



Kubernetes

Cloud Insights

NetApp
April 08, 2024

This PDF was generated from https://docs.netapp.com/us-en/cloudinsights/kubernetes_landing_page.html on April 08, 2024. Always check docs.netapp.com for the latest.

Table of Contents

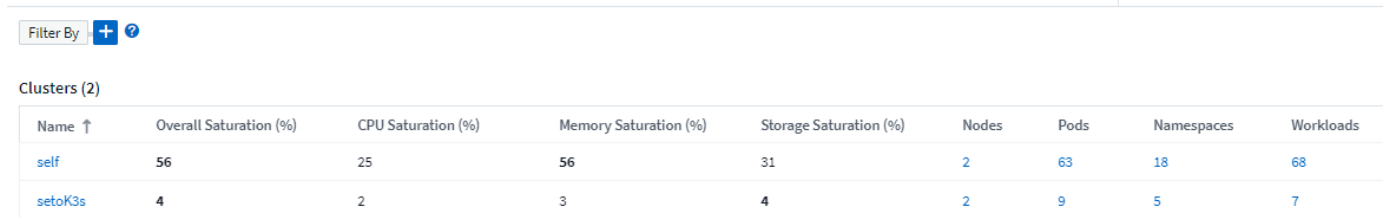
- Kubernetes 1
 - Kubernetes Cluster Overview 1
 - Before Installing or Upgrading the NetApp Kubernetes Monitoring Operator 2
 - Kubernetes Monitoring Operator Installation and Configuration 7
 - NetApp Kubernetes Monitoring Operator Configuration Options 25



Kubernetes

Kubernetes Cluster Overview

The Cloud Insights Kubernetes Explorer is a powerful tool for displaying the overall health and usage of your Kubernetes clusters and allows you to easily drill down into areas of investigation.

Clicking on **Dashboards > Kubernetes Explorer** opens the Kubernetes Cluster list page. This overview page contains table of the Kubernetes clusters in your environment.



Filter By  

Clusters (2)

Name ↑	Overall Saturation (%)	CPU Saturation (%)	Memory Saturation (%)	Storage Saturation (%)	Nodes	Pods	Namespaces	Workloads
self	56	25	56	31	2	63	18	68
setoK3s	4	2	3	4	2	9	5	7

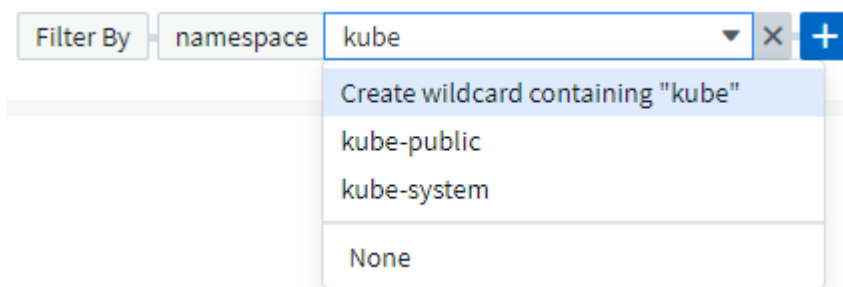
Cluster list



The cluster list displays the following information for each cluster in your environment:

- Cluster **Name**. Clicking on a cluster name will open the [detail page](#) for that cluster.
- **Saturation** percentages. Overall Saturation is the highest of CPU, Memory, or Storage Saturation.
- Number of **Nodes** in the cluster. Clicking this number will open the Node list page.
- Number of **Pods** in the cluster. Clicking this number will open the Pod list page.
- Number of **Namespaces** in the cluster. Clicking this number will open the Namespace list page.
- Number of **Workloads** in the cluster. Clicking this number will open the Workload list page.

Refining the Filter

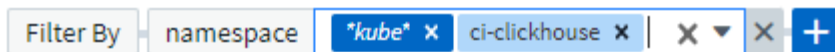
When you are filtering, as you begin typing you are presented with the option to create a **wildcard filter** based on the current text. Selecting this option will return all results that match the wildcard expression. You can also create **expressions** using NOT or AND, or you can select the "None" option to filter for null values in the field.



Filter By namespace kube  

- Create wildcard containing "kube"
- kube-public
- kube-system
- None

Filters based on wildcards or expressions (e.g. NOT, AND, "None", etc.) display in dark blue in the filter field. Items that you select directly from the list are displayed in light blue.



Kubernetes filters are contextual, meaning for example that if you are on a specific node page, the pod_name filter only lists pods related to that node. Moreover, if you apply a filter for a specific namespace, then the pod_name filter will list only pods on that node *and* in that namespace.

Note that Wildcard and Expression filtering works with text or lists but not with numerics, dates or booleans.

Before Installing or Upgrading the NetApp Kubernetes Monitoring Operator

Read this information before installing or upgrading your NetApp Kubernetes Monitoring Operator

Pre-requisites:

- If you are using a custom or private docker repository, follow the instructions in the Using a custom or private docker repository section
- NetApp Kubernetes Monitoring Operator installation is supported with Kubernetes version 1.20 or greater.
- When Cloud Insights is monitoring the backend storage and Kubernetes is used with the Docker container runtime, Cloud Insights can display pod-to-PV-to-storage mappings and metrics for NFS and iSCSI; other runtimes only show NFS.
- Beginning August 2022, the NetApp Kubernetes Monitoring Operator includes support for Pod Security Policy (PSP). You must upgrade to the latest NetApp Kubernetes Monitoring Operator if your environment uses PSP.
- If you are running on OpenShift 4.6 or higher, you must follow the OpenShift Instructions below in addition to ensuring these pre-requisites are met.
- Monitoring is only installed on Linux nodes
Cloud Insights supports monitoring of Kubernetes nodes that are running Linux, by specifying a Kubernetes node selector that looks for the following Kubernetes labels on these platforms:

Platform	Label
Kubernetes v1.20 and above	Kubernetes.io/os = linux
Rancher + cattle.io as orchestration/Kubernetes platform	cattle.io/os = linux

- The NetApp Kubernetes Monitoring Operator and its dependencies (telegraf, kube-state-metrics, fluentbit, etc.) are not supported on nodes that are running with Arm64 architecture.
- The following commands must be available: curl, kubectl. The docker command is required for an optional installation step. For best results, add these commands to the PATH. Note that kubectl needs to be configured with access to the following kubernetes objects at a minimum: agents, clusterroles, clusterrolebindings, customresourcedefinitions, deployments, namespaces, roles, rolebindings, secrets, serviceaccounts, and services. See here for an example .yaml file with these minimum clusterrole privileges.
- The host you will use for the NetApp Kubernetes Monitoring Operator installation must have kubectl configured to communicate with the target K8s cluster, and have Internet connectivity to your Cloud Insights environment.

