Qumulo Alerts Administrator Guide

Version 6.1.0

Qumulo Alerts Administrator Guide

Version 6.1.0

Table of Contents

Getting Started

How Qumulo Alerts Works 3
Supported Alarms and Alerts 7
Supported Language Locales 10

Installing and Configuring 13

Upgrading

Upgrading from a Previous Version 25
Upgrading from a Beta Version 27

Configuring Notifications

Configuring Integrations

Getting Started

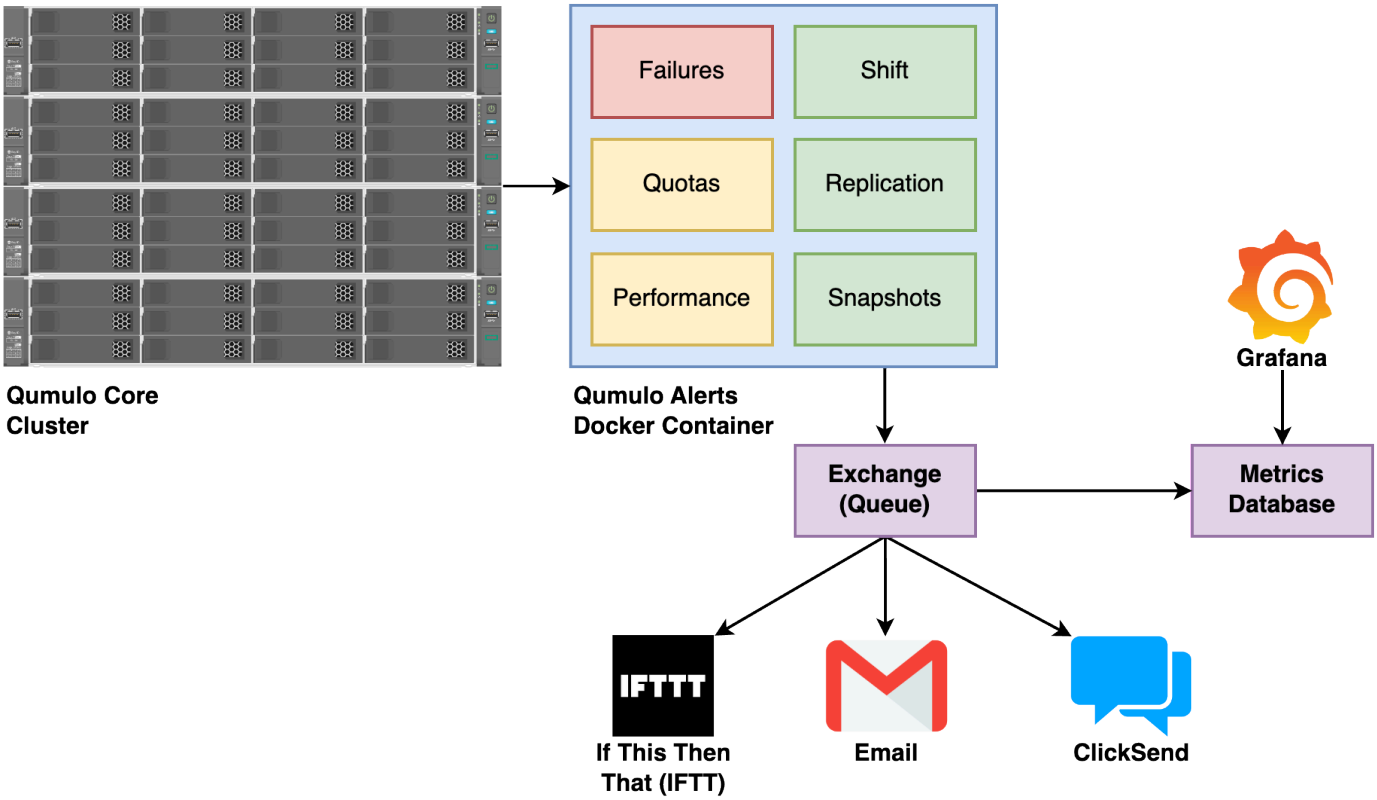
How Qumulo Alerts Works with Qumulo Core

This section explains how Qumulo Alerts monitors alarms and alerts for a Qumulo Cluster.

How Qumulo Alerts Works

Qumulo Alerts is a Docker-based system that comprises multiple containers. The main container uses a series of *plugins* to collect *hardware alarms* and *software alerts* from Qumulo clusters.

In Qumulo Alerts, *producers* are Docker containers that take data from various sources, pass it through *the Exchange*, a processing queue, and finally give the data to *consumers*, defined users or user groups. In addition to processing data, the Exchange facilitates the transfers between the producers and consumers.



Both producers and consumers use plugins that help process alarms and alerts from a Qumulo cluster. A *plugin* is a mechanism that processes a single function, such as fan failure, disk failure, or node failure. Plugins help with granular control over the information that Qumulo Alerts collects and processes.

Working with the alerts CLI

The `alerts` CLI lets you configure Qumulo Alerts. For more information, use the `--help` flag.

Qumulo Alerts includes a CLI for the following operating systems:

- Ubuntu 20 and 22
- Red Hat Enterprise (RHEL) 8
- macOS
- Windows Server 2019 and Windows 10 and 11

Known Limitations of Qumulo Alerts

This section lists the currenly known limitations for Qumulo Alerts.

- **Floating IP Addresses or Network Load Balancing (NLB):** To prevent overloading any node in a Qumulo cluster, Qumulo Alerts plugins connect to all nodes in the cluster by using floating IP addresses or an NLB.

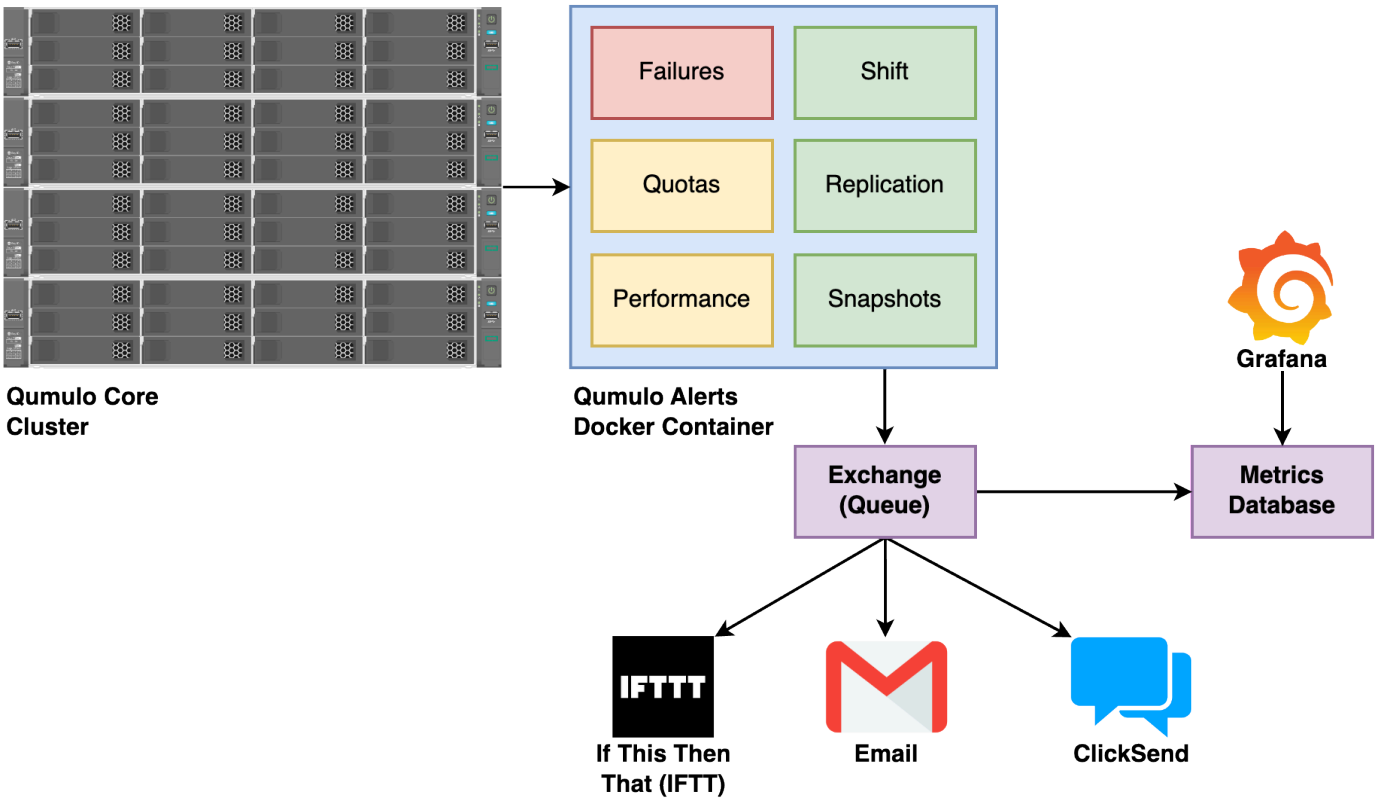
  > ⚠ Important
  > Qumulo Alerts can't function if neither IP addresses nor NLBs are configured.

- **Error Logging:** Qumulo Alerts generates a large number of error messages that can help you debug issues. However, currently, all logging remains within the Docker container and is therefore not accessible easily. For help with troubleshooting issues, contact the Qumulo Care team.

How Qumulo Alerts Works

Qumulo Alerts is a Docker-based system that comprises multiple containers. The main container uses a series of *plugins* to collect *hardware alarms* and *software alerts* from Qumulo clusters.

