose **Share dashboard** (the share icon) in the top navigation bar. This opens the **Share** dialog box, where you can get a link to the current dashboard with the current selected time range and template variables. If you have made changes to the dashboard, be sure to save those changes before you copy the link. You can share a dashboard in the following ways.

### Dashboard snapshot

A dashboard snapshot is an instant way to share an interactive dashboard publicly. When creating the snapshot, Amazon Managed Grafana strips sensitive data such as queries (metric, template, and annotation) and panel links, leaving only the visible metric data and series names embedded in your dashboard. Dashboard snapshots can be accessed by anyone who has the link and can reach the URL.

### Publish snapshots

You can publish snapshots to your local instance.

## Sharing a panel

Choose a panel title to open the panel menu, and then choose **Share** in the panel menu to open the **Share Panel** dialog box. You can copy the link, which will take you to exactly this panel with the current time range and selected template variables. You can share a panel in the following ways.

## Time range controls

Amazon Managed Grafana provides several ways to manage the time ranges of the data that are being visualized, both at the dashboard level and at the panel level.

This topic describes supported time units and relative ranges, the common time controls, dashboard-wide time settings, and panel-specific time settings.

### Time units and relative ranges

The following time units are supported:

- **s** (seconds)
- **m** (minutes)
- **h** (hours),
- **d** (days)
- **w** (weeks)
- **M** (months)
- **y** (years)

Use the minus operator to step back in time, relative to now. To display the full period of the unit (such as day, week, or month), append `/<time unit>`.

Use the plus operator to step forward in time relative to now. You can use this feature to look at predicted data for the future.

Here are some examples:

Example relative range	From	To
Last 5 minutes	now-5m	now
The day so far	now/d	now
This week	now/w	now/w
Week to date	now/w	now
Previous month	now-1M/M	now-1M/M

## Common time range controls

The dashboard and panel time controls have a common user interface, with the following options.

### Current time range

The current time range, also called the *time picker*, shows the time range currently displayed in the dashboard or panel that you are viewing.

Pause on a field to see the exact timestamps in the range and their source, such as the local browser.

To change the time range, choose on the current time range. You can change the current time using a *relative time range*, such as the last 15 minutes, or an *absolute time range*, such as 2020-05-14 00:00:00 to 2020-05-15 23:59:59.

### Relative time range

Select the relative time range from the **Relative time ranges** list. Here are some examples of relative time ranges:

- Last 30 minutes
- Last 12 hours
- Last 7 days
- Last 2 years
- Yesterday
- Day before yesterday
- This day last week
- Today so far
- This week so far
- This month so far

### Absolute time range

Set an absolute time range one of two ways:

- Enter values in the **From** and **To** fields. You can enter exact time values or relative values, such as now-24h, and then choose **Apply time range**.

- Choose the **From** or **To** field. Amazon Managed Grafana displays a calendar. Choose the day or days that you want to use as the current time range and then choose **Apply time range**.

Amazon Managed Grafana also displays recently used absolute ranges.

## Zoom out (Cmd+Z or Ctrl+Z)

To view a larger time range in the dashboard or panel visualization, choose the **Time range zoom out** icon.

## Zoom in (for graph visualizations only)

In the graph visualization, drag to select the time range that you want to view.

## Refresh dashboard

Choose the **Refresh dashboard** icon to run every query on the dashboard immediately and refresh the visualizations. Amazon Managed Grafana cancels any pending requests when a new refresh is started.

By default, Amazon Managed Grafana does not automatically refresh the dashboard. Queries run on their own schedule according to the panel settings. However, if you want to regularly refresh the dashboard, choose the down arrow next to the **Refresh dashboard** icon, and then select a refresh interval.

## Dashboard time settings

Time settings are saved on a per-dashboard basis.

To access the dashboard time settings, choose the **Dashboard settings** (gear) icon at the top of the screen. The settings are in the **Time Options** section of the **General** tab.

- **Timezone** – The local time zone of the service or system that you are monitoring. This can be helpful when you monitor a system or service that operates across several time zones.
  - **Default** – The default selected time zone for the user profile, team, or organization. If no time zone is specified for the user profile, a team that the user is a member of, or the organization, Amazon Managed Grafana uses local browser time.
  - **Browser Time** The time zone that is configured for the browser that is being used. This is usually the time zone that is set on the computer.
  - **Coordinated Universal Time** – Standard ISO 8601 time zones, including UTC. For more information, see a [list of time zones](#).
- **Auto-refresh** – Customizable options the relative time and auto-refresh settings. Entries are separated by commas and can be any valid time unit.
- **Now delay now-** – Time delay value that overrides the **now** value. Most commonly, this feature is used to avoid null values by accommodating known delays in data aggregation.
- **Hide time picker** – Option for not displaying the time picker.

## Panel time overrides and time shift

In [Query options \(p. 190\)](#), you can override the relative time range for individual panels, causing them to be different from what is selected in the dashboard time picker in the top navigation bar. You can show metrics from different time periods or days at the same time.

## Controlling the time range using a URL

You can control the time range of a dashboard by providing the following query parameters in the dashboard URL:

- `from` – Defines the lower limit of the time range, specified in ms epoch or relative time. For more information, see [Relative time range \(p. 245\)](#).
- `to` – Defines the upper limit of the time range, specified in ms epoch or relative time. For more information, see [Relative time range \(p. 245\)](#).
- `time` and `time.window` – Define a time range from `time-time.window/2` to `time+time.window/2`. Both parameters should be specified in milliseconds. For example, `?time=1500000000000&time.window=10000` results in a 10-second time range from 1499999995000 to 1500000005000

## Exporting and importing dashboards

Amazon Managed Grafana Dashboards can easily be exported and imported, either from the UI or from the [HTTP API]

### Exporting a dashboard

Dashboards are exported in Amazon Managed Grafana JSON format, and contain everything you need, including layout, variables, styles, data sources, and queries, to import the dashboard at a later time.

The export feature is accessed in the share window, which you open by choosing the share button in the dashboard menu.

#### Making a dashboard portable

When you export a dashboard for others to use, it's good to add template variables for values such as a metric prefix (use a constant variable) and a server name.

A template variable of the type `Constant` is automatically hidden in the dashboard. It is also added as a required input when the dashboard is imported. For more information about templating and template variables, see [Templates and variables \(p. 265\)](#).

### Importing a dashboard

To import a dashboard, choose the **+** icon in the side menu, and then choose **Import**.

You can upload a dashboard JSON file, paste a dashboard URL or paste dashboard JSON text directly into the text area.

In step 2 of the import process, you can change the name of the dashboard, specify the data source that you want the dashboard to use, and specify any metric prefixes (if the dashboard uses any).

### Discover dashboards on Grafana.com

Find dashboards for common server applications at [Grafana.com/dashboards](https://grafana.com/dashboards).

## Dashboard version history

Whenever you save a version of your dashboard, a copy of that version is saved so that previous versions of your dashboard are not lost. A list of these versions is available by choosing **Dashboard settings** and then selecting **Versions** in the left side menu.

The dashboard version history feature lets you compare and restore to previously saved dashboard versions.

## Comparing two dashboard versions

To compare two dashboard versions, select the two versions from the list that you want to compare. After you select two versions, choose **Compare versions** to open the diff view. By default, you'll see a textual summary of the changes, as in the following image.

To view the diff of the raw JSON that represents your dashboard, choose **View JSON Diff**.

To restore to the earlier version that you are comparing against, choose **Restore to version <x>**.

## Restoring to a previously saved dashboard version

If you need to restore to a previously saved dashboard version, you can do so either by choosing the "Restore" button on the right of a row in the dashboard version list or by choosing **Restore to version <x>** appearing in the diff view. After you choose to restore, a pop-up box will prompt you to confirm the restoration.

After restoring to a previous version, a new version will be created that containing the same exact data as the previous version, but with a different version number. This is indicated in the **Notes** column. This helps to ensure that your previous dashboard versions are not affected by the change.

## Keyboard shortcuts

Amazon Managed Grafana has a number of keyboard shortcuts available. To display all keyboard shortcuts available in your version of Amazon Managed Grafana, press **Shift + ?** on your keyboard.

**Amazon Managed Grafana includes the following popular shortcuts:**

- **Ctrl+S** saves the current dashboard.
- **Ctrl+F** opens the dashboard finder / search.
- **Ctrl+H** hides all controls (hiding controls works well for TV displays).
- **Escape** exits a graph when in full-screen or edit mode.

## Dashboard JSON model

A dashboard in Amazon Managed Grafana is represented by a JSON object, which stores metadata of its dashboard. Dashboard metadata includes dashboard properties, metadata from panels, template variables, and panel queries.

**To view the JSON of a dashboard**

1. Open a dashboard.
2. On the top navigation bar, choose on **Manage dashboard**.
3. Select **View JSON** from the dropdown menu.

## JSON fields

When a user creates a new dashboard, a new dashboard JSON object is initialized with the following fields.

### Note

In the following JSON, `id` is shown as `null`, which is the default value assigned to it until a dashboard is saved. After you save a dashboard, an integer value is assigned to the `id` field.

```
{
  "id": null,
  "uid": "cLV5GDCkz",
  "title": "New dashboard",
  "tags": [],
  "style": "dark",
  "timezone": "browser",
  "editable": true,
  "hideControls": false,
  "graphTooltip": 1,
  "panels": [],
  "time": {
    "from": "now-6h",
    "to": "now"
  },
  "timepicker": {
    "time_options": [],
    "refresh_intervals": []
  },
  "templating": {
    "list": []
  },
  "annotations": {
    "list": []
  },
  "refresh": "5s",
  "schemaVersion": 17,
  "version": 0,
  "links": []
}
```

The following table provides usage details for each field in the dashboard JSON.

Name	Usage
<b>id</b>	The unique numeric identifier for the dashboard (generated by the database).
<b>uid</b>	The unique dashboard identifier that can be generated by anyone. The <code>uid</code> is a string of 8-40 characters.
<b>title</b>	The current title of the dashboard.
<b>tags</b>	The tags that are associated with the dashboard. In the JSON, the tags are an array of strings.
<b>style</b>	The theme of the dashboard (for example, dark or light).
<b>timezone</b>	The timezone of dashboard (utc or browser).
<b>editable</b>	Whether a dashboard can be edited.
<b>graphTooltip</b>	<p>The tooltip style.</p> <ul style="list-style-type: none"> <li>0 for no shared crosshair or tooltip (default)</li> <li>1 for shared crosshair</li> <li>2 for shared crosshair and shared tooltip</li> </ul>
<b>time</b>	The time range for the dashboard (for example, last 6 hours, last 7 days).
<b>timepicker</b>	The timepicker metadata. For more information, see <a href="#">Time picker (p. 250)</a> .



Name	Usage
templating	The templating metadata. For more information, see <a href="#">Templates and variables (p. 265)</a> .
annotations	The annotations metadata. For more information, see <a href="#">Annotations (p. 238)</a> .
refresh	The auto-refresh interval.
schemaVersion	The version of the JSON schema (integer), which is incremented each time an Amazon Managed Grafana update changes the schema.
version	The version of the dashboard (integer), which is incremented each time the dashboard is updated.
panels	The panels array. For more information, see <a href="#">Panels (p. 250)</a> .

## Panels

Panels are the building blocks of a dashboard. It consists of data source queries, type of graphs, aliases, and other data. Panel JSON consists of an array of JSON objects, each representing a different panel. Most of the fields are common for all panels, but some fields depend on the panel type. The following example shows the panel JSON of a text panel.

```
"panels": [  
  {  
    "type": "text",  
    "title": "Panel Title",  
    "gridPos": {  
      "x": 0,  
      "y": 0,  
      "w": 12,  
      "h": 9  
    },  
    "id": 4,  
    "mode": "markdown",  
    "content": "# title"  
  }  
]
```

### Panel size and position

The `gridPos` property describes the panel size and position in grid coordinates:

- `w` – 1-24. The width of the dashboard is divided into 24 columns.
- `h` – In grid height units. Each grid height unit represents 30 pixels.
- `x` – The x position. The x position uses the in same column unit as `w`.
- `y` – The y position. The y position uses the same grid height unit as `h`.

The grid has a negative gravity that moves panels up if there is empty space above a panel.

### Time picker

The following example shows the `timepicker` options.

```
"timepicker": {
```

```
"collapse": false,
"enable": true,
"notice": false,
"now": true,
"refresh_intervals": [
  "5s",
  "10s",
  "30s",
  "1m",
  "5m",
  "15m",
  "30m",
  "1h",
  "2h",
  "1d"
],
"status": "Stable",
"type": "timepicker"
}
```

The following table provides usage details for timepicker.

Name	Usage
collapse	Whether timepicker is collapsed
enable	Whether timepicker is activated
notice	TODO
now	TODO
refresh_intervals	TODO
status	TODO
type	TODO

## Templating

The templating field contains an array of template variables with their saved values and other metadata. The following example shows templating metadata.

```
"templating": {
  "enable": true,
  "list": [
    {
      "allFormat": "wildcard",
      "current": {
        "tags": [],
        "text": "prod",
        "value": "prod"
      },
      "datasource": null,
      "includeAll": true,
      "name": "env",
      "options": [
        {
          "selected": false,
          "text": "All",

```

```
        "value": "*"
      },
      {
        "selected": false,
        "text": "stage",
        "value": "stage"
      },
      {
        "selected": false,
        "text": "test",
        "value": "test"
      }
    ],
    "query": "tag_values(cpu.utilization.average,env)",
    "refresh": false,
    "type": "query"
  },
  {
    "allFormat": "wildcard",
    "current": {
      "text": "apache",
      "value": "apache"
    },
    "datasource": null,
    "includeAll": false,
    "multi": false,
    "multiFormat": "glob",
    "name": "app",
    "options": [
      {
        "selected": true,
        "text": "tomcat",
        "value": "tomcat"
      },
      {
        "selected": false,
        "text": "cassandra",
        "value": "cassandra"
      }
    ],
    "query": "tag_values(cpu.utilization.average,app)",
    "refresh": false,
    "regex": "",
    "type": "query"
  }
]
}
```

The following table provides usage details for templating section.

Name	Usage
<b>enable</b>	Whether templating is activated.
<b>list</b>	An array of objects, each representing one template variable
<b>allFormat</b>	The format to use while fetching all values from the data source (for example, wildcard, glob, regex, and pipe).
<b>current</b>	Shows current selected variable text or value on the dashboard
<b>data source</b>	Shows the data source for the variables

Name	Usage
<b>includeAll</b>	Whether the all value option is available
<b>multi</b>	Whether multiple values can be selected from variable value list
<b>multiFormat</b>	The format to use while fetching timeseries from the data source
<b>name</b>	The name of a variable
<b>options</b>	The array of variable text/value pairs available for selection on dashboard
<b>query</b>	The data source query that is used to fetch values for a variable
<b>refresh</b>	TODO
<b>regex</b>	TODO
<b>type</b>	The type of variable (custom, query, or interval)

## Scripted dashboards

### Warning

This feature is deprecated and will be removed in a future release.

If you have many metric names that change (for example, new servers) in a defined pattern, it can be time consuming to constantly create new dashboards.

With scripted dashboards, you can dynamically create your dashboards using JavaScript. In the Grafana install folder, under `public/dashboards/`, there is a file named `scripted.js`. This file contains an example of a scripted dashboard. You can access it by using the URL: `http://grafana_url/dashboard/script/scripted.js?rows=3&name=myName`

When you open `scripted.js`, you can see how it reads URL parameters from the `ARGS` variable and then adds rows and panels.

### Example: scripted.js

```
var seriesName = 'argName';

if (!_.isUndefined(ARGS.name)) {
  seriesName = ARGS.name;
}

dashboard.panels.push({
  title: 'Events',
  type: 'graph',
  fill: 1,
  linewidth: 2,
  gridPos: {
    h: 10,
    w: 24,
    x: 0,
    y: 10,
  },
  targets: [
    {
      target: "randomWalk('" + seriesName + "')",
    },
  ],
});
```

```
{
  target: "randomWalk('random walk2')",
},
],
});

return dashboard;
```

## More examples

You can find more examples in the `public/dashboards/` directory of your Grafana installation.

# Explore

In a Grafana workspace, the dashboard UI provides tools for building dashboards for visualization. *Explore* strips away all the dashboard and panel options so that you can focus on the query. Iterate until you have a working query, and then plan and build a dashboard.

For infrastructure monitoring and incident response, you no longer need to switch to other tools to debug what went wrong. You can use Explore to dig deeper into your metrics and logs to find the cause.

Explore makes it easier to view your data without creating a dashboard. If your data source supports graph and table data, Explore shows the results both as a graph and as a table. This helps you to see trends in the data and more details at the same time.

## Start exploring

### Note

By default, users with the Viewer role cannot edit and do not have access to Explore.

The **Explore** icon on the left side menu opens an empty Explore tab.

To start with an existing query in a panel, choose the **Explore** option from the **Panel** menu. This opens an Explore tab that contains the query from the panel. You can then to tweak or iterate in the query outside of your dashboard.

Choose your data source from the dropdown list in the top left. Prometheus has a custom Explore implementation. The other data sources use their standard query editor.

In the query field, you can write your query and explore your data. There are three buttons beside the query field, a clear button (X), an add query button (+) and the remove query button (-).As in the panel query editor, you can add and remove multiple queries.

## Splitting and comparing

The split view feature is a way to compare graphs and tables side-by-side or to look at related data together on one page. Choose **Split** to duplicate the current query and split the page into two side-by-side queries. You have the option of selecting a different data source for the new query. This gives you the opportunity to compare the same query for two different servers or to compare the staging environment to the production environment.

In split view, time pickers for both panels can be linked (if you change one, the other gets changed as well) by choosing one of the time-sync buttons attached to the time pickers. Linking the time pickers helps keep the start and end times of the split view queries in sync, so that you're looking at the same time interval in both split panels.

You can close the newly created query by choosing **Close Split**.

## Sharing a shortened link

Use the **Share shortened link** capability to create smaller and simpler URLs of the format `/goto/:uid` instead of sharing longer URLs that contain complex query parameters. You can create a shortened link by choosing the **Share** option in the Explore toolbar. Any shortened links that are never used are automatically deleted after 7 days.

## Query history

Query history is a list of queries that you have used in Explore. The history is local to your browser and is not shared. To open and interact with your history, choose **Query history** in Explore.

### Viewing the query history

In the query history, you can do the following:

- Run a query.
- Create or edit a comment.
- Copy a query to the clipboard.
- Copy a shortened link with the query to the clipboard.
- Star a query.

### Managing favorite queries

All queries that have been starred in the Query history tab are displayed on the Starred tab. You can access your favorite queries faster and reuse those queries without retyping them from.

### Sort query history

By default, query history shows you the most recent queries. You can sort your history by date or by data source name in ascending or descending order.

On the right side of the query history, in the dropdown list, choose one of the following options: field.

- Newest first
- Oldest first
- Data source A-Z
- Data source Z-A

#### Note

If you are in split view, the sorting mode applies only to the active panel.

### Filter query history

On the **Query history** and **Starred** tabs, you can filter the query history by data source name.

1. Choose **Filter queries for specific data source(s)**.
2. Select the data source that you want to use to filter your history. You can select multiple data sources.

On the **Query history** tab, you can use the vertical slider to filter queries by date:

- Drag the lower handle to adjust the start date.
- Drag the upper handle to adjust the end date.

**Note**

If you are in split view, filters are applied only to the active panel.

## Searching in the query history

You can search in your history across queries and your comments. Search is possible for queries in the **Query history** and **Starred** tabs.

1. Choose the **Search queries** field.
2. In the search field, enter your search term.

## Query history settings

You can customize the query history in the **Settings** tab. The following table lists the available options.

Setting	Default value
Specify how long Grafana will save your query history.	1 week
Change the default active tab.	Query history tab
Show queries only for the data source that is currently active in Explore.	True
Clear query history.	(Choose <b>Clear query history</b> to permanently delete all stored queries.)

**Note**

Query history settings are global, and they are applied to both panels in split mode.

## Prometheus-specific features

The first version of Explore features a custom querying experience for Prometheus. When you run a query, Grafana actually runs two queries: a normal Prometheus query for the graph and an Instant Query for the table. An Instant Query returns the last value for each time series, which shows a good summary of the data shown in the graph.

### Metrics explorer

On the left side of the query field, choose **Metrics** to open the Metric Explorer. This shows a hierarchical menu with metrics grouped by their prefix. For example, all Alertmanager metrics are grouped under the `alertmanager` prefix. This is a good starting point for exploring which metrics are available.

### Query field

The Query field supports automatic completion for metric names, function and works mostly the same way as the standard Prometheus query editor. Press **Enter** to run a query.

The Autocomplete menu can be accessed by pressing **Ctrl+Space**. The Autocomplete menu contains a new History section with a list of recently run queries.

Suggestions can appear under the Query field. Choose a suggestion to update your query with the suggested change.

- For counters (monotonically increasing metrics), a rate function is suggested.
- For buckets, a histogram function is suggested.
- For recording rules, possible to expand the rules.

## Table filters

Choose the **Filter** button in the **label** column of a table panel to add filters to the query expression. You can add filters for multiple queries as well. The filter is added for all the queries.

## Logs integration

You can also use Explore to investigate your logs with the following data sources:

- InfluxDB
- Elasticsearch

## Logs visualization

Results of log queries are shown as histograms in the graph, and individual logs are displayed below. If the data source does not send histogram data for the requested time range, the logs model computes a time series based on the log row counts bucketed by an automatically calculated time interval. The start of the histogram is then anchored by the first log row's timestamp from the result. The end of the time series is anchored to the time picker's **To** range.

### Log level

For logs where a **level** label is specified, Grafana uses the value of the label to determine the log level and update the color accordingly. If the log doesn't have a level label specified, Grafana parses the log to find out whether its content matches any of the supported expressions. The log level is always determined by the first match. If Grafana is not able to determine a log level, it will be visualized with the **unknown** log level. The following table lists log levels and the mapping of log level abbreviations and expressions.

Supported expressions	Log level	Color
emerg	critical	purple
fatal	critical	purple
alert	critical	purple
crit	critical	purple
critical	critical	purple
err	error	red
error	error	red



Supported expressions	Log level	Color
error	error	red
warn	warning	yellow
warning	warning	yellow
info	info	green
information	info	green
notice	info	green
debug	debug	blue
debug	debug	blue
trace	trace	light blue
*	unknown	grey

## Visualization options

You can customize how logs are displayed and select which columns are shown.

### Time

This option shows or hides the time column. This is the timestamp that is associated with the log line as reported from the data source.

### Unique labels

This option shows or hides the unique labels column, which includes only non-common labels. All common labels are displayed above.

### Wrap lines

To use line wrapping in the display, set this to **True**. Setting this option to **False** results in horizontal scrolling.

### Deduping

Log data can be very repetitive. Explore can help by hiding duplicate log lines. You can choose from different deduplication algorithms:

- **Exact** – Exact matches are done on the whole line except for date fields.
- **Numbers** – Matches are done on the line after stripping out numbers such as durations, IP addresses, and so on.
- **Signature** – The most aggressive deduping, this strips all letters and numbers. Matches are done on the remaining whitespace and punctuation.

### Flip results order

You can change the order of received logs from the default descending order (newest first) to ascending order (oldest first).

## Labels and detected fields

Each log row has an extendable area with its labels and detected fields, for more robust interaction. For all labels, you can filter for (positive filter) and filter out (negative filter) selected labels. Each field or label also has a stats icon to display one-time statistics in relation to all displayed logs.

## Toggle detected fields

If your logs are structured in JSON or logfmt, you can show or hide detected fields. Expand a log line, and then choose the eye icon to show or hide fields.

```
{{< docs-imagebox img="/img/docs/explore/parsed-fields-7-2.gif" max-width="800px"
caption="Toggling detected fields in Explore" >}}
```

## Tracing integration

You can visualize traces from tracing data sources in Explore. Data sources currently supported:

- [Jaeger \(p. 89\)](#)
- [Tempo \(p. 136\)](#)
- [AWS X-Ray \(p. 65\)](#)
- [Zipkin \(p. 137\)](#)

For information on using the query editor, see the documentation for the specific data source.

## Header

The header includes the following items:

- Header title, which shows the name of the root span and trace ID
- Search, which highlights spans that contain the searched text
- Metadata about the trace

## Minimap

Minimap shows a condensed view of the trace timeline. Drag your mouse over the minimap to zoom into a smaller time range. Zooming will also update the main timeline, so it's easy to see shorter spans. If you pause on the minimap, when zoomed, you can see the **Reset Selection** button, which resets the zoom.

## Timeline

The timeline shows a list of spans within the trace. Each span row consists of the following components:

- **Expand children** button: Expands or collapses all the children spans of selected span
- Service name: Name of the service logged the span
- Operation name: Name of the operation that this span represents
- Span duration bar: Visual representation of the operation duration within the trace

Choosing anywhere on the span row shows span details.

## Span details

The span details include the following items:

- Operation name
- Span metadata
- Tags (any tags associated with this span)
- Process metadata (metadata about the process that logged this span)
- Logs: List of logs logged by this span and associated key values. In case of Zipkin logs section shows Zipkin annotations.

## Navigating between Explore and a dashboard

To help accelerate workflows that involve regularly switching from Explore to a dashboard and vice-versa, we've added the ability to return to the origin dashboard after navigating to Explore from the panel's dropdown.

After you've navigated to Explore, you should notice a "Back" button in the Explore toolbar.

Simply choosing the button will return you to the origin dashboard, or, if you'd like to bring changes you make in Explore back to the dashboard, simply choose the arrow next to the button to reveal a "Return to panel with changes" menu item.

## Query inspector

To help with debugging queries, Explore allows you to investigate query requests and responses, as well as query statistics, via the Query inspector. This functionality is similar to the panel inspector **Stats** tab and **Query** tab. For more information, see [Inspect query performance \(p. 236\)](#) and [View raw request and response to data source \(p. 236\)](#).

## Linking

You can use links to navigate between commonly used dashboards or to connect others to your visualizations. Links let you create shortcuts to other dashboards, panels, and even external websites.

Amazon Managed Grafana supports dashboard links, panel links, and data links. Dashboard links are displayed at the top of the dashboard. Panel links are accessible by choosing an icon on the top-left corner of the panel.

## Which link should you use?

Start by examining how you're currently navigating between dashboards. If you're often jumping between a set of dashboards and struggling to find the same context in each, links can help optimize your workflow.

The next step is to figure out which link type is right for your workflow. Although all the link types in Grafana are used to create shortcuts to other dashboards or external websites, they work in different contexts.

- To add links that relate to most or all of the panels in the dashboard, use [Dashboard links \(p. 261\)](#).
- To drill down into specific panels, use [Panel links \(p. 262\)](#).
- To link to external sites, you can use either a dashboard link or a panel link.
- To drill down into a specific series, or even a single measurement, use [Data links \(p. 263\)](#).

## Controlling time range using the URL

You can control the time range of a panel or dashboard by providing the following query parameters in the dashboard URL:

- `from` defines lower limit of the time range, specified in ms epoch.
- `to` defines upper limit of the time range, specified in ms epoch.
- `time` and `time.window` define a time range from `time-time.window/2` to `time+time.window/2`. Both parameters should be specified in milliseconds. For example, `?time=1500000000000&time.window=10000` will result in a 10-second time range from 1499999995000 to 1500000005000

## Dashboard links

When you create a dashboard link, you can include the time range and current template variables to directly jump to the same context in another dashboard. This helps to ensure that the person you send the link to is looking at the right data. For other types of links, see [Data link variables \(p. 264\)](#).

After you add a dashboard link, it appears in the upper-right corner of your dashboard.

## Adding links to dashboards

Add links to other dashboards at the top of your current dashboard.

1. While viewing the dashboard that you want to add a link to, choose the gear icon at the top of the screen to open **Dashboard settings**.
2. Choose **Links**, and then choose **Add Dashboard Link** or **New**.
3. In **Type**, select **dashboards**.
4. Select link options:
  - **With tags** – Enter tags to limit the linked dashboards to only the ones with the tags you enter. Otherwise, the Grafana workspace includes links to all other dashboards.
  - **As dropdown** – Select this option if you are linking to many dashboards, and add an optional title to the dropdown list. If this option is not selected, the Grafana workspace displays the dashboard links side by side across the top of your dashboard.
  - **Time range** – Select this option to include the dashboard time range in the link. When the user chooses the link, the linked dashboard opens with the indicated time range already set.
  - **Variable values** – Select this option to include the template variables that are currently used as query parameters in the link. When the user chooses the link, any matching templates in the linked dashboard are set to the values from the link.
  - **Open in new tab** – Select this option to open the dashboard link in a new tab or window.
5. Choose **Add**.

## Adding a URL link to a dashboard

Add a link to a URL at the top of your current dashboard. You can link to any available URL, including dashboards, panels, or external sites. You can even control the time range to ensure that the user sees the specific data in the Grafana workspace.

1. While viewing the dashboard that you want to link to, choose the gear icon at the top of the screen to open **Dashboard settings**.
2. Choose **Links**, and then choose **Add Dashboard Link** or **New**.