- If you are behind a proxy during installation, or when operating the K8s cluster to be monitored, follow the instructions in the [Configuring Proxy Support](#) section.
- The NetApp Kubernetes Monitoring Operator installs its own kube-state-metrics to avoid conflict with any other instances.
For accurate audit and data reporting, it is strongly recommended to synchronize the time on the Agent machine using Network Time Protocol (NTP) or Simple Network Time Protocol (SNTP).
- If you are re-deploying the Operator (i.e. you are updating or replacing it), there is no need to create a *new* API token; you can re-use the previous token.
- Also note that if you have a recent NetApp Kubernetes Monitoring Operator installed and are using an API access token that is renewable, expiring tokens will automatically be replaced by new/refreshed API access tokens.
- Network monitoring:
 - Requires Linux kernel version 4.18.0 and above
 - Photon OS is not supported.

Configuring the Operator

In newer versions of the operator, most commonly modified settings can be configured in the *AgentConfiguration* custom resource. You can edit this resource before deploying the operator by editing the *operator-config.yaml* file. This file includes commented out examples of some settings. See the list of [available settings](#) for the most recent version of the operator.

You can also edit this resource after the operator has been deployed using the following command:

```
kubectl -n netapp-monitoring edit AgentConfiguration
```

To determine if your deployed version of the operator supports *AgentConfiguration*, run the following command:

```
kubectl get crd agentconfigurations.monitoring.netapp.com
```

If you see an “Error from server (NotFound)” message, your operator must be upgraded before you can use the *AgentConfiguration*.

Important Things to Note Before You Start

If you are running with a [proxy](#), have a [custom repository](#), or are using [OpenShift](#), read the following sections carefully.

Also read about [Permissions](#).

If you are upgrading from a previous installation, read the [Upgrading](#) information.

Configuring Proxy Support

There are two places where you may use a proxy in your environment in order to install the NetApp Kubernetes Monitoring Operator. These may be the same or separate proxy systems:

- Proxy needed during execution of the installation code snippet (using "curl") to connect the system where the snippet is executed to your Cloud Insights environment
- Proxy needed by the target Kubernetes cluster to communicate with your Cloud Insights environment

If you use a proxy for either or both of these, to install the NetApp Kubernetes Operating Monitor you must first ensure that your proxy is configured to allow good communication to your Cloud Insights environment. For example, from the servers/VMs from which you wish to install the Operator, you need to be able to access Cloud Insights and be able to download binaries from Cloud Insights.

For the proxy used to install the NetApp Kubernetes Operating Monitor, before installing the Operator, set the *http_proxy*/*https_proxy* environment variables. For some proxy environments, you may also need to set the *no_proxy* environment variable.

To set the variable(s), perform the following steps on your system **before** installing the NetApp Kubernetes Monitoring Operator:

1. Set the *https_proxy* and/or *http_proxy* environment variable(s) for the current user:
 - a. If the proxy being setup does not have Authentication (username/password), run the following command:

```
export https_proxy=<proxy_server>:<proxy_port>
```

- b. If the proxy being setup does have Authentication (username/password), run this command:

```
export
http_proxy=<proxy_username>:<proxy_password>@<proxy_server>:<proxy_port>
```

For the proxy used for your Kubernetes cluster to communicate with your Cloud Insights environment, install the NetApp Kubernetes Monitoring Operator after reading all of these instructions.

Configure the proxy section of AgentConfiguration in operator-config.yaml before deploying the NetApp Kubernetes Monitoring Operator.

```

agent:
  ...
  proxy:
    server: <server for proxy>
    port: <port for proxy>
    username: <username for proxy>
    password: <password for proxy>

    # In the noproxy section, enter a comma-separated list of
    # IP addresses and/or resolvable hostnames that should bypass
    # the proxy
    noproxy: <comma separated list>

    isTelegrafProxyEnabled: true
    isFluentbitProxyEnabled: <true or false> # true if Events Log enabled
    isCollectorsProxyEnabled: <true or false> # true if Network
Performance and Map enabled
    isAuProxyEnabled: <true or false> # true if AU enabled
  ...
  ...

```

Using a custom or private docker repository

By default, the NetApp Kubernetes Monitoring Operator will pull container images from the Cloud Insights repository. If you have a Kubernetes cluster used as the target for monitoring, and that cluster is configured to only pull container images from a custom or private Docker repository or container registry, you must configure access to the containers needed by the NetApp Kubernetes Monitoring Operator.

Run the “Image Pull Snippet” from the NetApp Monitoring Operator install tile. This command will log into the Cloud Insights repository, pull all image dependencies for the operator, and log out of the Cloud Insights repository. When prompted, enter the provided repository temporary password. This command downloads all images used by the operator, including for optional features. See below for which features these images are used for.

Core Operator Functionality and Kubernetes Monitoring

- netapp-monitoring
- kube-rbac-proxy
- kube-state-metrics
- telegraf
- distroless-root-user

Events Log

- fluent-bit
- kubernetes-event-exporter

Network Performance and Map

- ci-net-observer

Push the operator docker image to your private/local/enterprise docker repository according to your corporate policies. Ensure that the image tags and directory paths to these images in your repository are consistent with those in the Cloud Insights repository.

Edit the monitoring-operator deployment in `operator-deployment.yaml`, and modify all image references to use your private Docker repository.

```
image: <docker repo of the enterprise/corp docker repo>/kube-rbac-  
proxy:<kube-rbac-proxy version>  
image: <docker repo of the enterprise/corp docker repo>/netapp-  
monitoring:<version>
```

Edit the AgentConfiguration in `operator-config.yaml` to reflect the new docker repo location. Create a new imagePullSecret for your private repository, for more details see <https://kubernetes.io/docs/tasks/configure-pod-container/pull-image-private-registry/>

```
agent:  
  ...  
  # An optional docker registry where you want docker images to be pulled  
  # from as compared to CI's docker registry  
  # Please see documentation link here: https://docs.netapp.com/us-  
  # en/cloudinsights/task_config_telegraf_agent_k8s.html#using-a-custom-or-  
  # private-docker-repository  
  dockerRepo: your.docker.repo/long/path/to/test  
  # Optional: A docker image pull secret that maybe needed for your  
  # private docker registry  
  dockerImagePullSecret: docker-secret-name
```

OpenShift Instructions

If you are running on OpenShift 4.6 or higher, you must edit the AgentConfiguration in `operator-config.yaml` to enable the `runPrivileged` setting:

```
# Set runPrivileged to true SELinux is enabled on your kubernetes nodes  
runPrivileged: true
```

Openshift may implement an added level of security that may block access to some Kubernetes components.