In Qumulo Alerts, *producers* are Docker containers that take data from various sources, pass it through *the Exchange*, a processing queue, and finally give the data to *consumers*, defined users or user groups. In addition to processing data, the Exchange facilitates the transfers between the producers and consumers.



Both producers and consumers use plugins that help process alarms and alerts from a Qumulo cluster. A *plugin* is a mechanism that processes a single function, such as fan failure, disk failure, or node failure. Plugins help with granular control over the information that Qumulo Alerts collects and processes.

Working with the alerts CLI

The `alerts` CLI lets you configure Qumulo Alerts. For more information, use the `--help` flag.

Qumulo Alerts includes a CLI for the following operating systems:

- Ubuntu 20 and 22
- Red Hat Enterprise (RHEL) 8
- macOS
- Windows Server 2019 and Windows 10 and 11

Known Limitations of Qumulo Alerts

This section lists the currenly known limitations for Qumulo Alerts.

- **Floating IP Addresses or Network Load Balancing (NLB):** To prevent overloading any node in a Qumulo cluster, Qumulo Alerts plugins connect to all nodes in the cluster by using floating IP addresses or an NLB.

> ⚠ Important
> Qumulo Alerts can't function if neither IP addresses nor NLBs are configured.

- Error Logging: Qumulo Alerts generates a large number of error messages that can help you debug issues. However, currently, all logging remains within the Docker container and is therefore not accessible easily. For help with troubleshooting issues, contact the Qumulo Care team.

What Alarms and Alerts Qumulo Alerts Supports

This section lists the alarms and alerts that Qumulo Alerts collects and processes.

Alarms

The following alarms report hardware changes in a Qumulo cluster.

Plugin Name	Description
CPU	Temperature deviation
Disks	Failure, state change
Fans	Speed deviation, failure
Network	Link failure
Nodes	Addition, failure

Alerts

The following alerts report software changes and changes in environmental conditions for a Qumulo cluster.

Plugin Name	Description
AD	Joining or leaving an Active Directory domain
Audit	Auditing enabled or disabled
Capacity	Change in cluster capacity (configured percentage of the entire cluster)
Exports	NFS exports created, modified, or deleted
FTP	FTP enabled or disabled
Groups	Local groups added, modified, or deleted
Monitoring	Cloud-based monitoring enabled, disabled, or unreachable
Quotas	Quota notification (configured percentage for specified directories)
Restriper	Restriper started, stopped, or percentage complete

Plugin Name	Description
Shares	SMB shares added, modified, or deleted
Softquotas	Soft quota notification (configured percentage for specified directories)
Users	Local users added, modified, or deleted

Informational

The following informational notifications show performance and status information for a Qumulo cluster.

Plugin Name	Description
Metrics	Performance metrics (throughput, IOPS, and latency)
OSUpgrade	Qumulo Core upgrade

Alarms

The following alarms report hardware changes in a Qumulo cluster.

Plugin Name	Description
CPU	Temperature deviation
Disks	Failure, state change
Fans	Speed deviation, failure
Network	Link failure
Nodes	Addition, failure

Alerts

The following alerts report software changes and changes in environmental conditions for a Qumulo cluster.

Plugin Name	Description
AD	Joining or leaving an Active Directory domain
Audit	Auditing enabled or disabled

Plugin Name	Description
Capacity	Change in cluster capacity (configured percentage of the entire cluster)
Exports	NFS exports created, modified, or deleted
FTP	FTP enabled or disabled
Groups	Local groups added, modified, or deleted
Monitoring	Cloud-based monitoring enabled, disabled, or unreachable
Quotas	Quota notification (configured percentage for specified directories)
Restriper	Restriper started, stopped, or percentage complete
Shares	SMB shares added, modified, or deleted
Softquotas	Soft quota notification (configured percentage for specified directories)
Users	Local users added, modified, or deleted

Informational

The following informational notifications show performance and status information for a Qumulo cluster.

Plugin Name	Description
Metrics	Performance metrics (throughput, IOPS, and latency)
OSUpgrade	Qumulo Core upgrade

What Language Locales Qumulo Alerts Supports

This section lists the language locales that Qumulo Alerts supports for notifying users through email, IFTTT, and SMS (ClickSend).

Language Locales

The consumer processes for email (page 49), IFTTT (page 52), and SMS (ClickSend) (page 54) integrations translate messages into the recipient's native language.

Code	Description
`de_AT`	German (Austria)
`de_CH`	German (Switzerland)
`de_DE`	German (Germany)
`en_GB`	English (Great Britain)
`en_US`	English (USA)
`es_ES`	Spanish (Spain)
`fr_BE`	French (Belgium)
`fr_CA`	French (Canada)
`fr_CH`	French (Switzerland)
`fr_FR`	French (France)
`hu_HU`	Hungarian (Hungary)
`it_CH`	Italian (Switzerland)
`it_IT`	Italian (Italy)
`ja_JP`	Japanese (Japan)
`ko_KR`	Korean (Korea)
`pl_PL`	Polish (Poland)
`sk_SK`	Slovak (Slovakia)

Code	Description
`tr_TR`	Turkish (Turkey)
`zh_TW`	Traditional Chinese (Taiwan)

Converting Time Zones

Each message that the Exchange processes contains a timestamp encoded in UTC time by default. This timestamp must match the recipient's time zone. If you don't use the `--timezone` flag when you create a user by using the `alerts` CLI, Qumulo Alerts uses `UTC` time.

Each translated message that a user receives includes a time zone in the `Continent/City` format (for example, `America/Los_Angeles`). For more information, see List of TZ Database Time Zones.

> **Note**
> `UTC` doesn't follow the `Continent/City` format.

Language Locales

The consumer processes for email (page 49), IFTTT (page 52), and SMS (ClickSend) (page 54) integrations translate messages into the recipient's native language.

Code	Description
`de_AT`	German (Austria)
`de_CH`	German (Switzerland)
`de_DE`	German (Germany)
`en_GB`	English (Great Britain)
`en_US`	English (USA)
`es_ES`	Spanish (Spain)
`fr_BE`	French (Belgium)
`fr_CA`	French (Canada)
`fr_CH`	French (Switzerland)
`fr_FR`	French (France)

Code	Description
`hu_HU`	Hungarian (Hungary)
`it_CH`	Italian (Switzerland)
`it_IT`	Italian (Italy)
`ja_JP`	Japanese (Japan)
`ko_KR`	Korean (Korea)
`pl_PL`	Polish (Poland)
`sk_SK`	Slovak (Slovakia)
`tr_TR`	Turkish (Turkey)
`zh_TW`	Traditional Chinese (Taiwan)

Converting Time Zones

Each message that the Exchange processes contains a timestamp encoded in UTC time by default. This timestamp must match the recipient's time zone. If you don't use the `--timezone` flag when you create a user by using the `alerts` CLI, Qumulo Alerts uses `UTC` time.

Each translated message that a user receives includes a time zone in the `Continent/City` format (for example, `America/Los_Angeles`). For more information, see List of TZ Database Time Zones.

> **Note**
> `UTC` doesn't follow the `Continent/City` format.

Installing and Configuring Qumulo Alerts

This section explains how to install, start and stop, and configure Qumulo Alerts.

Prerequisites

This section lists the prerequisites for Qumulo Alerts.

Firewall Ports

Qumulo Alerts requires the following firewall ports to be open from the Qumulo Alerts instance

Port	Target
`25` , `587` , or `465`	Email server
`3000`	Any client that queries or views Grafana dashboards
`8000`	Qumulo cluster

System Requirements

We recommend the following system requirements for Qumulo Alerts.

- 4-core processor
- 16 GB memory
- 500 GB disk space

Tools

Before you install Qumulo Alerts, make sure you have the following tools:

- Git (You can also browse the QumuloAlerts GitHub repository.)
- Docker
- Docker Compose Plugin

> **Important**
> Qumulo Alerts requires the Docker Compose Plugin to operate correctly.

Configuration Details

Before you connect Qumulo Alerts to a Qumulo cluster, collect the information that can help you configure Qumulo Alerts to monitor your cluster.

- **Cluster Address:** Use a fully qualified domain name (FQDN) rather than an IP address.
- **Traffic Distribution:** Will your Qumulo Alerts installation use a network load balancer or a floating IP address?
- **Default Plugin Frequency**: What should be the default frequency for plugin execution?

(You can specify the frequency in minutes or seconds.)

- **Alarm and Alert Types:** Which alarms and alerts will Qumulo Alerts will collect from your Qumulo cluster?

Installing Qumulo Alerts

This section explains how to install Qumulo Alerts on your machine.

Step 1: Clone the Qumulo Alerts Repository

Navigate to the directory where you want Git to download files and run the following command.

```
git clone https://github.com/Qumulo/QumuloAlerts.git
```

Git creates a directory called `QumuloAlerts` and places the necessary files in it.

Step 2: Create a Local User for Qumulo Alerts

To be able to generate access tokens, you must create a local user for Qumulo Alerts.

1. Use SSH to log in to any node in your cluster.
2. To create a local user, use the `auth_add_user` command and specify a name and password.

   ```
   qq auth_add_user --name QumuloAlerts --password <password>
   ```

3. You need the user ID that appears in the command output to create a role for Qumulo Alerts.

   In the following example, the user ID is `1234`.