3. In **Type**, select **link**.
4. Select link options:
  - **Url** – Enter the URL that you want to link to. Depending on the target, you might want to include field values.
  - **Title** – Enter the title that you want the link to display.
  - **Tooltip** – Enter the tooltip that you want the link to display when the user pauses over it.
  - **Icon** – Choose the icon that you want to display with the link.
  - **Time range** – Select this option to include the dashboard time range in the link. When the user chooses the link, the linked dashboard opens with the indicated time range already set.
    - `from` defines the lower limit of the time range, specified in ms epoch.
    - `to` defines the upper limit of the time range, specified in ms epoch.
    - `time` and `time.window` define a time range from `time-time.window/2` to `time+time.window/2`. Both parameters should be specified in milliseconds. For example, `?time=1500000000000&time.window=10000` results in a 10-second time range from 1499999995000 to 1500000005000.
  - **Variable values** – Select this option to include the template variables that are currently used as query parameters in the link. When the user chooses the link, any matching templates in the linked dashboard are set to the values from the link; for example, `https://play.grafana.org/d/000000074/alerting?var-app=backend&var-server=backend_01&var-server=backend_03&var-interval=1h`
  - **Open in new tab** – Select this option to open the dashboard link in a new tab or window.
5. Choose **Add**.

## Updating a dashboard link

To change or update an existing dashboard link, use the following procedure.

1. In **Dashboard Settings**, on the **Links** tab, choose the existing link that you want to edit.
2. Change the settings, and then choose **Update**.

## Duplicating a dashboard link

To duplicate an existing dashboard link, choose the duplicate icon next to the existing link that you want to duplicate.

## Deleting a dashboard link

To delete an existing dashboard link, choose the trash can icon for the link that you want to delete.

## Panel links

Each panel can have its own set of links that are shown in the upper-left corner of the panel. You can link to any available URLs, including dashboards, panels, or external sites. You can even control the time range to ensure the user sees the specific data in the Grafana workspace.

Choose the icon in the top-left corner of a panel to see available panel links.

## Adding a panel link

1. Pause on the panel that you want to add a link to, and then press **e**. Or choose the dropdown arrow next to the panel title, and then choose **Edit**.

2. On the **Panel** tab, scroll down to the **Links** section.
3. Expand **Links**, and then choose **Add link**.
4. Enter a **Title** for the link. The title will be displayed in the UI.
5. Enter the **URL** that you want to link to. You can include one of the template variables that are defined in the dashboard. Press **Ctrl+Space** or **Cmd+Space**, and then choose the **URL** field to see the available variables. When you add template variables to your panel link, the link sends the user to the right context, with the relevant variables already set. You can also use time variables
  - `from` defines the lower limit of the time range, specified in ms epoch.
  - `to` defines the upper limit of the time range, specified in ms epoch.
  - `time` and `time.window` define a time range from `time-time.window/2` to `time+time.window/2`. Both parameters should be specified in milliseconds. For example, `?time=1500000000000&time.window=10000` results in a 10-second time range from 1499999995000 to 1500000005000.
6. To open in a new tab, select **Open in a new tab**.
7. Choose **Save** to save changes and close the window.
8. Choose **Save** in the upper right to save your changes to the dashboard.

## Updating a panel link

1. On the **Panel** tab, find the link that you want to make changes to.
2. Choose the **Edit** (pencil) icon to open the **Edit link** window.
3. Make any necessary changes.
4. Choose **Save** to save changes and close the window.
5. Choose **Save** in the upper right to save your changes to the dashboard.

## Deleting a panel link

1. On the **Panel** tab, find the link that you want to delete.
2. Choose the **X** icon next to the link that you want to delete.
3. Choose **Save** in the upper right to save your changes to the dashboard.

## Data links

Data links provide more granular context to your links. You can create links that include the series name or even the value. For example, if your visualization shows four servers, you can add a data link to one or two of them.

The link itself is accessible in different ways, depending on the visualization. For the graph panel, you need to choose a data point or line. For a panel such as stat, gauge, or bar gauge, you can choose anywhere on the visualization to open the context menu.

You can use variables in data links to send people to a detailed dashboard with preserved data filters. For example, you can use variables to specify a time range, series, and variable selection. For more information, see [Data link variables \(p. 264\)](#).

## Typeahead suggestions

When you create or update a data link, press **Ctrl+Space** or **Cmd+Space** on your keyboard to open the typeahead suggestions to more easily add variables to your URL.

## Adding a data link

1. Pause on the panel that you want to add a link to, and then press **e**. Or choose the dropdown arrow next to the panel title, and then choose **Edit**.
2. On the **Field** tab, scroll down to the **Data links** section.
3. Expand **Data links**, and then choose **Add link**.
4. Enter a **Title** for the link. The title will be displayed in the UI.
5. Enter the **URL** that you want to link to.

You can add one of the template variables that are defined in the dashboard. Choose the **URL** field, and then type **\$**, or press **Ctrl+Space** or **Cmd+Space** to see a list of available variables. When you add template variables to your panel link, the link sends the user to the right context, with the relevant variables already set. For more information, see [Data link variables \(p. 264\)](#).

6. To open in a new tab, select **Open in a new tab**.
7. Choose **Save** to save changes and close the window.
8. Choose **Save** in the upper right to save your changes to the dashboard.

## Updating a data link

1. On the **Field** tab, find the link that you want to make changes to.
2. Choose the **Edit** (pencil) icon to open the **Edit link** window.
3. Make any necessary changes.
4. Choose **Save** to save changes and close the window.
5. Choose **Save** in the upper right to save your changes to the dashboard.

## Deleting a data link

1. On the **Field** tab, find the link that you want to delete.
2. Choose the **X** icon next to the link that you want to delete.
3. Choose **Save** in the upper right to save your changes to the dashboard.

## Data link variables

You can use variables in data links to see series fields, labels, and values. For more information about data links, see [Data links \(p. 263\)](#).

To see a list of available variables, enter **\$** in the data link **URL** field.

You can also use template variables in your data links URLs. For more information, see [Templates and variables \(p. 265\)](#).

## Time range panel variables

You can use the following variables to include the current time range in the data link URL:

- `__url_time_range` – The current dashboard's time range; for example, `?from=now-6h&to=now`
- `$__from` and `$__to` – For more information, see [\[Global variables\]\({{< relref "../variables/variable-types/global-variables.md#\\_\\_from-and-\\_\\_to" >}}\)](#).

## Series variables

Series-specific variables are available under the `__series` namespace:

- `__series.name` – Adds the series name to the URL
- `__series.labels.<LABEL>` – Adds the label's value to the URL. If your label contains dots, use `__series.labels["<LABEL>"]` syntax.

## Field variables

Field-specific variables are available under the `__field` namespace:

- `__field.name` – The name of the field

## Value variables

Value-specific variables are available under the `__value` namespace:

- `__value.time` – The value's timestamp (Unix ms epoch) to the URL; for example, `?time=1560268814105`
- `__value.raw` – The raw value
- `__value.numeric` – The numeric representation of a value
- `__value.text` – The text representation of a value
- `__value.calc` – The calculation name if the value is result of calculation

## Template variables

When linking to another dashboard that uses template variables, select variable values for whoever chooses the link.

Use `var-myvar=${myvar}`, where `myvar` is a name of the template variable that matches one in the current dashboard that you want to use.

To add all of the current dashboard's variables to the URL, use `__all_variables`.

# Templates and variables

A variable is a placeholder for a value. You can use variables in metric queries and in panel titles. Variables give you the ability to create more interactive and dynamic dashboards. Instead of hardcoding things like server, application, and sensor names in your metric queries, you can use variables in their place.

Variables are displayed as dropdown lists at the top of the dashboard. When you change the value by using the dropdown list at the top of the dashboard, your panel's metric queries reflect the new value.

These can be especially useful for administrators who want to allow viewers to adjust visualizations quickly but do not want to give them full editing permissions. Grafana viewers can use variables.

By using variables and templates, you can single-source dashboards. If you have multiple identical data sources or servers, you can make one dashboard and use variables to change what you are viewing. This simplifies maintenance and upkeep.



For a list of supported variable types, and instructions for adding each type of variable, see [Variable types \(p. 266\)](#)

## Templates

A *template* is any query that contains a variable.

For example, if you were administering a dashboard to monitor several servers, you could make a dashboard for each server. Or you could create one dashboard and use panels with template queries, as shown in the following example.

```
wmi_system_threads{instance=~"$server"}
```

Variable values are always synced to the URL by using the syntax `var-<varname>=value`.

## Variable best practices

Variable dropdown lists are displayed in the order they are listed in the variable list in **Dashboard settings**.

Put the variables that you will change often at the top, so that they will be shown first, at the far left on the dashboard.

## Variable syntax

Panel titles and metric queries can see variables by using two different syntaxes:

- `$varname` This syntax is easier to read, as in the following example: `apps.frontend.$server.requests.count`. However, you cannot use a variable in the middle of a word.
- `${var_name}` Use this syntax when you want to interpolate a variable in the middle of an expression.
- `${var_name:<format>}` This format gives you more control over how Grafana interpolates values. For more information, see [Advanced variable format options \(p. 276\)](#).

Before queries are sent to your data source, the query is *interpolated*, meaning that the variable is replaced with its current value. During interpolation, the variable value might be *escaped* to conform to the syntax of the query language and where it is used. For example, a variable that is used in a regex expression in a Prometheus query will be regex-escaped. Read the data source-specific documentation topic for details on value escaping during interpolation.

For information about advanced syntax to override data source default formatting, see [Advanced variable format options \(p. 276\)](#).

## Variable types

Grafana uses several types of variables.

Variable type	Description
Query	Query-generated list of values such as metric names, server names, sensor IDs, data centers, and so on. For more information, see <a href="#">Adding a query variable (p. 267)</a> .

Variable type	Description
Custom	Define the variable options manually using a comma-separated list. For more information, see <a href="#">Adding a custom variable (p. 268)</a> .
Text box	Display a text input field with an optional default value. For more information, see <a href="#">Adding a text box variable (p. 269)</a> .
Constant	Define a hidden constant. For more information, see <a href="#">Adding a constant variable (p. 270)</a> .
Data source	Quickly change the data source for an entire dashboard. For more information, see <a href="#">Adding a data source variable (p. 271)</a> .
Interval	Interval variables represent time spans. For more information, see <a href="#">Adding an interval variable (p. 271)</a> .
Ad hoc filters	Key/value filters that are automatically added to all metric queries for a data source (InfluxDB, Prometheus, and OpenSearch only). For more information, see <a href="#">Adding ad hoc filters (p. 272)</a> .
Global variables	Built-in variables that can be used in expressions in the query editor. For more information, see <a href="#">Global variables (p. 274)</a> .
Chained variables	Variable queries can contain other variables. For more information, see <a href="#">Chained variables (p. 273)</a> .

## Adding a query variable

Using query variables, you can write a data source query that returns a list of metric names, tag values, or keys. For example, a query variable might return a list of server names, sensor IDs, or data centers. The variable values change as they dynamically fetch options with a data source query.

Query expressions can contain references to other variables and, in effect, create linked variables. Grafana detects this and automatically refreshes a variable when one of its linked variables change.

### Query expressions

Query expressions are different for each data source. For more information, see the documentation for your data source at [Data sources \(p. 39\)](#).

### Entering general options

#### To enter general options for a query variable

1. Navigate to the dashboard that you want to make a variable for, and then choose the **Dashboard settings** (gear) icon at the top of the page.
2. On the **Variables** tab, choose **New**.
3. Enter a **Name** for your variable.

4. In the **Type** list, select **Query**.
5. (Optional) For **Label**, enter the display name of the variable dropdown list. If you don't enter a display name, the dropdown label will be the variable name.
6. Choose a **Hide** option:
  - **No selection (blank)** – The variable dropdown list displays the variable **Name** or **Label** value. This is the default.
  - **Label** – The variable dropdown list displays only the selected variable value and a down arrow.
  - **Variable** – No variable dropdown list is displayed on the dashboard.

## Entering query options

### To enter query options for a query variable

1. In the **Data source** list, select the target data source for the query. For more information about data sources, see [Data sources \(p. 39\)](#).
2. In the **Refresh** list, select when the variable should update options.
  - **Never** - Caches variable queries, and values are not updated. This is fine if the values never change, but problematic if they are dynamic and change a lot.
  - **On Dashboard Load** - Queries the data source every time the dashboard loads. This slows down dashboard loading, because the variable query must to be completed before dashboard can be initialized.
  - **On Time Range Change** - Queries the data source when the dashboard time range changes. Use this option only if your variable options query contains a time range filter or is dependent on the dashboard time range.
3. In the **Query** field, enter a query.
  - The query field varies according to your data source. Some data sources have custom query editors.
  - If you need more room in a single input field query editor, pause on the lines in the lower right corner of the field and drag downward to expand.
4. (Optional) In the **Regex** field, type a regex expression to filter or capture specific parts of the names returned by your data source query. For examples, see [Filtering variables with regex \(p. 280\)](#).
5. In the **Sort** list, select the sort order for values to be displayed in the dropdown list. The default option, **Disabled**, means that the order of options returned by your data source query will be used.
6. (Optional) Enter **Selection Options**. For more information, see [Entering variable selection options \(p. 276\)](#).
7. In **Preview of values**, the Grafana workspace displays a list of the current variable values. Review them to ensure they match what you expect.
8. Choose **Add** to add the variable to the dashboard.

## Adding a custom variable

Use a *custom* variable for values that do not change. This might be numbers, strings, or even other variables.

For example, if you have server names or region names that do not change, you can create them as custom variables rather than query variables. Because they do not change, you might use them in chained variables rather than other query variables. That would reduce the number of queries Grafana must send when chained variables are updated. For more information about chained variables, see [Chained variables \(p. 273\)](#).

## Entering general options

### To enter query options for a custom variable

1. Navigate to the dashboard you want to make a variable for and then choose the **Dashboard settings** (gear) icon at the top of the page.
2. On the **Variables** tab, choose **New**.
3. Enter a **Name** for your variable.
4. In the **Type** list, choose **Custom**.
5. (Optional) For **Label**, enter the display name of the variable dropdown list. If you don't enter a display name, the dropdown label will be the variable name.
6. Choose a **Hide** option:
  - **No selection (blank)** – The variable dropdown list displays the variable **Name** or **Label** value. This is the default.
  - **Label** – The variable list dropdown displays only the selected variable value and a down arrow.
  - **Variable** – No variable dropdown list is displayed on the dashboard.

## Entering custom options

### To enter custom options for a custom variable

1. In the **Values separated by comma** list, enter the values for this variable in a comma-separated list. You can include numbers, strings, other variables, or key-value pairs separated by a colon.
2. (Optional) Enter **Selection Options**. For more information, see [Entering variable selection options](#) (p. 276).
3. In **Preview of values**, the Grafana workspace displays a list of the current variable values. Review them to ensure they match what you expect.
4. Choose **Add** to add the variable to the dashboard.

## Adding a text box variable

*Text box* variables display a text input field with an optional default value. This is the most flexible variable, because you can enter any value. Use this type of variable if you have metrics with high cardinality or if you want to update multiple panels in a dashboard at the same time.

## Entering general options

### To enter general options for a text box variable

1. Navigate to the dashboard you want to make a variable for and then choose the **Dashboard settings** (gear) icon at the top of the page.
2. On the **Variables** tab, choose **New**.
3. Enter a **Name** for your variable.
4. In the **Type** list, select **Text box**.
5. (Optional) For **Label**, enter the display name of the variable dropdown list. If you don't enter a display name, then the dropdown label will be the variable name.
6. Choose a **Hide** option:
  - **No selection (blank)** – The variable dropdown list displays the variable **Name** or **Label** value. This is the default.
  - **Label** – The variable dropdown list displays only the selected variable value and a down arrow.

- **Variable** – No variable dropdown list is displayed on the dashboard.

## Entering text options

### To enter text options for a text box variable

1. (Optional) In the **Default value** field, select the default value for the variable. If you do not enter anything in this field, then Grafana displays an empty text box that you can type text into.
2. In **Preview of values**, Grafana displays a list of the current variable values. Review them to ensure they match what you expect.
3. Choose **Add** to add the variable to the dashboard.

## Adding a constant variable

To define a hidden constant, use *constant* variables. Constant variables are useful for metric path prefixes for dashboards you want to share. When you export a dashboard, constant variables are converted to import options.

Constant variables are not flexible. Each constant variable holds only one value. To update it, you must update the variable settings.

Constant variables are useful when you have complex values that you must include in queries but don't want to retype in every single query. For example, if you had a server path called `i-0b6a61efe2ab843gg`, you could replace it with a variable called `$path_gg`.

## Entering general options

### To enter general options for a constant variable

1. Navigate to the dashboard that you want to make a variable for and then choose the **Dashboard settings** (gear) icon at the top of the page.
2. On the **Variables** tab, choose **New**.
3. Enter a **Name** for your variable.
4. In the **Type** list, select **Constant**.
5. (Optional) For **Label**, enter the display name of the variable dropdown list. If you don't enter a display name, then the dropdown label will be the variable name.
6. Choose a **Hide** option:
  - **Variable** – No variable dropdown list is displayed on the dashboard. This is the default.
  - **No selection (blank)** – The variable dropdown list displays the variable **Name** or **Label** value.
  - **Label** – The variable dropdown list displays only the selected variable value and a down arrow.

## Entering constant options

### To enter constant options for a constant variable

1. In the **Value** field, enter the variable value. You can enter letters, numbers, and symbols. If you use advanced variable format options, you can even use wild cards. For more information, see [Advanced variable format options \(p. 276\)](#).
2. In **Preview of values**, the Grafana workspace displays the current variable value. Review it to ensure it matches what you expect.
3. Choose **Add** to add the variable to the dashboard.

## Adding a data source variable

To change the data source for an entire dashboard quickly, you can use *data source* variables. They are useful if you have multiple instances of a data source, perhaps in different environments.

### Entering general options

#### To enter general options for a data source variable

1. Navigate to the dashboard you want to make a variable for, and then choose the **Dashboard settings** (gear) icon at the top of the page.
2. On the **Variables** tab, choose **New**.
3. Enter a **Name** for your variable.
4. In the **Type** list, select **Datasource**.
5. (Optional) For **Label**, enter the display name of the variable dropdown list. If you don't enter a display name, the dropdown label will be the variable name.
6. Choose a **Hide** option:
  - **No selection (blank)** – The variable dropdown list displays the variable **Name** or **Label** value. This is the default.
  - **Label** – The variable dropdown list displays only the selected variable value and a down arrow.
  - **Variable** – No variable dropdown list is displayed on the dashboard.

### Entering data source options

#### To enter data source options for a data source variable

1. In the **Type** list, select the target data source for the variable. For more information about data sources, see [Data sources](#) (p. 39).
2. (Optional) For **Instance name filter**, enter a regex filter for which data source instances to choose from in the variable value drop-down list. Keep this field empty to display all instances.
3. (Optional) Enter **Selection Options**. For more information, see [Entering variable selection options](#) (p. 276).
4. In **Preview of values**, Grafana displays a list of the current variable values. Review them to ensure they match what you expect.
5. Choose **Add** to add the variable to the dashboard.

## Adding an interval variable

Use an *interval* variable to represent time spans such as 1m, 1h, 1d. You can think of them as a dashboard-wide group-by-time command. Interval variables change how the data is grouped in the visualization. You can also use the Auto option to return a set number of data points per time span.

You can use an interval variable as a parameter to group by time (for InfluxDB), date histogram interval (for OpenSearch), or as a summarize function parameter (for Graphite).

### Entering general options

#### To enter general options for an interval variable

1. Navigate to the dashboard you want to make a variable for and then choose the **Dashboard settings** (gear) icon at the top of the page.
2. On the **Variables** tab, choose **New**.

3. Enter a **Name** for your variable.
4. In the **Type** list, select **Interval**.
5. (Optional) For **Label**, enter the display name of the variable dropdownlist . If you don't enter a display name, the dropdown label will be the variable name.
6. Choose a **Hide** option:
  - **No selection (blank)** – The variable dropdown list displays the variable **Name** or **Label** value. This is the default.
  - **Label** – The variable dropdown list displays only the selected variable value and a down arrow.
  - **Variable** – No variable dropdown list is displayed on the dashboard.

## Entering interval options

### To enter interval options for an interval variable

1. In the **Values** field, enter the time range intervals that you want to appear in the variable dropdown list. The following time units are supported: s (seconds), m (minutes), h (hours), d (days), w (weeks), M (months), and y (years). You can also accept or edit the default values: 1m, 10m, 30m, 1h, 6h, 12h, 1d, 7d, 14d, 30d.
2. (Optional) Turn on **Auto Option** if you want to add the auto option to the list. Use this option to specify how many times the current time range should be divided to calculate the current auto time span. If you turn it on, then two more options appear:
  - **Step count** – Select the number of times that the current time range will be divided to calculate the value, similar to the **Max data points** query option. For example, if the current visible time range is 30 minutes, then the auto interval groups the data into 30 one-minute increments. The default value is 30 steps.
  - **Min Interval** – The minimum threshold below which the step count intervals will not divide the time. To continue the 30-minute example, if the minimum interval is set to 2m, Grafana groups the data into 15 2-minute increments.
3. In **Preview of values**, Grafana displays a list of the current variable values. Review them to ensure they match what you expect.
4. Choose **Add** to add the variable to the dashboard.

## Interval variable examples

Example using the template variable `myinterval` in a Graphite function:

```
summarize($myinterval, sum, false)
```

A more complex Graphite example:

```
groupByNode(summarize(movingAverage(apps.$app.$server.counters.requests.count, 5),  
'$interval', 'sum', false), 2, 'sum')
```

## Adding ad hoc filters

You can use one-time, or *ad hoc* filters to add key/value filters that are automatically added to all metric queries that use the specified data source. Unlike other variables, you do not use one-time filters in queries. Instead, you use them to write filters for existing queries.

**Note**

**Note:** One-time, or ad hoc, filter variables work only with InfluxDB, Prometheus, and OpenSearch data sources.

## Entering general options

### To enter general options for an ad hoc filter

1. Navigate to the dashboard that you want to make a variable for, and then choose the **Dashboard settings** (gear) icon at the top of the page.
2. On the **Variables** tab, choose **New**.
3. Enter a **Name** for your variable.
4. In the **Type** list, select **Ad hoc filters**.
5. (Optional) For **Label**, enter the display name of the variable dropdown list. If you don't enter a display name, the dropdown label will be the variable name.
6. Choose a **Hide** option:
  - **No selection (blank)** – The variable dropdown list displays the variable **Name** or **Label** value. This is the default.
  - **Label** – The variable dropdown list displays only the selected variable value and a down arrow.
  - **Variable** – No variable dropdown list is displayed on the dashboard.

## Entering options

### To enter options for an ad hoc filter

1. In the **Data source** list, select the target data source. For more information about data sources, see [Data sources \(p. 39\)](#).
2. Choose **Add** to add the variable to the dashboard.

## Creating ad hoc filters

Ad hoc filters are one of the most complex and flexible variable options available. Instead of a regular list of variable options, this variable enables the construction of a dashboard-wide ad hoc query. Filters that you apply in this manner are applied to all panels on the dashboard.

## Chained variables

*Chained variables*, also called *linked variables* or *nested variables*, are query variables with one or more other variables in their variable query. This section explains how chained variables work, and it provides links to example dashboards that use chained variables.

Chained variable queries are different for each data source, but the premise is the same for all. You can use chained variable queries in any data source that supports them.

You can build complex linked, templated dashboards, 5 or 10 levels deep. Technically, there is no limit to how deep or complex you can go, but the more links you have, the greater the query load.

## Best practices and tips

The following practices will make your dashboards and variables easier to use.

### Creating new chained variables

- Chaining variables creates parent-child dependencies. You can envision them as a ladder or a tree.



- The easiest way to create a new chained variable is to copy the variable that you want to base the new one on. In the variable list, choose the **Duplicate variable** icon to the right of the variable entry to create a copy. You can then add on to the query for the parent variable.
- New chained variables that you create this way appear at the bottom of the list. To give the list a logical order, drag the variable to a different position in the list.

### Variable order

To change the order of variables in the dashboard variable list, choose the up and down arrows on the right side of each entry. The Grafana workspace lists variable dropdown lists left to right according to this list, displaying the variable at the top of the list on the far left.

- List variables that do not have dependencies at the top, before their child variables.
- Each variable should follow the one that it is dependent on.
- The UI doesn't indicate which variables have dependency relationships. List the variables in a logical order to make it clearer to end users (and yourself).

### Complexity consideration

The more layers of dependency you have in variables, the longer it takes to update dashboards after you change variables.

For example, if you have a series of four linked variables (country, region, server, metric) and you change a root variable value (country), the Grafana workspace must run queries for all the dependent variables before it updates the visualizations in the dashboard.

## Global variables

Grafana has global built-in variables that can be used in expressions in the query editor. This topic lists them in alphabetical order and defines them. These variables are useful in queries, dashboard links, panel links, and data links.

### \$\_\_dashboard

This variable is the name of the current dashboard.

### \$\_\_from and \$\_\_to

Grafana has two built in time range variables: \$\_\_from and \$\_\_to. They are currently always interpolated as epoch milliseconds by default but you can control date formatting.

Syntax	Example result	Description
<code>\${__from}</code>	1594671549254	Unix millisecond epoch
<code>\${__from:date}</code>	2020-07-13T20:19:09.254Z	No args, defaults to ISO 8601/RFC 3339
<code>\${__from:date:iso}</code>	2020-07-13T20:19:09.254Z	ISO 8601/RFC 3339
<code>\${__from:date:seconds}</code>	1594671549	Unix seconds epoch

Syntax	Example result	Description
<code>\${__from:date:YYYY-MM}</code>	2020-07	Any custom data format. For more information, see <a href="#">Display</a> .

The above syntax works with `${__to}` as well.

You can use this variable in URLs as well. For example, to send an end user to a dashboard that shows a time range from six hours ago until now, use the following URL: `https://play.grafana.org/d/000000012/grafana-play-home?viewPanel=2&orgId=1?from=now-6h&to=now`

## `$__interval`

You can use the `$__interval` variable as a parameter to group by time (for InfluxDB, MySQL, PostgreSQL, MSSQL), Date histogram interval (for OpenSearch), or as a *summarize* function parameter (for Graphite).

The Grafana workspace automatically calculates an interval that can be used to group by time in queries. When there are more data points than can be shown on a graph, queries can be made more efficient by grouping by a larger interval. For example, it is more efficient to group by 1 day than by 10s when looking at 3 months of data. The graph will look the same, and the query will be faster. The `$__interval` is calculated by using the time range and the width of the graph (the number of pixels).

Approximate Calculation:  $(\text{from} - \text{to}) / \text{resolution}$

For example, when the time range is 1 hour and the graph is full screen, the interval might be calculated to 2m; points are grouped in 2-minute intervals. If the time range is 6 months and the graph is full screen, the interval might be 1d (1 day); points are grouped by day.

In the InfluxDB data source, the legacy variable `$interval` is the same variable. Use `$__interval` instead.

The InfluxDB and OpenSearch data sources have `Group by time interval` fields that are used to hardcode the interval or to set the minimum limit for the `$__interval` variable by using the `>` syntax - `> >10m`.

## `$__interval_ms`

This variable is the `$__interval` variable in milliseconds, not a time interval formatted string. For example, if the `$__interval` is 20m then the `$__interval_ms` is 1200000.

## `$__name`

This variable is only available in the Singlestat panel and can be used in the prefix or suffix fields on the **Options** tab. The variable will be replaced with the series name or alias.

## `$__org`

This variable is the ID of the current organization. The variable `${__org.name}` is the name of the current organization.

## `$__user`

The variable `${__user.id}` is the ID of the current user. The variable `${__user.login}` is the login handle of the current user. The variable `${__user.email}` is the email for the current user.

## `$__range`

This variable is currently supported only for Prometheus data sources. This variable represents the range for the current dashboard. It is calculated by `to - from`. It has millisecond and second representations called `$__range_ms` and `$__range_s`.

## `$timeFilter` or `$__timeFilter`

The `$timeFilter` variable returns the currently selected time range as an expression. For example, the time range interval `Last 7 days` expression is `time > now() - 7d`.

This is used in several places, including:

- The WHERE clause for the InfluxDB data source. Grafana adds it automatically to InfluxDB queries when in **Query Editor** mode. You can add it manually in **Text Editor** mode: `WHERE $timeFilter`.
- Log Analytics queries in the Azure Monitor data source.
- SQL queries in MySQL, Postgres, and MSSQL.
- The `$__timeFilter` variable is used in the MySQL data source.

# Other variable options

This section explains the other variable options that are available.

## Entering variable selection options

You can use **Selection Options** to manage variable option selections. All selection options are optional, and they are off by default.

### Multi-value

If you turn this option on, the variable dropdown list supports the selection of multiple options at the same time. For more information, see [Formatting multi-value variables \(p. 279\)](#).

### Include All option

The Grafana workspace adds an **All** option to the variable dropdown list. If an end user selects this option, all variable options are selected.

### Custom all value

This option is visible only if the **Include All option** is selected.

To define the value of the **All** option, enter regex, glob, or Lucene syntax in the **Custom all value** field.

By default, the **All** value includes all options in combined expression. This can become very long and can have performance problems. Sometimes it can be better to specify a custom all value, like a wild card regex.