Permissions

If the cluster you are monitoring contains Custom Resources which do not have a ClusterRole which [aggregates to view](#), you will need to manually grant the operator access to these resources to monitor them with Event Logs.

1. Edit `operator-additional-permissions.yaml` before installing, or after installing edit the resource `ClusterRole/<namespace>-additional-permissions`
2. Create a new rule for the desired apiGroups and resources with the verbs ["get", "watch", "list"]. See <https://kubernetes.io/docs/reference/access-authn-authz/rbac/>
3. Apply your changes to the cluster

Tolerations and Taints

The `netapp-ci-teleggraf-ds`, `netapp-ci-fluent-bit-ds`, and `netapp-ci-net-observer-l4-ds` DaemonSets must schedule a pod on every node in your cluster in order to correctly collect data on all nodes. The operator has been configured to tolerate some well known **taints**. If you have configured any custom taints on your nodes, thus preventing pods from running on every node, you can create a **toleration** for those taints in the [AgentConfiguration](#). If you have applied custom taints to all nodes in your cluster, you must also add the necessary tolerations to the operator deployment to allow the operator pod to be scheduled and executed.

Learn More about Kubernetes [Taints and Tolerations](#).

Return to the [NetApp Kubernetes Monitoring Operator Installation page](#)

Kubernetes Monitoring Operator Installation and Configuration

Cloud Insights offers the **NetApp Kubernetes Monitoring Operator** (NKMO) for Kubernetes collection. When adding a data collector, simply choose the "Kubernetes" tile.



If you have Cloud Insights Federal Edition, your installation and configuration instructions may be different than the instructions on this page. Follow the instructions in Cloud Insights to install the NetApp Kubernetes Monitoring Operator.

Choose a Data Collector to Monitor

kubernetes

kubernetes

Kubernetes

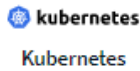
The Operator and the data collectors are downloaded from the Cloud Insights Docker Registry. Once installed, NKMO then manages any Operator-compatible collectors deployed in the Kubernetes cluster nodes to acquire data, including managing the life cycle of those collectors. Following this chain, data is acquired from the collectors and sent through to Cloud Insights.

Before installing the NetApp Kubernetes Monitoring Operator



Read the [Before Installing or Upgrading](#) pre-requisites documentation before installing or upgrading the NetApp Kubernetes Monitoring Operator.

Installing the NetApp Kubernetes Monitoring Operator



Deploy NetApp Monitoring Operator

Quickly install and configure a Kubernetes Operator to send cluster information to Cloud Insights.

Select existing API Access Token or create a new one

KEY2024 (...vw6NdM) ▼

[+ API Access Token](#)

Production Best Practices [?](#)

Installation Instructions

Need Help?

Please review the [pre-requisites](#) for installing the NetApp Kubernetes Monitoring Operator.
To update an existing operator installation please follow [these steps](#).

- 1 Define Kubernetes cluster name and namespace
- Provide the Kubernetes cluster name and specify a namespace for deploying the monitoring components.

Cluster	Namespace
<input type="text" value="clustername"/>	<input type="text" value="netapp-monitoring"/>

- 2 Download the operator YAML files
- Execute the following download command in a *bash* prompt.

[Copy Download Command Snippet](#)

[⊞ Reveal Download Command Snippet](#)

This snippet includes a unique access key that is valid for 24 hours.

3 Optional: Upload the operator images to your private repository

By default, the operator pulls container images from the Cloud Insights repository. To use a private repository, download the required images using the Image Pull command. Then upload them to your private repository maintaining the same tags and directory structure. Finally, update the image paths in `operator-deployment.yaml` and the docker repository settings in `operator-config.yaml`. For more information review [the documentation](#).

Copy Image Pull Snippet

⊞ Reveal Image Pull Snippet

Copy Repository Password

⊞ Reveal Repository Password

This password is valid for 24 hours.

4 Optional: Review available configuration options

Configure custom options such as proxy and private repository settings. Review the [instructions and available options](#).

5 Deploy the operator (create new or upgrade existing)

Execute the `kubectl` snippet to apply the following operator YAML files.

- `operator-setup.yaml` - Create the operator's dependencies.
- `operator-secrets.yaml` - Create secrets holding your API key.
- `operator-deployment.yaml`, `operator-cr.yaml` - Deploy the NetApp Kubernetes Monitoring Operator.
- `operator-config.yaml` - Apply the configuration settings if not already present.

Copy kubectl Apply Snippet

⊞ Reveal kubectl Apply Snippet

After deploying the operator, **delete or securely store `operator-secrets.yaml`**.

6

Next

Steps to install NetApp Kubernetes Monitoring Operator agent on Kubernetes:

1. Enter a unique cluster name and namespace. If you are [upgrading](#) from a previous Kubernetes Operator, use the same cluster name and namespace.
2. Once these are entered, you can copy the Download Command snippet to the clipboard.
3. Paste the snippet into a `bash` window and execute it. The Operator installation files will be downloaded. Note that the snippet has a unique key and is valid for 24 hours.
4. If you have a custom or private repository, copy the optional Image Pull snippet, paste it into a `bash` shell and execute it. Once the images have been pulled, copy them to your private repository. Be sure to maintain the same tags and folder structure. Update the paths in `operator-deployment.yaml` as well as the docker repository settings in `operator-config.yaml`.
5. If desired, review available configuration options such as proxy or private repository settings. You can read more about [configuration options](#).
6. When you are ready, deploy the Operator by copying the `kubectl` Apply snippet, downloading it, and executing it.
7. The installation proceeds automatically. When it is complete, click the *Next* button.
8. When installation is complete, click the *Next* button. Be sure to also delete or securely store the `operator-secrets.yaml` file.

Read more about [configuring proxy](#).

Read more about [using a custom/private docker repository](#).

Kubernetes EMS log collection is enabled by default when installing the NetApp Kubernetes Monitoring Operator. To disable this collection following installation, click the **Modify Deployment** button at the top of the Kubernetes cluster detail page, and un-select "Log collection".

kubernetes
Kubernetes

Modify Deployment

Cluster Information

Kubernetes Cluster	Log Collection
k3s-2nodes	Enabled - Online

Deployment Options [Need Help?](#)

☒ Log Collection

[Cancel](#) [Complete Modification](#)

This screen also shows current Log Collection status. Below are the possible states:

- Disabled
- Enabled
- Enabled - Installation in progress
- Enabled - Offline
- Enabled - Online
- Error - API Key has insufficient permissions

Upgrading

Upgrading to the latest NetApp Kubernetes Monitoring Operator

Determine whether an AgentConfiguration exists with the existing Operator (if your namespace is not the default *netapp-monitoring*, substitute the appropriate namespace):

```
kubectl -n netapp-monitoring get agentconfiguration netapp-monitoring-configuration
```

If an AgentConfiguration exists:

- [Install](#) the latest Operator over the existing Operator.
 - Ensure you are [pulling the latest container images](#) if you are using a custom repository.