   ```
   {
     "can_change_password": true,
     "home_directory": null,
     "id": "1234",
     "name": "QumuloAlerts",
     "primary_group": "999",
     "sid": "S-1-5-21-1234567890-098765432-1234567890-1234",
     "uid": ""
   }
   ```

Step 3: Create a Qumulo Core Role for Qumulo Alerts

1. Log in to the Qumulo Core Web UI and then click **Cluster** > **Role Management**.
2. On the **Role Management** page, click **Create Role**.
3. On the **Create Role** page:
    a. Enter `QumuloAlerts`.

    > ⚠ Important
    > Because Qumulo Alerts verifies that it has sufficient role permissions before starting, this name is required.

    b. Enter a description, for example `This account lets an administrator restrict the privileges of the QumuloAlerts user.`
4. For **Privileges**, click all of the following:

    - ACCESS_TOKENS_READ: View any access tokens present in the system
    - AD_READ: Read Qumulo Active Directory Settings
    - ANALYTICS_READ: Read cluster analytics
    - AUDIT_READ: Read audit settings
    - CHECKSUMMING_READ: View the status of checksumming
    - CLUSTER_READ: View nodes, disks, protection status, and SSL certificate
    - DNS_READ: Read DNS setting
    - ENCRYPTION_READ: View the status of at rest encryption
    - FILE_FULL_ACCESS: Provides full access to all files regardless of permissions
    - FS_ATTRIBUTES_READ: Read file system statistics
    - FS_DELETE_TREE_READ: View the status of directory tree delete operations
    - FS_KEY_MANAGEMENT_READ: Read and list public keys for various FS security features
    - FS_LOCK_READ: View NLM and SMB locks and waiters
    - FS_SETTINGS_READ: View file system permissions settings
    - FTP_READ: View FTP status and settings
    - IDENTITY_MAPPING_READ: Get AD/LDAP User Defined Mappings
    - IDENTITY_READ: Use Qumulo's identity lookup and translation APIs
    - KERBEROS_KEYTAB_READ: View Kerberos keytab
    - KERBEROS_SETTINGS_READ: Read Kerberos settings
    - LDAP_READ: View LDAP settings
    - LOCAL_GROUP_READ: View local groups and members
    - LOCAL_USER_READ: Get information about local users
    - METRICS_READ: Get all metrics
    - NETWORK_IP_ALLOCATION_READ: View network IP address allocations
    - NETWORK_READ: Read network status and settings
    - NFS_EXPORT_READ: Read network status and settings

- NFS_SETTINGS_READ: Internal-Only: View NFS server settings
- QUOTA_READ: View all file system quotas
- REBOOT_READ: View Reboot Status
- RECONCILER_READ: View reconciler status and metrics
- REPLICATION_OBJECT_READ: View object store relationship settings and status
- REPLICATION_SOURCE_READ: View source relationship settings and status
- REPLICATION_TARGET_READ: View target relationship settings and status
- ROLE_READ: View roles and assignments
- S3_BUCKETS_READ: View all S3 buckets present in the system
- S3_CREDENTIALS_READ: View any S3 access key present in the system
- S3_SETTINGS_READ: View S3 server settings
- S3_UPLOADS_READ: View all S3 uploads present in the system.
- SAML_SETTINGS_READ: View SAML integration settings
- SMB_FILE_HANDLE_READ: List open SMB file handles
- SMB_SESSION_READ: List logged on SMB sessions
- SMB_SHARE_READ: View configuration of SMB shares and SMB server settings
- SNAPSHOT_CALCULATE_USED_CAPACITY_READ: Recalculate capacity usage of snapshots
- SNAPSHOT_DIFFERENCE_READ: View the changes between snapshots
- SNAPSHOT_POLICY_READ: View snapshot policies and status
- SNAPSHOT_READ: List snapshots and view their status and cached capacity.
- SUPPORT_READ: View support configuration and status
- TENANT_READ: View any tenant information
- TIME_READ: View time and time settings
- UNCONFIGURED_NODE_READ: List unconfigured Qumulo nodes
- UPGRADE_READ: View upgrade configuration and status

5. Click **Save**.

Step 4: Assign the Qumulo Alerts Role to Your Local Qumulo Core User

1. In the Qumulo Core Web UI, click **Cluster** > **Role Management**.
2. On the **Role Management** page, in the **QumuloAlerts** section, click **Add Member**.
3. In the **Add Member to Administrators** dialog box, for **Trustee**, enter the local username you have created earlier (for example, `QumuloAlerts`) and then click **Yes, Add Member**.

Step 5: Create a Long-Lived Access Token

Use the `auth_create_access_token` command and specify the ID of the local user. For example:

```
qq auth_create_access_token auth_id:1234
```

The `auth_create_access_token` command returns a JSON response that contains the bearer token body and the access token ID, which you can use to manage the access token.

```
{
  "bearer_token": "access-v1:abAcde...==",
  "id": "12345678901234567890123"
}
```

> ⚠ Important
> As soon as you receive your bearer token, record it in a safe place. If you misplace the bearer token, you can't retrieve it at a later time. You must create a new access token.

For more information, see Using Qumulo Core Access Tokens in the Qumulo On-Premises Administrator Guide.

Starting and Stopping Qumulo Alerts

- To start Qumulo Alerts, run the `./start-docker-qumulo-alerts.sh` command from the Qumulo Alerts directory.
- To stop Qumulo Alerts, run the `./stop-docker-qumulo-alerts.sh` command from the Qumulo Alerts directory.

Configuring Qumulo Alerts

This section explains how to use the `alerts` CLI and how to configure Qumulo Alerts

Step 1: Configure the alerts CLI for Your Operating System

Qumulo Alerts comes with the following binaries for Linux, macOS, and Windows.

- `alerts.macos-latest`
- `alerts.redhat-8`
- `alerts.ubuntu-20.04`
- `alerts.ubuntu-latest`
- `alerts.windows-latest.exe`

To Configure the alerts CLI for Linux

1. Link the binary for your operating system to the `alerts` CLI. For example:

   ```
   ln -s alerts.ubuntu-20.04 alerts
   ```

2. Make the binary file executable. For example:

   ```
   chmod a+x alerts.ubuntu-20.04
   ```

To Configure the alerts CLI for Windows

1. Copy `alerts.windows-latest.exe` to your Windows machine.
2. Rename the file to `alerts.exe`

Step 2: Log in to Qumulo Alerts

1. To log in to Qumulo Alerts, run the `./alerts login -u admin` command.
2. When prompted, enter the following:
    - Login: `admin`
    - Password: `Admin123`

Step 3: Configure Qumulo Alerts

1. Configure integration with an email server (page 49) or with SMS (ClickSend) (page 54).
2. Configure alarm and alert notifications (page 29).
3. Configure Collection of Alarms and Alerts from a Qumulo Cluster (page 57).

Prerequisites

This section lists the prerequisites for Qumulo Alerts.

Firewall Ports

Qumulo Alerts requires the following firewall ports to be open from the Qumulo Alerts instance

Port	Target
`25` , `587` , or `465`	Email server
`3000`	Any client that queries or views Grafana dashboards
`8000`	Qumulo cluster

System Requirements

We recommend the following system requirements for Qumulo Alerts.

- 4-core processor
- 16 GB memory
- 500 GB disk space

Tools

Before you install Qumulo Alerts, make sure you have the following tools:

- Git (You can also browse the QumuloAlerts GitHub repository.)

- Docker
- Docker Compose Plugin

> ⚠ Important
> Qumulo Alerts requires the Docker Compose Plugin to operate correctly.

Configuration Details

Before you connect Qumulo Alerts to a Qumulo cluster, collect the information that can help you configure Qumulo Alerts to monitor your cluster.

- **Cluster Address:** Use a fully qualified domain name (FQDN) rather than an IP address.
- **Traffic Distribution:** Will your Qumulo Alerts installation use a network load balancer or a floating IP address?
- **Default Plugin Frequency**: What should be the default frequency for plugin execution? (You can specify the frequency in minutes or seconds.)
- **Alarm and Alert Types:** Which alarms and alerts will Qumulo Alerts will collect from your Qumulo cluster?

Installing Qumulo Alerts

This section explains how to install Qumulo Alerts on your machine.

Step 1: Clone the Qumulo Alerts Repository

Navigate to the directory where you want Git to download files and run the following command.

```
git clone https://github.com/Qumulo/QumuloAlerts.git
```

Git creates a directory called `QumuloAlerts` and places the necessary files in it.

Step 2: Create a Local User for Qumulo Alerts

To be able to generate access tokens, you must create a local user for Qumulo Alerts.

1. Use SSH to log in to any node in your cluster.
2. To create a local user, use the `auth_add_user` command and specify a name and password.

```
qq auth_add_user --name QumuloAlerts --password <password>
```

3. You need the user ID that appears in the command output to create a role for Qumulo Alerts.

   In the following example, the user ID is `1234`.

```
{
  "can_change_password": true,
  "home_directory": null,
  "id": "1234",
  "name": "QumuloAlerts",
  "primary_group": "999",
  "sid": "S-1-5-21-1234567890-098765432-1234567890-1234",
  "uid": ""
}
```

Step 3: Create a Qumulo Core Role for Qumulo Alerts

1. Log in to the Qumulo Core Web UI and then click **Cluster** > **Role Management**.
2. On the **Role Management** page, click **Create Role**.
3. On the **Create Role** page:
   a. Enter `QumuloAlerts`.