When you use custom regex, glob, or Lucene syntax in the **Custom all value** option, it is never escaped, so you must consider what is a valid value for your data source.

## Advanced variable format options

The formatting of the variable interpolation depends on the data source, but there are some situations where you might want to change the default formatting.

For example, the default for the MySQL data source is to join multiple values as comma-separated with quotes: 'server01', 'server02'. In some cases, you might want to have a comma-separated string without quotes: server01,server02. To do this, use the following advanced variable formatting options.

## General syntax

Syntax: `${var_name:option}`

If any invalid formatting option is specified, `glob` is the default, or `fallback`, option.

## CSV

Formats variables with multiple values as a comma-separated string.

```
servers = ['test1', 'test2']  
String to interpolate: '${servers:csv}'  
Interpolation result: 'test1,test2'
```

## Distributed - OpenTSDB

Formats variables with multiple values in custom format for OpenTSDB.

```
servers = ['test1', 'test2']  
String to interpolate: '${servers:distributed}'  
Interpolation result: 'test1,servers=test2'
```

## Doublequote

Formats single-value and multi-value variables into a comma-separated string, escapes " in each value by \", and quotes each value with ".

```
servers = ['test1', 'test2']  
String to interpolate: '${servers:doublequote}'  
Interpolation result: '"test1","test2"'
```

## Glob - Graphite

Formats variables with multiple values into a glob (for Graphite queries).

```
servers = ['test1', 'test2']  
String to interpolate: '${servers:glob}'  
Interpolation result: '{test1,test2}'
```

## JSON

Formats variables with multiple values as a comma-separated string.

```
servers = ['test1', 'test2']  
String to interpolate: '${servers:json}'  
Interpolation result: '["test1", "test2"]'
```

## Lucene - OpenSearch

Formats variables with multiple values in Lucene format for OpenSearch.

```
servers = ['test1', 'test2']  
String to interpolate: '${servers:lucene}'  
Interpolation result: '("test1" OR "test2")'
```

## Percentencode

Formats single-value and multi-value variables for use in URL parameters.

```
servers = ['foo()bar BAZ', 'test2']  
String to interpolate: '${servers:percentencode}'  
Interpolation result: 'foo%28%29bar%20BAZ%20test2'
```

## Pipe

Formats variables with multiple values into a pipe-separated string.

```
servers = ['test1.', 'test2']  
String to interpolate: '${servers:pipe}'  
Interpolation result: 'test1.|test2'
```

## Raw

Turns off data source–specific formatting, such as single quotes in an SQL query.

```
servers = ['test1.', 'test2']  
String to interpolate: '${var_name:raw}'  
Interpolation result: '{test.1,test2}'
```

## Regex

Formats variables with multiple values into a regex string.

```
servers = ['test1.', 'test2']  
String to interpolate: '${servers:regex}'  
Interpolation result: '(test1\.|test2)'
```

## Singlequote

Formats single- and multi-value variables into a comma-separated string, escapes ' in each value by \ ' and quotes each value with '.

```
servers = ['test1', 'test2']  
String to interpolate: '${servers:singlequote}'  
Interpolation result: "'test1','test2'"
```

## Sqlstring

Formats single- and multi-value variables into a comma-separated string, escapes ' in each value by ' ' and quotes each value with ' '.

```
servers = ["test'1", "test2"]  
String to interpolate: '${servers:sqlstring}'  
Interpolation result: "'test' '1', 'test2'"
```

## Text

Formats single-value and multi-value variables into their text representation. For a single variable, it will just return the text representation. For multi-value variables, it will return the text representation combined with +.

```
servers = ["test1", "test2"]  
String to interpolate: '${servers:text}'  
Interpolation result: "test1 + test2"
```

## Formatting multi-value variables

Interpolating a variable with multiple values selected is tricky as it is not straight forward how to format the multiple values into a string that is valid in the given context where the variable is used. Grafana tries to solve this by enabling each data source plugin to inform the templating interpolation engine what format to use for multiple values.

### Note

The **Custom all value** option on the variable must be blank for Grafana to format all values into a single string. If leave it blank, then the Grafana concatenates (adds together) all the values in the query. Something like `value1,value2,value3`. If a custom all value is used, then instead the value will be something like `*` or `all`.

## Multi-value variables with a Graphite data source

Graphite uses glob expressions. A variable with multiple values would, in this case, be interpolated as `{host1,host2,host3}` if the current variable value was `host1`, `host2`, and `host3`.

## Multi-value variables with a Prometheus or InfluxDB data source

InfluxDB and Prometheus use regex expressions, so the same variable would be interpolated as `(host1|host2|host3)`. Every value would also be regex escaped. If not, a value with a regex control character would break the regex expression.

## Multi-value variables with an Elastic data source

Amazon OpenSearch uses Lucene query syntax, so the same variable would be formatted as `("host1" OR "host2" OR "host3")`. In this case, every value must be escaped so that the value contains only Lucene control words and quotation marks.

## Troubleshooting formatting

Automatic escaping and formatting can cause problems. It can be tricky to grasp the logic behind a problem, especially for InfluxDB and Prometheus, where the use of regex syntax requires that the variable is used in regex operator context.

If you do not want Grafana to do this automatic regex escaping and formatting, you must do one of the following:

- Turn off the **Multi-value Include All option** options.
- Use the [raw variable format]({{< relref "advanced-variable-format-options.md#raw" >}}).

## Filtering variables with regex

Using the Regex Query option, you can filter the list of options returned by the variable query or modify the options returned.

This section shows how to use regex to filter and modify values in the variable dropdown list.

Using the Regex Query option, you filter the list of options returned by the Variable query or modify the options returned. For more information, see [Regular expressions](#).

Examples of filtering on the following list of options:

```
backend_01  
backend_02  
backend_03  
backend_04
```

### Filtering so that only the options that end with 01 or 02 are returned

Regex:

```
/.*[01|02]/
```

Result:

```
backend_01  
backend_02
```

### Filtering and modifying the options using a regex capture group to return part of the text

Regex:

```
/.*(01|02)/
```

Result:

```
01  
02
```

### Filtering and modifying - Prometheus Example

List of options:

```
up{instance="demo.robustperception.io:9090",job="prometheus"} 1 1521630638000
up{instance="demo.robustperception.io:9093",job="alertmanager"} 1 1521630638000
up{instance="demo.robustperception.io:9100",job="node"} 1 1521630638000
```

Regex:

```
/.*instance="([^\"]*)"/
```

Result:

```
demo.robustperception.io:9090
demo.robustperception.io:9093
demo.robustperception.io:9100
```

## Filtering and modifying using named text and value capture groups

Using named capture groups, you can capture separate "text" and "value" parts from the options returned by the variable query. The variable dropdown list can contain a friendly name for each value that can be selected.

For example, when querying the `node_hwmon_chip_names` Prometheus metric, the `chip_name` is friendlier than the `chip` value. Start with the following variable query result.

```
node_hwmon_chip_names{chip="0000:d7:00_0_0000:d8:00_0",chip_name="enp216s0f0np0"} 1
node_hwmon_chip_names{chip="0000:d7:00_0_0000:d8:00_1",chip_name="enp216s0f0np1"} 1
node_hwmon_chip_names{chip="0000:d7:00_0_0000:d8:00_2",chip_name="enp216s0f0np2"} 1
node_hwmon_chip_names{chip="0000:d7:00_0_0000:d8:00_3",chip_name="enp216s0f0np3"} 1
```

Pass it through the following Regex.

```
/chip_name="(?(<text>[^\"]+)|chip="(?(<value>[^\"]+))/g
```

The following dropdown list is produced.

Display Name	Value
enp216s0f0np0	0000:d7:00_0_0000:d8:00_0
enp216s0f0np1	0000:d7:00_0_0000:d8:00_1
enp216s0f0np2	0000:d7:00_0_0000:d8:00_2
enp216s0f0np3	0000:d7:00_0_0000:d8:00_3

**Note:** Only text and value capture group names are supported.

## Repeating panels or rows

You can create dynamic dashboards using *template variables*. All variables in your queries expand to the current value of the variable before the query is sent to the database. With variables, you can reuse a single dashboard for all your services.

Template variables can be very useful for dynamically changing your queries across a whole dashboard. If you want Grafana to dynamically create new panels or rows based on the values that you have selected, you can use the *Repeat* feature.



## Repeating panels

If you have a variable with `Multi-value` or `Include all value` options turned on, you can choose one panel and have Grafana repeat that panel for every selected value. You can find the *Repeat* feature under the *General tab* in panel edit mode.

The `direction` controls how the panels are arranged.

If you choose `horizontal`, the panels are arranged side-by-side. Grafana automatically adjusts the width of each repeated panel so that the whole row is filled. Currently, you cannot mix other panels on a row with a repeated panel.

Set `Max per row` to tell Grafana how many panels per row you want at most. It defaults to 4.

If you choose `vertical`, the panels are arranged from top to bottom in a column. The width of the repeated panels is the same as of the first panel (the original template) that is being repeated.

Make changes only to the first panel (the original template). For the changes take effect on all panels, you need to start a dynamic dashboard re-build. You can do this by either changing the variable value (that is, the basis for the repeat) or reloading the dashboard.

### Note

Repeating panels require variables to have one or more items selected. You cannot repeat a panel zero times to hide it.

## Repeating rows

As seen above with the panels you can also repeat rows if you have variables set with `Multi-value` or `Include all value` selection option.

To turn on this feature, you must first add a new *Row* by using the *Add Panel* menu. Then pause on the row title and choose the cog button to access the *Row Options* configuration panel. You can then select the variable you want to repeat the row for.

A best practice is to use a variable in the row title as well.

# Alerts

Use alerts to identify problems in your system moments after they occur. By quickly identifying unintended changes in your system, you can minimize disruptions to your services.

Alerts consist of two parts:

- Alert rules – When the alert is triggered. Alert rules are defined by one or more conditions that are regularly evaluated by Grafana.
- Notification channel – How the alert is delivered. When the conditions of an alert rule are met, the Grafana notifies the channels configured for that alert.

Currently, only the graph panel visualization supports alerts.

## Alert configuration

You can configure alerts in your Amazon Managed Grafana workspace.

- Add or edit an alert notification channel. For more information, see [Notifications \(p. 283\)](#).
- Create an alert rule. For more information, see [Creating alerts \(p. 286\)](#).

- View existing alert rules and their current state. For more information, see [Viewing existing alert rules \(p. 289\)](#).
- Test alert rules and troubleshoot. For more information, see [Troubleshooting alerts \(p. 290\)](#).

## Clustering

Currently, alerting supports a limited form of high availability. Alert notifications are deduplicated when you run multiple workspaces. This means that all alerts are run on every server, but no duplicate alert notifications are sent due to the deduping logic.

## Notifications

You can create alert rules with detailed messages including information such as how you might solve the issue, a link to a runbook, and so on.

The actual notifications are configured and shared between multiple alerts.

## Alert execution

Alert rules are evaluated in Amazon Managed Grafana in a scheduler and query execution engine.

## Alert notifications

When an alert changes state, it sends out notifications. Each alert rule can have multiple notifications. To add a notification to an alert rule, you first must add and configure a notification channel.

This is done from the Notification channels page.

## Adding a notification channel

1. In the side bar, pause on the **Alerting** (bell) icon, and then choose **Notification channels**.
2. Choose **Add channel**.
3. Fill out the fields or select options described in the following sections.

## New notification channel fields

### Default (send on all alerts)

- **Name** – Enter a name for this channel. It will be displayed when users add notifications to alert rules.
- **Type** – Select the channel type. For more information, see [List of supported notifiers \(p. 284\)](#).
- **Default (send on all alerts)** – When selected, this option sends a notification on this channel for all alert rules.
- **Disable Resolve Message** – When selected, this option disables the resolve message [OK] that is sent when the alerting state returns to false.
- **Send reminders** – When this option is selected, additional notifications (reminders) will be sent for alerts. You can specify how often reminders should be sent by using the number of seconds (s), minutes (m), or hours (h); for example, 30s, 3m, 5m or 1h.

### Important

Alert reminders are sent after rules are evaluated. Therefore, a reminder can't be sent more frequently than a configured alert rule evaluation interval.

The following examples show how often and when reminders are sent for a triggered alert.

Alert rule evaluation interval	Send reminders every	Reminder sent every (after last alert notification)
30s	15s	~30 seconds
1m	5m	~5 minutes
5m	15m	~15 minutes
6m	20m	~24 minutes
1h	15m	~1 hour
1h	2h	~2 hours

## List of supported notifiers

Name	Type	Supports images	Supports alert rule tags
<a href="#">Amazon Simple Notification Service (p. 284)</a>		No	No
OpsGenie	opsgenie	No	Yes
<a href="#">PagerDuty (p. 285)</a>	pagerduty	No	Yes
<a href="#">Slack (p. 284)</a>	slack	No	No
VictorOps	victorops	No	No

### Amazon Simple Notification Service

If you have enabled service-managed permissions and included Amazon SNS as a notification channel for your workspace, you only need to provide the SNS Topic ARN when you create your notification channel. In the **Name** field, provide the name of the SNS topic that you have created. If you created the workspace using service-managed permissions, the SNS topic name must be prefixed with `grafana` for notifications to successfully publish to the topic. If you selected customer-managed permissions when you created the workspace, the SNS Topic name does not need to be prefixed with `grafana`. In the **Topic** field, copy and paste the ARN of the SNS topic.

If you use customer-managed permissions for the workplace, the IAM role that you supply should include SNS Publish permissions for your SNS Topic.

### Slack

To set up Slack, you must configure an incoming Slack webhook URL. For more information, see [Sending messages using Incoming Webhooks](#).

To include screenshots of the firing alerts in the Slack messages, you must configure either the external image destination in Grafana or a bot integration via Slack Apps. For more information about setting up a Slack bot integration, see [Follow Slack's guide to set up a bot integration](#). Use the token provided, which starts with "xoxb".

Setting	Description
Url	Slack incoming webhook URL, or eventually the <a href="#">chat.postMessage</a> Slack API endpoint.
Username	Set the user name for the bot's message.
Recipient	Use this to override the Slack recipient. You must provide either a channel Slack ID, a user Slack ID, a user name reference (@<user>, all lowercase, no whitespace), or a channel reference (#<channel>, all lowercase, no white space). If you use the <code>chat.postMessage</code> Slack API endpoint, this is required.
Icon emoji	Provide an emoji to use as the icon for the bot's message. For example, <code>:smile:</code>
Icon URL	Provide a URL to an image to use as the icon for the bot's message.
Mention Users	Optionally mention one or more users in the Slack notification sent by Grafana. To see users, comma-separated, via their corresponding Slack IDs, choose the overflow button on each user's Slack profile.
Mention Groups	Optionally mention one or more groups in the Slack notification sent by Grafana. You can see groups, comma-separated, via their corresponding Slack IDs (which you can get from each group's Slack profile URL).
Mention Channel	Optionally mention either all channel members or only active ones.
Token	If provided, Amazon Managed Grafana will upload the generated image via the Slack <code>file.upload</code> API operation, not the external image destination. If you use the <code>chat.postMessage</code> Slack API endpoint, this is required.

If you are using the token for a slack bot, you have to invite the bot to the channel that you want to send notifications. Then add the channel to the recipient field.

## PagerDuty

To set up PagerDuty, provide an integration key.

Setting	Description
Integration Key	Integration key for PagerDuty.
Severity	Level for dynamic notifications; default is <code>critical</code> (1).
Auto resolve incidents	Resolve incidents in PagerDuty after the alert goes back to ok.

Setting	Description
Message in details	Removes the Alert message from the PD summary field and puts it into custom details instead (2).

**Note**

The tags `Severity`, `Class`, `Group`, `dedup_key`, and `Component` have special meaning in the [PagerDuty Common Event Format – PD-CEF](#). If an alert panel defines these tag keys, they are transposed to the root of the event sent to PagerDuty. This means they will be available within the PagerDuty UI and Filtering tools. A `Severity` tag set on an alert overrides the global `Severity` set on the notification channel if it's a valid level.

**Note**

Using `Message In Details` will change the structure of the `custom_details` field in the PagerDuty Event. This might break custom event rules in your PagerDuty rules if you rely on the fields in `payload.custom_details`. Move any existing rules that use `custom_details.myMetric` to `custom_details.queries.myMetric`.

**Note**

Using `dedup_key` tag will override the Grafana generated `dedup_key` with a custom key.

## Configuring the link back to Grafana from alert notifications

All alert notifications contain a link back to the triggered alert in the Grafana workspace.

## Creating alerts

When you use Amazon Managed Grafana alerting, you can attach rules to your dashboard panels. When you save the dashboard, Amazon Managed Grafana extracts the alert rules into a separate alert rule storage and schedules them for evaluation.

On the **Alert** tab of the graph panel, you can configure how often the alert rule should be evaluated and the conditions that must be met for the alert to change state and initiate its notifications.

Currently, only the graph panel supports alert rules.

## Adding or editing an alert rule

1. Navigate to the panel where add or edit an alert rule, choose the title, and then choose **Edit**.
2. On the **Alert** tab, choose **Create Alert**. If an alert already exists for this panel, you can edit the fields on the **Alert** tab.
3. Fill out the fields. For more information, see [Alert rule fields \(p. 286\)](#).
4. When you have finished writing your rule, choose **Save** in the upper right corner to save the alert rule and the dashboard.
5. (Optional but recommended) To make sure that the rule returns the results you expect, choose **Test rule**.

## Deleting an alert rule

To delete an alert, scroll to the bottom of the alert, and then choose **Delete**.

## Alert rule fields

This section describes the fields that you fill out to create an alert.

## Rule

- **Name** – Enter a descriptive name. The name will be displayed in the **Alert Rules** list.
- **Evaluate every** – Specify how often the scheduler should evaluate the alert rule. This is referred to as the *evaluation interval*.
- **For** – Specify how long the query must violate the configured thresholds before the alert notification triggers.

### Warning

Do not use `For` with the `If no data or all values are null` setting set to `No Data`. The triggering of `No Data` will trigger instantly and not take `For` into consideration. This may also result in that an OK notification not being sent if alert transitions from `No Data` -> `Pending` -> `OK`.

If an alert rule has a configured `For` and the query violates the configured threshold, it will first go from `OK` to `Pending`. Going from `OK` to `Pending`, Amazon Managed Grafana does not send any notifications. When the alert rule has been firing for more than the `For` duration, it will change to `Alerting` and send alert notifications.

Typically, we recommend using this setting because it's often worse to get false positive than to wait a few minutes before the alert notification initiates. Looking at the `Alert list` or `Alert list` panels, you will be able to see alerts that are in the pending state.

## Conditions

Currently, the only existing condition type is a `query` condition that allows you to specify a query letter, a time range, and an aggregation function.

### Query condition example

```
avg() OF query(A, 15m, now) IS BELOW 14
```

- `avg()` Controls how the values for **each** series should be reduced to a value that can be compared against the threshold. Choose the function to change it to another aggregation function.
- `query(A, 15m, now)` The letter defines what query to run from the **Metrics** tab. The second two parameters define the time range: `15m`, `now` means 15 minutes ago to now. You can also use `10m`, `now-2m` to define a time range that will be 10 minutes ago to 2 minutes ago. This is useful if you want to ignore the last 2 minutes of data.
- `IS BELOW 14` Defines the type of threshold and the threshold value. You can choose `IS BELOW` to change the type of threshold.

The query used in an alert rule cannot contain any template variables. Currently, we support only `AND` and `OR` operators between conditions, and they are run serially. For example, we have three conditions in the following order: `condition:A(evaluates to: TRUE) OR condition:B(evaluates to: FALSE) AND condition:C(evaluates to: TRUE)` so the result will be calculated as `((TRUE OR FALSE) AND TRUE) = TRUE`.

### Multiple series

If a query returns multiple series, the aggregation function and threshold check will be evaluated for each series. Currently, Amazon Managed Grafana does not track the alert rule state **per series**. The implications of this are detailed in the following scenario.

- An alert condition with query that returns two series: **server1** and **server2**.

- The **server1** series causes the alert rule to fire and switch to state `Alerting`.
- Notifications are sent out with message: *load peaking (server1)*
- In a subsequent evaluation of the same alert rule, the **server2** series also causes the alert rule to fire.
- No new notifications are sent because the alert rule is already in state `Alerting`.

As you can see from the previous scenario, if the rule already is in state `Alerting`, Grafana doesn't send out notifications when other series cause the alert to fire.

#### Note

You can configure reminders to be sent for triggered alerts. This will send additional notifications when an alert continues to fire. If other series (such as `server2` in the previous example) also cause the alert rule to fire, they are included in the reminder notification. Depending on which notification channel you're using, you might be able to take advantage of this feature for identifying new or existing series that are causing alerts to fire.

## No data and error handling

The following table contains conditions for controlling how the rule evaluation engine handles queries that return no data or only null values.

No Data Option	Description
No Data	Set alert rule state to <code>NoData</code> .
Alerting	Set alert rule state to <code>Alerting</code> .
Keep Last State	Keep the current alert rule state, whatever it is.
Ok	Supported, but usually not useful.

## Execution errors or timeouts

The following options tell Amazon Managed Grafana how to handle execution or timeout errors.

Error or timeout option	Description
Alerting	Set alert rule state to <code>Alerting</code> .
Keep Last State	Keep the current alert rule state, whatever it is.

If you have an unreliable time series store from which queries sometimes time out or fail randomly, you can set this option to `Keep Last State` to basically ignore them.

## Notifications

On **Alert** tab, you can also specify alert rule notifications and a detailed message about the alert rule. The message can contain anything: information about how you might solve the issue, link to runbook, and so on.

The actual notifications are configured and shared between multiple alerts. For information on how to configure and set up notifications, see [Alert notifications \(p. 283\)](#).

- **Send to** – Select an alert notification channel if you have one set up.

- **Message** – Enter a text message to be sent on the notification channel. Some alert notifiers support transforming the text to HTML or other rich formats.
- **Tags** – Specify a list of tags (key-value) to be included in the notification. It is supported by only some notifiers.

## Alert state history and annotations

Alert state changes are recorded in the internal annotation table in the Amazon Managed Grafana database. The state changes are visualized as annotations in the graph panel of the alert rule. You can also go into the `State history` submenu on the **Alert** tab to view and clear state history.

## Pausing an alert rule

Pausing the evaluation of an alert rule can sometimes be useful. For example, during a maintenance window, pausing alert rules can avoid initiating a flood of alerts.

1. In the Grafana side bar, pause on the **Alerting** (bell) icon and then choose **Alert Rules**. All configured alert rules are listed, along with their current state.
2. Find your alert in the list, and choose the **Pause** icon on the right. The **Pause** icon turns into a **Play** icon.
3. Choose the **Play** icon to resume evaluation of your alert.

## Viewing existing alert rules

Amazon Managed Grafana stores individual alert rules in the panels where they are defined, but you can also view a list of all existing alert rules and their current state.

In the Grafana side bar, pause on the **Alerting** (bell) icon, and then choose **Alert Rules**. All configured alert rules are listed, along with their current state.

While viewing alerts, you can do the following:

- **Filter alerts by name** – Type an alert name in the **Search alerts** field.
- **Filter alerts by state** – In **States**, select which alert states you want to see. All others will be hidden.
- **Pause or resume an alert** – choose the **Pause** or **Play** icon next to the alert to pause or resume evaluation.
- **Access alert rule settings** – Choose the alert name or the **Edit alert rule** (gear) icon. Amazon Managed Grafana opens the **Alert** tab of the panel where the alert rule is defined. This is helpful when an alert is firing, but you don't know which panel it is defined in.

## Notification templating

The alert notification template feature allows you to take the label value from an alert query and inject that into alert notifications.

Labels that exist from the evaluation of the alert query can be used in the alert rule name and in the alert notification message fields. The alert label data is injected into the notification fields when the alert is in the alerting state. When there are multiple unique values for the same label, the values are comma-separated.

### To add alert label data into your alert notification

1. Navigate to the panel that you want to add or edit an alert rule for.



2. Choose the panel title and choose **Edit**.
3. On the **Alert** tab, choose **Create Alert**. If an alert already exists for this panel, you can edit it directly.
4. Refer to the alert query labels in the alert rule name or the alert notification message field by using the `${Label1}` syntax.
5. Choose **Save** in the upper right corner.

## Troubleshooting alerts

If alerts are not behaving as you expect, the following steps can help you troubleshoot and figure out what is going wrong.

The first level of troubleshooting that you can do is choose **Test Rule**. You can expand the result to the point where you can see the raw data that was returned from your query.

## Change your preferences

You can perform several tasks on the **Preferences** tab. You can edit your profile, change your Amazon Managed Grafana preferences, and view information about your profile and Amazon Managed Grafana usage.

### Edit your Amazon Managed Grafana profile

Your profile includes your name, user name, and email address.

#### To edit your profile

1. Pause on your user icon in the lower left corner of the screen, and then choose **Preferences**.
2. In the **Edit Profile** section, you can edit any of the following:
  - **Name** – Edit this field to change the display name associated with your profile.
  - **Email** – Edit this field to change the email address associated with your profile.
  - **Username** – Edit this field to change your user name.
3. Choose **Save**.

### Edit your preferences

Your preferences include whether uses the dark or light theme, your home dashboard, and your timezone.

#### Note

Settings on your personal instance override settings made by your administrator at the instance or team level.

#### To change your preferences

1. Pause on your user icon in the lower left corner of the screen, and then choose **Preferences**.
2. In the Preferences section, you can edit any of the following:
  - **UI Theme** – To set a theme, choose **Dark** or **Light**. **Default** is either the dark theme or the theme selected by your Grafana administrator.
  - **Home Dashboard**

- **Timezone** – Choose to select an option in the **Timezone** list. **Default** is either the browser local timezone or the timezone selected by your Grafana administrator. For more information, see [Time range controls \(p. 244\)](#)..
3. Choose **Save**.

## View your Amazon Managed Grafana sessions

Amazon Managed Grafana logs your sessions in each Grafana workspace. If you suspect someone has misused your Amazon Managed Grafana credentials you can review this section.

### To view your sessions information

1. Pause on your user icon in the lower left corner of the screen, and then choose **Preferences**.
2. Scroll down to the **Sessions** section. Grafana displays the following:
  - **Last seen** – How long ago you logged on.
  - **Logged on** – The date you logged on to the current Grafana instance.
  - **IP address** – The IP address that you logged on from.
  - **Browser & OS** – The web browser and operating system used to log on to Grafana.
  - If you are a Grafana admin for the instance, you can revoke a session by choosing the red signout icon in the session row.

# Using Grafana HTTP APIs

You can use Grafana HTTP APIs with Amazon Managed Grafana workspaces. The following sections list which Grafana APIs are supported.

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana *API key*, which is also called an *API token*. To create an API key, use the following procedure. An API key is valid for a limited time that you specify when you create it, up to 30 days.

When you create an API key, you specify a *role* for the key. The role determines the level of administrative power that users of the key have.

The following tables show the permissions granted to the Admin, Editor, and Viewer roles. The first table shows general organizational permissions. In this table, **Full** means the ability to view, edit, add permissions, and delete permissions. The **Explore** column shows whether the role can use the *Explore* view. The **Other** permissions column shows whether the role has permissions for managing users, teams, plug-ins, and organizational settings.

Role	Dashboards	Playlists	Folders	Explore	Data sources	Other permissions
<b>Viewer</b>	View	View	No	No	No	No
<b>Editor</b>	Full	Full	Full	Yes	No	No
<b>Admin</b>	Full	Full	Full	Yes	Full	Full

The following table shows the additional dashboard- and folder-level permissions that you can set. These are different than the Admin, Editor, and Viewer roles.

Role	Dashboards	Folders	Change permissions
<b>View</b>	View	View	No
<b>Edit</b>	Create, edit	View	No
<b>Admin</b>	Create, edit, delete	Create, edit, delete	Yes

## Note

A more scoped permission with a lower permission level does not have effect if a more general rule with more permission exists. For example, if you give a user the organizational **Editor** role and then assign that user only the **View** permissions for a dashboard, the more restrictive **View** permission has no effect because the user has full **Edit** access because of their **Editor** role.

## To create an API key to use with Grafana APIs

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the upper left corner of the page, choose the menu icon and then choose **All workspaces**.
3. Choose the name of the Amazon Managed Grafana workspace.
4. In the workspace details page, choose the URL displayed under **Grafana workspace URL**.
5. In the Grafana console side menu, pause on the **Configuration** (gear) icon, then choose **API Keys**.
6. Choose **New API Key**.

7. Enter a unique name for the key.
8. For **Role**, select the access level that the key is to be granted. Select **Admin** to allow a user with this key to use APIs at the broadest, most powerful administrative level. Select **Editor** or **Viewer** to limit the key's users to those levels of power. For more information, see the previous tables.
9. For **Time to live**, specify how long you want the key to be valid. The maximum is 30 days (one month). You enter a number and a letter. The valid letters are **s** for seconds, **m** for minutes, **h** for hours, **d** for days, **w** for weeks, and **M** for month. For example, **12h** is 12 hours and **1M** is 1 month (30 days).

We strongly recommend that you set the key's time to live for a shorter time, such as a few hours or less. This creates much less risk than having API keys that are valid for a long time.