If the AgentConfiguration does not exist:

- Make note of your cluster name as recognized by Cloud Insights (if your namespace is not the default netapp-monitoring, substitute the appropriate namespace):

```
kubectl -n netapp-monitoring get agent -o
jsonpath='{.items[0].spec.cluster-name}'
```

- Create a backup of the existing Operator (if your namespace is not the default netapp-monitoring, substitute the appropriate namespace):

```
kubectl -n netapp-monitoring get agent -o yaml > agent_backup.yaml
```

- [Uninstall](#) the existing Operator.
- [Install](#) the latest Operator.
 - Use the same cluster name.
 - After downloading the latest Operator YAML files, port any customizations found in agent_backup.yaml to the downloaded operator-config.yaml before deploying.
 - Ensure you are [pulling the latest container images](#) if you are using a custom repository.

Stopping and Starting the Netapp Kubernetes Monitoring Operator

To stop the Netapp Kubernetes Monitoring Operator:

```
kubectl -n netapp-monitoring scale deploy monitoring-operator --replicas=0
```

To start the Netapp Kubernetes Monitoring Operator:

```
kubectl -n netapp-monitoring scale deploy monitoring-operator --replicas=1
```

Uninstalling

To remove the NetApp Kubernetes Monitoring Operator

Note that the default namespace for the NetApp Kubernetes Monitoring Operator is "netapp-monitoring". If you have set your own namespace, substitute that namespace in these and all subsequent commands and files.

Newer versions of the monitoring operator can be uninstalled with the following commands:

```
kubectl -n <NAMESPACE> delete agent -l installed-by=nkmo-<NAMESPACE>
kubectl -n <NAMESPACE> delete
clusterrole,clusterrolebinding,crd,svc,deploy,role,rolebinding,secret,sa
-l installed-by=nkmo-<NAMESPACE>
```

If the monitoring operator was deployed in its own dedicated namespace, delete the namespace:

```
kubectl delete ns <NAMESPACE>
```

If the first command returns “No resources found”, use the following instructions to uninstall older versions of the monitoring operator.

Execute each of the following commands in order. Depending on your current installation, some of these commands may return ‘object not found’ messages. These messages may be safely ignored.

```
kubectl -n <NAMESPACE> delete agent agent-monitoring-netapp
kubectl delete crd agents.monitoring.netapp.com
kubectl -n <NAMESPACE> delete role agent-leader-election-role
kubectl delete clusterrole agent-manager-role agent-proxy-role agent-
metrics-reader <NAMESPACE>-agent-manager-role <NAMESPACE>-agent-proxy-role
<NAMESPACE>-cluster-role-privileged
kubectl delete clusterrolebinding agent-manager-rolebinding agent-proxy-
rolebinding agent-cluster-admin-rolebinding <NAMESPACE>-agent-manager-
rolebinding <NAMESPACE>-agent-proxy-rolebinding <NAMESPACE>-cluster-role-
binding-privileged
kubectl delete <NAMESPACE>-psp-nkmo
kubectl delete ns <NAMESPACE>
```

If a Security Context Constraint was previously-created:

```
kubectl delete scc telegraf-hostaccess
```

About Kube-state-metrics

The NetApp Kubernetes Monitoring Operator installs kube-state-metrics automatically; no user interaction is needed.

kube-state-metrics Counters

Use the following links to access information for these kube state metrics counters:

1. [ConfigMap Metrics](#)
2. [DaemonSet Metrics](#)
3. [Deployment Metrics](#)
4. [Ingress Metrics](#)
5. [Namespace Metrics](#)
6. [Node Metrics](#)
7. [Persistent Volume Metrics](#)
8. [Persistent Volume Claim Metrics](#)
9. [Pod Metrics](#)

10. [ReplicaSet metrics](#)
11. [Secret metrics](#)
12. [Service metrics](#)
13. [StatefulSet metrics](#)

```
== Configuring the Operator
```

In newer versions of the operator, most commonly modified settings can be configured in the *AgentConfiguration* custom resource. You can edit this resource before deploying the operator by editing the *operator-config.yaml* file. This file includes commented out examples of some settings. See the list of [available settings](#) for the most recent version of the operator.

You can also edit this resource after the operator has been deployed using the following command:

```
kubectl -n netapp-monitoring edit AgentConfiguration
```

To determine if your deployed version of the operator supports AgentConfiguration, run the following command:

```
kubectl get crd agentconfigurations.monitoring.netapp.com
```

If you see an "Error from server (NotFound)" message, your operator must be upgraded before you can use the AgentConfiguration.

Configuring Proxy Support

There are two places where you may use a proxy in your environment in order to install the NetApp Kubernetes Monitoring Operator. These may be the same or separate proxy systems:

- Proxy needed during execution of the installation code snippet (using "curl") to connect the system where the snippet is executed to your Cloud Insights environment
- Proxy needed by the target Kubernetes cluster to communicate with your Cloud Insights environment

If you use a proxy for either or both of these, in order to install the NetApp Kubernetes Operating Monitor you must first ensure that your proxy is configured to allow good communication to your Cloud Insights environment. If you have a proxy and can access Cloud Insights from the server/VM from which you wish to install the Operator, then your proxy is likely configured properly.

For the proxy used to install the NetApp Kubernetes Operating Monitor, before installing the Operator, set the *http_proxy*/*https_proxy* environment variables. For some proxy environments, you may also need to set the *no_proxy* environment variable.

To set the variable(s), perform the following steps on your system **before** installing the NetApp Kubernetes Monitoring Operator:

1. Set the *https_proxy* and/or *http_proxy* environment variable(s) for the current user:

- a. If the proxy being setup does not have Authentication (username/password), run the following command:

```
export https_proxy=<proxy_server>:<proxy_port>
```

- b. If the proxy being setup does have Authentication (username/password), run this command:

```
export
http_proxy=<proxy_username>:<proxy_password>@<proxy_server>:<proxy_port>
```

For the proxy used for your Kubernetes cluster to communicate with your Cloud Insights environment, install the NetApp Kubernetes Monitoring Operator after reading all of these instructions.