   > ⚠ Important
   > Because Qumulo Alerts verifies that it has sufficient role permissions before starting, this name is required.

   b. Enter a description, for example `This account lets an administrator restrict the privileges of the QumuloAlerts user.`
4. For **Privileges**, click all of the following:
   - ACCESS_TOKENS_READ: View any access tokens present in the system
   - AD_READ: Read Qumulo Active Directory Settings
   - ANALYTICS_READ: Read cluster analytics
   - AUDIT_READ: Read audit settings
   - CHECKSUMMING_READ: View the status of checksumming
   - CLUSTER_READ: View nodes, disks, protection status, and SSL certificate
   - DNS_READ: Read DNS setting
   - ENCRYPTION_READ: View the status of at rest encryption

- FILE_FULL_ACCESS: Provides full access to all files regardless of permissions
- FS_ATTRIBUTES_READ: Read file system statistics
- FS_DELETE_TREE_READ: View the status of directory tree delete operations
- FS_KEY_MANAGEMENT_READ: Read and list public keys for various FS security features
- FS_LOCK_READ: View NLM and SMB locks and waiters
- FS_SETTINGS_READ: View file system permissions settings
- FTP_READ: View FTP status and settings
- IDENTITY_MAPPING_READ: Get AD/LDAP User Defined Mappings
- IDENTITY_READ: Use Qumulo's identity lookup and translation APIs
- KERBEROS_KEYTAB_READ: View Kerberos keytab
- KERBEROS_SETTINGS_READ: Read Kerberos settings
- LDAP_READ: View LDAP settings
- LOCAL_GROUP_READ: View local groups and members
- LOCAL_USER_READ: Get information about local users
- METRICS_READ: Get all metrics
- NETWORK_IP_ALLOCATION_READ: View network IP address allocations
- NETWORK_READ: Read network status and settings
- NFS_EXPORT_READ: Read network status and settings
- NFS_SETTINGS_READ: Internal-Only: View NFS server settings
- QUOTA_READ: View all file system quotas
- REBOOT_READ: View Reboot Status
- RECONCILER_READ: View reconciler status and metrics
- REPLICATION_OBJECT_READ: View object store relationship settings and status
- REPLICATION_SOURCE_READ: View source relationship settings and status
- REPLICATION_TARGET_READ: View target relationship settings and status
- ROLE_READ: View roles and assignments
- S3_BUCKETS_READ: View all S3 buckets present in the system
- S3_CREDENTIALS_READ: View any S3 access key present in the system

- S3_SETTINGS_READ: View S3 server settings
- S3_UPLOADS_READ: View all S3 uploads present in the system.
- SAML_SETTINGS_READ: View SAML integration settings
- SMB_FILE_HANDLE_READ: List open SMB file handles
- SMB_SESSION_READ: List logged on SMB sessions
- SMB_SHARE_READ: View configuration of SMB shares and SMB server settings
- SNAPSHOT_CALCULATE_USED_CAPACITY_READ: Recalculate capacity usage of snapshots
- SNAPSHOT_DIFFERENCE_READ: View the changes between snapshots
- SNAPSHOT_POLICY_READ: View snapshot policies and status
- SNAPSHOT_READ: List snapshots and view their status and cached capacity.
- SUPPORT_READ: View support configuration and status
- TENANT_READ: View any tenant information
- TIME_READ: View time and time settings
- UNCONFIGURED_NODE_READ: List unconfigured Qumulo nodes
- UPGRADE_READ: View upgrade configuration and status

5. Click **Save**.

Step 4: Assign the Qumulo Alerts Role to Your Local Qumulo Core User

1. In the Qumulo Core Web UI, click **Cluster** > **Role Management**.
2. On the **Role Management** page, in the **QumuloAlerts** section, click **Add Member**.
3. In the **Add Member to Administrators** dialog box, for **Trustee**, enter the local username you have created earlier (for example, `QumuloAlerts`) and then click **Yes, Add Member**.

Step 5: Create a Long-Lived Access Token

Use the `auth_create_access_token` command and specify the ID of the local user. For example:

```
qq auth_create_access_token auth_id:1234
```

The `auth_create_access_token` command returns a JSON response that contains the bearer token body and the access token ID, which you can use to manage the access token.

```
{
  "bearer_token": "access-v1:abAcde...==",
  "id": "12345678901234567890123"
}
```

> ⚠ Important
> As soon as you receive your bearer token, record it in a safe place. If you misplace the bearer token, you can't retrieve it at a later time. You must create a new access token.

For more information, see Using Qumulo Core Access Tokens in the Qumulo On-Premises Administrator Guide.

Starting and Stopping Qumulo Alerts

- To start Qumulo Alerts, run the `./start-docker-qumulo-alerts.sh` command from the Qumulo Alerts directory.
- To stop Qumulo Alerts, run the `./stop-docker-qumulo-alerts.sh` command from the Qumulo Alerts directory.

Configuring Qumulo Alerts

This section explains how to use the `alerts` CLI and how to configure Qumulo Alerts

Step 1: Configure the alerts CLI for Your Operating System

Qumulo Alerts comes with the following binaries for Linux, macOS, and Windows.

- `alerts.macos-latest`
- `alerts.redhat-8`
- `alerts.ubuntu-20.04`
- `alerts.ubuntu-latest`
- `alerts.windows-latest.exe`

To Configure the alerts CLI for Linux

1. Link the binary for your operating system to the `alerts` CLI. For example:

   ```
   ln -s alerts.ubuntu-20.04 alerts
   ```

2. Make the binary file executable. For example:

```
chmod a+x alerts.ubuntu-20.04
```

To Configure the alerts CLI for Windows

1. Copy `alerts.windows-latest.exe` to your Windows machine.
2. Rename the file to `alerts.exe`

Step 2: Log in to Qumulo Alerts

1. To log in to Qumulo Alerts, run the `./alerts login -u admin` command.
2. When prompted, enter the following:
    - Login: `admin`
    - Password: `Admin123`

Step 3: Configure Qumulo Alerts

1. Configure integration with an email server (page 49) or with SMS (ClickSend) (page 54).
2. Configure alarm and alert notifications (page 29).
3. Configure Collection of Alarms and Alerts from a Qumulo Cluster (page 57).

Upgrading

Upgrading Qumulo Alerts from a Previous Public Version

This section explains how to upgrade Qumulo Alerts from a previous public version to the latest one.

To Upgrade Qumulo Alerts to the Latest Public Version

1. To shut down Qumulo Alerts, navigate to its directory and run the `./stop-docker-qumulo-alerts.sh` command.

   > **Note**
   > This process might take up to 60 seconds. The `Alerts` Docker container must shut down and then verify that all Qumulo Alerts Docker containers are also shut down correctly.

2. In the Qumulo Alerts directory, run the `git pull` command.
3. To remove all existing Qumulo Alerts Docker images from your machine, use the `docker system prune -a -f` command.

   > **Note**
   > This release of Qumulo Alerts adds new Docker containers, making it necessary to remove all existing images.

4. To restart the Docker containers for Qumulo Alerts, pull new Docker images from the Qumulo Docker repository, and restart all Docker containers, use the `./start-docker-qumulo-alerts.sh` command.

To Upgrade Qumulo Alerts to the Latest Public Version

1. To shut down Qumulo Alerts, navigate to its directory and run the `./stop-docker-qumulo-alerts.sh` command.

   > **Note**
   > This process might take up to 60 seconds. The `Alerts` Docker container must shut down and then verify that all Qumulo Alerts Docker containers are also shut down correctly.

2. In the Qumulo Alerts directory, run the `git pull` command.

3. To remove all existing Qumulo Alerts Docker images from your machine, use the `docker system prune -a -f` command.

   > **Note**
   > This release of Qumulo Alerts adds new Docker containers, making it necessary to remove all existing images.

4. To restart the Docker containers for Qumulo Alerts, pull new Docker images from the Qumulo Docker repository, and restart all Docker containers, use the `./start-docker-qumulo-alerts.sh` command.

Upgrading Qumulo Alerts from the Beta Version

This section explains how to upgrade Qumulo Alerts from the beta version.

> **Note**
>
> - Whereas the beta version of Qumulo Alerts uses JSON files for configuration, the public version of Qumulo Alerts uses its API or the `alerts` CLI to store configuration information in a database.
> - It isn't possible to upgrade from the beta version of Qumulo Alerts to the public version automatically. To enable upgrades from a beta version, you must perform the following manual steps.

To Prepare for Upgrading Qumulo Alerts from the Beta Version

1. To shut down Qumulo Alerts, navigate to its directory and run the `./stop-docker-qumulo-alerts.sh` command.
2. Copy the information from the `user_token` field located in the `QumuloAlerts/config/alerts/QumuloAlerts.json` file.
3. Rename the directory of the beta version of Qumulo Alerts, for example to `QumuloAlerts.beta`.
4. Install the latest public version of Qumulo Alerts (page 13).
5. When you configure alarm and alert collection from your Qumulo cluster (page 57), use the information from the `user_token` field.

> **Note**
>
> - Whereas the beta version of Qumulo Alerts uses JSON files for configuration, the public version of Qumulo Alerts uses its API or the `alerts` CLI to store configuration information in a database.
> - It isn't possible to upgrade from the beta version of Qumulo Alerts to the public version automatically. To enable upgrades from a beta version, you must perform the following manual steps.

To Prepare for Upgrading Qumulo Alerts from the Beta Version

1. To shut down Qumulo Alerts, navigate to its directory and run the `./stop-docker-qumulo-alerts.sh` command.

2. Copy the information from the `user_token` field located in the `QumuloAlerts/config/alerts/QumuloAlerts.json` file.
3. Rename the directory of the beta version of Qumulo Alerts, for example to `QumuloAlerts.beta`.
4. Install the latest public version of Qumulo Alerts (page 13).
5. When you configure alarm and alert collection from your Qumulo cluster (page 57), use the information from the `user_token` field.