10. Choose **Add**.

### Topics

- [Alerting API \(p. 293\)](#)
- [Alerting Notification Channels API \(p. 296\)](#)
- [Annotations API \(p. 302\)](#)
- [Authentication API \(p. 306\)](#)
- [Dashboard API \(p. 308\)](#)
- [Dashboard Permissions API \(p. 315\)](#)
- [Dashboard Versions API \(p. 317\)](#)
- [Data Source API \(p. 322\)](#)
- [Data Source Permissions API \(p. 332\)](#)
- [External Group Synchronization API \(p. 336\)](#)
- [Folder API \(p. 338\)](#)
- [Folder/Dashboard Search API \(p. 343\)](#)
- [Folder Permissions API \(p. 345\)](#)
- [Organization API \(p. 347\)](#)
- [Playlist API \(p. 351\)](#)
- [Preferences API \(p. 355\)](#)
- [Snapshot API \(p. 357\)](#)
- [Team API \(p. 361\)](#)
- [User API \(p. 367\)](#)

## Alerting API

Use the Preferences API to get information about legacy dashboard alerts and their states. However, you can't use this API to modify the alert. To create new alerts or modify them you need to update the dashboard JSON that contains the alerts.

### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Get alerts

```
GET /api/alerts
```

### Example request

```
GET /api/alerts HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

### Querystring parameters:

These parameters are used as querystring parameters. For example: `/api/alerts?dashboardId=1`

- **dashboardId**— Limit the responses to alerts in the specified dashboards value. You can specify multiple dashboards. For example, `dashboardId=23&dashboardId=35`
- **panelId**— Limit the response to alert for a specified panel on a dashboard.
- **query**— Limit the response to alerts having a name like this value.
- **state**— Return the alerts that have one or more of the following alert states: ALL, alerting, ok, no\_data, paused, or pending. To specify multiple states, use the following format: `?state=paused&state=alerting`
- **limit**— Limit the response to X number of alerts.
- **folderId**— Limit the response to alerts of dashboards in the specified folders. You can specify multiple folders. For example, `folderId=23&folderId=35`
- **dashboardQuery**— Limit the responses to alerts having a dashboard name like this value.
- **dashboardTag**— Limit the response alerts of dashboards with specified tags. To do "AND" filtering with multiple tags, specify the tags parameter multiple times. For example, `dashboardTag=tag1&dashboardTag=tag2`. Note that these are Grafana tags, not AWS tags.

### Example response

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "id": 1,
    "dashboardId": 1,
    "dashboardUid": "ABcdEFghij",
    "dashboardSlug": "sensors",
    "panelId": 1,
    "name": "fire place sensor",
    "state": "alerting",
    "newStateDate": "2018-05-14T05:55:20+02:00",
    "evalDate": "0001-01-01T00:00:00Z",
    "evalData": null,
    "executionError": "",
    "url": "http://grafana.com/dashboard/db/sensors"
  }
]
```

## Get alert by Id

```
GET /api/alerts/:id
```

### Example request

```
GET /api/alerts/1 HTTP/1.1
Accept: application/json
```

```
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "id": 1,
  "dashboardId": 1,
  "dashboardUID": "ABcdEFghij",
  "dashboardSlug": "sensors",
  "panelId": 1,
  "name": "fire place sensor",
  "state": "alerting",
  "message": "Someone is trying to break in through the fire place",
  "newStateDate": "2018-05-14T05:55:20+02:00",
  "evalDate": "0001-01-01T00:00:00Z",
  "evalData": "evalMatches": [
    {
      "metric": "movement",
      "tags": {
        "name": "fireplace_chimney"
      },
      "value": 98.765
    }
  ],
  "executionError": "",
  "url": "http://grafana.com/dashboard/db/sensors"
}
```

### Important

evalMatches data is cached in the database when and only when the state of the alert changes. If data from one server triggers the alert first and, before that server is seen leaving the alerting state, a second server also enters a state that would trigger the alert, the second server is not visible in the evalMatches data.

## Pause alert by Id

```
POST /api/alerts/:id/pause
```

### Example request

```
POST /api/alerts/1/pause HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdE9ZWmNrMkZYbk

{
  "paused": true
}
```

The `:id` query parameter is the Id of the alert to be paused or unpaused. `paused` can be `true` to pause an alert or `false` to unpause the alert.

### Example response

```
HTTP/1.1 200
```

```
Content-Type: application/json
```

```
{
  "alertId": 1,
  "state": "Paused",
  "message": "alert paused"
}
```

## Alerting Notification Channels API

Use the Alerting Notification Channels API to create, update, delete, and retrieve notification channels.

The identifier (id) of a notification channel is an auto-incrementing numeric value and is only unique per workspace. The unique identifier (uid) of a notification channel can be used to uniquely identify a folder between multiple workspaces. It's automatically generated if you don't provide one when you create a notification channel. The uid allows having consistent URLs for accessing notification channels and when synchronizing notification channels between multiple Amazon Managed Grafana workspaces.

### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Get all notification channels

Returns all notification channels that the authenticated user has permission to view.

```
GET /api/alert-notifications
```

### Example request

```
GET /api/alert-notifications HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVadFsNFZXdeE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "id": 1,
    "uid": "team-a-email-notifier",
    "name": "Team A",
    "type": "email",
    "isDefault": false,
    "sendReminder": false,
    "disableResolveMessage": false,
    "settings": {
      "addresses": "dev@grafana.com"
    },
    "created": "2018-04-23T14:44:09+02:00",
    "updated": "2018-08-20T15:47:49+02:00"
  }
]
```

```
] ]
```

## Get all notification channels (lookup)

Returns all notification channels, but with less detailed information. Accessible by any authenticated user and is mainly used to provide alert notification channels in the Grafana workspace console UI when configuring alert rules.

```
GET /api/alert-notifications/lookup
```

### Example request

```
GET /api/alert-notifications/lookup HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "id": 1,
    "uid": "0000000001",
    "name": "Test",
    "type": "email",
    "isDefault": false
  },
  {
    "id": 2,
    "uid": "0000000002",
    "name": "Slack",
    "type": "slack",
    "isDefault": false
  }
]
```

## Get all notification channels by UID

```
GET /api/alert-notifications/uid/:uid
```

### Example request

```
GET /api/alert-notifications/uid/team-a-email-notifier HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json
```



```
{
  "id": 1,
  "uid": "team-a-email-notifier",
  "name": "Team A",
  "type": "email",
  "isDefault": false,
  "sendReminder": false,
  "disableResolveMessage": false,
  "settings": {
    "addresses": "dev@grafana.com"
  },
  "created": "2018-04-23T14:44:09+02:00",
  "updated": "2018-08-20T15:47:49+02:00"
}
```

## Get all notification channels by Id

```
GET /api/alert-notifications/:id
```

### Example request

```
GET /api/alert-notifications/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVadFsNFZXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "id": 1,
  "uid": "team-a-email-notifier",
  "name": "Team A",
  "type": "email",
  "isDefault": false,
  "sendReminder": false,
  "disableResolveMessage": false,
  "settings": {
    "addresses": "dev@grafana.com"
  },
  "created": "2018-04-23T14:44:09+02:00",
  "updated": "2018-08-20T15:47:49+02:00"
}
```

## Create notification channel

To see what notification channels are supported by Amazon Managed Grafana, see [List of supported notifiers \(p. 284\)](#).

```
POST /api/alert-notifications
```

### Example request

```
POST /api/alert-notifications HTTP/1.1
Accept: application/json
```

```
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdE9ZWmNrMkZYbk

{
  "uid": "new-alert-notification", // optional
  "name": "new alert notification", //Required
  "type": "email", //Required
  "isDefault": false,
  "sendReminder": false,
  "settings": {
    "addresses": "dev@grafana.com"
  }
}
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "id": 1,
  "uid": "new-alert-notification",
  "name": "new alert notification",
  "type": "email",
  "isDefault": false,
  "sendReminder": false,
  "settings": {
    "addresses": "dev@grafana.com"
  },
  "created": "2018-04-23T14:44:09+02:00",
  "updated": "2018-08-20T15:47:49+02:00"
}
```

## Update notification channel by UID

```
PUT /api/alert-notifications/uid/:uid
```

#### Example request

```
PUT /api/alert-notifications/uid/cIBgcSjkk HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdE9ZWmNrMkZYbk

{
  "uid": "new-alert-notification", // optional
  "name": "new alert notification", //Required
  "type": "email", //Required
  "isDefault": false,
  "sendReminder": true,
  "frequency": "15m",
  "settings": {
    "addresses": "dev@grafana.com"
  }
}
```

#### Example response

```
HTTP/1.1 200
```

```
Content-Type: application/json
```

```
{
  "id": 1,
  "uid": "new-alert-notification",
  "name": "new alert notification",
  "type": "email",
  "isDefault": false,
  "sendReminder": true,
  "frequency": "15m",
  "settings": {
    "addresses": "dev@grafana.com"
  },
  "created": "2017-01-01 12:34",
  "updated": "2017-01-01 12:34"
}
```

## Update notification channel by Id

```
PUT /api/alert-notifications/:id
```

### Example request

```
PUT /api/alert-notifications/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVadFsNFZXXde9ZWmNrMkZYbk
```

```
{
  "id": 1,
  "uid": "new-alert-notification", // optional
  "name": "new alert notification", //Required
  "type": "email", //Required
  "isDefault": false,
  "sendReminder": true,
  "frequency": "15m",
  "settings": {
    "addresses": "dev@grafana.com"
  }
}
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "id": 1,
  "uid": "new-alert-notification",
  "name": "new alert notification",
  "type": "email",
  "isDefault": false,
  "sendReminder": true,
  "frequency": "15m",
  "settings": {
    "addresses": "dev@grafana.com"
  },
  "created": "2017-01-01 12:34",
  "updated": "2017-01-01 12:34"
}
```

## Delete notification channel by UID

```
DELETE /api/alert-notifications/uid/:uid
```

### Example request

```
DELETE /api/alert-notifications/uid/team-a-email-notifier HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "message": "Notification deleted"
}
```

## Delete notification channel by Id

```
DELETE /api/alert-notifications/:id
```

### Example request

```
DELETE /api/alert-notifications/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "message": "Notification deleted"
}
```

## Test notification channel

Sends a test notification message for the given notification channel type and settings.

```
POST /api/alert-notifications/test
```

### Example request

```
POST /api/alert-notifications/test HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdE9ZWmNrMkZYbk
```

```
{
  "type": "email",
  "settings": {
    "addresses": "dev@grafana.com"
  }
}
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "message": "Test notification sent"
}
```

## Annotations API

Use the Annotations API to create, update, delete, and work with annotations in the Amazon Managed Grafana workspace.

Annotations are saved in the workspace's Grafana database (sqlite, mysql or postgres). Annotations can be global annotations that can be shown on any dashboard by configuring an annotation data source. Annotations are filtered by tags. Or they can be tied to a panel on a dashboard, and displayed only on that panel.

#### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Find annotations

```
GET /api/annotations?from=1506676478816&to=1507281278816&tags=tag1&tags=tag2&limit=100
```

#### Example request

```
GET /api/annotations?from=1506676478816&to=1507281278816&tags=tag1&tags=tag2&limit=100
HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

Query parameters:

- **from**— (Optional) Epoch datetime in milliseconds.
- **to**— (Optional) Epoch datetime in milliseconds.
- **limit**— (Optional) Maximum number of results returned. The default is 100.
- **alertid**— (Optional) Find annotations for the specified alert.
- **dashboardId**— (Optional) Find annotations that are scoped to the specified dashboard.
- **panelId**— (Optional) Find annotations that are scoped to the specified panel.
- **userId**— (Optional) Find annotations created by the specified user.

- **type**— (Optional) Specify to return alerts or user-created annotations. Value values are `alert` and `annotation`.
- **tags**— (Optional) Use this to filter global annotations. Global annotations are annotations from an annotation data source that are not connected specifically to a dashboard or panel. To do an “AND” filtering with multiple tags, specify the `tags` parameter multiple times. For example, `tags=tag1&tags=tag2`. These are Grafana tags, not AWS tags.

### Example response

```
HTTP/1.1 200
Content-Type: application/json
[
  {
    "id": 1124,
    "alertId": 0,
    "dashboardId": 468,
    "panelId": 2,
    "userId": 1,
    "userName": "",
    "newState": "",
    "prevState": "",
    "time": 1507266395000,
    "timeEnd": 1507266395000,
    "text": "test",
    "metric": "",
    "tags": [
      "tag1",
      "tag2"
    ],
    "data": {}
  },
  {
    "id": 1123,
    "alertId": 0,
    "dashboardId": 468,
    "panelId": 2,
    "userId": 1,
    "userName": "",
    "newState": "",
    "prevState": "",
    "time": 1507265111000,
    "text": "test",
    "metric": "",
    "tags": [
      "tag1",
      "tag2"
    ],
    "data": {}
  }
]
```

## Create annotation

```
POST /api/annotations
```

Creates an annotation in the workspace's Grafana database. The `dashboardId` and `panelId` fields are optional. If they are not specified, a global annotation is created and can be queried in any dashboard that adds the Grafana annotations data source. When creating a region annotation, be sure to include the `timeEnd` property.

The format for `time` and `timeEnd` should be epoch numbers in millisecond resolution.

#### Example request

```
POST /api/annotations HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVadFsNFZXXdeE9ZWmNrMkZYbk

{
  "dashboardId":468,
  "panelId":1,
  "time":1507037197339,
  "timeEnd":1507180805056,
  "tags":["tag1","tag2"],
  "text":"Annotation Description"
}
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "message":"Annotation added",
  "id": 1,
}
```

## Create annotation in graphite format

```
POST /api/annotations/graphite
```

Creates an annotation by using a Graphite-compatible event format. The `when` and `data` fields are optional. If `when` is not specified, the current time is used as annotation's timestamp. The `tags` field can also be in prior to Graphite 0.10.0 format (string with multiple tags being separated by a space).

#### Example request

```
POST /api/annotations/graphite HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVadFsNFZXXdeE9ZWmNrMkZYbk

{
  "what": "Event - deploy",
  "tags": ["deploy", "production"],
  "when": 1467844481,
  "data": "deploy of master branch happened at Wed Jul 6 22:34:41 UTC 2016"
}
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "message":"Graphite annotation added",
  "id": 1
}
```

## Update annotation

```
PUT /api/annotations/:id
```

Updates all properties of an annotation that matches the specified id. To only update certain properties, use the Patch Annotation operation.

### Example request

```
PUT /api/annotations/1141 HTTP/1.1
Accept: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
Content-Type: application/json

{
  "time":1507037197339,
  "timeEnd":1507180805056,
  "text":"Annotation Description",
  "tags":["tag3","tag4","tag5"]
}
```

### Example response:

```
HTTP/1.1 200
Content-Type: application/json

{
  "message":"Annotation updated"
}
```

## Patch annotation

```
PATCH /api/annotations/:id
```

Updates one or more properties of an annotation that matches the specified id. This operation currently supports updating the text, tags, time, and timeEnd properties.

### Example request:

```
PATCH /api/annotations/1145 HTTP/1.1
Accept: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
Content-Type: application/json

{
  "text":"New Annotation Description",
  "tags":["tag6","tag7","tag8"]
}
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "message":"Annotation patched"
}
```



```
}
```

## Delete annotation by Id

```
DELETE /api/annotations/:id
```

Deletes the annotation that matches the specified Id.

### Example request

```
DELETE /api/annotations/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcGlpU1Y2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "message": "Annotation deleted"
}
```

## Authentication API

Use the Authentication API to work with authentication keys in an Amazon Managed Grafana workspace.

### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Get API keys

```
GET /api/auth/keys
```

### Example request

```
GET /api/auth/keys HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcGlpU1Y2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

### Query parameter:

- **includeExpired**— (Optional) Boolean parameter that specifies whether to include expired keys in the returned results. The default is `false`.

### Example response

```
HTTP/1.1 200
```

```
Content-Type: application/json
```

```
[
  {
    "id": 3,
    "name": "API",
    "role": "Admin"
  },
  {
    "id": 1,
    "name": "TestAdmin",
    "role": "Admin",
    "expiration": "2019-06-26T10:52:03+03:00"
  }
]
```

## Create API key

```
POST /api/auth/keys
```

### Example request

```
POST /api/auth/keys HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk

{
  "name": "mykey",
  "role": "Admin",
  "secondsToLive": 86400
}
```

JSON body schema:

- **name**— The name for the key.
- **role**— Sets the access level (Grafana role) for the key. Valid values are `Admin`, `Editor`, or `Viewer`.
- **secondsToLive**— Sets the amount of time before the key expires. It must be 2592000 (30 days) or less.

### Example response

```
yJrIjoiWHZiSWd3NzdCYUZnNUtibE9obUpESmE3bzJYNDRic0UiLCJuIjoibXlrZXkiLCJpZCI6MX1=, "id": 1}
```

Error statuses:

- **400**— `secondsToLive` is greater than 2592000
- **500**— The key couldn't be stored in the database.

## Delete API key

```
DELETE /api/auth/keys/:id
```

### Example request

```
DELETE /api/auth/keys/3 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
```

```
Content-Type: application/json  
{ "message": "API key deleted" }
```

## Dashboard API

Use the Dashboard API to create, update, delete, and work with dashboards in the Amazon Managed Grafana workspace.

The identifier (id) of a dashboard is an auto-incrementing numeric value and is only unique per workspace. The unique identifier (uid) of a dashboard can be used to uniquely identify a dashboard between multiple Amazon Managed Grafana workspaces. It's automatically generated if you don't provide one when you create a dashboard. The uid allows having consistent URLs for accessing dashboards and when synchronizing dashboards between multiple workspaces. The use of the uid means that changing the title of a dashboard does not break any bookmarked links to that dashboard.

The uid can have a maximum length of 40 characters.

### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Create/Update dashboard

```
POST /api/dashboards/db
```

Creates a new dashboard or updates an existing dashboard.

### Example request to create a new dashboard

```
POST /api/dashboards/db HTTP/1.1  
Accept: application/json  
Content-Type: application/json  
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdE9ZWmNrMkZYbk  
  
{  
  "dashboard": {  
    "id": null,  
    "uid": null,  
    "title": "Production Overview",  
    "tags": [ "templated" ],  
    "timezone": "browser",  
    "schemaVersion": 16,  
    "version": 0,  
    "refresh": "25s"  
  },  
  "folderId": 0,  
  "folderUid": "l3KqBxCMz",  
  "message": "Made changes to xyz",  
  "overwrite": false  
}
```

JSON body schema:

- **dashboard**— The complete dashboard model. Use null to create a new dashboard.
- **dashboard.id**— Use null to create a new dashboard.

- **dashboard.uid**— Optional unique identifier when you use this to create a new dashboard. If null, a new uid is generated.
- **folderid**— The id of the folder to save the dashboard in.
- **folderUid**— The Uid of the folder to save the dashboard in. Overrides the value of `folderid`
- **overwrite**— Specify `true` to overwrite an existing dashboard with a newer version, same dashboard title in folder or same dashboard uid.
- **message**— Set a commit message for the version history.
- **refresh**— Set the dashboard refresh interval. If this is lower than the minimum refresh interval, it is ignored and the minimum refresh interval is used.

To add or update an alert rule for a dashboard panel, declare a `dashboard.panels.alert` block.

#### Example request to update a dashboard alert rule

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 78

{
  "dashboard": {
    "id": 104,
    "panels": [
      {
        "alert": {
          "alertRuleTags": {},
          "conditions": [
            {
              "evaluator": {
                "params": [
                  25
                ],
                "type": "gt"
              },
              "operator": {
                "type": "and"
              },
              "query": {
                "params": [
                  "A",
                  "5m",
                  "now"
                ]
              },
              "reducer": {
                "params": [],
                "type": "avg"
              },
              "type": "query"
            }
          ],
          "executionErrorState": "alerting",
          "for": "5m",
          "frequency": "1m",
          "handler": 1,
          "name": "Panel Title alert",
          "noDataState": "no_data",
          "notifications": []
        },
        "aliasColors": {},
        "bars": false,
        "dashLength": 10,
```

```

"dashes": false,
"datasource": null,
"fieldConfig": {
  "defaults": {
    "custom": {}
  },
  "overrides": []
},
"fill": 1,
"fillGradient": 0,
"gridPos": {
  "h": 9,
  "w": 12,
  "x": 0,
  "y": 0
},
"hiddenSeries": false,
"id": 2,
"legend": {
  "avg": false,
  "current": false,
  "max": false,
  "min": false,
  "show": true,
  "total": false,
  "values": false
},
"lines": true,
"linewidth": 1,
"nullPointMode": "null",
"options": {
  "dataLinks": []
},
"percentage": false,
"pointRadius": 2,
"points": false,
"renderer": "flot",
"seriesOverrides": [],
"spaceLength": 10,
"stack": false,
"steppedLine": false,
"targets": [
  {
    "refId": "A",
    "scenarioId": "random_walk"
  }
],
"thresholds": [
  {
    "colorMode": "critical",
    "fill": true,
    "line": true,
    "op": "gt",
    "value": 50
  }
],
"timeFrom": null,
"timeRegions": [],
"timeShift": null,
"title": "Panel Title",
"tooltip": {
  "shared": true,
  "sort": 0,
  "value_type": "individual"
},
"type": "graph",

```

```

        "xaxis": {
            "buckets": null,
            "mode": "time",
            "name": null,
            "show": true,
            "values": []
        },
        "yaxes": [
            {
                "format": "short",
                "label": null,
                "logBase": 1,
                "max": null,
                "min": null,
                "show": true
            },
            {
                "format": "short",
                "label": null,
                "logBase": 1,
                "max": null,
                "min": null,
                "show": true
            }
        ],
        "yaxis": {
            "align": false,
            "alignLevel": null
        }
    },
    "title": "Update alert rule via API",
    "uid": "dHEquNzGz",
    "version": 1
}

```

### Example response

```

HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 78

{
  "id": 1,
  "uid": "cIBgcSjkk",
  "url": "/d/cIBgcSjkk/production-overview",
  "status": "success",
  "version": 1,
  "slug": "production-overview" //deprecated in Grafana v5.0
}

```

### Status Codes:

- **200**— Created
- **400**— Error such as invalid JSON, invalid or missing fields
- **401**— Unauthorized
- **403**— Access denied
- **412**— Precondition failed

The **412** status code is used to explain why the dashboard can't be created.

- The dashboard has been changed by someone else `status=version-mismatch`
- A dashboard with the same name in the folder already exists `status=name-exists`
- A dashboard with the same uid already exists `status=name-exists`
- The dashboard belongs to plugin `plugin title status=plugin-dashboard`

The response body has the following properties. If another dashboard has the same title, the `status` value will be `name-exists`.

```
HTTP/1.1 412 Precondition Failed
Content-Type: application/json; charset=UTF-8
Content-Length: 97

{
  "message": "The dashboard has been changed by someone else",
  "status": "version-mismatch"
}
```

## Get dashboard by uid

```
GET /api/dashboards/uid/:uid
```

Returns the dashboard matching the uid. The metadata returned might contain information about the UID of the folder that contains the dashboard.

### Example request

```
GET /api/dashboards/uid/cIBgcSjkk HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoitOtcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "dashboard": {
    "id": 1,
    "uid": "cIBgcSjkk",
    "title": "Production Overview",
    "tags": [ "templated" ],
    "timezone": "browser",
    "schemaVersion": 16,
    "version": 0
  },
  "meta": {
    "isStarred": false,
    "url": "/d/cIBgcSjkk/production-overview",
    "folderId": 2,
    "folderUid": "l3KqBxCMz",
    "slug": "production-overview" //deprecated in Grafana v5.0
  }
}
```

Status Codes:

- **200**— Found
- **401**— Unauthorized
- **403**— Access denied
- **404**— Not found

## Delete dashboard by uid

```
DELETE /api/dashboards/uid/:uid
```

Deletes the dashboard matching the uid.

### Example request

```
DELETE /api/dashboards/uid/cIBgcSjkk HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdeE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "title": "Production Overview",
  "message": "Dashboard Production Overview deleted",
  "id": 2
}
```

Status Codes:

- **200**— Deleted
- **401**— Unauthorized
- **403**— Access denied
- **404**— Not found

## Gets the home dashboard

```
GET /api/dashboards/home
```

Returns the home dashboard.

### Example request

```
GET /api/dashboards/home HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdeE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
```



```
Content-Type: application/json

{
  "dashboard": {
    "editable": false,
    "hideControls": true,
    "nav": [
      {
        "enable": false,
        "type": "timepicker"
      }
    ],
    "style": "dark",
    "tags": [],
    "templating": {
      "list": [
      ]
    },
    "time": {
    },
    "timezone": "browser",
    "title": "Home",
    "version": 5
  },
  "meta": {
    "isHome": true,
    "canSave": false,
    "canEdit": false,
    "canStar": false,
    "url": "",
    "expires": "0001-01-01T00:00:00Z",
    "created": "0001-01-01T00:00:00Z"
  }
}
```

## Get dashboard tags

```
GET /api/dashboards/tags
```

Returns all tags of dashboards.

### Example request

```
GET /api/dashboards/tags HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "term": "tag1",
    "count": 1
  },
  {
    "term": "tag2",
    "count": 4
  }
]
```

```
}  
]
```

## Dashboard Permissions API

Use the Dashboard Permissions API to update or retrieve the permissions for a dashboard.

Permissions with `dashboardId=-1` are the default permissions for users with the Viewer and Editor roles. Permissions can be set for a user, a team or a role (Viewer or Editor). Permissions cannot be set for Admins - they always have access to everything.

The permission levels for the `permission` field are as follows:

- 1 = View
- 2 = Edit
- 4 = Admin

### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Get permissions for a dashboard

```
GET /api/dashboards/id/:dashboardId/permissions
```

Gets all existing permissions for the dashboard with the given `dashboardId`.

### Example request

```
GET /api/dashboards/id/1/permissions HTTP/1.1  
Accept: application/json  
Content-Type: application/json  
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200 OK  
Content-Type: application/json; charset=UTF-8  
Content-Length: 551  
  
[  
  {  
    "id": 1,  
    "dashboardId": -1,  
    "created": "2017-06-20T02:00:00+02:00",  
    "updated": "2017-06-20T02:00:00+02:00",  
    "userId": 0,  
    "userLogin": "",  
    "userEmail": "",  
    "teamId": 0,  
    "team": "",  
    "role": "Viewer",  
    "permission": 1,  
  },  
]
```

```
    "permissionName": "View",
    "uid": "",
    "title": "",
    "slug": "",
    "isFolder": false,
    "url": ""
  },
  {
    "id": 2,
    "dashboardId": -1,
    "created": "2017-06-20T02:00:00+02:00",
    "updated": "2017-06-20T02:00:00+02:00",
    "userId": 0,
    "userLogin": "",
    "userEmail": "",
    "teamId": 0,
    "team": "",
    "role": "Editor",
    "permission": 2,
    "permissionName": "Edit",
    "uid": "",
    "title": "",
    "slug": "",
    "isFolder": false,
    "url": ""
  }
]
```

Status Codes:

- **200**— OK
- **401**— Unauthorized
- **403**— Access denied
- **404**— Dashboard not found

## Update permissions for a dashboard

```
POST /api/dashboards/id/:dashboardId/permissions
```

Updates the permissions for a dashboard. This operation removes existing permissions if they are not included in the request.

### Example request

```
POST /api/dashboards/id/1/permissions
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVadFsNFZXdE9ZWmNrMkZYbk

{
  "items": [
    {
      "role": "Viewer",
      "permission": 1
    },
    {
      "role": "Editor",
      "permission": 2
    }
  ],
}
```

```
{
  {
    "teamId": 1,
    "permission": 1
  },
  {
    "userId": 11,
    "permission": 4
  }
}
```

JSON body schema:

- **items**— The permission items to add or update. Existing items that are omitted from the list will be removed.

#### Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 35

{"message": "Dashboard permissions updated"}
```

Status Codes:

- **200**— Created
- **401**— Unauthorized
- **403**— Access denied
- **404**— Dashboard not found

## Dashboard Versions API

Use the Dashboard Versions API to retrieve dashboard versions and restore a dashboard to a specified version.

#### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Get all dashboard versions

```
GET /api/dashboards/id/:dashboardId/versions
```

Gets all existing dashboard versions for the dashboard with the given `dashboardId`.

Query parameters:

- **limit**— Maximum number of results to return.
- **start**— Version to start from when returning queries.

#### Example request

```
GET /api/dashboards/id/1/versions?limit=2?start=0 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 428

[
  {
    "id": 2,
    "dashboardId": 1,
    "parentVersion": 1,
    "restoredFrom": 0,
    "version": 2,
    "created": "2017-06-08T17:24:33-04:00",
    "createdBy": "admin",
    "message": "Updated panel title"
  },
  {
    "id": 1,
    "dashboardId": 1,
    "parentVersion": 0,
    "restoredFrom": 0,
    "version": 1,
    "created": "2017-06-08T17:23:33-04:00",
    "createdBy": "admin",
    "message": "Initial save"
  }
]
```

### Status Codes:

- **200**— OK
- **400**— Errors
- **401**— Unauthorized
- **404**— Dashboard version not found

## Get dashboard version

```
GET /api/dashboards/id/:dashboardId/versions/:id
```

Get the dashboard version with the given id, for the dashboard with the given dashboardId.