Configure the proxy section of AgentConfiguration in operator-config.yaml before deploying the NetApp Kubernetes Monitoring Operator.

```
agent:
  ...
  proxy:
    server: <server for proxy>
    port: <port for proxy>
    username: <username for proxy>
    password: <password for proxy>

    # In the noproxy section, enter a comma-separated list of
    # IP addresses and/or resolvable hostnames that should bypass
    # the proxy
    noproxy: <comma separated list>

    isTelegrafProxyEnabled: true
    isFluentbitProxyEnabled: <true or false> # true if Events Log enabled
    isCollectorsProxyEnabled: <true or false> # true if Network
Performance and Map enabled
    isAuProxyEnabled: <true or false> # true if AU enabled
  ...
  ...
```

Using a custom or private docker repository

By default, the NetApp Kubernetes Monitoring Operator will pull container images from the Cloud Insights repository. If you have a Kubernetes cluster used as the target for monitoring, and that cluster is configured to only pull container images from a custom or private Docker repository or container registry, you must configure access to the containers needed by the NetApp Kubernetes Monitoring Operator.

Run the “Image Pull Snippet” from the NetApp Monitoring Operator install tile. This command will log into the Cloud Insights repository, pull all image dependencies for the operator, and log out of the Cloud Insights repository. When prompted, enter the provided repository temporary password. This command downloads all images used by the operator, including for optional features. See below for which features these images are used for.

Core Operator Functionality and Kubernetes Monitoring

- netapp-monitoring
- ci-kube-rbac-proxy
- ci-ksm
- ci-teleggraf
- distroless-root-user

Events Log

- ci-fluent-bit
- ci-kubernetes-event-exporter

Network Performance and Map

- ci-net-observer

Push the operator docker image to your private/local/enterprise docker repository according to your corporate policies. Ensure that the image tags and directory paths to these images in your repository are consistent with those in the Cloud Insights repository.

Edit the monitoring-operator deployment in operator-deployment.yaml, and modify all image references to use your private Docker repository.

```
image: <docker repo of the enterprise/corp docker repo>/kube-rbac-  
proxy:<ci-kube-rbac-proxy version>  
image: <docker repo of the enterprise/corp docker repo>/netapp-  
monitoring:<version>
```

Edit the AgentConfiguration in operator-config.yaml to reflect the new docker repo location. Create a new imagePullSecret for your private repository, for more details see <https://kubernetes.io/docs/tasks/configure-pod-container/pull-image-private-registry/>

```
agent:
  ...
  # An optional docker registry where you want docker images to be pulled
  # from as compared to CI's docker registry
  # Please see documentation link here: https://docs.netapp.com/us-
  # en/cloudinsights/task_config_telegraf_agent_k8s.html#using-a-custom-or-
  # private-docker-repository
  dockerRepo: your.docker.repo/long/path/to/test
  # Optional: A docker image pull secret that maybe needed for your
  # private docker registry
  dockerImagePullSecret: docker-secret-name
```

OpenShift Instructions

If you are running on OpenShift 4.6 or higher, you must edit the AgentConfiguration in *operator-config.yaml* to enable the *runPrivileged* setting:

```
# Set runPrivileged to true SELinux is enabled on your kubernetes nodes
runPrivileged: true
```

Openshift may implement an added level of security that may block access to some Kubernetes components.

A Note About Secrets

To remove permission for the NetApp Kubernetes Monitoring Operator to view secrets cluster-wide, delete the following resources from the *operator-setup.yaml* file before installing:

```
ClusterRole/netapp-ci-<namespace>-agent-secret-clusterrole
ClusterRoleBinding/netapp-ci-<namespace>-agent-secret-clusterrolebinding
```

If this is an upgrade, also delete the resources from your cluster:

```
kubectl delete ClusterRole/netapp-ci-<namespace>-agent-secret-clusterrole
kubectl delete ClusterRoleBinding/netapp-ci-<namespace>-agent-secret-
clusterrolebinding
```

If Change Analysis is enabled, modify the *AgentConfiguration* or *operator-config.yaml* to uncomment the change-management section and include *kindsToIgnoreFromWatch*: *"secrets"* under the change-management section. Note the presence and position of single and double quotes in this line.

```
# change-management:
...
# # A comma separated list of kinds to ignore from watching from the
default set of kinds watched by the collector
# # Each kind will have to be prefixed by its apigroup
# # Example: '"networking.k8s.io.networkpolicies,batch.jobs",
"authorization.k8s.io.subjectaccessreviews"'
kindsToIgnoreFromWatch: '"secrets"'
...
```

Verifying Kubernetes Checksums

The Cloud Insights agent installer performs integrity checks, but some users may want to perform their own verifications before installing or applying downloaded artifacts. To perform a download-only operation (as opposed to the default download-and-install), these users can edit the agent installation command obtained from the UI and remove the trailing “install” option.

Follow these steps:

1. Copy the Agent Installer snippet as directed.
2. Instead of pasting the snippet into a command window, paste it into a text editor.
3. Remove the trailing “--install” from the command.
4. Copy the entire command from the text editor.
5. Now paste it into your command window (in a working directory) and run it.

- Download and install (default):

```
installerName=cloudinsights-rhel_centos.sh ... && sudo -E -H
./$installerName --download --install
```

- Download-only:

```
installerName=cloudinsights-rhel_centos.sh ... && sudo -E -H
./$installerName --download
```

The download-only command will download all required artifacts from Cloud Insights to the working directory. The artifacts include, but may not be limited to:

- an installation script
- an environment file
- YAML files
- a signed checksum file (sha256.signed)
- a PEM file (netapp_cert.pem) for signature verification

The installation script, environment file, and YAML files can be verified using visual inspection.

The PEM file can be verified by confirming its fingerprint to be the following:

```
1A918038E8E127BB5C87A202DF173B97A05B4996
```

More specifically,

```
openssl x509 -fingerprint -sha1 -noout -inform pem -in netapp_cert.pem
```

The signed checksum file can be verified using the PEM file:

```
openssl smime -verify -in sha256.signed -CAfile netapp_cert.pem -purpose any
```

Once all of the artifacts have been satisfactorily verified, the agent installation can be initiated by running:

```
sudo -E -H ./<installation_script_name> --install
```

Troubleshooting

Some things to try if you encounter problems setting up the NetApp Kubernetes Monitoring Operator:

Problem:	Try this:
I do not see a hyperlink/connection between my Kubernetes Persistent Volume and the corresponding back-end storage device. My Kubernetes Persistent Volume is configured using the hostname of the storage server.	Follow the steps to uninstall the existing Telegraf agent, then re-install the latest Telegraf agent. You must be using Telegraf version 2.0 or later, and your Kubernetes cluster storage must be actively monitored by Cloud Insights.