Configuring Notifications

Configuring Alarm and Alert Notifications to an Administrative Account in Qumulo Alerts

This section explains how to configure Qumulo Alerts to send alarm and alert notifications from a Qumulo cluster to an administrative account.

You must first add the account as a Qumulo Alerts user, create a notification group and configure its notifications, and then add the user to the notification group.

Step 1: Add an Administrative Account as a Qumulo Alerts User

Use the `./alerts user_add` command and specify the administrator's full name, username, password, email address, language, and time zone. For example:

```
./alerts user_add \
  --full_name "Jane Johnson" \
  --username jjohnson \
  --password MyPassword123 \
  --email jjohnson@example.com \
  --language en_US \
  --timezone "America/Los_Angeles"
```

> **Note**
>
> - For the `--language` flag, see What Language Locales Qumulo Alerts Supports (page 10). The consumer processes for email (page 49), IFTTT (page 52), and SMS (ClickSend) (page 54) integrations translate messages into the recipient's native language.
> - For the `--timezone` flag, see Converting Time Zones (page 11).

The following is example JSON output from the command.

```
[{
  "disabled": false,
  "email": "jjohnson@example.com",
  "full_name": "Jane Johnson",
  "id": 3,
  "ifttt_event": null,
  "language": "en_US",
  "phone": null,
  "timezone": "America/Los_Angeles",
  "username": "jjohnson"
}]
```

Step 2: Create and Configure a Notification Group

Use the `./alerts notification_group_add` command and specify the notification group's name, description, and the events for which the notification group receives notifications. In the following example, the `NotifyOnHardwareChange` group receives notifications for all hardware state change events.

```
./alerts notification_group_add \
  --name NotifyOnHardwareChange \
  --description "Send a notification when any hardware changes state" \
  --event NOTIFY_FANS \
  --event NOTIFY_CPU \
  --event NOTIFY_DISKS \
  --event NOTIFY_NETWORK \
  --event NOTIFY_NODES
```

The following is example JSON output from the command.

```
[{
  "description": "Send a notification when any hardware changes state",
  "id": 2,
  "name": "NotifyOnHardwareChange"
}]
```

Step 3: Add a Qumulo Alerts User to a Notification Group

Use the `./alerts notification_group_add_user` command and specify the notification group name and the Qumulo Alerts user name to add to the notification group. For example:

```
./alerts notification_group_add_user \
  --name NotifyOnHardwareChange \
  --username jjohnson
```

The following is example JSON output from the command.

```
[{
  "description": "Notify when certain hardware changes state",
  "id": 2,
  "name": "NotifyOnHardwareChange",
  "users": [{
    "can_change_password": true,
    "disabled": false,
    "email": "jjohnson@example.com",
    "full_name": "Jane Johnson",
    "id": 3,
    "ifttt_event": null,
    "language": "en_US",
    "phone": null,
    "timezone": "America/Los_Angeles",
    "username": "jjohnson"
  }]
}]
```

You must first add the account as a Qumulo Alerts user, create a notification group and configure its notifications, and then add the user to the notification group.

Step 1: Add an Administrative Account as a Qumulo Alerts User

Use the `./alerts user_add` command and specify the administrator's full name, username, password, email address, language, and time zone. For example:

```
./alerts user_add \
  --full_name "Jane Johnson" \
  --username jjohnson \
  --password MyPassword123 \
  --email jjohnson@example.com \
  --language en_US \
  --timezone "America/Los_Angeles"
```

> **Note**
>
> - For the `--language` flag, see What Language Locales Qumulo Alerts Supports (page 10). The consumer processes for email (page 49), IFTTT (page 52), and SMS (ClickSend) (page 54) integrations translate messages into the recipient's native language.
> - For the `--timezone` flag, see Converting Time Zones (page 11).

The following is example JSON output from the command.

```
[{
  "disabled": false,
  "email": "jjohnson@example.com",
  "full_name": "Jane Johnson",
  "id": 3,
  "ifttt_event": null,
  "language": "en_US",
  "phone": null,
  "timezone": "America/Los_Angeles",
  "username": "jjohnson"
}]
```

Step 2: Create and Configure a Notification Group

Use the `./alerts notification_group_add` command and specify the notification group's name, description, and the events for which the notification group receives notifications. In the following example, the `NotifyOnHardwareChange` group receives notifications for all hardware state change events.

```
./alerts notification_group_add \
  --name NotifyOnHardwareChange \
  --description "Send a notification when any hardware changes state" \
  --event NOTIFY_FANS \
  --event NOTIFY_CPU \
  --event NOTIFY_DISKS \
  --event NOTIFY_NETWORK \
  --event NOTIFY_NODES
```

The following is example JSON output from the command.

```
[{
  "description": "Send a notification when any hardware changes state",
  "id": 2,
  "name": "NotifyOnHardwareChange"
}]
```

Step 3: Add a Qumulo Alerts User to a Notification Group

Use the `./alerts notification_group_add_user` command and specify the notification group name and the Qumulo Alerts user name to add to the notification group. For example:

```
./alerts notification_group_add_user \
  --name NotifyOnHardwareChange \
  --username jjohnson
```

The following is example JSON output from the command.

```
[{
  "description": "Notify when certain hardware changes state",
  "id": 2,
  "name": "NotifyOnHardwareChange",
  "users": [{
    "can_change_password": true,
    "disabled": false,
    "email": "jjohnson@example.com",
    "full_name": "Jane Johnson",
    "id": 3,
    "ifttt_event": null,
    "language": "en_US",
    "phone": null,
    "timezone": "America/Los_Angeles",
    "username": "jjohnson"
  }]
}]
```

Configuring Default Quota Notifications in Qumulo Alerts

This section explains how to configure default quota notifications in Qumulo Alerts.

Qumulo Alerts lets an administrator configure notifications that inherit a template from one of the following default quotas.

- No-Path Quota: This quota type has no defined file system path. It is the most common quota type and it applies thresholds to every quota defined for a Qumulo cluster.
- Inherited-Path Quotas: This quota type lets an administrator specify a default path for every quota defined for a Qumulo cluster. Every quota created under the default path inherits its thresholds from this quota.

You can configure quota monitoring by using *thresholds*.

- For the `--warning` flag, the threshold must be lower than the thresholds of both the `--error` and `--critical` flags.
- For the `--error` flag, the threshold must be lower than the threshold of the `--critical` flag.
- For the `--critical` flag, the threshold must be greater than the thresholds of both the `--warning` and `--error` flags.

For more information about how quotas work, see Configuring Quota Notifications to an Administrative Account (page 0) and Configuring Quota Notifications to a User Account (page 0).

To List the Predefined No-Path Quota

Qumulo Alerts comes with a predefined no-path quota. To get information about this quota, use the `./alerts default_quota_list` command.

The following is example JSON output from the command.

```
[{
  "items": [{
    "admin_notification": true,
    "critical": 95,
    "error": 85,
    "id": 1,
    "quota_prefix": "",
    "user_mode": "owner",
    "user_notification": false,
    "warning": 75
  }],
  "page": 1,
  "pages": 1,
  "size": 50,
  "total": 1
}]
```

To Configure an Inherited-Path Quota

Use the `./alerts default_quota_add` command and specify the default path and thresholds. For example:

```
./alerts default_quota_add \
  --quota-prefix /Home \
  --warning 80 \
  --error 90 \
  --critical 98
```

The following is example JSON output from the command.

```
[{
  "admin_notification": true,
  "critical": 98,
  "error": 90,
  "id": 2,
  "quota_prefix": "/Home/",
  "user_mode": "owner",
  "user_notification": false,
  "warning": 80
}]
```

Qumulo Alerts lets an administrator configure notifications that inherit a template from one of the following default quotas.

- No-Path Quota: This quota type has no defined file system path. It is the most common quota type and it applies thresholds to every quota defined for a Qumulo cluster.
- Inherited-Path Quotas: This quota type lets an administrator specify a default path for every quota defined for a Qumulo cluster. Every quota created under the default path inherits its thresholds from this quota.

You can configure quota monitoring by using *thresholds*.

- For the `--warning` flag, the threshold must be lower than the thresholds of both the `--error` and `--critical` flags.
- For the `--error` flag, the threshold must be lower than the threshold of the `--critical` flag.
- For the `--critical` flag, the threshold must be greater than the thresholds of both the `--warning` and `--error` flags.

For more information about how quotas work, see Configuring Quota Notifications to an Administrative Account (page 0) and Configuring Quota Notifications to a User Account (page 0).

To List the Predefined No-Path Quota

Qumulo Alerts comes with a predefined no-path quota. To get information about this quota, use the `./alerts default_quota_list` command.

The following is example JSON output from the command.


```
[{
  "items": [{
    "admin_notification": true,
    "critical": 95,
    "error": 85,
    "id": 1,
    "quota_prefix": "",
    "user_mode": "owner",
    "user_notification": false,
    "warning": 75
  }],
  "page": 1,
  "pages": 1,
  "size": 50,
  "total": 1
}]
```

To Configure an Inherited-Path Quota

Use the `./alerts default_quota_add` command and specify the default path and thresholds. For example:

```
./alerts default_quota_add \
  --quota-prefix /Home \
  --warning 80 \
  --error 90 \
  --critical 98
```

The following is example JSON output from the command.

```
[{
  "admin_notification": true,
  "critical": 98,
  "error": 90,
  "id": 2,
  "quota_prefix": "/Home/",
  "user_mode": "owner",
  "user_notification": false,
  "warning": 80
}]
```

Configuring Quota Notifications to an Administrative Account in Qumulo Alerts

This section explains how to configure Qumulo Alerts to send quota notifications from a Qumulo cluster to an administrative account.