### Example request

```
GET /api/dashboards/id/1/versions/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200 OK
```

Content-Type: application/json; charset=UTF-8  
Content-Length: 1300

```
{
  "id": 1,
  "dashboardId": 1,
  "parentVersion": 0,
  "restoredFrom": 0,
  "version": 1,
  "created": "2017-04-26T17:18:38-04:00",
  "message": "Initial save",
  "data": {
    "annotations": {
      "list": [

    ]
    },
    "editable": true,
    "gnetId": null,
    "graphTooltip": 0,
    "hideControls": false,
    "id": 1,
    "links": [

  ],
  "rows": [
    {
      "collapse": false,
      "height": "250px",
      "panels": [

    ],
      "repeat": null,
      "repeatIteration": null,
      "repeatRowId": null,
      "showTitle": false,
      "title": "Dashboard Row",
      "titleSize": "h6"
    }
  ],
  "schemaVersion": 14,
  "style": "dark",
  "tags": [

  ],
  "templating": {
    "list": [

  ]
  },
  "time": {
    "from": "now-6h",
    "to": "now"
  },
  "timepicker": {
    "refresh_intervals": [
      "5s",
      "10s",
      "30s",
      "1m",
      "5m",
      "15m",
      "30m",
      "1h",
      "2h",
      "1d"
    ]
  }
}
```

```
    ],
    "time_options": [
      "5m",
      "15m",
      "1h",
      "6h",
      "12h",
      "24h",
      "2d",
      "7d",
      "30d"
    ]
  },
  "timezone": "browser",
  "title": "test",
  "version": 1
},
"createdBy": "admin"
}
```

Status Codes:

- **200**— OK
- **401**— Unauthorized
- **404**— Dashboard version not found

## Restore dashboard

```
POST /api/dashboards/id/:dashboardId/restore
```

Restores a dashboard to the dashboard version that you specify.

### Example request

```
POST /api/dashboards/id/1/restore
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk

{
  "version": 1
}
```

JSON body schema:

- **version**— The dashboard version to restore to.

### Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 67

{
  "slug": "my-dashboard",
  "status": "success",
  "version": 3
}
```

```
}

```

JSON response body schema:

- **slug**— The URL-friendly slug of the dashboard's title.
- **status**— Whether the restore was successful or not.
- **version**— The new dashboard version following the restore.

Status Codes:

- **200**— Created
- **401**— Unauthorized
- **404**— Dashboard or dashboard version not found
- **500**— Internal server error (indicates issue retrieving dashboard tags from database)

Example error response:

```
HTTP/1.1 404 Not Found
Content-Type: application/json; charset=UTF-8
Content-Length: 46

{
  "message": "Dashboard version not found"
}
```

JSON response body schema:

- **message**— A message explaining the reason for the failure.

## Compare dashboard versions

```
POST /api/dashboards/calculate-diff

```

Compares two dashboard versions by calculating the JSON diff of them.

**Example request**

```
POST /api/dashboards/calculate-diff HTTP/1.1
Accept: text/html
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZlXde9ZWmNrMkZYbk

{
  "base": {
    "dashboardId": 1,
    "version": 1
  },
  "new": {
    "dashboardId": 1,
    "version": 2
  },
  "diffType": "json"
}
```



JSON body schema:

- **base**— An object representing the base dashboard version.
- **new**— An object representing the new dashboard version.
- **difftype**— The type of diff to return. Valid values are `json` and `basic`.

Example response (JSON diff)

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8

<p id="l1" class="diff-line diff-json-same">
  <!-- Diff omitted -->
</p>
```

The response is a textual representation of the diff, with the dashboard values being in JSON, similar to the diffs seen on sites like GitHub or GitLab.

Status Codes:

- **200**— OK
- **200**— Bad request, invalid JSON sent
- **401**— Unauthorized
- **404**— Not found

Example response (Basic diff)

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8

<div class="diff-group">
  <!-- Diff omitted -->
</div>
```

The response is a summary of the changes, derived from the diff between the two JSON objects.

Status Codes:

- **200**— OK
- **200**— Bad request, invalid JSON sent
- **401**— Unauthorized
- **404**— Not found

## Data Source API

Use the Data Source API to create, update, delete, and work with data sources in the Amazon Managed Grafana workspace.

### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Get all data sources

```
GET /api/datasources
```

### Example request

```
GET /api/datasources HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "id": 1,
    "orgId": 1,
    "uid": "H8joYFVGz",
    "name": "datasource_elastic",
    "type": "elasticsearch",
    "typeLogoUrl": "public/app/plugins/datasource/elasticsearch/img/elasticsearch.svg",
    "access": "proxy",
    "url": "http://mydatasource.com",
    "password": "",
    "user": "",
    "database": "grafana-dash",
    "basicAuth": false,
    "isDefault": false,
    "jsonData": {
      "esVersion": 5,
      "logLevelField": "",
      "logMessageField": "",
      "maxConcurrentShardRequests": 256,
      "timeField": "@timestamp"
    },
    "readOnly": false
  }
]
```

## Get a single data source by Id

```
GET /api/datasources/:datasourceId
```

### Example request

```
GET /api/datasources/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json
```

```
{
  "id": 1,
  "uid": "kLtEtcRGk",
  "orgId": 1,
  "name": "test_datasource",
  "type": "graphite",
  "typeLogoUrl": "",
  "access": "proxy",
  "url": "http://mydatasource.com",
  "password": "",
  "user": "",
  "database": "",
  "basicAuth": false,
  "basicAuthUser": "",
  "basicAuthPassword": "",
  "withCredentials": false,
  "isDefault": false,
  "jsonData": {
    "graphiteType": "default",
    "graphiteVersion": "1.1"
  },
  "secureJsonFields": {},
  "version": 1,
  "readOnly": false
}
```

## Get a single data source by UID

```
GET /api/datasources/uid/:uid
```

### Example request

```
GET /api/datasources/uid/kLtEtcRGk HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "id": 1,
  "uid": "kLtEtcRGk",
  "orgId": 1,
  "name": "test_datasource",
  "type": "graphite",
  "typeLogoUrl": "",
  "access": "proxy",
  "url": "http://mydatasource.com",
  "password": "",
  "user": "",
  "database": "",
  "basicAuth": false,
  "basicAuthUser": "",
  "basicAuthPassword": "",
  "withCredentials": false,
  "isDefault": false,
  "jsonData": {
```

```
{
  "graphiteType": "default",
  "graphiteVersion": "1.1"
},
"secureJsonFields": {},
"version": 1,
"readOnly": false
}
```

## Get a single data source by name

```
GET /api/datasources/name/:name
```

### Example request

```
GET /api/datasources/name/test_datasource HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "id": 1,
  "uid": "kLtEtCRGk",
  "orgId": 1,
  "name": "test_datasource",
  "type": "graphite",
  "typeLogoUrl": "",
  "access": "proxy",
  "url": "http://mydatasource.com",
  "password": "",
  "user": "",
  "database": "",
  "basicAuth": false,
  "basicAuthUser": "",
  "basicAuthPassword": "",
  "withCredentials": false,
  "isDefault": false,
  "jsonData": {
    "graphiteType": "default",
    "graphiteVersion": "1.1"
  },
  "secureJsonFields": {},
  "version": 1,
  "readOnly": false
}
```

## Get data source Id by name

```
GET /api/datasources/id/:name
```

### Example request

```
GET /api/datasources/id/test_datasource HTTP/1.1
```

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVadFsNFZXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "id":1
}
```

## Create a data source

```
POST /api/datasources
```

### Example Graphite request

```
POST /api/datasources HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVadFsNFZXde9ZWmNrMkZYbk

{
  "name": "test_datasource",
  "type": "graphite",
  "url": "http://mydatasource.com",
  "access": "proxy",
  "basicAuth": false
}
```

### Example Graphite response

```
HTTP/1.1 200
Content-Type: application/json

{
  "datasource": {
    "id": 1,
    "orgId": 1,
    "name": "test_datasource",
    "type": "graphite",
    "typeLogoUrl": "",
    "access": "proxy",
    "url": "http://mydatasource.com",
    "password": "",
    "user": "",
    "database": "",
    "basicAuth": false,
    "basicAuthUser": "",
    "basicAuthPassword": "",
    "withCredentials": false,
    "isDefault": false,
    "jsonData": {},
    "secureJsonFields": {},
    "version": 1,
    "readOnly": false
  },
}
```

```
"id": 1,  
"message": "Datasource added",  
"name": "test_datasource"  
}
```

### Note

When you define password and basicAuthPassword within secureJsonData, Amazon Managed Grafana encryptes them securely as an encrypted blob in the database. The response then lists the encrypte fields in secureJsonFields.

### Example Graphite request with basic auth enabled

```
POST /api/datasources HTTP/1.1  
Accept: application/json  
Content-Type: application/json  
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVdFsnFZXdE9ZWmNrMkZYbk  
  
{  
  "name": "test_datasource",  
  "type": "graphite",  
  "url": "http://mydatasource.com",  
  "access": "proxy",  
  "basicAuth": true,  
  "basicAuthUser": "basicuser",  
  "secureJsonData": {  
    "basicAuthPassword": "basicpassword"  
  }  
}
```

### Example response with basic auth enabled

```
HTTP/1.1 200  
Content-Type: application/json  
  
{  
  "datasource": {  
    "id": 1,  
    "orgId": 1,  
    "name": "test_datasource",  
    "type": "graphite",  
    "typeLogoUrl": "",  
    "access": "proxy",  
    "url": "http://mydatasource.com",  
    "password": "",  
    "user": "",  
    "database": "",  
    "basicAuth": true,  
    "basicAuthUser": "basicuser",  
    "basicAuthPassword": "",  
    "withCredentials": false,  
    "isDefault": false,  
    "jsonData": {},  
    "secureJsonFields": {  
      "basicAuthPassword": true  
    },  
    "version": 1,  
    "readOnly": false  
  },  
  "id": 102,  
  "message": "Datasource added",  
  "name": "test_datasource"  
}
```

### Example CloudWatch request

```
POST /api/datasources HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVadFsNFZXde9ZWmNrMkZYbk

{
  "name": "test_datasource",
  "type": "cloudwatch",
  "url": "http://monitoring.us-west-1.amazonaws.com",
  "access": "proxy",
  "jsonData": {
    "authType": "keys",
    "defaultRegion": "us-west-1"
  },
  "secureJsonData": {
    "accessKey": "O14pIDpeKSA6XikgO14p",
    "secretKey": "dGVzdCBBrZXkgYmxlYXNlIGRvbid0IHNOZWFs"
  }
}
```

## Update an existing data source

```
PUT /api/datasources/:datasourceId
```

### Example request

```
PUT /api/datasources/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVadFsNFZXde9ZWmNrMkZYbk

{
  "id": 1,
  "orgId": 1,
  "name": "test_datasource",
  "type": "graphite",
  "access": "proxy",
  "url": "http://mydatasource.com",
  "password": "",
  "user": "",
  "database": "",
  "basicAuth": true,
  "basicAuthUser": "basicuser",
  "secureJsonData": {
    "basicAuthPassword": "basicpassword"
  },
  "isDefault": false,
  "jsonData": null
}
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "datasource": {
    "id": 1,
```

```
"orgId": 1,
"name": "test_datasource",
"type": "graphite",
"typeLogoUrl": "",
"access": "proxy",
"url": "http://mydatasource.com",
"password": "",
"user": "",
"database": "",
"basicAuth": true,
"basicAuthUser": "basicuser",
"basicAuthPassword": "",
"withCredentials": false,
"isDefault": false,
"jsonData": {},
"secureJsonFields": {
  "basicAuthPassword": true
},
"version": 1,
"readOnly": false
},
"id": 102,
"message": "Datasource updated",
"name": "test_datasource"
}
```

#### Note

We recommend that you define password and basicAuthPassword within secureJsonData so that they are stored securely as an encrypted blob in the database. The response then lists the encrypt fields in secureJsonFields.

## Delete data source by Id

```
DELETE /api/datasources/:datasourceId
```

#### Example request

```
DELETE /api/datasources/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUlyY2RnVKZTFVadFsNFZXdeE9ZWmNrMkZYbk
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "Data source deleted"}
```

## Delete data source by UID

```
DELETE /api/datasources/uid/:uid
```

#### Example request

```
DELETE /api/datasources/uid/kLtEtcRGk HTTP/1.1
```



```
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "Data source deleted"}
```

## Delete data source by name

```
DELETE /api/datasources/name/:datasourceName
```

#### Example request

```
DELETE /api/datasources/name/test_datasource HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "message": "Data source deleted",
  "id": 1
}
```

## Data source proxy calls

```
GET /api/datasources/proxy/:datasourceId/*
```

Proxies all calls to the actual data source.

## Query data source by Id

```
POST /api/tsdb/query
```

Queries a data source having backend implementation. Most built-in data sources have backend implementation.

#### Example request

```
POST /api/tsdb/query HTTP/1.1
Accept: application/json
Content-Type: application/json

{
```

```
{
  "from": "1420066800000",
  "to": "1575845999999",
  "queries": [
    {
      "refId": "A",
      "intervalMs": 86400000,
      "maxDataPoints": 1092,
      "datasourceId": 86,
      "rawSql": "SELECT 1 as valueOne, 2 as valueTwo",
      "format": "table"
    }
  ]
}
```

### Note

The `from`, `to`, and `queries` properties are required.

JSON body schema:

- **from/to**— Must be either absolute in epoch timestamps in milliseconds, or relative using Grafana time units. For example, `now-1h`.
- **queries.refId**— (Optional) Specifies an identifier for the query. The default is `A`.
- **queries.datasourceId**— Specifies the data source to be queried. Each query in the request must have a unique `datasourceId`.
- **queries.maxDataPoints**— (Optional) Specifies the maximum amount of data points that a dashboard panel can render. The default is 100.
- **queries.intervalMs**— (Optional) Specifies the time interval in milliseconds of time series. The default is 1000

### Example request for the MySQL data source:

```
POST /api/tsdb/query HTTP/1.1
Accept: application/json
Content-Type: application/json

{
  "from": "1420066800000",
  "to": "1575845999999",
  "queries": [
    {
      "refId": "A",
      "intervalMs": 86400000,
      "maxDataPoints": 1092,
      "datasourceId": 86,
      "rawSql": "SELECT\n  time,\n  sum(opened) AS \"Opened\",\n  sum(closed) AS \"Closed\"\nFROM\n  issues_activity\nWHERE\n  $__unixEpochFilter(time) AND\n  period = 'm' AND\n  repo IN('grafana/grafana') AND\n  opened_by IN('Contributor','Grafana Labs')\nGROUP BY\n  1\nORDER BY 1",
      "format": "time_series"
    }
  ]
}
```

### Example response for the MySQL data source request:

```
HTTP/1.1 200
Content-Type: application/json

{
```

```
"results": {
  "A": {
    "refId": "A",
    "meta": {
      "rowCount": 0,
      "sql": "SELECT\n time,\n sum(opened) AS \"Opened\",\n sum(closed) AS \"Closed\n\nFROM\n issues_activity\nWHERE\n time <= 1420066800 AND time >= 1575845999 AND\n period = 'm' AND\n repo IN('grafana/grafana') AND\n opened_by IN('Contributor','Grafana Labs')\nGROUP BY 1\nORDER BY 1\n",
    },
    "series": [
      {
        "name": "Opened",
        "points": [
          [
            109,
            1420070400000
          ],
          [
            122,
            1422748800000
          ]
        ]
      },
      {
        "name": "Closed",
        "points": [
          [
            89,
            1420070400000
          ]
        ]
      }
    ]
  }
}
```

## Data Source Permissions API

Use the Data Source Permissions API to enable, disable, list, add, and remove permissions for data sources.

You can set permissions for a user or a team. Permissions can't be set for Admins, because they always have access to everything.

The permission levels for the permission field are as follows:

- 1 = Query

### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Enable permissions for a data source

```
POST /api/datasources/:id/enable-permissions
```

Enables permissions for the data source with the given id. No one except Org Admins will be able to query the data source until permissions have been added to permit certain users or teams to query the data source.

#### Example request

```
POST /api/datasources/1/enable-permissions
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

#### Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 35

{"message": "Datasource permissions enabled"}
```

Status Codes:

- **200**— Created
- **400**— Permissions can't be enabled, see the response body for details.
- **401**— Unauthorized
- **403**— Access denied
- **404**— Data source not found

## Disable permissions for a data source

```
POST /api/datasources/:id/disable-permissions
```

Disables permissions for the data source with the given id. All existing permissions will be removed and anyone will be able to query the data source.

#### Example request

```
POST /api/datasources/1/disable-permissions
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk

{}
```

#### Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 35

{"message": "Datasource permissions disabled"}
```

Status Codes:

- **200**— Ok

- **400**— Permissions can't be disabled, see the response body for details.
- **401**— Unauthorized
- **403**— Access denied
- **404**— Data source not found

## Get permissions for a data source

```
GET /api/datasources/:id/permissions
```

Gets all existing permissions for the data source with the given id.

### Example request

```
GET /api/datasources/1/permissions HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 551

{
  "datasourceId": 1,
  "enabled": true,
  "permissions": [
    {
      "id": 1,
      "datasourceId": 1,
      "userId": 1,
      "userLogin": "user",
      "userEmail": "user@test.com",
      "userAvatarUrl": "/avatar/46d229b033af06a191ff2267bca9ae",
      "permission": 1,
      "permissionName": "Query",
      "created": "2017-06-20T02:00:00+02:00",
      "updated": "2017-06-20T02:00:00+02:00",
    },
    {
      "id": 2,
      "datasourceId": 1,
      "teamId": 1,
      "team": "A Team",
      "teamAvatarUrl": "/avatar/46d229b033af06a191ff2267bca9ae",
      "permission": 1,
      "permissionName": "Query",
      "created": "2017-06-20T02:00:00+02:00",
      "updated": "2017-06-20T02:00:00+02:00",
    }
  ]
}
```

### Status Codes:

- **200**— Ok

- **401**— Unauthorized
- **403**— Access denied
- **404**— Data source not found

## Add permission for a data source

```
POST /api/datasources/:id/permissions
```

Adds a user permission for the data source with the given id.

### Example request to add user permission

```
POST /api/datasources/1/permissions
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk

{
  "userId": 1,
  "permission": 1
}
```

### Example response for adding a user permission

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 35

{"message":"Datasource permission added"}
```

### Example request to add team permission

```
POST /api/datasources/1/permissions
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk

{
  "teamId": 1,
  "permission": 1
}
```

### Example response for adding a team permission

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 35

{"message":"Datasource permission added"}
```

Status Codes:

- **200**— Ok
- **400**— Permission can't be added, see response body for details.
- **401**— Unauthorized

- **403**— Access denied
- **404**— Data source not found

## Remove permission for a data source

```
DELETE /api/datasources/:id/permissions/:permissionId
```

Removes the permission with the given `permissionId` for the data source with the given `id`.

### Example request

```
DELETE /api/datasources/1/permissions/2
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 35

{"message": "Datasource permission removed"}
```

Status Codes:

- **200**— Ok
- **401**— Unauthorized
- **403**— Access denied
- **404**— Data source not found or permission not found

## External Group Synchronization API

### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Get external groups

```
GET /api/teams/:teamId/groups
```

### Example request

```
GET /api/teams/1/groups HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk]
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "orgId": 1,
    "teamId": 1,
    "groupId": "cn=editors,ou=groups,dc=grafana,dc=org"
  }
]
```

Status Codes:

- **200**— Ok
- **401**— Unauthorized
- **403**— Access denied

## Add external group

```
POST /api/teams/:teamId/groups
```

### Example request

```
POST /api/teams/1/members HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdE9ZWmNrMkZYbk]

{
  "groupId": "cn=editors,ou=groups,dc=grafana,dc=org"
}
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "Group added to Team"}
```

Status Codes:

- **200**— Ok
- **400**— Group is already added to this team
- **401**— Unauthorized
- **403**— Permission denied
- **404**— Team not found

## Remove external group

```
DELETE /api/teams/:teamId/groups/:groupId
```

### Example request



```
DELETE /api/teams/1/groups/cn=editors,ou=groups,dc=grafana,dc=org HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdE9ZWmNrMkZYbk]
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "Team Group removed"}
```

#### Status Codes:

- **200**— Ok
- **401**— Unauthorized
- **403**— Permission denied
- **404**— Team not found or group not found

## Folder API

Use the Folder API to work with folders in the Amazon Managed Grafana workspace.

The identifier (id) of a folder is an auto-incrementing numeric value and is only unique per workspace. The unique identifier (uid) of a folder can be used to uniquely identify a folder between multiple workspaces. It's automatically generated if you don't provide one when you create a folder. The uid allows having consistent URLs for accessing folder and when synchronizing folder between multiple Amazon Managed Grafana workspaces. The use of the uid means that changing the title of a folder does not break any bookmarked links to that folder.

The uid can have a maximum length of 40 characters.

Folders can't be nested.

#### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

The **General** folder, with an id of 0, is not part of the Folder API. You can't use the Folder API to retrieve information about the general folder.

## Create folder

```
POST /api/folders
```

Creates a new folder.

#### Example request

```
POST /api/folders HTTP/1.1
Accept: application/json
```

```
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk

{
  "uid": "nErXDvCkzz",
  "title": "Department ABC"
}
```

JSON body schema:

- **uid**— Optional unique identifier. If null, a new uid is generated.
- **title**— The title for the folder.

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "id": 1,
  "uid": "nErXDvCkzz",
  "title": "Department ABC",
  "url": "/dashboards/f/nErXDvCkzz/departments-abc",
  "hasAcl": false,
  "canSave": true,
  "canEdit": true,
  "canAdmin": true,
  "createdBy": "admin",
  "created": "2018-01-31T17:43:12+01:00",
  "updatedBy": "admin",
  "updated": "2018-01-31T17:43:12+01:00",
  "version": 1
}
```

Status Codes:

- **200**— Created
- **400**— Error such as invalid JSON, invalid or missing fields
- **401**— Unauthorized
- **403**— Access denied

## Update folder

```
PUT /api/folders/:uid
```

Updates the existing folder that matches the uid.

### Example request

```
PUT /api/folders/nErXDvCkzz HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk

{
  "title": "Department DEF",
}
```

```
"version": 1
}
```

JSON body schema:

- **uid**— Changes the unique identifier, if provided.
- **title**— The title of the folder.
- **version**— Provide the current version to be able to overwrite the folder. Not needed if `overwrite=true`.
- **overwrite**— Set to `true` to overwrite the existing folder with a newer version.

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "id": 1,
  "uid": "nErXDvCkzz",
  "title": "Department DEF",
  "url": "/dashboards/f/nErXDvCkzz/department-def",
  "hasAcl": false,
  "canSave": true,
  "canEdit": true,
  "canAdmin": true,
  "createdBy": "admin",
  "created": "2018-01-31T17:43:12+01:00",
  "updatedBy": "admin",
  "updated": "2018-01-31T17:43:12+01:00",
  "version": 1
}
```

Status Codes:

- **200**— Created
- **400**— Error such as invalid JSON, invalid or missing fields
- **401**— Unauthorized
- **403**— Access denied
- **404**— Folder not found
- **412**— Precondition failed

The **412** status code is used to explain why the folder can't be updated.

- The folder has been changed by someone else `status=version-mismatch`

The response body has the following properties:

```
HTTP/1.1 412 Precondition Failed
Content-Type: application/json; charset=UTF-8
Content-Length: 97

{
  "message": "The folder has been changed by someone else",
  "status": "version-mismatch"
}
```

## Get all folders

```
GET /api/folders
```

Returns all folders that you have permission to view. You can control the maximum number of folders returned by using the `limit` query parameter. The default is 1000.

### Example request

```
GET /api/folders?limit=10 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "id": 1,
    "uid": "nErXDvCkzz",
    "title": "Department ABC"
  },
  {
    "id": 2,
    "uid": "k3S1cklGk",
    "title": "Department RND"
  }
]
```

## Get folder by uid

```
GET /api/folders/:uid
```

Returns all folders that matches the given uid.

### Example request

```
GET /api/folders/nErXDvCkzz HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "id": 1,
  "uid": "nErXDvCkzz",
  "title": "Department ABC",
  "url": "/dashboards/f/nErXDvCkzz/departments-abc",
  "hasAcl": false,
}
```

```
"canSave": true,
"canEdit": true,
"canAdmin": true,
"createdBy": "admin",
"created": "2018-01-31T17:43:12+01:00",
"updatedBy": "admin",
"updated": "2018-01-31T17:43:12+01:00",
"version": 1
}
```

Status Codes:

- **200**— Found
- **401**— Unauthorized
- **403**— Access denied
- **404**— Not found

## Get folder by id

```
GET /api/folders/id/:id
```

Returns the folder that matches the given id.

### Example request

```
GET /api/folders/id/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "id": 1,
  "uid": "nErXDvCkzz",
  "title": "Department ABC",
  "url": "/dashboards/f/nErXDvCkzz/departement-abc",
  "hasAcl": false,
  "canSave": true,
  "canEdit": true,
  "canAdmin": true,
  "createdBy": "admin",
  "created": "2018-01-31T17:43:12+01:00",
  "updatedBy": "admin",
  "updated": "2018-01-31T17:43:12+01:00",
  "version": 1
}
```

Status Codes:

- **200**— Found
- **401**— Unauthorized
- **403**— Access denied

- **404**— Not found

## Delete folder by uid

```
DELETE /api/folders/:uid
```

Deletes the folder matching the uid, and also deletes all dashboards stored in the folder. This operation can't be reverted.

### Example request

```
DELETE /api/folders/nErXDvCkzz HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "message": "Folder deleted",
  "id": 2
}
```

Status Codes:

- **200**— Deleted
- **401**— Unauthorized
- **403**— Access denied
- **404**— Not found

## Folder/Dashboard Search API

Use the FolderDashboard-Search API to search folders and dashboards in an Amazon Managed Grafana workspace.

### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Search folders and dashboards

```
GET /api/search/
```

Query parameters:

- **query**— Search query
- **tag**— List of tags to search for. These are Grafana tags, not AWS tags.

- **type**— The type to search for, either `dash-folder` or `dash-db`.
- **dashboardIds**— List of dashboard Id's to search for.
- **folderIds**— List of dashboard Id's to search for in dashboards.
- **starred**— Flag to specify that only starred dashboards are to be returned.
- **limit**— Limit the number of returned results (maximum is 5000).
- **page**— Use this parameter to access hits beyond limit. Numbering starts at 1. The `limit` parameter acts as page size.

#### Example request for retrieving folders and dashboards of the general folder

```
GET /api/search?folderIds=0&query=&starred=false HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

#### Example response for retrieving folders and dashboards of the general folder

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "id": 163,
    "uid": "000000163",
    "title": "Folder",
    "url": "/dashboards/f/000000163/folder",
    "type": "dash-folder",
    "tags": [],
    "isStarred": false,
    "uri": "db/folder" // deprecated in Grafana v5.0
  },
  {
    "id": 1,
    "uid": "cIBgcSjkk",
    "title": "Production Overview",
    "url": "/d/cIBgcSjkk/production-overview",
    "type": "dash-db",
    "tags": [prod],
    "isStarred": true,
    "uri": "db/production-overview" // deprecated in Grafana v5.0
  }
]
```

#### Example request for searching for starred dashboards

```
GET /api/search?query=Production%20Overview&starred=true&tag=prod HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

#### Example response for searching for starred dashboards

```
HTTP/1.1 200
Content-Type: application/json

[HTTP/1.1 200
Content-Type: application/json
```

```
[
  {
    "id": 1,
    "uid": "cIBgcSjkk",
    "title": "Production Overview",
    "url": "/d/cIBgcSjkk/production-overview",
    "type": "dash-db",
    "tags": [prod],
    "isStarred": true,
    "folderId": 2,
    "folderUid": "000000163",
    "folderTitle": "Folder",
    "folderUrl": "/dashboards/f/000000163/folder",
    "uri": "db/production-overview" // deprecated in Grafana v5.0
  }
]
```

## Folder Permissions API

Use the Folder API to update or retrieve the permissions for a folder.

Permissions with `folderId=-1` are the default permissions for users with the Viewer and Editor roles. Permissions can be set for a user, a team or a role (Viewer or Editor). Permissions cannot be set for Admins - they always have access to everything.

The permission levels for the `permission` field are as follows:

- 1 = View
- 2 = Edit
- 4 = Admin

### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Get permissions for a folder

```
GET /api/folders/:uid/permissions
```

Gets all existing permissions for the folder with the given `uid`.

### Example request

```
GET /api/folders/nErXDvCkzz/permissions HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 551
```



```
[
  {
    "id": 1,
    "folderId": -1,
    "created": "2017-06-20T02:00:00+02:00",
    "updated": "2017-06-20T02:00:00+02:00",
    "userId": 0,
    "userLogin": "",
    "userEmail": "",
    "teamId": 0,
    "team": "",
    "role": "Viewer",
    "permission": 1,
    "permissionName": "View",
    "uid": "nErXDvCkzz",
    "title": "",
    "slug": "",
    "isFolder": false,
    "url": ""
  },
  {
    "id": 2,
    "dashboardId": -1,
    "created": "2017-06-20T02:00:00+02:00",
    "updated": "2017-06-20T02:00:00+02:00",
    "userId": 0,
    "userLogin": "",
    "userEmail": "",
    "teamId": 0,
    "team": "",
    "role": "Editor",
    "permission": 2,
    "permissionName": "Edit",
    "uid": "",
    "title": "",
    "slug": "",
    "isFolder": false,
    "url": ""
  }
]
```

Status Codes:

- **200**— OK
- **401**— Unauthorized
- **403**— Access denied
- **404**— Folder not found

## Update permissions for a folder

```
POST /api/folders/:uid/permissions
```

Updates the permissions for a folder. This operation removes existing permissions if they are not included in the request.