Problem:	Try this:
<p>I'm seeing messages in the logs resembling the following:</p> <pre>E0901 15:21:39.962145 1 reflector.go:178] k8s.io/kube-state-metrics/internal/store/builder.go:352: Failed to list *v1.MutatingWebhookConfiguration: the server could not find the requested resource E0901 15:21:43.168161 1 reflector.go:178] k8s.io/kube-state-metrics/internal/store/builder.go:352: Failed to list *v1.Lease: the server could not find the requested resource (get leases.coordination.k8s.io) etc.</pre>	<p>These messages may occur if you are running kube-state-metrics version 2.0.0 or above with Kubernetes versions below 1.20.</p> <p>To get the Kubernetes version:</p> <pre>kubectl version</pre> <p>To get the kube-state-metrics version:</p> <pre>kubectl get deploy/kube-state-metrics -o jsonpath='{..image}'</pre> <p>To prevent these messages from happening, users can modify their kube-state-metrics deployment to disable the following Leases:</p> <pre>mutatingwebhookconfigurations validatingwebhookconfigurations volumeattachments resources</pre> <p>More specifically, they can use the following CLI argument:</p> <pre>resources=certificatesigningrequests,configmaps,cron jobs,daemonsets, deployments,endpoints,horizontalpodautoscalers,ingr esses,jobs,limitranges, namespaces,networkpolicies,nodes,persistentvolume claims,persistentvolumes, poddisruptionbudgets,pods,replicasets,replicationcont rollers,resourcequotas, secrets,services,statefulsets,storageclasses</pre> <p>The default resource list is:</p> <pre>"certificatesigningrequests,configmaps,cronjobs,daem onsets,deployments, endpoints,horizontalpodautoscalers,ingresses,jobs,lea ses,limitranges, mutatingwebhookconfigurations,namespaces,network policies,nodes, persistentvolumeclaims,persistentvolumes,poddisrupti onbudgets,pods,replicasets, replicationcontrollers,resourcequotas,secrets,services, statefulsets,storageclasses, validatingwebhookconfigurations,volumeattachments"</pre>

Problem:	Try this:
<p>I see error messages from Telegraf resembling the following, but Telegraf does start up and run:</p> <pre>Oct 11 14:23:41 ip-172-31-39-47 systemd[1]: Started The plugin-driven server agent for reporting metrics into InfluxDB. Oct 11 14:23:41 ip-172-31-39-47 telegraf[1827]: time="2021-10-11T14:23:41Z" level=error msg="failed to create cache directory. /etc/telegraf/.cache/snowflake, err: mkdir /etc/telegraf/.ca che: permission denied. ignored\n" func="gosnowflake.(*defaultLogger).Errorf" file="log.go:120" Oct 11 14:23:41 ip-172-31-39-47 telegraf[1827]: time="2021-10-11T14:23:41Z" level=error msg="failed to open. Ignored. open /etc/telegraf/.cache/snowflake/ocsp_response_cache.j son: no such file or directory\n" func="gosnowflake.(*defaultLogger).Errorf" file="log.go:120" Oct 11 14:23:41 ip-172-31-39-47 telegraf[1827]: 2021- 10-11T14:23:41Z !! Starting Telegraf 1.19.3</pre>	<p>This is a known issue. Refer to This GitHub article for more details. As long as Telegraf is up and running, users can ignore these error messages.</p>
<p>On Kubernetes, my Telegraf pod(s) are reporting the following error:</p> <pre>"Error in processing mountstats info: failed to open mountstats file: /hostfs/proc/1/mountstats, error: open /hostfs/proc/1/mountstats: permission denied"</pre>	<p>If SELinux is enabled and enforcing, it is likely preventing the Telegraf pod(s) from accessing the /proc/1/mountstats file on the Kubernetes node. To overcome this restriction, edit the agentconfiguration, and enable the runPrivileged setting. For more details, refer to: https://docs.netapp.com/us-en/cloudinsights/task_config_telegraf_agent_k8s.html#openshift-instructions.</p>
<p>On Kubernetes, my Telegraf ReplicaSet pod is reporting the following error:</p> <pre>[inputs.prometheus] Error in plugin: could not load keypair /etc/kubernetes/pki/etcd/server.crt:/etc/kubernetes/pki/ etcd/server.key: open /etc/kubernetes/pki/etcd/server.crt: no such file or directory</pre>	<p>The Telegraf ReplicaSet pod is intended to run on a node designated as a master or for etcd. If the ReplicaSet pod is not running on one of these nodes, you will get these errors. Check to see if your master/etcd nodes have taints on them. If they do, add the necessary tolerations to the Telegraf ReplicaSet, telegraf-rs.</p> <p>For example, edit the ReplicaSet...</p> <pre>kubectrl edit rs telegraf-rs</pre> <p>...and add the appropriate tolerations to the spec. Then, restart the ReplicaSet pod.</p>

Problem:	Try this:
<p>I have a PSP/PSA environment. Does this affect my monitoring operator?</p>	<p>If your Kubernetes cluster is running with Pod Security Policy (PSP) or Pod Security Admission (PSA) in place, you must upgrade to the latest NetApp Kubernetes Monitoring Operator. Follow these steps to upgrade to the current NKMO with support for PSP/PSA:</p> <ol style="list-style-type: none"> 1. Uninstall the previous monitoring operator: <pre>kubect! delete agent agent-monitoring-netapp -n netapp-monitoring kubect! delete ns netapp-monitoring kubect! delete crd agents.monitoring.netapp.com kubect! delete clusterrole agent-manager-role agent-proxy-role agent-metrics-reader kubect! delete clusterrolebinding agent-manager-rolebinding agent-proxy-rolebinding agent-cluster-admin-rolebinding</pre> <ol style="list-style-type: none"> 2. Install the latest version of the monitoring operator.
<p>I ran into issues trying to deploy the NKMO, and I have PSP/PSA in use.</p>	<ol style="list-style-type: none"> 1. Edit the agent using the following command: <pre>kubect! -n <name-space> edit agent</pre> <ol style="list-style-type: none"> 2. Mark 'security-policy-enabled' as 'false'. This will disable Pod Security Policies and Pod Security Admission and allow the NKMO to deploy. Confirm by using the following commands: <pre>kubect! get psp (should show Pod Security Policy removed) kubect! get all -n <namespace> grep -i psp (should show that nothing is found)</pre>
<p>"ImagePullBackoff" errors seen</p>	<p>These errors may be seen if you have a custom or private docker repository and have not yet configured the NetApp Kubernetes Monitoring Operator to properly recognize it. Read more about configuring for custom/private repo.</p>