You can configure quota monitoring by using *thresholds*.

- For the `--warning` flag, the threshold must be lower than the thresholds of both the `--error` and `--critical` flags.
- For the `--error` flag, the threshold must be lower than the threshold of the `--critical` flag.
- For the `--critical` flag, the threshold must be greater than the thresholds of both the `--warning` and `--error` flags.

You can configure unattached quotas or attach them to a Qumulo cluster.

To Configure Quota Notifications with Two Thresholds

Use the `./alerts quota_add` command and specify the quota path to monitor. The following example specifies the warning threshold and the error threshold and doesn't attach the quota to a Qumulo cluster.

```
./alerts quota_add \
  --quotapath /Reports/Sales \
  --warning 80 \
  --error 85
```

The following is example JSON output from the command.

```
[{
  "admin_notification": true,
  "critical": 95,
  "error": 85,
  "id": 2,
  "quota_path": "/Reports/Sales/",
  "user_email": "",
  "user_mode": "direct",
  "user_notification": false,
  "warning": 80
}]
```

To Configure Quota Notifications with a Single Threshold

Use the `./alerts quota_add` command and specify the quota path. The following example specifies the error threshold and attaches the quota to the fully qualified domain name (FQDN) of a Qumulo cluster.

```
./alerts quota_add \
  --quotapath /Reports/Marketing \
  --error 90 \
  --cluster-include cluster.example.com
```

> **Note**
> When you add a quota and attach it to a Qumulo cluster, the `alerts` CLI doesn't list the cluster.

The following is example JSON output from the command.

```
[{
  "admin_notification": true,
  "critical": 95,
  "error": 90,
  "id": 3,
  "quota_path": "/Movies/Dutch/",
  "user_email": "",
  "user_mode": "direct",
  "user_notification": false,
  "warning": 75
}]
```

To List All Defined Quotas and Attached Clusters

Use the `./alerts quota_list` command.

The following is example JSON output from the command. In this example, the second quota is attached to the fully qualified domain name (FQDN) of a Qumulo cluster.

```
[{
  "items": [{
    "admin_notification": true,
    "clusters": [],
    "critical": 95,
    "error": 85,
    "id": 2,
    "quota_path": "/Reports/Sales/",
    "user_email": "",
    "user_mode": "direct",
    "user_notification": false,
    "warning": 80
  },{
    "admin_notification": true,
    "clusters": [{
      "frequency": 1,
      "name": "cluster.example.com",
      "nlb": false,
      "port": 8000
    }],
    "critical": 95,
    "error": 90,
    "id": 3,
    "quota_path": "/Reports/Marketing/",
    "user_email": "",
    "user_mode": "direct",
    "user_notification": false,
    "warning": 75
  }],
  "page": 1,
  "pages": 1,
  "size": 50,
  "total": 2
}]
```


You can configure quota monitoring by using *thresholds*.

- For the `--warning` flag, the threshold must be lower than the thresholds of both the `--error` and `--critical` flags.
- For the `--error` flag, the threshold must be lower than the threshold of the `--critical` flag.
- For the `--critical` flag, the threshold must be greater than the thresholds of both the `--warning` and `--error` flags.

You can configure unattached quotas or attach them to a Qumulo cluster.

To Configure Quota Notifications with Two Thresholds

Use the `./alerts quota_add` command and specify the quota path to monitor. The following example specifies the warning threshold and the error threshold and doesn't attach the quota to a Qumulo cluster.

```
./alerts quota_add \
  --quotapath /Reports/Sales \
  --warning 80 \
  --error 85
```

The following is example JSON output from the command.

```
[{
  "admin_notification": true,
  "critical": 95,
  "error": 85,
  "id": 2,
  "quota_path": "/Reports/Sales/",
  "user_email": "",
  "user_mode": "direct",
  "user_notification": false,
  "warning": 80
}]
```

To Configure Quota Notifications with a Single Threshold

Use the `./alerts quota_add` command and specify the quota path. The following example specifies the error threshold and attaches the quota to the fully qualified domain name (FQDN) of a Qumulo cluster.

```
./alerts quota_add \
  --quotapath /Reports/Marketing \
  --error 90 \
  --cluster-include cluster.example.com
```

> **Note**
> When you add a quota and attach it to a Qumulo cluster, the `alerts` CLI doesn't list the cluster.

The following is example JSON output from the command.

```
[{
  "admin_notification": true,
  "critical": 95,
  "error": 90,
  "id": 3,
  "quota_path": "/Movies/Dutch/",
  "user_email": "",
  "user_mode": "direct",
  "user_notification": false,
  "warning": 75
}]
```

To List All Defined Quotas and Attached Clusters

Use the `./alerts quota_list` command.

The following is example JSON output from the command. In this example, the second quota is attached to the fully qualified domain name (FQDN) of a Qumulo cluster.

```
[{
  "items": [{
    "admin_notification": true,
    "clusters": [],
    "critical": 95,
    "error": 85,
    "id": 2,
    "quota_path": "/Reports/Sales/",
    "user_email": "",
    "user_mode": "direct",
    "user_notification": false,
    "warning": 80
  },{
    "admin_notification": true,
    "clusters": [{
      "frequency": 1,
      "name": "cluster.example.com",
      "nlb": false,
      "port": 8000
    }],
    "critical": 95,
    "error": 90,
    "id": 3,
    "quota_path": "/Reports/Marketing/",
    "user_email": "",
    "user_mode": "direct",
    "user_notification": false,
    "warning": 75
  }],
  "page": 1,
  "pages": 1,
  "size": 50,
  "total": 2
}]
```

Configuring Quota Notifications to a User Account in Qumulo Alerts

This section explains how to configure Qumulo Alerts to send quota notifications from a Qumulo cluster to a user account.

Qumulo Alerts can notify an individual user's email address manually or use default quotas (page 0) to notify email addresses associated in Active Directory (AD) with the security identifier (SID) of the quota directory's owner automatically.

To Notify an Individual Email Address

Use the `./alerts quota_add` command and specify the quota path, the email address to notify, the email address to notify, and the fully qualified domain name (FQDN) of your Qumulo cluster. For example:

```
./alerts quota_add \
  --quotapath /Reports/Marketing \
  --user-notification True \
  --user-mode direct \
  --user-email jjohnson@example.com \
  --cluster-include cluster.example.com
```

> **Note**
> For the `--user-email` flag, you can specify a comma-delimited list of email addresses to notify, if you also specify `--user-notification True --user-mode direct`.

The following is example JSON output from the command.

```
[{
  "admin_notification": true,
  "critical": 95,
  "error": 85,
  "id": 1,
  "quota_path": "/Reports/Marketing/",
  "user_email": "jjohnson@example.com",
  "user_mode": "direct",
  "user_notification": true,
  "warning": 75
}]
```

Notifying Directory Owners Automatically

To ue this method, you must first add an AD server to Qumulo Alerts and then configure the default quota to use AD lookup to retrieve users' email addresses.

Step 1: Connect Qumulo Alerts to an Active Directory Server

Use the `./alerts ad_server_add` command and specify the AD server, AD login name, AD password, the search base for looking up users, and the fully qualified domain name (FQDN) of your Qumulo cluster. For example:

```
./alerts ad_server_add \
  --server-name "ad.example.com" \
  --login-name "example.com\LookupUser" \
  --password MyPassword123 \
  --search-base "CN=Users,DC=example,DC=com" \
  --cluster-include cluster.example.com
```

> ⚠ Important
> For maximum security, configure a specific AD user to issue lookup requests.

The following is example JSON output from the command.

```
[{
  "clusters": [{
    "frequency": 1,
    "name": "cluster.example.com",
    "nlb": false,
    "port": 8000
  }],
  "id": 2,
  "login_name": "example.com\\LookupUser",
  "search_base": "CN=Users,DC=example,DC=com",
  "server_name": "ad.example.com"
}]
```

Step 2: Configure a Default Quota to use Active Directory Lookup

Use the `./alerts default_quota_update` command, specify the default quota ID, and configure the quota to notify users. For example:

```
./alerts default_quota_update \
  --id 1 \
  --user-notification True \
  --admin-notification False
```

The following is example JSON output from the command.

```
[{
  "admin_notification": false,
  "critical": 95,
  "error": 85,
  "quota_prefix": "",
  "user_mode": "owner",
  "user_notification": true,
  "warning": 75
}]
```

Qumulo Alerts can notify an individual user's email address manually or use default quotas (page 0) to notify email addresses associated in Active Directory (AD) with the security identifier (SID) of the quota directory's owner automatically.

To Notify an Individual Email Address

Use the `./alerts quota_add` command and specify the quota path, the email address to notify, the email address to notify, and the fully qualified domain name (FQDN) of your Qumulo cluster. For example:

```
./alerts quota_add \
  --quotapath /Reports/Marketing \
  --user-notification True \
  --user-mode direct \
  --user-email jjohnson@example.com \
  --cluster-include cluster.example.com
```

> **Note**
> For the `--user-email` flag, you can specify a comma-delimited list of email addresses to notify, if you also specify `--user-notification True --user-mode direct`.

The following is example JSON output from the command.

```
[{
  "admin_notification": true,
  "critical": 95,
  "error": 85,
  "id": 1,
  "quota_path": "/Reports/Marketing/",
  "user_email": "jjohnson@example.com",
  "user_mode": "direct",
  "user_notification": true,
  "warning": 75
}]
```

Notifying Directory Owners Automatically

To ue this method, you must first add an AD server to Qumulo Alerts and then configure the default quota to use AD lookup to retrieve users' email addresses.