### Example request

```
POST /api/folders/nErXDvCkzz/permissions
Accept: application/json
```

```
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
{
  "items": [
    {
      "role": "Viewer",
      "permission": 1
    },
    {
      "role": "Editor",
      "permission": 2
    },
    {
      "teamId": 1,
      "permission": 1
    },
    {
      "userId": 11,
      "permission": 4
    }
  ]
}
```

JSON body schema:

- **items**— The permission items to add or update. Existing items that are omitted from the list will be removed.

#### Example response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 35

{"message": "Folder permissions updated", "id": 1, "title": "Department ABC"}
```

Status Codes:

- **200**— Created
- **401**— Unauthorized
- **403**— Access denied
- **404**— Dashboard not found

## Organization API

Use the Organization API to work with organizations in an Amazon Managed Grafana workspace.

#### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Get current organization

```
GET /api/org/
```

#### Example request

```
GET /api/org/ HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVadFsNFZXde9ZWmNrMkZYbk
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "id":1,
  "name":"Main Org."
}
```

## Get all users within the current organization

```
GET /api/org/users
```

Required permissions: the `org.users:read` action with the scope `users:*`

#### Example request

```
GET /api/org/users HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVadFsNFZXde9ZWmNrMkZYbk
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "orgId": 1,
    "userId": 1,
    "email": "admin@localhost",
    "avatarUrl": "/avatar/46d229b033af06a191ff2267bca9ae",
    "login": "admin",
    "role": "Admin",
    "lastSeenAt": "2019-08-09T11:02:49+02:00",
    "lastSeenAtAge": "< 1m"
  }
]
```

## Get all users within the current organization (lookup)

```
GET /api/org/users/lookup
```

Returns all users within the current organization, but with less detailed information. Accessible to users with org admin role, admin in any folder or admin of any team. Used mostly by the Grafana UI to provide a list of users when adding team members and when editing folder/dashboard permissions.

### Example request

```
GET /api/org/users/lookup HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "userId": 1,
    "login": "admin",
    "avatarUrl": "/avatar/46d229b033af06a191ff2267bca9ae"
  }
]
```

## Updates the given user

```
PATCH /api/org/users/:userId
```

Required permissions: the `org.users.role:update` action with the scope `users:*`

### Example request

```
PATCH /api/org/users/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk

{
  "role": "Viewer",
}
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message":"Organization user updated"}
```

## Deletes user in current organization

```
DELETE /api/org/users/:userId
```

Required permissions: the `org.users:remove` action with the scope `users:*`

### Example request

```
DELETE /api/org/users/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
```

```
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "User removed from organization"}
```

## Update the current organization

```
PUT /api/org
```

#### Example request

```
PUT /api/org HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk

{
  "name": "Main Org."
}
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "Organization updated"}
```

## Add user to the current organization

```
POST /api/org/users
```

Required permissions: the `org.users:add` action with the scope `users:*`

#### Example request

```
POST /api/org/users HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk

{
  "role": "Admin",
  "loginOrEmail": "admin"
}
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json
```

```
{"message": "User added to organization", "userId": 1}
```

## Playlist API

Use the Playlist API to work with playlists in the Amazon Managed Grafana workspace.

### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Search playlist

```
GET /api/playlists
```

Returns all playlists for the current Amazon Managed Grafana workspace, using pagination.

### Example request

```
GET /api/playlists HTTP/1.1
Accept: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

Querystring parameters:

- **query**— Limit the responses to playlists that have a name like this value.
- **limit**— Limit the response to X number of playlists.

### Example response

```
HTTP/1.1 200
Content-Type: application/json
[
  {
    "id": 1,
    "name": "my playlist",
    "interval": "5m"
  }
]
```

## Get one playlist

```
GET /api/playlists/:id
```

### Example request

```
GET /api/playlists/1 HTTP/1.1
Accept: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json
{
  "id" : 1,
  "name": "my playlist",
  "interval": "5m",
  "orgId": "my org",
  "items": [
    {
      "id": 1,
      "playlistId": 1,
      "type": "dashboard_by_id",
      "value": "3",
      "order": 1,
      "title": "my third dashboard"
    },
    {
      "id": 2,
      "playlistId": 1,
      "type": "dashboard_by_tag",
      "value": "myTag",
      "order": 2,
      "title": "my other dashboard"
    }
  ]
}
```

## Get playlist items

```
GET /api/playlists/:id/items
```

### Example request

```
GET /api/playlists/1/items HTTP/1.1
Accept: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json
[
  {
    "id": 1,
    "playlistId": 1,
    "type": "dashboard_by_id",
    "value": "3",
    "order": 1,
    "title": "my third dashboard"
  },
  {
    "id": 2,
    "playlistId": 1,
    "type": "dashboard_by_tag",
    "value": "myTag",
    "order": 2,
    "title": "my other dashboard"
  }
]
```

## Get playlist dashboards

```
GET /api/playlists/:id/dashboards
```

### Example request

```
GET /api/playlists/1/dashboards HTTP/1.1
Accept: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdeE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json
[
  {
    "id": 3,
    "title": "my third dashboard",
    "order": 1,
  },
  {
    "id": 5,
    "title": "my other dashboard",
    "order": 2,
  }
]
```

## Create a playlist

```
POST /api/playlists/
```

### Example request

```
PUT /api/playlists/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdeE9ZWmNrMkZYbk
{
  "name": "my playlist",
  "interval": "5m",
  "items": [
    {
      "type": "dashboard_by_id",
      "value": "3",
      "order": 1,
      "title": "my third dashboard"
    },
    {
      "type": "dashboard_by_tag",
      "value": "myTag",
      "order": 2,
      "title": "my other dashboard"
    }
  ]
}
```

### Example response



```
HTTP/1.1 200
Content-Type: application/json
{
  "id": 1,
  "name": "my playlist",
  "interval": "5m"
}
```

## Update a playlist

```
PUT /api/playlists/:id
```

### Example request

```
PUT /api/playlists/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcGlpU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
{
  "name": "my playlist",
  "interval": "5m",
  "items": [
    {
      "playlistId": 1,
      "type": "dashboard_by_id",
      "value": "3",
      "order": 1,
      "title": "my third dashboard"
    },
    {
      "playlistId": 1,
      "type": "dashboard_by_tag",
      "value": "myTag",
      "order": 2,
      "title": "my other dashboard"
    }
  ]
}
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json
{
  "id": 1,
  "name": "my playlist",
  "interval": "5m",
  "orgId": "my org",
  "items": [
    {
      "id": 1,
      "playlistId": 1,
      "type": "dashboard_by_id",
      "value": "3",
      "order": 1,
      "title": "my third dashboard"
    },
    {
      "id": 2,
      "playlistId": 1,

```

```
{
  "type": "dashboard_by_tag",
  "value": "myTag",
  "order": 2,
  "title": "my other dashboard"
}
```

## Delete a playlist

```
DELETE /api/playlists/:id
```

### Example request

```
DELETE /api/playlists/1 HTTP/1.1
Accept: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdeE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json
{}
```

# Preferences API

Use the Preferences API to work with user preferences in the Amazon Managed Grafana workspace.

Keys:

- **theme**— Valid values are `light`, `dark`, or an empty string to use the default theme.
- **homeDashboardId**— The numerical `:id` of a favorited dashboard. The default is 0.
- **timezone**— Valid values are `utc`, `browser`, or an empty string to use the default.

Omitting a key causes the current value to be replaced with the system default value.

### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Get current user preferences

```
GET /api/user/preferences
```

### Example request

```
GET /api/user/preferences HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdeE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"theme":"","homeDashboardId":0,"timezone":""}
```

## Update current user preferences

```
PUT /api/user/preferences
```

### Example request

```
PUT /api/user/preferences HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk

{
  "theme": "",
  "homeDashboardId": 0,
  "timezone": "utc"
}
```

### Example response

```
HTTP/1.1 200
Content-Type: text/plain; charset=utf-8

{"message":"Preferences updated"}
```

## Get current org preferences

```
GET /api/org/preferences
```

### Example request

```
GET /api/org/preferences HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"theme":"","homeDashboardId":0,"timezone":""}
```

## Update current org preferences

```
PUT /api/org/preferences
```

### Example request

```
PUT /api/org/preferences HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdE9ZWmNrMkZYbk

{
  "theme": "",
  "homeDashboardId":0,
  "timezone":"utc"
}
```

### Example response

```
HTTP/1.1 200
Content-Type: text/plain; charset=utf-8

{"message":"Preferences updated"}
```

## Snapshot API

Use the Snapshot API to work with snapshots in an Amazon Managed Grafana workspace.

### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Create new shapshot

```
POST /api/snapshots
```

### Example request

```
POST /api/snapshots HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXdE9ZWmNrMkZYbk

{
  "dashboard": {
    "editable":false,
    "hideControls":true,
    "nav":[
      {
        "enable":false,
        "type":"timepicker"
      }
    ],
    "rows": [
      {
      }
    ],
    "style":"dark",
    "tags":[],
  }
```

```
    "templating":{
      "list":[
      ]
    },
    "time":{
    },
    "timezone":"browser",
    "title":"Home",
    "version":5
  },
  "expires": 3600
}
```

JSON body schema:

- **dashboard**— (Required) The complete dashboard model.
- **name**— (Optional) A name for the snapshot.
- **expires**— (Optional) When the snapshot should expire, in seconds. The default is to never expire.
- **external**— (Optional) Save the snapshot on an external server rather than locally. Default is false.
- **key**— (Required if `external` is true) Define a unique key.
- **deletekey**— (Required if `external` is true) A unique key to be used to delete the snapshot. It is different than `key` so that only the creator can delete the snapshot.

#### Note

When creating a snapshot using the API, you have to provide the full dashboard payload including the snapshot data. This endpoint is designed for the Grafana UI.

#### Example response

```
HTTP/1.1 200
Content-Type: application/json
{
  "deleteKey":"XXXXXXX",
  "deleteUrl":"myurl/api/snapshots-delete/XXXXXXX",
  "key":"YYYYYYY",
  "url":"myurl/dashboard/snapshot/YYYYYYY",
  "id": 1,
}
```

Keys:

- **deleteKey**— A key generated to be used to delete the snapshot.
- **key**— A key generated to share the dashboard.

## Get list of snapshots

```
GET /api/dashboard/snapshots
```

Query parameters:

- **query**— Search query
- **limit**— Limit the number of returned results

#### Example request

```
GET /api/dashboard/snapshots HTTP/1.1
Accept: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "id": 8,
    "name": "Home",
    "key": "YYYYYYY",
    "orgId": 1,
    "userId": 1,
    "external": false,
    "externalUrl": "",
    "expires": "2200-13-32T25:23:23+02:00",
    "created": "2200-13-32T28:24:23+02:00",
    "updated": "2200-13-32T28:24:23+02:00"
  }
]
```

## Get snapshot by key

```
GET /api/snapshots/:key
```

### Example request

```
GET /api/snapshots/YYYYYYY HTTP/1.1
Accept: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "meta": {
    "isSnapshot": true,
    "type": "snapshot",
    "canSave": false,
    "canEdit": false,
    "canStar": false,
    "slug": "",
    "expires": "2200-13-32T25:23:23+02:00",
    "created": "2200-13-32T28:24:23+02:00"
  },
  "dashboard": {
    "editable": false,
    "hideControls": true,
    "nav": [
      {
        "enable": false,
        "type": "timepicker"
      }
    ]
  }
}
```

```
"rows": [
  {
  },
],
"style": "dark",
"tags": [],
"templating": {
  "list": [
  ],
},
"time": {
},
"timezone": "browser",
"title": "Home",
"version": 5
}
```

## Delete snapshot by key

```
DELETE /api/snapshots/:key
```

### Example request

```
DELETE /api/snapshots/YYYYYYY HTTP/1.1
Accept: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "Snapshot deleted. It might take an hour before it's cleared from any CDN caches.", "id": 1}
```

## Delete snapshot by deleteKey

This API call can be used without authentication by using the secret delete key for the snapshot.

```
GET /api/snapshots-delete/:deleteKey
```

### Example request

```
GET /api/snapshots-delete/XXXXXXX HTTP/1.1
Accept: application/json
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "Snapshot deleted. It might take an hour before it's cleared from any CDN caches.", "id": 1}
```

# Team API

Use the Team API to work with teams in an Amazon Managed Grafana workspace. All actions in this API require that you have the Admin role.

## Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Team search with pagination

```
GET /api/teams/search?perpage=50&page=1&query=myteam
```

or

```
GET /api/teams/search?name=myteam
```

### Example request

```
GET /api/teams/search?perpage=10&page=1&query=myteam HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

### Using the query parameter

The default value for the `perpage` parameter is 1000 and for the `page` parameter is 1.

The `totalCount` field in the response can be used for pagination of the teams list. For example, if `totalCount` is 100 teams and the `perpage` parameter is set to 10, then there are 10 pages of teams.

The `query` parameter is optional and returns results where the query value is contained in the `name` field. Query values with spaces need to be URL-encoded. For example, `query=my%20team`.

### Using the name parameter

The `name` parameter returns a single team if the parameter matches the `name` field.

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "totalCount": 1,
  "teams": [
    {
      "id": 1,
      "orgId": 1,
      "name": "MyTestTeam",
      "email": "",
      "avatarUrl": "\/avatar\/3f49c15916554246daa714b9bd0ee39",
      "memberCount": 1
    }
  ],
  "page": 1,
```



```
"perPage": 1000
```

#### Status Codes:

- **200**— Created
- **401**— Unauthorized
- **403**— Permission denied
- **404**— Team not found (if searching by name)

## Get team by Id

```
GET /api/teams/:id
```

#### Example request

```
GET /api/teams/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "id": 1,
  "orgId": 1,
  "name": "MyTestTeam",
  "email": "",
  "created": "2017-12-15T10:40:45+01:00",
  "updated": "2017-12-15T10:40:45+01:00"
}
```

## Add a team

The name of the team must be unique. The name field is required and the email and orgId fields are optional.

```
POST /api/teams
```

#### Example request

```
POST /api/teams HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXdE9ZWmNrMkZYbk

{
  "name": "MyTestTeam",
  "email": "email@test.com",
  "orgId": 2
}
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "Team created", "teamId": 2}
```

#### Status Codes:

- **200**— Created
- **401**— Unauthorized
- **403**— Permission denied
- **409**— Team name already exists

## Update team

```
PUT /api/teams/:id
```

Only the name and email fields can be updated.

### Example request

```
PUT /api/teams/2 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk

{
  "name": "MyTestTeam",
  "email": "email@test.com"
}
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "Team updated"}
```

#### Status Codes:

- **200**— Created
- **401**— Unauthorized
- **403**— Permission denied
- **404**— Team not found
- **409**— Team name already exists

## Delete team by Id

```
DELETE /api/teams/:id
```

### Example request

```
DELETE /api/teams/2 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "Team deleted"}
```

### Status Codes:

- **200**— Created
- **401**— Unauthorized
- **403**— Permission denied
- **404**— Team not found

## Get team members

```
GET /api/teams/:teamId/members
```

### Example request

```
GET /api/teams/1/members HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "orgId": 1,
    "teamId": 1,
    "userId": 3,
    "email": "user1@email.com",
    "login": "user1",
    "avatarUrl": "\/avatar\/1b3c32f6386b0185c40d359cdc733a7"
  },
  {
    "orgId": 1,
    "teamId": 1,
    "userId": 2,
    "email": "user2@email.com",
    "login": "user2",
    "avatarUrl": "\/avatar\/cad3c68da76e45d10269e8ef02f8e7"
  }
]
```

### Status Codes:

- **200**— Created
- **401**— Unauthorized
- **403**— Permission denied

## Add team member

```
POST /api/teams/:teamId/members
```

### Example request

```
POST /api/teams/1/members HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk

{
  "userId": 2
}
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "Member added to Team"}
```

Status Codes:

- **200**— Created
- **400**— User is already in team
- **401**— Unauthorized
- **403**— Permission denied
- **404**— Team not found

## Remove member from team

```
DELETE /api/teams/:teamId/members/:userId
```

### Example request

```
DELETE /api/teams/2/members/3 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "Team Member removed"}
```

Status Codes:

- **200**— Created
- **401**— Unauthorized
- **403**— Permission denied
- **404**— Team not found/team member not found

## Get team preferences

```
GET /api/teams/:teamId/preferences
```

### Example request

```
GET /api/teams/2/preferences HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "theme": "",
  "homeDashboardId": 0,
  "timezone": ""
}
```

## Update team preferences

```
PUT /api/teams/:teamId/preferences
```

### Example request

```
PUT /api/teams/2/preferences HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk

{
  "theme": "dark",
  "homeDashboardId": 39,
  "timezone": "utc"
}
```

JSON body schema:

- **theme**— Specify either `light`, `dark`, or an empty string to use the default theme.
- **homeDashboardId**— The numerical `:id` of a dashboard. The default is 0.
- **timezone**— Specify either `utc`, `browser`, or an empty string to use the default.

Omitting a parameter causes the current value to be replaced with the system default value.

### Example response

```
HTTP/1.1 200
Content-Type: text/plain; charset=utf-8

{
  "message": "Preferences updated"
}
```

## User API

Use the User API to work with users in an Amazon Managed Grafana workspace.

### Note

To use a Grafana API with your Amazon Managed Grafana workspace, you must have a valid Grafana API key. You include this key in the `Authorization` field in the API request. For information about how to create a Grafana API key, see [Using Grafana HTTP APIs \(p. 292\)](#).

## Get teams that the user is a member of

```
GET /api/user/teams
```

### Example request

```
GET /api/user/teams HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "id": 1,
    "orgId": 1,
    "name": "MyTestTeam",
    "email": "",
    "avatarUrl": "\/avatar\/3f49c15916554246daa714b9bd0ee3",
    "memberCount": 1
  }
]
```

## Get list of snapshots

Stars the given Dashboard for the actual user.

```
POST /api/user/stars/dashboard/:dashboardId
```

### Example request

```
POST /api/user/stars/dashboard/1 HTTP/1.1
```

```
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVadFsNFZXde9ZWmNrMkZYbk
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "Dashboard starred!"}
```

## Unstar a dashboard

Deletes the starring of the given Dashboard for the actual user.

```
DELETE /api/user/stars/dashboard/:dashboardId
```

#### Example request

```
DELETE /api/user/stars/dashboard/1 HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVadFsNFZXde9ZWmNrMkZYbk
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

{"message": "Dashboard unstarred"}
```

## Get auth tokens of the actual user

```
GET /api/user/auth-tokens
```

#### Example request

```
GET /api/user/auth-tokens HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pUly2RnVKZTFVadFsNFZXde9ZWmNrMkZYbk
```

#### Example response

```
HTTP/1.1 200
Content-Type: application/json

[
  {
    "id": 361,
    "isActive": true,
    "clientId": "127.0.0.1",
    "browser": "Chrome",
    "browserVersion": "72.0",
    "os": "Linux",
```

```
"osVersion": "",
"device": "Other",
"createdAt": "2019-03-05T21:22:54+01:00",
"seenAt": "2019-03-06T19:41:06+01:00"
},
{
  "id": 364,
  "isActive": false,
  "clientIp": "127.0.0.1",
  "browser": "Mobile Safari",
  "browserVersion": "11.0",
  "os": "iOS",
  "osVersion": "11.0",
  "device": "iPhone",
  "createdAt": "2019-03-06T19:41:19+01:00",
  "seenAt": "2019-03-06T19:41:21+01:00"
}
]
```

## Revoke an auth token of the actual user

```
POST /api/user/revoke-auth-token
```

Revokes the given auth token (device) for the actual user. User of issued auth token (device) will no longer be logged in and will be required to authenticate again at their next activity.

### Example request

```
POST /api/user/revoke-auth-token HTTP/1.1
Accept: application/json
Content-Type: application/json
Authorization: Bearer eyJrIjoiT0tTcG1pU1Y2RnVKZTFVaDFsNFZXXde9ZWmNrMkZYbk

{
  "authTokenId": 364
}
```

### Example response

```
HTTP/1.1 200
Content-Type: application/json

{
  "message": "User auth token revoked"
}
```



# Upgrade a workspace to Grafana Enterprise

You can use the Amazon Managed Grafana console to upgrade your account and one or more workspaces to Grafana Enterprise. You can start with a 30-day free trial, or purchase a subscription license. Upgrading gives you access to consultation and support directly from Grafana Labs. You also gain access to a variety of third-party independent software vendors (ISVs), including the following:

- AppDynamics
- Datadog
- Dynatrace
- GitLab
- Honeycomb
- Jira
- MongoDB
- New Relic
- Oracle Database
- Salesforce
- ServiceNow
- Snowflake
- Splunk
- Wavefront

For a complete list of Enterprise data source plugins available when you upgrade, see [Data sources available in Grafana Enterprise \(p. 138\)](#).

Each Amazon Managed Grafana workspace is eligible for a 30-day free trial of Grafana Enterprise. To retain access to Enterprise third-party plugins after the 30-day trial, purchase a Grafana Enterprise license for your workspace before the trial expires.

Grafana Enterprise fees are in addition to the active user prices for Amazon Managed Grafana. For detailed fee information, see the [Grafana Enterprise Pricing page](#).

To upgrade a workspace to Grafana Enterprise, you must be signed in to an AWS account for a user or role that has the **AWSMarketplaceManageSubscriptions** IAM policy attached, or the equivalent permissions.

## To upgrade a workspace and your account to Grafana Enterprise

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the navigation pane, choose the menu icon.
3. Choose **All workspaces**.
4. Choose the name of the workspace that you want to upgrade.
5. Choose **Upgrade to Grafana Enterprise**.

6. Choose **30-day free trial** or **Grafana Enterprise subscription**.

If you choose **Grafana Enterprise subscription**, choose whether you want the contract to be automatically renewed when it ends. You can change the renewal preference at any time. This is an account-level setting. Changing your auto-renewal choice for one workspace also updates it for all other workspaces in the account that are currently subscribed to a Grafana Enterprise contract.

7. Choose **Subscribe**.

8. To register your free trial or paid subscription with Grafana Labs, choose the <https://grafana.com/partners/amg/register> link and complete the resulting page. You need to do this only once for your AWS account.

When you upgrade a workspace to Grafana Enterprise, your entire AWS account gains the benefits of the Grafana Enterprise subscription. However, Grafana Enterprise is not automatically applied to other Amazon Managed Grafana workspaces in your account. You can follow the instructions in the preceding procedure to apply Grafana Enterprise to your other workspaces. You pay the subscription contract fee for Grafana Enterprise only once for your account during the billing cycle, but every workspace that you activate for Grafana Enterprise also incurs per-user fees to Grafana Labs. A typical billing cycle is 30 days, but you can also choose to enter a custom billing cycle through a private offer. For more information about a private offer, see [Private offers](#) in the AWS Marketplace Buyer Guide. For detailed fee information, see [Grafana Enterprise on Amazon Managed Service for Grafana](#).

#### **Purchasing a subscription during the free trial**

Anytime during a 30-day free trial, you can purchase a subscription to Grafana Enterprise to ensure that you don't lose access to Grafana Enterprise features. If you do this, the paid subscription time starts immediately when you purchase the subscription and ends the free trial for that workspace.

#### **Account-level Grafana Enterprise subscription**

You can also choose to subscribe to Grafana Enterprise at the account level. If you do this, you are billed the subscription contract fee at the account level, which shows up in the AWS Marketplace section of your AWS bill. You can then apply this Enterprise license to each of your workspaces through the workspace details page. After you upgrade a workspace, the upgraded workspace begins to incur per-user fees.

#### **To subscribe to Grafana Enterprise at the account level**

1. In the left navigation pane, choose **Grafana Enterprise license**.
2. Choose **Upgrade to Grafana Enterprise**.
3. Choose **Grafana Enterprise subscription, Subscribe**.
4. To register your free trial or paid subscription with Grafana Labs, choose the <https://grafana.com/partners/amg/register> link and complete the resulting page. You need to do this only once for your AWS account.

#### **Private marketplace**

If your account is using a private marketplace, you must add Grafana Enterprise products to your private marketplace before you begin the free trial or upgrade to the Enterprise subscription. Follow the private marketplace instructions to add either of the following two products:

- [Grafana Enterprise Trial on AWS](#)
- [Grafana Enterprise on Amazon Managed Grafana](#)

For more information about private marketplaces, see [Private marketplaces](#).

## Canceling Grafana Enterprise

You can cancel Grafana Enterprise either at the workspace level or the account level. Canceling using either method does not delete any Amazon Managed Grafana workspaces. You can continue to use these workspaces without Enterprise capabilities.

### To cancel Grafana Enterprise for a single workspace and continue using Grafana Enterprise in other workspaces

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the navigation pane, choose the menu icon.
3. Choose **All workspaces**.
4. Choose the name of the workspace where you want to cancel Grafana Enterprise.
5. In the **Enterprise license** section, choose **Remove**.

Doing so stops incurring Grafana Enterprise per-user fees for this workspace. Your account is still subscribed to the Grafana Enterprise product and you will still incur monthly subscription fees.

### To cancel Grafana Enterprise for your entire account and remove Enterprise features from all your workspaces

1. Open the Amazon Managed Grafana console at <https://console.aws.amazon.com/grafana/>.
2. In the navigation pane, choose the menu icon.
3. Choose **Grafana Enterprise license**.
4. Choose **Cancel auto renew**.
5. Read the information in the box and choose **Cancel auto renew**.

This cancels your subscription to Grafana Enterprise for the next billing cycle.

6. Remove the Grafana Enterprise license from each workspace by following the steps in the previous procedure. This immediately stops incurring Grafana Enterprise per-user fees per workspace.

# Security in Amazon Managed Grafana

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 Grafana, 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 Amazon Managed Grafana. The following topics show you how to configure Amazon Managed Grafana to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your Amazon Managed Grafana resources.

## Topics

- [Data protection in AWS](#) (p. 373)
- [Identity and Access Management for Amazon Managed Grafana](#) (p. 374)
- [Amazon Managed Grafana permissions and policies for AWS data sources and notification channels](#) (p. 390)
- [IAM permissions](#) (p. 397)
- [Compliance Validation for Amazon Managed Grafana](#) (p. 398)
- [Resilience in Amazon Managed Grafana](#) (p. 399)
- [Infrastructure Security in Amazon Managed Grafana](#) (p. 399)
- [Logging Amazon Managed Grafana API calls using AWS CloudTrail](#) (p. 399)
- [Security best practices](#) (p. 415)

## Data protection in AWS

Amazon Managed Grafana 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 Amazon Managed Grafana or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into Amazon Managed Grafana 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 protection in Amazon Managed Grafana

Amazon Managed Grafana collects and stores the following types of data:

- Customer-provided dashboard and alert configurations for Grafana workspaces.
- Grafana dashboard snapshots that you have saved to your workspace.
- A list of SSO users that have been granted access to the Grafana workspace, including the user names and email addresses of the users.

The data that Amazon Managed Grafana stores is encrypted with AWS Key Management Service. Data in transit is automatically encrypted with Secure Sockets Layer (SSL).

# Identity and Access Management for Amazon Managed Grafana

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 Amazon Managed Grafana resources. IAM is an AWS service that you can use with no additional charge.

### Topics

- [Audience \(p. 375\)](#)
- [Authenticating with identities \(p. 375\)](#)
- [Managing access using policies \(p. 377\)](#)
- [How Amazon Managed Grafana works with IAM \(p. 378\)](#)
- [Identity-based policy examples for Amazon Managed Grafana \(p. 383\)](#)
- [Troubleshooting Amazon Managed Grafana identity and access \(p. 388\)](#)

## Audience

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

**Service user** – If you use the Amazon Managed Grafana service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more Amazon Managed Grafana 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 Amazon Managed Grafana, see [Troubleshooting Amazon Managed Grafana identity and access](#) (p. 388).

**Service administrator** – If you're in charge of Amazon Managed Grafana resources at your company, you probably have full access to Amazon Managed Grafana. It's your job to determine which Amazon Managed Grafana 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 Amazon Managed Grafana, see [How Amazon Managed Grafana works with IAM](#) (p. 378).

**IAM administrator** – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to Amazon Managed Grafana. To view example Amazon Managed Grafana identity-based policies that you can use in IAM, see [Identity-based policy examples for Amazon Managed Grafana](#) (p. 383).

## 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 Grafana](#) in the *Service Authorization Reference*.
- **Service role** – A service role is an [IAM role](#) that a service assumes to perform actions on your behalf. 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 Grafana works with IAM

Before you use IAM to manage access to Amazon Managed Grafana, learn what IAM features are available to use with Amazon Managed Grafana.