Problem:	Try this:
<p>I am having an issue with my monitoring-operator deployment, and the current documentation does not help me resolve it.</p>	<p>Capture or otherwise note the output from the following commands, and contact the Technical Support team.</p> <pre> kubect1 -n netapp-monitoring get all kubect1 -n netapp-monitoring describe all kubect1 -n netapp-monitoring logs <monitoring-operator-pod> --all -containers=true kubect1 -n netapp-monitoring logs <telegraf-pod> --all -containers=true </pre>
<p>net-observer (Workload Map) pods in NKMO namespace are in CrashLoopBackOff</p>	<p>These pods correspond to Workload Map data collector for Network Observability. Try these:</p> <ul style="list-style-type: none"> • Check the logs of one of the pods to confirm minimum kernel version. For example: <pre> ---- {"ci-tenant-id":"your-tenant-id","collector-cluster":"your- k8s-cluster- name","environment":"prod","level":"error","msg":"faile d in validation. Reason: kernel version 3.10.0 is less than minimum kernel version of 4.18.0","time":"2022- 11-09T08:23:08Z"} ---- </pre> <ul style="list-style-type: none"> • Net-observer pods requires the Linux kernel version to be at least 4.18.0. Check the kernel version using the command “uname -r” and ensure they are >= 4.18.0
<p>Pods are running in NKMO namespace (default: netapp-monitoring), but no data is shown in UI for workload map or Kubernetes metrics in Queries</p>	<p>Check the time setting on the nodes of the K8S cluster. For accurate audit and data reporting, it is strongly recommended to synchronize the time on the Agent machine using Network Time Protocol (NTP) or Simple Network Time Protocol (SNTP).</p>
<p>Some of the net-observer pods in NKMO namespace are in Pending state</p>	<p>Net-observer is a DaemonSet and runs a pod in each Node of the k8s cluster.</p> <ul style="list-style-type: none"> • Note the pod which is in Pending state, and check if it is experiencing a resource issue for CPU or memory. Ensure the required memory and CPU is available in the node.

Problem:	Try this:
<p>I'm seeing the following in my logs immediately after installing the NetApp Kubernetes Monitoring Operator:</p> <pre>[inputs.prometheus] Error in plugin: error making HTTP request to http://kube-state- metrics.<namespace>.svc.cluster.local:8080/metrics: Get http://kube-state- metrics.<namespace>.svc.cluster.local:8080/metrics: dial tcp: lookup kube-state- metrics.<namespace>.svc.cluster.local: no such host</pre>	<p>This message is typically only seen when a new operator is installed and the <i>telegraf-rs</i> pod is up before the <i>krm</i> pod is up. These messages should stop once all pods are running.</p>
<p>I do see not any metrics being collected for the Kubernetes CronJobs that exist in my cluster.</p>	<p>Verify your Kubernetes version (i.e. <code>kubectl version</code>). If it is v1.20.x or below, this is an expected limitation. The kube-state-metrics release deployed with the Netapp Kubernetes Monitoring Operator only supports v1.CronJob. With Kubernetes 1.20.x and below, the CronJob resource is at v1beta.CronJob. As a result, kube-state-metrics cannot find the CronJob resource.</p>
<p>After installing the operator, the telegraf-ds pods enter CrashLoopBackOff and the pod logs indicate "su: Authentication failure".</p>	<p>Edit the telegraf section in <i>AgentConfiguration</i>, and set <i>dockerMetricCollectionEnabled</i> to false. For more details, refer to the operator's configuration options.</p> <p>NOTE: If you are using Cloud Insights Federal Edition, users with restrictions on the use of <i>su</i> will not be able to collect docker metrics because access to the docker socket requires either running the telegraf container as root or using <i>su</i> to add the telegraf user to the docker group. Docker metric collection and the use of <i>su</i> is enabled by default; to disable both, remove the <i>telegraf.docker</i> entry in the <i>AgentConfiguration</i> file:</p> <pre>... spec: ... telegraf: ... - name: docker run-mode: - DaemonSet substitutions: - key: DOCKER_UNIX_SOCKET_PLACEHOLDER value: unix:///run/docker.sock</pre>

Problem:	Try this:
<p>I see repeating error messages resembling the following in my Telegraf logs:</p> <pre>E! [agent] Error writing to outputs.http: Post "https://<tenant_url>/rest/v1/lake/ingest/influxdb": context deadline exceeded (Client.Timeout exceeded while awaiting headers)</pre>	<p>Edit the telegraf section in <i>AgentConfiguration</i>, and set <i>dockerMetricCollectionEnabled</i> to false. For more details, refer to the operator's configuration options.</p>
<p>I'm missing <i>involvedobject</i> data for some Event Logs.</p>	<p>Be sure you have followed the steps in the Permissions section above.</p>
<p>Why am I seeing two monitoring operator pods running, one named netapp-ci-monitoring-operator- <pod> and the other named monitoring-operator- <pod>?</p>	<p>As of October 12, 2023, Cloud Insights has refactored the operator to better serve our users; for those changes to be fully adopted, you must remove the old operator and install the new one.</p>
<p>My kubernetes events unexpectedly stopped reporting to Cloud Insights.</p>	<div> <p>Retrieve the name of the event-exporter pod:</p> <pre>`kubectl -n netapp-monitoring get pods grep event-exporter awk '{print \$1}' sed 's/event-exporter./event-exporter/'`</pre> </div> <p>It should be either "netapp-ci-event-exporter" or "event-exporter". Next, edit the monitoring agent <code>kubectl -n netapp-monitoring edit agent</code>, and set the value for <code>LOG_FILE</code> to reflect the appropriate event-exporter pod name found in the previous step. More specifically, <code>LOG_FILE</code> should be set to either <code>/var/log/containers/netapp-ci-event-exporter.log</code> or <code>/var/log/containers/event-exporter*.log</code></p> <div> <pre>fluent-bit: ... - name: event-exporter-ci substitutions: - key: LOG_FILE values: - /var/log/containers/netapp-ci-event-exporter*.log ...</pre> </div> <p>Alternatively, one can also uninstall and reinstall the agent.</p>

Problem:	Try this:
I'm seeing pod(s) deployed by the Netapp Kubernetes Monitoring Operator crash because of insufficient resources.	Refer to the Netapp Kubernetes Monitoring Operator configuration options to increase the CPU and/or memory limits as needed.

Additional information may be found from the [Support](#) page or in the [Data Collector Support Matrix](#).

NetApp Kubernetes Monitoring Operator Configuration Options

The [NetApp Kubernetes Monitoring Operator](#) installation and configuration can be customized.