Step 1: Connect Qumulo Alerts to an Active Directory Server

Use the `./alerts ad_server_add` command and specify the AD server, AD login name, AD password, the search base for looking up users, and the fully qualified domain name (FQDN) of your Qumulo cluster. For example:

```
./alerts ad_server_add \
  --server-name "ad.example.com" \
  --login-name "example.com\LookupUser" \
  --password MyPassword123 \
  --search-base "CN=Users,DC=example,DC=com" \
  --cluster-include cluster.example.com
```

> ⚠ Important
> For maximum security, configure a specific AD user to issue lookup requests.

The following is example JSON output from the command.

```
[{
  "clusters": [{
    "frequency": 1,
    "name": "cluster.example.com",
    "nlb": false,
    "port": 8000
  }],
  "id": 2,
  "login_name": "example.com\\LookupUser",
  "search_base": "CN=Users,DC=example,DC=com",
  "server_name": "ad.example.com"
}]
```

Step 2: Configure a Default Quota to use Active Directory Lookup

Use the `./alerts default_quota_update` command, specify the default quota ID, and configure the quota to notify users. For example:

```
./alerts default_quota_update \
  --id 1 \
  --user-notification True \
  --admin-notification False
```

The following is example JSON output from the command.

```
[{
  "admin_notification": false,
  "critical": 95,
  "error": 85,
  "quota_prefix": "",
  "user_mode": "owner",
  "user_notification": true,
  "warning": 75
}]
```

Configuring Integrations

Configuring Qumulo Alerts Integration with an Email Server

This section explains how to configure Qumulo Alerts to work with an email server.

> **Note**
> After May 2022, only organizations with access to the Google Admin Console can use SMTP relay. If your organization has this access, see Route outgoing SMTP relay messages through Google.

To Add a New Email Server to Qumulo Alerts

Use the `./alerts email_server_add` and specify the sender's email address, recipient's email address, email server hostname and port, language, and time zone. For example:

```
./alerts email_server_add \
  --from-addr alerts@example.com \
  --to-addr name@example.com \
  --server mail.example.com \
  --port 25
  --language en_US
  --timezone "America/Los_Angeles"
```

> **Note**
> - The `--login`, `--password`, and `--security` flags might be optional, depending on the type of SMTP email server that you use.
> - For the `--language` flag, see What Language Locales Qumulo Alerts Supports (page 10). The consumer processes for email (page 49), IFTTT (page 52), and SMS (ClickSend) (page 54) integrations translate messages into the recipient's native language.
> - For the `--timezone` flag, see Converting Time Zones (page 11).

The following is example JSON output from the command.

```
[{
  "from_address": "alerts@example.com",
  "language": "en_US",
  "login": null,
  "password": null,
  "port": 25,
  "security": null,
  "server": "mail.example.com",
  "timezone": "America/Los_Angeles",
  "to_address": "name@example.com"
}]
```

To Test Integration with Your Email Server

Use the `./alerts email_server_test` command.

A successful response returns the `[{ "ok": true }]` JSON output.

> **Note**
> After May 2022, only organizations with access to the Google Admin Console can use SMTP relay. If your organization has this access, see Route outgoing SMTP relay messages through Google.

To Add a New Email Server to Qumulo Alerts

Use the `./alerts email_server_add` and specify the sender's email address, recipient's email address, email server hostname and port, language, and time zone. For example:

```
./alerts email_server_add \
  --from-addr alerts@example.com \
  --to-addr name@example.com \
  --server mail.example.com \
  --port 25
  --language en_US
  --timezone "America/Los_Angeles"
```

> **Note**
>
> - The `--login`, `--password`, and `--security` flags might be optional, depending on the type of SMTP email server that you use.
> - For the `--language` flag, see What Language Locales Qumulo Alerts Supports (page 10). The consumer processes for email (page 49), IFTTT (page 52), and SMS (ClickSend) (page 54) integrations translate messages into the recipient's native language.
> - For the `--timezone` flag, see Converting Time Zones (page 11).

The following is example JSON output from the command.

```
[{
  "from_address": "alerts@example.com",
  "language": "en_US",
  "login": null,
  "password": null,
  "port": 25,
  "security": null,
  "server": "mail.example.com",
  "timezone": "America/Los_Angeles",
  "to_address": "name@example.com"
}]
```

To Test Integration with Your Email Server

Use the `./alerts email_server_test` command.

A successful response returns the `[{ "ok": true }]` JSON output.

Configuring Qumulo Alerts Integration with IFTTT

This section explains how to configure Qumulo Alerts to work with IFTTT.

IFTTT (If This Then That) is a paid, third-party service that provides delivery of messages by using Webhooks integrations and events. For more information, see the IFTTT documentation.

To Integrate IFTTT with Qumulo Alerts

Use the `./alerts ifttt_server_add` command and specify the IFTTT server token, language, and time zone. For example:

```
./alerts ifttt_server_add \
  --token abcABde12f3g4567CDE89 \
  --language en_US \
  --timezone "America/Phoenix"
```

> **Note**
>
> - For the `--language` flag, see What Language Locales Qumulo Alerts Supports (page 10). The consumer processes for email (page 49), IFTTT (page 52), and SMS (ClickSend) (page 54) integrations translate messages into the recipient's native language.
> - For the `--timezone` flag, see Converting Time Zones (page 11).

The following is example JSON output from the command.

```
[{
  "language": "en_US",
  "timezone": "America/Los_Angeles",
  "token": "abcABde12f3g4567CDE89"
}]
```

To Test Integration with IFTTT

Use the `./alerts ifttt_server_test` command.

A successful response returns the `[{ "ok": true }]` JSON output.

IFTTT (If This Then That) is a paid, third-party service that provides delivery of messages by using Webhooks integrations and events. For more information, see the IFTTT documentation.

To Integrate IFTTT with Qumulo Alerts

Use the `./alerts ifttt_server_add` command and specify the IFTTT server token, language, and time zone. For example:

```
./alerts ifttt_server_add \
  --token abcABde12f3g4567CDE89 \
  --language en_US \
  --timezone "America/Phoenix"
```

> **Note**
>
> - For the `--language` flag, see What Language Locales Qumulo Alerts Supports (page 10). The consumer processes for email (page 49), IFTTT (page 52), and SMS (ClickSend) (page 54) integrations translate messages into the recipient's native language.
> - For the `--timezone` flag, see Converting Time Zones (page 11).

The following is example JSON output from the command.

```
[{
  "language": "en_US",
  "timezone": "America/Los_Angeles",
  "token": "abcABde12f3g4567CDE89"
}]
```

To Test Integration with IFTTT

Use the `./alerts ifttt_server_test` command.

A successful response returns the `[{ "ok": true }]` JSON output.

Configuring Qumulo Alerts Integration with SMS (ClickSend)

This section explains how to configure Qumulo Alerts to work with SMS by using ClickSend.

ClickSend is a paid, third-party service that provides delivery of messages as SMS (and other formats). For more information, see How to get started with ClickSend in the ClickSend documentation.

> **Important**
> To be able to send SMS in the U.S. and Canada, you must sign up for a dedicated toll-free number (TFN).

To Integrate ClickSend with Qumulo Alerts

Use the `./alerts clicksend_server_add` command and specify the username, token, sender ID, and recipient's phone number.

```
./alerts clicksend_server_add \
  --username name@example.com \
  --token 12345678-ABCDEFGH-12345678-ABCDEFGH \
  --senderid "+15551234567" \
  --to-address "+15550987654"
```

> **Note**
> - For the `--username` and `--token` flags, see API Credentials in the ClickSend documentation.
> - The `--senderid` flag is mandatory for the U.S. and Canada. For more information, see Toll-Free Number (TFN) Verification in the ClickSend documentation.
> - For the `--language` flag, see What Language Locales Qumulo Alerts Supports (page 10). The consumer processes for email (page 49), IFTTT (page 52), and SMS (ClickSend) (page 54) integrations translate messages into the recipient's native language.
> - For the `--timezone` flag, see Converting Time Zones (page 11).

The following is example JSON output from the command.

```
[{
  "language": "en_GB",
  "senderid": "+15551234567",
  "timezone": "UTC",
  "to_address": "+15550987654",
  "username": "name@example.com"
}]
```

To Test Integration with ClickSend

Use the `./alerts clicksend_server_test` command.

> **Note**
> For integration testing to complete successfully, the `--to-address` flag must be configured already.

A successful response returns the `[{ "ok": true }]` JSON output. In addition, the recipient's phone number receives a test message.

ClickSend is a paid, third-party service that provides delivery of messages as SMS (and other formats). For more information, see How to get started with ClickSend in the ClickSend documentation.

> **Important**
> To be able to send SMS in the U.S. and Canada, you must sign up for a dedicated toll-free number (TFN).

To Integrate ClickSend with Qumulo Alerts

Use the `./alerts clicksend_server_add` command and specify the username, token, sender ID, and recipient's phone number.

```
./alerts clicksend_server_add \
  --username name@example.com \
  --token 12345678-ABCDEFGH-12345678-ABCDEFGH \
  --senderid "+15551234567" \
  --to-address "+15550987654"
```

> **Note**
>
> - For the `--username` and `--token` flags, see API Credentials in the ClickSend documentation.
> - The `--senderid` flag is mandatory for the U.S. and Canada. For more information, see Toll-Free Number (TFN) Verification in the ClickSend documentation.
> - For the `--language` flag, see What Language Locales Qumulo Alerts Supports (page 10). The consumer processes for email (page 49), IFTTT (page 52), and SMS (ClickSend) (page 54) integrations translate messages into the recipient's native language.
> - For the `--timezone` flag, see Converting Time Zones (page 11).