### IAM features you can use with Amazon Managed Grafana

IAM feature	Amazon Managed Grafana support
<a href="#">Identity-based policies</a> (p. 379)	Yes

IAM feature	Amazon Managed Grafana support
<a href="#">Resource-based policies (p. 379)</a>	No
<a href="#">Policy actions (p. 380)</a>	Yes
<a href="#">Policy resources (p. 380)</a>	No
<a href="#">Policy condition keys (p. 381)</a>	No
<a href="#">ACLs (p. 381)</a>	No
<a href="#">ABAC (tags in policies) (p. 382)</a>	No
<a href="#">Temporary credentials (p. 382)</a>	Yes
<a href="#">Principal permissions (p. 382)</a>	Yes
<a href="#">Service roles (p. 383)</a>	Yes
<a href="#">Service-linked roles (p. 383)</a>	No

To get a high-level view of how Amazon Managed Grafana 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 Amazon Managed Grafana

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 Amazon Managed Grafana

To view examples of Amazon Managed Grafana identity-based policies, see [Identity-based policy examples for Amazon Managed Grafana \(p. 383\)](#).

## Resource-based policies within Amazon Managed Grafana

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 Amazon Managed Grafana

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 Amazon Managed Grafana actions, see [Actions Defined by Amazon Managed Grafana](#) in the *Service Authorization Reference*.

Policy actions in Amazon Managed Grafana use the following prefix before the action:

```
grafana
```

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

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

To view examples of Amazon Managed Grafana identity-based policies, see [Identity-based policy examples for Amazon Managed Grafana \(p. 383\)](#).

## Policy resources for Amazon Managed Grafana

Supports policy resources	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 **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 Amazon Managed Grafana resource types and their ARNs, see [Resources defined by Amazon Managed Grafana](#) in the *Service Authorization Reference*. To learn with which actions you can specify the ARN of each resource, see [Actions defined by Amazon Managed Grafana](#).

To view examples of Amazon Managed Grafana identity-based policies, see [Identity-based policy examples for Amazon Managed Grafana \(p. 383\)](#).

## Policy condition keys for Amazon Managed Grafana

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 Amazon Managed Grafana condition keys, see [Condition keys for Amazon Managed Grafana](#) in the *Service Authorization Reference*. To learn with which actions and resources you can use a condition key, see [Actions defined by Amazon Managed Grafana](#).

To view examples of Amazon Managed Grafana identity-based policies, see [Identity-based policy examples for Amazon Managed Grafana \(p. 383\)](#).

## Access control lists (ACLs) in Amazon Managed Grafana

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 Amazon Managed Grafana

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 Amazon Managed Grafana

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](#).

## Cross-service principal permissions for Amazon Managed Grafana

Supports principal permissions	Yes
--------------------------------	-----

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 Grafana](#) in the *Service Authorization Reference*.

## Service roles for Amazon Managed Grafana

Supports service roles	Yes
------------------------	-----

A service role is an [IAM role](#) that a service assumes to perform actions on your behalf. 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 Amazon Managed Grafana functionality. Edit service roles only when Amazon Managed Grafana provides guidance to do so.

## Service-linked roles for Amazon Managed Grafana

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 Grafana

By default, IAM users and roles don't have permission to create or modify Amazon Managed Grafana 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](#) (p. 383)
- [Using the Amazon Managed Grafana console](#) (p. 384)
- [Sample policies for Amazon Managed Grafana](#) (p. 384)
- [Contents of AWS-managed policies for Amazon Managed Grafana](#) (p. 386)

## Policy best practices

Identity-based policies are very powerful. They determine whether someone can create, access, or delete Amazon Managed Grafana 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 Amazon Managed Grafana 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 Amazon Managed Grafana console

To access the console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the 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.

## Sample policies for Amazon Managed Grafana

This section contains identity-based policies that are useful for several Amazon Managed Grafana scenarios.

### Grafana administrator using SAML

If you use SAML for your user authentication, the administrator who creates and manages Amazon Managed Grafana needs the following policies:

- **AWSGrafanaAccountAdministrator** or the equivalent permissions to create and manage Amazon Managed Grafana workspaces.
- The **AWSMarketplaceManageSubscriptions** policy or equivalent permissions, if you want to upgrade an Amazon Managed Grafana workspace to Grafana Enterprise.

### Grafana administrator in a management account using AWS SSO

To grant an IAM user or an IAM role the permissions to create and manage Amazon Managed Grafana workspaces across an entire organization, and to enable dependencies such as AWS SSO, assign the **AWSGrafanaAccountAdministrator**, **AWSSSOMasterAccountAdministrator** and the **AWSSSODirectoryAdministrator** policies to that IAM user or IAM role. Additionally, to upgrade an Amazon Managed Grafana workspace to Grafana Enterprise, a user must have the **AWSMarketplaceManageSubscriptions** IAM policy or the equivalent permissions.

If you want to use service-managed permissions when you create an Amazon Managed Grafana workspace, the user who creates the workspace must also have the `iam:CreateRole`, `iam:CreatePolicy`, and `iam:AttachRolePolicy` permissions. These are required to use AWS CloudFormation StackSets to deploy policies that enable you to read data sources in the organization's accounts.



### Important

Granting a user the `iam:CreateRole`, `iam:CreatePolicy`, and `iam:AttachRolePolicy` permissions gives that user full administrative access to your AWS account. For example, a user with these permissions can create a policy that has full permissions for all resources, and attach that policy to any role. Be very careful about who you grant these permissions to.

To see the permissions granted to **AWSGrafanaAccountAdministrator**, see [AWSGrafanaAccountAdministrator policy contents \(p. 386\)](#)

### Grafana administrator in a member account using AWS SSO

To grant permissions to create and manage Amazon Managed Grafana workspaces in the member account of an organization, assign the **AWSGrafanaAccountAdministrator**, **AWSSSOMemberAccountAdministrator** and the **AWSSSODirectoryAdministrator** policies to that IAM user or IAM role. Additionally, to upgrade an Amazon Managed Grafana workspace to Grafana Enterprise, a user must have the **AWSMarketplaceManageSubscriptions** IAM policy or the equivalent permissions.

If you want to use service-managed permissions when you create an Amazon Managed Grafana workspace, the user who creates the workspace must also have the `iam:CreateRole`, `iam:CreatePolicy`, and `iam:AttachRolePolicy` permissions. These are required to enable the user to read data sources in the account.

### Important

Granting a user the `iam:CreateRole`, `iam:CreatePolicy`, and `iam:AttachRolePolicy` permissions gives that user full administrative access to your AWS account. For example, a user with these permissions can create a policy that has full permissions for all resources, and attach that policy to any role. Be very careful about who you grant these permissions to.

To see the permissions granted to **AWSGrafanaAccountAdministrator**, see [AWSGrafanaAccountAdministrator policy contents \(p. 386\)](#)

### Create and manage Amazon Managed Grafana workspaces and users in a single standalone account using AWS SSO

A standalone AWS account is an account that is not yet a member of an organization. For more information about organizations, see [What is AWS Organizations?](#)

To grant an IAM user or an IAM role permission to create and manage Amazon Managed Grafana workspaces and users in a standalone account, assign the **AWSGrafanaAccountAdministrator**, **AWSSSOMemberAccountAdministrator** and the **AWSSSODirectoryAdministrator** policies to that IAM user or IAM role. Additionally, to upgrade an Amazon Managed Grafana workspace to Grafana Enterprise, a user must have the **AWSMarketplaceManageSubscriptions** IAM policy or the equivalent permissions.

### Important

Granting a user the `iam:CreateRole`, `iam:CreatePolicy`, and `iam:AttachRolePolicy` permissions gives that user full administrative access to your AWS account. For example, a user with these permissions can create a policy that has full permissions for all resources, and attach that policy to any role. Be very careful about who you grant these permissions to.

To see the permissions granted to **AWSGrafanaAccountAdministrator**, see [AWSGrafanaAccountAdministrator policy contents \(p. 386\)](#)

### Assign and unassign users access to Amazon Managed Grafana

To grant permissions to an IAM user or an IAM role to manage other users' access to Amazon Managed Grafana workspaces in the account, including granting Grafana admin permissions to those users for the workspaces, assign the **AWSGrafanaWorkspacePermissionManagement** policy to that user. If you are using AWS SSO to manage users in this workspace, the user also needs the **AWSSSORedOnly** and **AWSSSODirectoryReadOnly** policies.



To see the permissions granted to **AWSGrafanaWorkspacePermissionManagement**, see [AWSGrafanaWorkspacePermissionManagement policy contents \(p. 387\)](#)

### Amazon Managed Grafana read-only permissions

To grant an IAM user or IAM role the permissions for read actions, such as listing and viewing workspaces and opening the Grafana workspace console, assign the **AWSGrafanaConsoleReadOnlyAccess**, **AWSSSORedOnly** and **AWSSSODirectoryReadOnly** policies to that IAM user or IAM role.

To see the permissions granted to **AWSGrafanaConsoleReadOnlyAccess**, see [AWSGrafanaConsoleReadOnlyAccess policy contents \(p. 387\)](#).

## Contents of AWS-managed policies for Amazon Managed Grafana

This section lists the built-in IAM policies for Amazon Managed Grafana.

### AWSGrafanaAccountAdministrator policy contents

The contents of the **AWSGrafanaAccountAdministrator** policy is the following.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSGrafanaOrganizationAdmin",
      "Effect": "Allow",
      "Action": [
        "iam:ListRoles"
      ],
      "Resource": "*"
    },
    {
      "Sid": "GrafanaIAMGetRolePermission",
      "Effect": "Allow",
      "Action": "iam:GetRole",
      "Resource": "arn:aws:iam::*:role/*"
    },
    {
      "Sid": "AWSGrafanaPermissions",
      "Effect": "Allow",
      "Action": [
        "grafana:*"
      ],
      "Resource": "arn:aws:grafana:*:*/workspaces*"
    },
    {
      "Sid": "GrafanaIAMPassRolePermission",
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::*:role/*",
      "Condition": {
        "StringLike": {
          "iam:PassedToService": "grafana.amazonaws.com"
        }
      }
    },
    {
      "Sid": "SSOSLRPermission",
      "Effect": "Allow",
      "Action": [
```

```
        "iam:CreateServiceLinkedRole"
      ],
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "sso.amazonaws.com"
        }
      },
      "Resource": "arn:aws:iam::*:role/aws-service-role/sso.amazonaws.com/AWSServiceRoleForSSO"
    ]
  }
}
```

## AWSGrafanaWorkspacePermissionManagement policy contents

The **AWSGrafanaWorkspacePermissionManagement** policy includes the following permissions.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSGrafanaPermissions",
      "Effect": "Allow",
      "Action": [
        "grafana:DescribeWorkspace",
        "grafana:DescribeWorkspaceAuthentication",
        "grafana:UpdatePermissions",
        "grafana:ListPermissions",
        "grafana:ListWorkspaces"
      ],
      "Resource": "arn:aws:grafana:*:*/workspaces*"
    }
  ]
}
```

## AWSGrafanaConsoleReadOnlyAccess policy contents

The **AWSGrafanaConsoleReadOnlyAccess** policy includes the following permissions.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSGrafanaConsoleReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "grafana:Describe*",
        "grafana:List*"
      ],
      "Resource": "arn:aws:grafana:*:*/workspaces*"
    }
  ]
}
```

## Amazon Managed Grafana updates to AWS managed policies

View details about updates to AWS managed policies for Amazon Managed Grafana since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the CloudWatch Document history page.

Change	Description	Date
<a href="#">AWSGrafanaWorkspacePermissionManagement</a> – Update to an existing policy	<p>Amazon Managed Grafana added new permissions to <b>AWSGrafanaWorkspacePermissionManagement</b> so that users with this policy can see the authentication methods associated with workspaces.</p> <p>The <code>grafana:DescribeWorkspaceAuthentication</code> permission was added.</p>	September 21, 2021
<a href="#">AWSGrafanaConsoleReadOnlyAccess</a> – Update to an existing policy	<p>Amazon Managed Grafana added new permissions to <b>AWSGrafanaConsoleReadOnlyAccess</b> so that users with this policy can see the authentication methods associated with workspaces.</p> <p>The <code>grafana:Describe*</code> and <code>grafana:List*</code> permissions were added to the policy, and they replace the previous narrower permissions <code>grafana:DescribeWorkspace</code>, <code>grafana:ListPermissions</code>, and <code>grafana:ListWorkspaces</code>.</p>	September 21, 2021
Amazon Managed Grafana started tracking changes	Amazon Managed Grafana started tracking changes for its AWS managed policies.	September 9, 2021

## Troubleshooting Amazon Managed Grafana identity and access

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

### Topics

- [I am not authorized to perform an action in Amazon Managed Grafana \(p. 389\)](#)
- [I am not authorized to perform iam:PassRole \(p. 389\)](#)
- [I want to view my access keys \(p. 389\)](#)
- [I'm an administrator and want to allow others to access Amazon Managed Grafana \(p. 390\)](#)
- [I want to allow people outside of my AWS account to access my Amazon Managed Grafana resources \(p. 390\)](#)

## I am not authorized to perform an action in Amazon Managed Grafana

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 `grafana:GetWidget` permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
grafana: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 `grafana: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 Amazon Managed Grafana.

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 Amazon Managed Grafana. 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 Amazon Managed Grafana

To allow others to access Amazon Managed Grafana, 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 Amazon Managed Grafana.

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 Amazon Managed Grafana 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 Amazon Managed Grafana supports these features, see [How Amazon Managed Grafana works with IAM](#) (p. 378).
- 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*.

# Amazon Managed Grafana permissions and policies for AWS data sources and notification channels

Amazon Managed Grafana offers three permission modes:

- Service-managed permissions for current account
- Service-managed permissions for organizations
- Customer-managed permissions

When you create a workspace, you choose which permission mode to use. You can also change this later if you want.

In either of the service-managed permission modes, Amazon Managed Grafana creates roles and policies that are needed to access and discover AWS data sources in your account or organization. You can then edit these policies in the IAM console if you choose.

## Service-managed permissions for a single account

In this mode, Amazon Managed Grafana creates a role called **AmazonGrafanaServiceRole-*random-id***. Amazon Managed Grafana then attaches a policy to this role for each AWS service that you select to access from the Amazon Managed Grafana workspace.

### CloudWatch

Amazon Managed Grafana creates a customer-managed policy with the name **AmazonGrafanaCloudWatchPolicy-*random-id***. The contents of the policy are as follows:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowReadingMetricsFromCloudWatch",
      "Effect": "Allow",
      "Action": [
        "cloudwatch:DescribeAlarmsForMetric",
        "cloudwatch:DescribeAlarmHistory",
        "cloudwatch:DescribeAlarms",
        "cloudwatch:ListMetrics",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:GetMetricData"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowReadingLogsFromCloudWatch",
      "Effect": "Allow",
      "Action": [
        "logs:DescribeLogGroups",
        "logs:GetLogGroupFields",
        "logs:StartQuery",
        "logs:StopQuery",
        "logs:GetQueryResults",
        "logs:GetLogEvents"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowReadingTagsInstancesRegionsFromEC2",
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeTags",
        "ec2:DescribeInstances",
        "ec2:DescribeRegions"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AllowReadingResourcesForTags",
      "Effect": "Allow",
      "Action": "tag:GetResources",
      "Resource": "*"
    }
  ]
}
```

### Amazon OpenSearch Service

Amazon Managed Grafana creates a customer-managed policy with the name **AmazonGrafanaOpenSearchPolicy-*random-id***. The Get/Post permissions are needed for data

source access. The List/Describe permissions are used by Amazon Managed Grafana for data source discovery, but they aren't required for the data source plugin to work. The contents of the policy are as follows:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "es:ESHttpGet",
        "es:DescribeElasticsearchDomains",
        "es:ListDomainNames"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "es:ESHttpPost",
      "Resource": [
        "arn:aws:es::*:domain/*/_msearch*",
        "arn:aws:es::*:domain/*/_opendistro/_ppl"
      ]
    }
  ]
}
```

### AWS IoT SiteWise

Amazon Managed Grafana attaches the AWS managed policy **AWSIoTSiteWiseReadOnlyAccess**.

### Amazon Managed Service for Prometheus

Amazon Managed Grafana creates a customer-managed policy with the name **AmazonGrafanaPrometheusPolicy-*random-id***. The List/Describe permissions are used by Amazon Managed Grafana for data source discovery, they aren't required for the plugin to work. The contents of the policy are as follows:

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

### Amazon SNS

Amazon Managed Grafana creates a customer-managed policy with the name **AmazonGrafanaSNSPolicy-*random-id***. The policy restricts you to only using SNS topics in your account that start with the string grafana. This is not necessary if you create your own policy. The contents of the policy are as follows:

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

### Timestream

Amazon Managed Grafana attaches the AWS managed policy **AmazonTimestreamReadOnlyAccess**.

### X-Ray

Amazon Managed Grafana attaches the AWS managed policy **AmazonXrayReadOnlyAccess**.

## Service-managed permissions for an organization

This mode is supported only for workspaces created in management accounts or delegated administrator accounts in an organization. For more information about delegated administrator accounts, see [Register a delegated administrator](#).

### Note

Creating resources such as Amazon Managed Grafana workspaces in the management account of an organization is against AWS security best practices.

In this mode, Amazon Managed Grafana creates all the IAM roles that are necessary to access AWS resources in other accounts in your AWS organization. In each account in the Organizational Units that you select, Amazon Managed Grafana creates a role called **AmazonGrafanaOrgMemberRole-*random-id***. This role creation is performed through an integration with AWS CloudFormation StackSets.

This role will have a policy attached for each AWS data source that you select to use in the workspace. For the contents of these data policies, see [Service-managed permissions for a single account](#) (p. 391).

Amazon Managed Grafana also creates a role called **AmazonGrafanaOrgAdminRole-*random-id*** in the organization's management account. This role allows the Amazon Managed Grafana workspace permission to access other accounts in the organization. AWS service notification channel policies will also get attached to this role. Use the **AWS Data Source** menu in your workspace to quickly provision data sources for each account that your workspace can access

To use this mode, you must enable AWS CloudFormation Stacksets as a trusted service in your AWS organization. For more information, see [Enable trusted access with AWS Organizations](#).

Here is the content of the **AmazonGrafanaStackSet-*random-id*** stack set:

```
Parameters:
  IncludePrometheusPolicy:
    Description: Whether to include Amazon Prometheus access in the role
    Type: String
    AllowedValues:
      - true
      - false
```



```
    Default: false
  IncludeAESPolicies:
    Description: Whether to include Amazon Elasticsearch access in the role
    Type: String
    AllowedValues:
      - true
      - false
    Default: false
  IncludeCloudWatchPolicy:
    Description: Whether to include CloudWatch access in the role
    Type: String
    AllowedValues:
      - true
      - false
    Default: false
  IncludeTimestreamPolicy:
    Description: Whether to include Amazon Timestream access in the role
    Type: String
    AllowedValues:
      - true
      - false
    Default: false
  IncludeXrayPolicy:
    Description: Whether to include AWS X-Ray access in the role
    Type: String
    AllowedValues:
      - true
      - false
    Default: false
  IncludeSiteWisePolicy:
    Description: Whether to include AWS IoT SiteWise access in the role
    Type: String
    AllowedValues:
      - true
      - false
    Default: false
  RoleName:
    Description: Name of the role to create
    Type: String
  AdminAccountId:
    Description: Account ID of the Amazon Grafana org admin
    Type: String
  Conditions:
    addPrometheus: !Equals [!Ref IncludePrometheusPolicy, true]
    addAES: !Equals [!Ref IncludeAESPolicies, true]
    addCloudWatch: !Equals [!Ref IncludeCloudWatchPolicy, true]
    addTimestream: !Equals [!Ref IncludeTimestreamPolicy, true]
    addXray: !Equals [!Ref IncludeXrayPolicy, true]
    addSiteWise: !Equals [!Ref IncludeSiteWisePolicy, true]

  Resources:
    PrometheusPolicy:
      Type: AWS::IAM::Policy
      Condition: addPrometheus
      Properties:
        Roles:
          - !Ref GrafanaMemberServiceRole
        PolicyName: AmazonGrafanaPrometheusPolicy
        PolicyDocument:
          Version: '2012-10-17'
          Statement:
            - Effect: Allow
              Action:
                - aps:QueryMetrics
                - aps:GetLabels
                - aps:GetSeries
```

```
    - aps:GetMetricMetadata
    - aps:ListWorkspaces
    - aps:DescribeWorkspace
  Resource: '*'

AESPPolicy:
  Type: AWS::IAM::Policy
  Condition: addAES
  Properties:
    Roles:
      - !Ref GrafanaMemberServiceRole
    PolicyName: AmazonGrafanaElasticsearchPolicy
    PolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Sid: AllowReadingESDomains
          Effect: Allow
          Action:
            - es:ESHttpGet
            - es:ESHttpPost
            - es:ListDomainNames
            - es:DescribeElasticsearchDomains
          Resource: '*'

CloudWatchPolicy:
  Type: AWS::IAM::Policy
  Condition: addCloudWatch
  Properties:
    Roles:
      - !Ref GrafanaMemberServiceRole
    PolicyName: AmazonGrafanaCloudWatchPolicy
    PolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Sid: AllowReadingMetricsFromCloudWatch
          Effect: Allow
          Action:
            - cloudwatch:DescribeAlarmsForMetric
            - cloudwatch:DescribeAlarmHistory
            - cloudwatch:DescribeAlarms
            - cloudwatch:ListMetrics
            - cloudwatch:GetMetricStatistics
            - cloudwatch:GetMetricData
          Resource: "*"
        - Sid: AllowReadingLogsFromCloudWatch
          Effect: Allow
          Action:
            - logs:DescribeLogGroups
            - logs:GetLogGroupFields
            - logs:StartQuery
            - logs:StopQuery
            - logs:GetQueryResults
            - logs:GetLogEvents
          Resource: "*"
        - Sid: AllowReadingTagsInstancesRegionsFromEC2
          Effect: Allow
          Action:
            - ec2:DescribeTags
            - ec2:DescribeInstances
            - ec2:DescribeRegions
          Resource: "*"
        - Sid: AllowReadingResourcesForTags
          Effect: Allow
          Action:
            - tag:GetResources
          Resource: "*"

```

```
GrafanaMemberServiceRole:
  Type: 'AWS::IAM::Role'
  Properties:
    RoleName: !Ref RoleName
    AssumeRolePolicyDocument:
      Version: '2012-10-17'
      Statement:
        - Effect: Allow
          Principal:
            AWS: !Sub arn:aws:iam::${AdminAccountId}:root
          Action:
            - 'sts:AssumeRole'
      Path: /service-role/
    ManagedPolicyArns:
      - !If [addTimestream, arn:aws:iam::aws:policy/AmazonTimestreamReadOnlyAccess, !Ref
AWS::NoValue]
      - !If [addXray, arn:aws:iam::aws:policy/AWSXrayReadOnlyAccess, !Ref AWS::NoValue]
      - !If [addSitewise, arn:aws:iam::aws:policy/AWSIoTSiteWiseReadOnlyAccess, !Ref
AWS::NoValue]
```

Here is the content of **AmazonGrafanaOrgAdminPolicy-*random-id***.

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": [
      "organizations:ListAccountsForParent",
      "organizations:ListOrganizationalUnitsForParent"
    ],
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "aws:PrincipalOrgID": "o-organizationId"
      }
    }
  },
  {
    "Effect": "Allow",
    "Action": [
      "sts:AssumeRole"
    ],
    "Resource": "arn:aws:iam::*:role/service-role/AmazonGrafanaOrgMemberRole-random-
Id"
  }]
}
```

## Customer-managed permissions

If you choose to use customer-managed permissions, you specify an existing IAM role in your account when you create an Amazon Managed Grafana workspace. The role must have a trust policy which trusts `grafana.amazonaws.com`.

The following is an example of such a policy:

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

```
    },
    "Action": "sts:AssumeRole"
  }
]
```

For that role to access AWS data sources or notification channels in that account, it must have the permissions in the policies listed earlier in this section. For example, to use the CloudWatch data source, it must have the permissions in the CloudWatch policy listed in [Service-managed permissions for a single account](#) (p. 391).

The `List` and `Describe` permissions in the policies for Amazon OpenSearch Service and Amazon Managed Service for Prometheus shown in [Service-managed permissions for a single account](#) (p. 391) are only needed for the data source discovery and provisioning to work correctly. They aren't needed if you just want to set up these data sources manually.

### Cross-account access

When a workspace is created in account 111111111111, a role in account 111111111111 must be supplied. For this example, we will call this role *WorkspaceRole*. To access data in account 999999999999, you must create a role in account 999999999999, which we will call *DataSourceRole*. You must then establish a trust relationship between *WorkspaceRole* and *DataSourceRole*. For more information about establishing trust between two roles, see [IAM Tutorial: Delegate access across AWS accounts using IAM roles](#).

*DataSourceRole* needs to contain the policy statements listed earlier in this section for each data source that you want to use. After the trust relationship is established, you can specify the ARN of *DataSourceRole* (arn:aws:iam::999999999999:role:DataSourceRole) in the **Assume Role ARN** field on the data source configuration page of any AWS data source in your workspace. The data source will then access account 999999999999 with the permissions that are defined in *DataSourceRole*.

## IAM permissions

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

## Amazon Managed Grafana permissions

The following table displays possible Amazon Managed Grafana actions and their required permissions:

Action	Required permission
Create an Amazon Managed Grafana workspace. A workspace is a logically isolated Grafana server used to create and visualize metrics, logs, and traces.	grafana:CreateWorkspace
Delete an Amazon Managed Grafana workspace.	grafana:DeleteWorkspace
Retrieve detailed information about an Amazon Managed Grafana workspace.	grafana:DescribeWorkspace

Action	Required permission
Retrieve the authentication configuration associated with a workspace.	<code>grafana:DescribeWorkspaceAuthentication</code>
Retrieve a list of permissions associated with workspace users and groups.	<code>grafana:ListPermissions</code>
Retrieve a list of the Amazon Managed Grafana workspaces that exist in the account.	<code>grafana:ListWorkspaces</code>
Update the permissions associated with workspace users and groups.	<code>grafana:UpdatePermissions</code>
Update Amazon Managed Grafana workspaces.	<code>grafana:UpdateWorkspace</code>
Update the authentication configuration associated with a workspace.	<code>grafana:UpdateWorkspaceAuthentication</code>
Associate a Grafana enterprise license with a workspace.	<code>grafana:AssociateLicense</code>

## Compliance Validation for Amazon Managed Grafana

Third-party auditors assess the security and compliance of AWS services as part of multiple AWS compliance programs, such as SOC, PCI, FedRAMP, and HIPAA.

To learn whether Amazon Managed Grafana 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 Grafana

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, Amazon Managed Grafana offers several features to help support your data resiliency and backup needs.

## Infrastructure Security in Amazon Managed Grafana

As a managed service, Amazon Managed Grafana 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 Amazon Managed Grafana through the network. Clients must support Transport Layer Security (TLS) 1.0 or later. We recommend 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.

## Logging Amazon Managed Grafana API calls using AWS CloudTrail

Amazon Managed Grafana is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, a role, or an AWS service in Amazon Managed Grafana. CloudTrail captures all API calls for Amazon Managed Grafana as events. The calls that are captured include calls from the Amazon Managed Grafana console and code calls to the Amazon Managed Grafana API operations. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for Amazon Managed Grafana. 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 Amazon Managed Grafana and additional details, such as the IP address that the request was made from, who made the request, and when it was made.

Amazon Managed Grafana also captures some calls that use Grafana APIs. The calls captured are those that change data, such as calls that create, update, or delete resources. For more information about Grafana APIs that are supported in Amazon Managed Grafana, see [Using Grafana HTTP APIs \(p. 292\)](#).

To learn more about CloudTrail, see the [AWS CloudTrail User Guide](#).

## Amazon Managed Grafana information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When activity occurs in Amazon Managed Grafana, 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 Amazon Managed Grafana, 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 Amazon Managed Grafana actions in the following list are logged by CloudTrail. These APIs are not yet publicly available for you to use programmatically. They will be available in a future release. CloudTrail logs the following API actions that result from actions that you take in the AMG console.