The following is example JSON output from the command.

```
[{
  "language": "en_GB",
  "senderid": "+15551234567",
  "timezone": "UTC",
  "to_address": "+15550987654",
  "username": "name@example.com"
}]
```

To Test Integration with ClickSend

Use the `./alerts clicksend_server_test` command.

> **Note**
> For integration testing to complete successfully, the `--to-address` flag must be configured already.

A successful response returns the `[{ "ok": true }]` JSON output. In addition, the recipient's phone number receives a test message.

Configuring Alarm and Alert Collection from a Qumulo Cluster

This section explains how to configure Qumulo Alerts to collect alarms and alerts from a Qumulo Cluster.

Collecting Information about Specific Alarms

Use the `./alerts cluster_add` command and specify the fully qualified domain name (FQDN) of your Qumulo cluster, your long-lived access token for the Qumulo REST API, and the plugins or plugin categories to include or exclude from monitoring.

In the following example, we include the plugins `Disks` and `Nodes`.

```
./alerts cluster_add \
  --name cluster.example.com \
  --token 12345678901234567890 \
  -pi Disks \
  -pi Nodes
```

The following is example JSON output from the command.

```
[{
  "frequency": 1,
  "id": 1,
  "name": "cluster.example.com",
  "nlb": false,
  "plugins": [{
    "category": "Alarms",
    "description": "Get Disk State Information",
    "frequency": null,
    "name": "Disks"
  },{
    "category": "Alarms",
    "description": "Get Cluster Node Failures",
    "frequency": null,
    "name": "Nodes"
  }],
  "port": 8000
}]
```

> **Note**
>
> - For the `--nlb` flag, the `false` setting requires floating IP address configuration.
> - To prevent spreading the load of a plugin's API requests across all nodes in a Qumulo cluster, each alarm or alert plugin that you configure communicates with your cluster by using either a network load balancer or floating IPs. You can configure *one*—but not both—of these communication methods.

Collecting Information about All Alarms

Use the `./alerts cluster_add` command and specify the fully qualified domain name (FQDN) of your Qumulo cluster, your long-lived access token for the Qumulo REST API, and the plugins or plugin categories to include or exclude from monitoring.

In the following example, we include the `Alarms` category.

```
./alerts cluster_add \
  --name cluster.example.com \
  --token 12345678901234567890 \
  -pc Alarms
```

The following is example JSON output from the command. This example output is truncated.

```
[{
  "frequency": 1,
  "id": 1,
  "name": "cluster.example.com",
  "nlb": false,
  "plugins": [{
    "category": "Alarms",
    "description": "Get Disk State Information",
    "frequency": null,
    "name": "Disks"
  },{
    "category": "Alarms",
    "description": "Get Cluster Node Failures",
    "frequency": null,
    "name": "Nodes"
  },{
    "category": "Alarms",
    "description": "Get Fan Failures",
    "frequency": null,
    "name": "Fans"
  },{
    "category": "Alarms",
    "description": "Get CPU Overtemp",
    "frequency": null,
    "name": "CPU"
  },
  ...
  ],
  "port": 8000
}]
```

Collecting Information about All Alarms, Alerts, and Informational Messages

Use the `./alerts cluster_add` command and specify the fully qualified domain name (FQDN) of your Qumulo cluster, your long-lived access token for the Qumulo REST API, and the plugins or plugin categories to include or exclude from monitoring.

In the following example, we include the `Alarms`, `Alerts`, and `Informational` categories.

```
./alerts cluster_add \
  --name cluster.example.com \
  --token 12345678901234567890 \
  -pc Alarms \
  -pc Alerts \
  -pc Informational
```

The following is example JSON output from the command. This example output is truncated.

```
[{
  "frequency": 1,
  "id": 1,
  "name": "cluster.example.com",
  "nlb": false,
  "plugins": [{
    "category": "Alarms",
    "description": "Get Disk State Information",
    "frequency": null,
    "name": "Disks"
  },{
    "category": "Alarms",
    "description": "Get Cluster Node Failures",
    "frequency": null,
    "name": "Nodes"
  },{
    "category": "Alerts",
    "description": "Get Active Directory State",
    "frequency": null,
    "name": "AD"
  },{
    "category": "Alerts",
    "description": "Get Audit Status",
    "frequency": null,
    "name": "Audit"
  },{
    "category": "Alerts",
    "description": "Get Cluster Volume Capacity",
    "frequency": null,
    "name": "Capacity"
  },
  ...
  ],
  "port": 8000
}]
```

Collecting Information about Specific Alarms

Use the `./alerts cluster_add` command and specify the fully qualified domain name (FQDN) of your Qumulo cluster, your long-lived access token for the Qumulo REST API, and the plugins or plugin categories to include or exclude from monitoring.

In the following example, we include the plugins `Disks` and `Nodes`.

```
./alerts cluster_add \
  --name cluster.example.com \
  --token 12345678901234567890 \
  -pi Disks \
  -pi Nodes
```

The following is example JSON output from the command.

```
[{
  "frequency": 1,
  "id": 1,
  "name": "cluster.example.com",
  "nlb": false,
  "plugins": [{
    "category": "Alarms",
    "description": "Get Disk State Information",
    "frequency": null,
    "name": "Disks"
  },{
    "category": "Alarms",
    "description": "Get Cluster Node Failures",
    "frequency": null,
    "name": "Nodes"
  }],
  "port": 8000
}]
```

Note

- For the `--nlb` flag, the `false` setting requires floating IP address configuration.
- To prevent spreading the load of a plugin's API requests across all nodes in a Qumulo cluster, each alarm or alert plugin that you configure communicates with your cluster by using either a network load balancer or floating IPs. You can configure *one*—but not both—of these communication methods.

Collecting Information about All Alarms

Use the `./alerts cluster_add` command and specify the fully qualified domain name (FQDN) of your Qumulo cluster, your long-lived access token for the Qumulo REST API, and the plugins or plugin categories to include or exclude from monitoring.

In the following example, we include the `Alarms` category.

```
./alerts cluster_add \
  --name cluster.example.com \
  --token 12345678901234567890 \
  -pc Alarms
```

The following is example JSON output from the command. This example output is truncated.

```
[{
  "frequency": 1,
  "id": 1,
  "name": "cluster.example.com",
  "nlb": false,
  "plugins": [{
    "category": "Alarms",
    "description": "Get Disk State Information",
    "frequency": null,
    "name": "Disks"
  },{
    "category": "Alarms",
    "description": "Get Cluster Node Failures",
    "frequency": null,
    "name": "Nodes"
  },{
    "category": "Alarms",
    "description": "Get Fan Failures",
    "frequency": null,
    "name": "Fans"
  },{
    "category": "Alarms",
    "description": "Get CPU Overtemp",
    "frequency": null,
    "name": "CPU"
  },
  ...
  ],
  "port": 8000
}]
```


Collecting Information about All Alarms, Alerts, and Informational Messages

Use the `./alerts cluster_add` command and specify the fully qualified domain name (FQDN) of your Qumulo cluster, your long-lived access token for the Qumulo REST API, and the plugins or plugin categories to include or exclude from monitoring.

In the following example, we include the `Alarms`, `Alerts`, and `Informational` categories.

```
./alerts cluster_add \
  --name cluster.example.com \
  --token 12345678901234567890 \
  -pc Alarms \
  -pc Alerts \
  -pc Informational
```

The following is example JSON output from the command. This example output is truncated.

```
[{
  "frequency": 1,
  "id": 1,
  "name": "cluster.example.com",
  "nlb": false,
  "plugins": [{
    "category": "Alarms",
    "description": "Get Disk State Information",
    "frequency": null,
    "name": "Disks"
  },{
    "category": "Alarms",
    "description": "Get Cluster Node Failures",
    "frequency": null,
    "name": "Nodes"
  },{
    "category": "Alerts",
    "description": "Get Active Directory State",
    "frequency": null,
    "name": "AD"
  },{
    "category": "Alerts",
    "description": "Get Audit Status",
    "frequency": null,
    "name": "Audit"
  },{
    "category": "Alerts",
    "description": "Get Cluster Volume Capacity",
    "frequency": null,
    "name": "Capacity"
  },
  ...
  ],
  "port": 8000
}]
```

Connecting to Grafana to View Visualizations of Qumulo Alerts Data

This section explains how to connect to the Qumulo Alerts instance of Grafana to view visualizations and information about your Qumulo cluster from prebuilt dashboards.

To Connect to the Grafana Endpoint

1. In a browser, navigate to the hostname of your running Grafana instance on port 3000. Grafana was started when you executed the script

   ```
   ./start-docker-qumulo-alerts.sh
   ```

   For example, if your docker instance of Grafana is on IP address 203.0.113.0:

   ```
   http://203.0.113.0:3000
   ```

2. When prompted, enter the following:

   - Login: `qumulo`
   - Password: `Admin123`.

   Grafana displays visualizations and information about your cluster.

3. Change the default Grafana password.

To Connect to the Grafana Endpoint

1. In a browser, navigate to the hostname of your running Grafana instance on port 3000. Grafana was started when you executed the script

   ```
   ./start-docker-qumulo-alerts.sh
   ```

   For example, if your docker instance of Grafana is on IP address 203.0.113.0:

   ```
   http://203.0.113.0:3000
   ```

2. When prompted, enter the following:

- Login: `qumulo`
- Password: `Admin123`.

Grafana displays visualizations and information about your cluster.

3. Change the default Grafana password.