- AssociateLicense
- CreateWorkspace
- DeleteWorkspace
- DescribeWorkspace
- DisassociateLicense
- ListPermissions
- ListWorkspaces
- UpdatePermissions
- UpdateWorkspace

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 AWS account root user credentials 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 Amazon Managed Grafana 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 for a CreateWorkspace action.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "ANPAJ2UCCR6DPCEXAMPLE:sdbt-Isengard",
    "arn": "arn:aws:sts::123456789012:assumed-role/Admin/sdbt-Isengard",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "ANPAJ2UCCR6DPCEXAMPLE",
        "arn": "arn:aws:iam::123456789012:role/Admin",
        "accountId": "123456789012",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2020-11-26T20:59:21Z"
      }
    }
  },
  "eventTime": "2020-11-26T21:10:48Z",
  "eventSource": "grafana.amazonaws.com",
  "eventName": "CreateWorkspace",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "205.251.233.179",
  "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:82.0) Gecko/20100101 Firefox/82.0",
  "requestParameters": {
    "permissionType": "Service Managed",
    "workspaceNotificationDestinations": [
      "SNS"
    ],
    "workspaceDescription": "",
    "clientToken": "12345678-abcd-1234-5678-111122223333",
    "workspaceDataSources": [
      "SITEWISE",
      "XRAY",
      "CLOUDWATCH",
      "ELASTICSEARCH",
      "PROMETHEUS",
      "TIMESTREAM"
    ],
    "accountAccessType": "CURRENT_ACCOUNT",
    "workspaceName": "CloudTrailTest",
    "workspaceRoleArn": "arn:aws:iam::123456789012:role/service-role/AmazonGrafanaServiceRole-2705976ol"
  },
  "responseElements": {
    "Access-Control-Expose-Headers": "x-amzn-RequestId,x-amzn-ErrorType,x-amzn-ErrorMessage,Date",
    "workspace": {
      "accountAccessType": "CURRENT_ACCOUNT",
      "created": 1606425045.22,
      "dataSources": [
        "SITEWISE",
        "XRAY",
        "CLOUDWATCH",
```



```
        "ELASTICSEARCH",
        "PROMETHEUS",
        "TIMESTREAM"
    ],
    "description": "",
    "grafanaVersion": "7.3.1",
    "id": "g-a187c473d3",
    "modified": 1606425045.22,
    "name": "CloudTrailTest",
    "notificationDestinations": [
        "SNS"
    ],
    "permissionType": "Service Managed",
    "status": "CREATING",
    "workspaceRoleArn": "arn:aws:iam::123456789012:role/service-role/AmazonGrafanaServiceRole-2705976ol"
  }
},
"requestID": "12345678-5533-4e10-b486-e9c7b219f2fd",
"eventID": "12345678-2710-4359-ad90-b902dbfb606b",
"readOnly": false,
"eventType": "AwsApiCall",
"managementEvent": true,
"eventCategory": "Management",
"recipientAccountId": "123456789012"
}
```

The following example shows a CloudTrail log entry for an UpdateWorkspaceAuthentication action.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAU2UJBF3NRO35YZ3GV:CODETEST_Series_GrafanaApiTestHydraCanary12-o6aeXqaXS_1090259374",
    "arn": "arn:aws:sts::332073610971:assumed-role/HydraInvocationRole-4912743f1277b7c3c67cb29518f8bc413ae/CODETEST_Series_GrafanaApiTestHydraCanary12-o6aeXqaXS_1090259374",
    "accountId": "111122223333",
    "accessKeyId": "AIDACKCEVSQ6C2EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAU2UJBF3NRO35YZ3GV",
        "arn": "arn:aws:iam::111122223333:role/HydraInvocationRole-4912743f1277b7c3c67cb29518f8bc413ae",
        "accountId": "332073610971",
        "userName": "TestInvocationRole-4912743f1277b7c3c67cb29518f8bc413ae"
      },
      "webIdFederationData": {},
      "attributes": {
        "creationDate": "2021-08-04T20:50:24Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2021-08-04T21:29:25Z",
  "eventSource": "gamma-grafana.amazonaws.com",
  "eventName": "UpdateWorkspaceAuthentication",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "34.215.72.249",
  "userAgent": "aws-internal/3 aws-sdk-java/1.11.1030 Linux/4.14.231-180.360.amzn2.x86_64 OpenJDK_64-Bit_Server_VM/11.0.11+9-LTS java/11.0.11 vendor/Amazon.com_Inc. cfg/retry-mode/legacy exec-env/AWS_Lambda_java11",
  "requestParameters": {

```

```
    "authenticationProviders": [
      "AWS_SSO",
      "SAML"
    ],
    "samlConfiguration": {
      "idpMetadata": {
        "url": "https://portal.sso.us-east-1.amazonaws.com/saml/metadata/
NjMwMDg2NDc4OTA3X2lucy1jY2E2ZGU3ZDlmYjdiM2Vh"
      }
    },
    "workspaceId": "g-84ea23c1b4"
  },
  "responseElements": {
    "authentication": {
      "awsSso": {
        "ssoClientId": "gAROCWGs9-LoqCMIQ56XyEXAMPLE"
      },
      "providers": [
        "AWS_SSO",
        "SAML"
      ],
      "saml": {
        "configuration": {
          "idpMetadata": {
            "url": "https://portal.sso.us-east-1.amazonaws.com/saml/metadata/
NjMwMDg2NDc4OTA3X2lucy1jY2E2ZGU3ZDlmYjdiM2Vh"
          },
          "loginValidityDuration": 60
        },
        "status": "CONFIGURED"
      }
    }
  },
  "requestID": "96adb1de-7fa5-487e-b6c6-6b0d4495cb71",
  "eventID": "406bc825-bc52-475c-9c91-4c0d8a07c1fa",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "111122223333",
  "eventCategory": "Management"
}
```

## Understanding Grafana API log file entries

Amazon Managed Grafana also logs some Grafana API calls in CloudTrail. The calls captured are those that change data, such as calls that create, update, or delete resources. For more information about Grafana APIs that are supported in Amazon Managed Grafana, see [Using Grafana HTTP APIs \(p. 292\)](#).

### Example CloudTrail entries from Grafana APIs

#### User signs in to Amazon Managed Grafana workspace using AWS Single Sign-On

```
{
  "Records": [
    {
      "eventVersion": "1.08",
      "userIdentity": {
        "type": "SAMLUser",
        "userName": "johndoe"
      },
      "eventTime": "2021-07-09T02:31:59Z",
      "eventSource": "grafana.amazonaws.com",
      "eventName": "login-auth.sso",
```

```
      "awsRegion": "us-west-2",
      "sourceIPAddress": "192.0.2.0,198.51.100.0",
      "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.114 Safari/537.36",
      "requestParameters": null,
      "responseElements": null,
      "eventID": "176bf326-0302-4190-8dbf-dfdf481d8198",
      "readOnly": false,
      "eventType": "AwsServiceEvent",
      "managementEvent": true,
      "eventCategory": "Management",
      "recipientAccountId": "111122223333",
      "serviceEventDetails": {
        "timestamp": "2021-07-09T02:31:59.045984031Z",
        "user": {
          "userId": 1,
          "orgId": 1,
          "name": "johndoe",
          "isAnonymous": false
        },
        "action": "login-auth.sso",
        "requestUri": "",
        "request": {
          "query": {
            "code": [
              "eyJraWQiOiJrZXktMTU2Njk2ODEyMSIsImFsZyI6IkhTMzg0In0.eyJwbGFpbnRleHQiOiJZUzEwYWhpBZUowTDlQcW5ROGFmZ",
              ],
              "state": [
                "QUFBQURtdGx1UzB4TlRZNE9UVTF0ekkyM2RUWUFUaHZHYXcyOU9ULUVaWHhNUXAwX184N25RVGVWMD0enFpVE1iWlRlRVOM0X09Ha",
                ]
              }
            },
            "result": {
              "statusType": "failure"
            },
            "ipAddress": "192.0.2.0,198.51.100.0",
            "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.114 Safari/537.36",
            "grafanaVersion": "7.5.7",
            "additionalData": {
              "GiraffeCustomerAccount": "111122223333",
              "GiraffeWorkspaceId": "g-123EXAMPLE",
              "extUserInfo": "{\"OAuthToken\":null,\"AuthModule\": \"auth.sso\",
                \"AuthId\": \"92670be4c1-e524608b-82f2-452d-a707-161c1e5f4706\", \"UserId\": 0, \"Email\": \"\",
                \"Login\": \"johndoe\", \"Name\": \"johndoe\", \"Groups\": null, \"OrgRoles\": {\"1\": \"Admin\"},
                \"IsGrafanaAdmin\": false, \"IsDisabled\": false}"
              }
            }
          }
        ]
      }
    }
```

#### Grafana API POST /api/auth/keys

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "Unknown",
    "userName": "api_key"
  },
  "eventTime": "2021-07-09T02:16:32Z",
  "eventSource": "grafana.amazonaws.com",
  "eventName": "create",
}
```

```
"awsRegion": "us-west-2",
"sourceIPAddress": "192.0.2.0,198.51.100.1",
"userAgent": "python-requests/2.24.0",
"errorCode": "200",
"requestParameters": null,
"responseElements": null,
"eventID": "157bbf19-6ba4-4704-bc3b-d3e334b3a2b8",
"readOnly": false,
"eventType": "AwsServiceEvent",
"managementEvent": true,
"eventCategory": "Management",
"recipientAccountId": "111122223333",
"serviceEventDetails": {
  "timestamp": "2021-07-09T02:16:32.419795511Z",
  "user": {
    "orgId": 1,
    "orgRole": "Admin",
    "name": "api_key",
    "apiKeyId": "23",
    "isAnonymous": false
  },
  "action": "create",
  "resources": [
    {
      "ID": 0,
      "type": "api-key"
    }
  ],
  "requestUri": "",
  "request": {
    "body": "{\"name\":\"keyname\",\"role\":\"Admin\",\"secondsToLive\":60}"
  },
  "result": {
    "statusType": "success",
    "statusCode": "200"
  },
  "ipAddress": "192.0.2.0,198.51.100.1",
  "userAgent": "python-requests/2.24.0",
  "grafanaVersion": "7.5.7",
  "additionalData": {
    "GiraffeCustomerAccount": "111122223333",
    "GiraffeWorkspaceId": "g-123EXAMPLE"
  }
}
}
```

#### Grafana API DELETE /api/auth/keys/:id

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "Unknown",
    "userName": "api_key"
  },
  "eventTime": "2021-07-09T02:16:33Z",
  "eventSource": "grafana.amazonaws.com",
  "eventName": "delete",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0,198.51.100.2",
  "userAgent": "python-requests/2.24.0",
  "errorCode": "200",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "df1aafb3-28c6-4836-a64b-4d34538edc51",
  "readOnly": false,
```

```
"eventType": "AwsServiceEvent",
"managementEvent": true,
"eventCategory": "Management",
"recipientAccountId": "111122223333",
"serviceEventDetails": {
  "timestamp": "2021-07-09T02:16:33.045041594Z",
  "user": {
    "orgId": 1,
    "orgRole": "Admin",
    "name": "api_key",
    "apiKeyId": "23",
    "isAnonymous": false
  },
  "action": "delete",
  "resources": [
    {
      "ID": 0,
      "type": "api-key"
    }
  ],
  "requestUri": "",
  "request": {
    "params": {
      "id": "24"
    }
  },
  "result": {
    "statusType": "success",
    "statusCode": "200"
  },
  "ipAddress": "192.0.2.0,198.51.100.2",
  "userAgent": "python-requests/2.24.0",
  "grafanaVersion": "7.5.7",
  "additionalData": {
    "GiraffeCustomerAccount": "111122223333",
    "GiraffeWorkspaceId": "g-123EXAMPLE"
  }
}
}
```

#### Grafana API POST /api/alerts/:id/pause

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "Unknown",
    "userName": "api_key"
  },
  "eventTime": "2021-07-09T02:16:40Z",
  "eventSource": "grafana.amazonaws.com",
  "eventName": "pause",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0,198.51.100.3",
  "userAgent": "python-requests/2.24.0",
  "errorCode": "200",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "d533a7ba-f193-45ac-a88c-75ed0594509b",
  "readOnly": false,
  "eventType": "AwsServiceEvent",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "111122223333",
  "serviceEventDetails": {
    "timestamp": "2021-07-09T02:16:40.261226856Z",
```

```
    "user": {
      "orgId": 1,
      "orgRole": "Admin",
      "name": "api_key",
      "apiKeyId": "23",
      "isAnonymous": false
    },
    "action": "pause",
    "resources": [
      {
        "ID": 0,
        "type": "alert"
      }
    ],
    "requestUri": "",
    "request": {
      "params": {
        ":alertId": "1"
      },
      "body": "{\"paused\":true}"
    },
    "result": {
      "statusType": "success",
      "statusCode": "200"
    },
    "ipAddress": "192.0.2.0,198.51.100.3",
    "userAgent": "python-requests/2.24.0",
    "grafanaVersion": "7.5.7",
    "additionalData": {
      "GiraffeCustomerAccount": "111122223333",
      "GiraffeWorkspaceId": "g-123EXAMPLE"
    }
  }
}
```

#### Grafana POST /api/alerts/test

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "Unknown",
    "userName": "api_key"
  },
  "eventTime": "2021-07-09T02:16:39Z",
  "eventSource": "grafana.amazonaws.com",
  "eventName": "test",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0,10.0.42.208",
  "userAgent": "python-requests/2.24.0",
  "errorCode": "400",
  "errorMessage": "The dashboard needs to be saved at least once before you can test an alert rule",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "7094644d-8230-4774-a092-8a128eb6dec9",
  "readOnly": false,
  "eventType": "AwsServiceEvent",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "111122223333",
  "serviceEventDetails": {
    "timestamp": "2021-07-09T02:16:39.622607860Z",
    "user": {
      "orgId": 1,
      "orgRole": "Admin",

```

```
        "name": "api_key",
        "apiKeyId": "23",
        "isAnonymous": false
    },
    "action": "test",
    "resources": [
        {
            "ID": 0,
            "type": "panel"
        }
    ],
    "requestUri": "",
    "request": {},
    "result": {
        "statusType": "failure",
        "statusCode": "400",
        "failureMessage": "The dashboard needs to be saved at least once before you
test an alert rule"
    },
    "ipAddress": "192.0.2.0, 10.0.42.208",
    "userAgent": "python-requests/2.24.0",
    "grafanaVersion": "7.5.7",
    "additionalData": {
        "GiraffeCustomerAccount": "111122223333",
        "GiraffeWorkspaceId": "g-123EXAMPLE"
    }
}
}
```

#### Grafana API POST /api/alert-notifications

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Unknown",
        "userName": "api_key"
    },
    "eventTime": "2021-07-09T02:16:40Z",
    "eventSource": "grafana.amazonaws.com",
    "eventName": "create",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0,198.51.100.0",
    "userAgent": "python-requests/2.24.0",
    "errorCode": "200",
    "requestParameters": null,
    "responseElements": null,
    "eventID": "1ce099b3-c427-4338-9f42-d38d1ef64efe",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "111122223333",
    "serviceEventDetails": {
        "timestamp": "2021-07-09T02:16:40.888295790Z",
        "user": {
            "orgId": 1,
            "orgRole": "Admin",
            "name": "api_key",
            "apiKeyId": "23",
            "isAnonymous": false
        }
    },
    "action": "create",
    "resources": [
        {
            "ID": 0,
```

```
        "type": "alert-notification"
      }
    ],
    "requestUri": "",
    "request": {
      "body": "{\"name\":\"alert notification name\",\"type\":\"Slack\"}"
    },
    "result": {
      "statusType": "success",
      "statusCode": "200"
    },
    "ipAddress": "192.0.2.0,198.51.100.0",
    "userAgent": "python-requests/2.24.0",
    "grafanaVersion": "7.5.7",
    "additionalData": {
      "GiraffeCustomerAccount": "111122223333",
      "GiraffeWorkspaceId": "g-123EXAMPLE"
    }
  }
}
```

#### Grafana API PUT /api/alert-notifications/uid/:uid

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "Unknown",
    "userName": "api_key"
  },
  "eventTime": "2021-07-09T02:16:42Z",
  "eventSource": "grafana.amazonaws.com",
  "eventName": "update",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0,198.51.100.3",
  "userAgent": "python-requests/2.24.0",
  "errorCode": "200",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "cebfef38-5007-495c-bd29-c8077797acac",
  "readOnly": false,
  "eventType": "AwsServiceEvent",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "111122223333",
  "serviceEventDetails": {
    "timestamp": "2021-07-09T02:16:42.792652648Z",
    "user": {
      "orgId": 1,
      "orgRole": "Admin",
      "name": "api_key",
      "apiKeyId": "23",
      "isAnonymous": false
    },
    "action": "update",
    "resources": [
      {
        "ID": 0,
        "type": "alert-notification"
      }
    ],
    "requestUri": "",
    "request": {
      "params": {
        ":uid": "WvDWDSinz"
      }
    }
  }
}
```



```
        "body": "{\n\"name\": \"DIFFERENT alert notification name\", \"type\": \"AWS SNS\"}"
      },
      "result": {
        "statusType": "success",
        "statusCode": "200"
      },
      "ipAddress": "192.0.2.0,198.51.100.3",
      "userAgent": "python-requests/2.24.0",
      "grafanaVersion": "7.5.7",
      "additionalData": {
        "GiraffeCustomerAccount": "111122223333",
        "GiraffeWorkspaceId": "g-123EXAMPLE"
      }
    }
  }
}
```

### Grafana API POST /api/annotations

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "Unknown",
    "userName": "api_key"
  },
  "eventTime": "2021-07-09T02:16:45Z",
  "eventSource": "grafana.amazonaws.com",
  "eventName": "create",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0,198.51.100.1",
  "userAgent": "python-requests/2.24.0",
  "errorCode": "200",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "13bf3bef-966c-4913-a760-ade365a4a08f",
  "readOnly": false,
  "eventType": "AwsServiceEvent",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "111122223333",
  "serviceEventDetails": {
    "timestamp": "2021-07-09T02:16:45.394513179Z",
    "user": {
      "orgId": 1,
      "orgRole": "Admin",
      "name": "api_key",
      "apiKeyId": "23",
      "isAnonymous": false
    },
    "action": "create",
    "resources": [
      {
        "ID": 0,
        "type": "annotation"
      }
    ],
    "requestUri": "",
    "request": {
      "body": "{\n\"dashboardId\":36,\n\"panelId\":2,\n\"tags\":[\n\"tag1\", \"tag2\"],\n\"what\":\n\"Event Name\"}"
    },
    "result": {
      "statusType": "success",
      "statusCode": "200"
    },
    "ipAddress": "192.0.2.0,198.51.100.1",
  }
}
```

```
    "userAgent": "python-requests/2.24.0",
    "grafanaVersion": "7.5.7",
    "additionalData": {
      "GiraffeCustomerAccount": "111122223333",
      "GiraffeWorkspaceId": "g-123EXAMPLE"
    }
  }
}
```

#### Grafana API DELETE /api/dashboards/uid/:uid

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "Unknown",
    "userName": "api_key"
  },
  "eventTime": "2021-07-09T02:17:09Z",
  "eventSource": "grafana.amazonaws.com",
  "eventName": "delete",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0,198.51.100.7",
  "userAgent": "python-requests/2.24.0",
  "errorCode": "200",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "d6ad9134-5fbc-403c-a76d-4ed9a81065b6",
  "readOnly": false,
  "eventType": "AwsServiceEvent",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "111122223333",
  "serviceEventDetails": {
    "timestamp": "2021-07-09T02:17:09.200112003Z",
    "user": {
      "orgId": 1,
      "orgRole": "Admin",
      "name": "api_key",
      "apiKeyId": "23",
      "isAnonymous": false
    },
    "action": "delete",
    "resources": [
      {
        "ID": 0,
        "type": "dashboard"
      }
    ],
    "requestUri": "",
    "request": {
      "params": {
        ":uid": "GLzWvIi7z"
      }
    },
    "result": {
      "statusType": "success",
      "statusCode": "200"
    },
    "ipAddress": "192.0.2.0,198.51.100.7",
    "userAgent": "python-requests/2.24.0",
    "grafanaVersion": "7.5.7",
    "additionalData": {
      "GiraffeCustomerAccount": "111122223333",
      "GiraffeWorkspaceId": "g-123EXAMPLE"
    }
  }
}
```

```
}  
}
```

#### Grafana API PUT /api/datasources/:datasourceId

```
{  
  "eventVersion": "1.08",  
  "userIdentity": {  
    "type": "Unknown",  
    "userName": "api_key"  
  },  
  "eventTime": "2021-07-09T02:16:36Z",  
  "eventSource": "grafana.amazonaws.com",  
  "eventName": "update",  
  "awsRegion": "us-west-2",  
  "sourceIPAddress": "192.0.2.0,10.0.108.94",  
  "userAgent": "python-requests/2.24.0",  
  "errorCode": "200",  
  "requestParameters": null,  
  "responseElements": null,  
  "eventID": "92877483-bdf6-44f5-803e-1ac8ad997113",  
  "readOnly": false,  
  "eventType": "AwsServiceEvent",  
  "managementEvent": true,  
  "eventCategory": "Management",  
  "recipientAccountId": "111122223333",  
  "serviceEventDetails": {  
    "timestamp": "2021-07-09T02:16:36.918660585Z",  
    "user": {  
      "orgId": 1,  
      "orgRole": "Admin",  
      "name": "api_key",  
      "apiKeyId": "23",  
      "isAnonymous": false  
    },  
    "action": "update",  
    "resources": [  
      {  
        "ID": 0,  
        "type": "datasource"  
      }  
    ],  
    "requestUri": "",  
    "request": {  
      "params": {  
        "id": "108"  
      },  
      "body": "{\"access\": \"proxy\", \"basicAuth\": false, \"name\":  
\\\"test_amp_datasource_NEW_name\\\", \"type\": \"Amazon Managed Prometheus\\\", \"url\": \"http://  
amp.amazonaws.com\"}"  
    },  
    "result": {  
      "statusType": "success",  
      "statusCode": "200"  
    },  
    "ipAddress": "192.0.2.0,10.0.108.94",  
    "userAgent": "python-requests/2.24.0",  
    "grafanaVersion": "7.5.7",  
    "additionalData": {  
      "GiraffeCustomerAccount": "111122223333",  
      "GiraffeWorkspaceId": "g-123EXAMPLE"  
    }  
  }  
}
```

### Grafana API DELETE /api/teams/:teamId/groups/:groupId

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "Unknown",
    "userName": "api_key"
  },
  "eventTime": "2021-07-09T02:17:07Z",
  "eventSource": "grafana.amazonaws.com",
  "eventName": "delete",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.0,198.51.100.2",
  "userAgent": "python-requests/2.24.0",
  "errorCode": "200",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "b41d3967-daab-44d1-994a-a437556add82",
  "readOnly": false,
  "eventType": "AwsServiceEvent",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "111122223333",
  "serviceEventDetails": {
    "timestamp": "2021-07-09T02:17:07.296142539Z",
    "user": {
      "orgId": 1,
      "orgRole": "Admin",
      "name": "api_key",
      "apiKeyId": "23",
      "isAnonymous": false
    },
    "action": "delete",
    "resources": [
      {
        "ID": 0,
        "type": "team"
      }
    ],
    "requestUri": "",
    "request": {
      "params": {
        ":groupId": "cn=editors,ou=groups,dc=grafana,dc=org",
        ":teamId": "35"
      }
    },
    "result": {
      "statusType": "success",
      "statusCode": "200"
    },
    "ipAddress": "192.0.2.0,198.51.100.2",
    "userAgent": "python-requests/2.24.0",
    "grafanaVersion": "7.5.7",
    "additionalData": {
      "GiraffeCustomerAccount": "111122223333",
      "GiraffeWorkspaceId": "g-123EXAMPLE"
    }
  }
}
```

### Grafana API PUT /api/folders/:uid

```
{
  "eventVersion": "1.08",
```

```
"userIdentity": {
  "type": "Unknown",
  "userName": "api_key"
},
"eventTime": "2021-07-09T02:16:56Z",
"eventSource": "grafana.amazonaws.com",
"eventName": "update",
"awsRegion": "us-west-2",
"sourceIPAddress": "192.0.2.0,198.51.100.1",
"userAgent": "python-requests/2.24.0",
"errorCode": "412",
"errorMessage": "the folder has been changed by someone else",
"requestParameters": null,
"responseElements": null,
"eventId": "414c98c8-aa53-45e4-940d-bea55716eaf6",
"readOnly": false,
"eventType": "AwsServiceEvent",
"managementEvent": true,
"eventCategory": "Management",
"recipientAccountId": "11112223333",
"serviceEventDetails": {
  "timestamp": "2021-07-09T02:16:56.382646826Z",
  "user": {
    "orgId": 1,
    "orgRole": "Admin",
    "name": "api_key",
    "apiKeyId": "23",
    "isAnonymous": false
  },
  "action": "update",
  "resources": [
    {
      "ID": 0,
      "type": "folder"
    }
  ],
  "requestUri": "",
  "request": {
    "params": {
      "uid": "lnsZvSi7z"
    },
    "body": "{\"title\":\"NEW Folder Name\"}"
  },
  "result": {
    "statusType": "failure",
    "statusCode": "412",
    "failureMessage": "the folder has been changed by someone else"
  },
  "ipAddress": "192.0.2.0,198.51.100.1",
  "userAgent": "python-requests/2.24.0",
  "grafanaVersion": "7.5.7",
  "additionalData": {
    "GiraffeCustomerAccount": "11112223333",
    "GiraffeWorkspaceId": "g-123EXAMPLE"
  }
}
}
```

#### Grafana API POST /api/teams

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "Unknown",
    "userName": "api_key"
```

```

    },
    "eventTime": "2021-07-09T02:17:02Z",
    "eventSource": "grafana.amazonaws.com",
    "eventName": "create",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "192.0.2.0,10.0.40.206",
    "userAgent": "python-requests/2.24.0",
    "errorCode": "200",
    "requestParameters": null,
    "responseElements": null,
    "eventID": "8d40bd79-76a8-490c-b7bb-74205253b707",
    "readOnly": false,
    "eventType": "AwsServiceEvent",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "111122223333",
    "serviceEventDetails": {
      "timestamp": "2021-07-09T02:17:02.845022379Z",
      "user": {
        "orgId": 1,
        "orgRole": "Admin",
        "name": "api_key",
        "apiKeyId": "23",
        "isAnonymous": false
      },
      "action": "create",
      "resources": [
        {
          "ID": 0,
          "type": "team"
        }
      ],
      "requestUri": "",
      "request": {
        "body": "{\"name\":\"TeamName\"}"
      },
      "result": {
        "statusType": "success",
        "statusCode": "200"
      },
      "ipAddress": "192.0.2.0,10.0.40.206",
      "userAgent": "python-requests/2.24.0",
      "grafanaVersion": "7.5.7",
      "additionalData": {
        "GiraffeCustomerAccount": "111122223333",
        "GiraffeWorkspaceId": "g-123EXAMPLE"
      }
    }
  }
}

```

## Security best practices

The topics in this section explain the best practices to follow to best maintain security in your Amazon Managed Grafana deployment.

### Use short-lived API keys

To use Grafana APIs in an Amazon Managed Grafana workspace, you must first create an API key to use for authorization. When you create the key, you specify the **Time to live** for the key, which defines how long the key is valid, up to a maximum of 30 days. We strongly recommend that you set the key's time to

live for a shorter time, such as a few hours or less. This creates much less risk than having API keys that are valid for a long time.

We also recommend that you treat API keys as passwords, in terms of securing them. For example, do not store them in plain text.

## Migrating from self-managed Grafana

This section is relevant for you if you are migrating an existing self-managed Grafana or Grafana Enterprise deployment to Amazon Managed Grafana. This applies to both on-premises Grafana and to a Grafana deployment on AWS, in your own account.

If you are running Grafana on-premises or in your own AWS account, you have likely defined users and teams and potentially organization roles to manage access. In Amazon Managed Grafana, users and groups are managed outside of Amazon Managed Grafana, using AWS SSO or directly from your identity provider (IdP) via SAML 2.0 integration. With Amazon Managed Grafana, you can assign certain permissions as necessary for carrying out a task— for example viewing dashboards. For more information about user management in Amazon Managed Grafana, see [Managing workspaces, users, and policies \(p. 19\)](#).

Additionally, when you run on-premises Grafana you're using long-lived keys or secret credentials to access data sources. We strongly recommend that when you migrate to Amazon Managed Grafana, you replace these IAM users with IAM roles. For an example, see [Manually add CloudWatch as a data source \(p. 42\)](#).

# Amazon Managed Grafana service quotas

Amazon Managed Grafana has the following quotas. Currently, you can't request increases to any of the quotas.

Resource	Default Quota
Alerts	100 per workspace.
API operations. The APIs are not yet available publicly, but the following quotas apply to API operations that result from your actions in the Amazon Managed Grafana console.	CreateWorkspace and DeleteWorkspace: 1 transaction per second (TPS), with a burst limit of 5 TPS.  DescribeWorkspace and ListWorkspaces: 5 transaction per second (TPS), with a burst limit of 10 TPS.  UpdateWorkspace, ListPermissions, and UpdatePermissions: 10 transaction per second (TPS), with a burst limit of 25 TPS.  AssociateLicense and DisassociateLicense: 1 transaction per second (TPS), with a burst limit of 5 TPS.  DescribeWorkspaceAuthentication and UpdateWorkspaceAuthentication: 1 transaction per second (TPS), with a burst limit of 5 TPS.
Dashboards	100 per workspace.
Data sources	100 per workspace.
Users	500 per workspace.
Workspaces per Region per account	5



# Document history for User Guide

The following table describes the important changes to the documentation since the last release of Amazon Managed Grafana. For notification about updates to this documentation, you can subscribe to an RSS feed.

update-history-change	update-history-description	update-history-date
<a href="#">Amazon Managed Grafana preview feature enhancements.</a>	The preview of Amazon Managed Grafana supports Grafana version 7.5 and supports upgrade to Grafana Enterprise via AWS Marketplace integration. The Amazon Elasticsearch Service data source has also been upgraded to support Open Distro for Elasticsearch.	April 16, 2021
<a href="#">Amazon Managed Grafana preview released.</a>	The preview of Amazon Managed Grafana is released.	December 15, 2020

# AWS glossary

For the latest AWS terminology, see the [AWS glossary](#) in the *AWS General Reference*.