The table below lists the possible options for the AgentConfiguration file:

Component	Option	Description
agent		Configuration options that are common to all components that the operator can install. These can be considered as "global" options.
	dockerRepo	A dockerRepo override to pull images from customers private docker repos as compared to Cloud Insights docker repo. Default is cloud insights docker repo
	dockerImagePullSecret	Optional: A secret for the customers private repo
	clusterName	Free text field that uniquely identifies a cluster across all customers clusters. This should be unique across a cloud insights tenant. Default is what the customer enters in the UI for the "Cluster Name" field
	proxy Format: proxy: server: port: username: password: noProxy: isTelegrafProxyEnabled: isAuProxyEnabled: isFluentbitProxyEnabled: isCollectorProxyEnabled:	Optional to set proxy. This is usually the customer's corporate proxy.
telegraf		Configuration options that can customize the telegraf installation of the Operator
	collectionInterval	Metrics collection interval, in seconds (Max=60s)
	dsCpuLimit	CPU Limit for telegraf ds

Component	Option	Description
	dsMemLimit	Memory limit for telegraf ds
	dsCpuRequest	CPU request for telegraf ds
	dsMemRequest	Memory request for telegraf ds
	rsCpuLimit	CPU Limit for telegraf rs
	rsMemLimit	Memory limit for telegraf rs
	rsCpuRequest	CPU request for telegraf rs
	rsMemRequest	Memory request for telegraf rs
	dockerMountPoint	an override for dockerMountPoint path. This is for non standard docker installations on k8s platforms like cloud foundry
	dockerUnixSocket	an override for dockerUnixSocket path. This is for non standard docker installations on k8s platforms like cloud foundry.
	crioSockPath	an override for crioSockPath path. This is for non standard docker installations on k8s platforms like cloud foundry.
	runPrivileged	Run the telegraf container in privileged mode. Set this to true if SELinux is enabled on your k8s nodes
	batchSize	See Telegraf configuration documentation
	bufferLimit	See Telegraf configuration documentation
	roundInterval	See Telegraf configuration documentation
	collectionJitter	See Telegraf configuration documentation
	precision	See Telegraf configuration documentation
	flushInterval	See Telegraf configuration documentation
	flushJitter	See Telegraf configuration documentation
	outputTimeout	See Telegraf configuration documentation
	dockerMetricCollectionEnabled	Collect Docker metrics. By default, this is set to true and docker metrics will be collected for on-premise, docker-based k8s deployments. To disable docker metric collection, set this to false.
	dsTolerations	telegraf-ds additional tolerations.
	rsTolerations	telegraf-rs additional tolerations.
kube-state-metrics		Configuration options that can customize the kube state metrics installation of the Operator
	cpuLimit	CPU limit for kube-state-metrics deployment
	memLimit	Mem limit for kube-state-metrics deployment
	cpuRequest	CPU request for kube state metrics deployment

Component	Option	Description
	memRequest	Mem request for kube state metrics deployment
	resources	a comma separated list of resources to capture. example: cronjobs,daemonsets,deployments,ingresses,jobs,namespaces,nodes,persistentvolumeclaims,persistentvolumes,pods,replicasets,resourcequotas,services,statefulsets
	tolerations	kube-state-metrics additional tolerations.
	labels	a comma separated list of resources that kube-state-metrics should capture example: cronjobs=[*],daemonsets=[*],deployments=[*],ingresses=[*],jobs=[*],namespaces=[*],nodes=[*],persistentvolumeclaims=[*],persistentvolumes=[*],pods=[*],replicasets=[*],resourcequotas=[*],services=[*],statefulsets=[*]
logs		Configuration options that can customize logs collection and installation of the Operator
	readFromHead	true/false, should fluent bit read the log from head
	timeout	timeout, in secs
	dnsMode	TCP/UDP, mode for DNS
	fluent-bit-tolerations	fluent-bit-ds additional tolerations.
	event-exporter-tolerations	event-exporter additional tolerations.
workload-map		Configuration options that can customize the workload map collection and installation of the Operator
	cpuLimit	CPU limit for net observer ds
	memLimit	mem limit for net observer ds
	cpuRequest	CPU request for net observer ds
	memRequest	mem request for net observer ds
	metricAggregationInterval	metric aggregation interval, in seconds
	bpfPollInterval	BPF poll interval, in seconds
	enableDNSLookup	true/false, enable DNS lookup
	l4-tolerations	net-observer-l4-ds additional tolerations.
	runPrivileged	true/false - Set runPrivileged to true if SELinux is enabled on your Kubernetes nodes.
change-management		Configuration options for Kubernetes Change Management and Analysis

Component	Option	Description
	cpuLimit	CPU limit for change-observer-watch-rs
	memLimit	Mem limit for change-observer-watch-rs
	cpuRequest	CPU request for change-observer-watch-rs
	memRequest	mem request for change-observer-watch-rs
	failureDeclarationInterval Mins	Interval in minutes after which a non-successful deployment of a workload will be marked as failed
	deployAggrIntervalSeconds	Frequency at which workload deployment in-progress events are sent
	nonWorkloadAggrIntervalSeconds	Frequency at which non-workload deployments are combined and sent
	termsToRedact	A set of regular expressions used in env names and data maps whose value will be redacted Example terms:"pwd", "password", "token", "apikey", "api-key", "jwt"
	additionalKindsToWatch	A comma separated list of additional kinds to watch from the default set of kinds watched by the collector
	kindsToIgnoreFromWatch	A comma separated list of kinds to ignore from watching from the default set of kinds watched by the collector
	logRecordAggrIntervalSeconds	Frequency with which log records are sent to CI from the collector
	watch-tolerations	change-observer-watch-ds additional tolerations. Abbreviated single line format only. Example: '{key: taint1, operator: Exists, effect: NoSchedule},{key: taint2, operator: Exists, effect: NoExecute}'

Sample AgentConfiguration file

Below is a sample AgentConfiguration file.

```
apiVersion: monitoring.netapp.com/v1alpha1
kind: AgentConfiguration
metadata:
  name: netapp-monitoring-configuration
  namespace: "NAMESPACE_PLACEHOLDER"
  labels:
    installed-by: nkmo-NAMESPACE_PLACEHOLDER

spec:
  # # You can modify the following fields to configure the operator.
  # # Optional settings are commented out and include default values for
  reference
```

```

# # To update them, uncomment the line, change the value, and apply
the updated AgentConfiguration.
agent:
  # # [Required Field] A uniquely identifiable user-friendly
clustername.
  # # clusterName must be unique across all clusters in your Cloud
Insights environment.
  clusterName: "CLUSTERNAME_PLACEHOLDER"

  # # Proxy settings. The proxy that the operator should use to send
metrics to Cloud Insights.
  # # Please see documentation here: https://docs.netapp.com/us-en/cloudinsights/task\_config\_telegraf\_agent\_k8s.html#configuring-proxy-support
  # proxy:
  #   server:
  #   port:
  #   noproxy:
  #   username:
  #   password:
  #   isTelegrafProxyEnabled:
  #   isFluentbitProxyEnabled:
  #   isCollectorsProxyEnabled:

  # # [Required Field] By default, the operator uses the CI repository.
  # # To use a private repository, change this field to your repository
name.
  # # Please see documentation here: https://docs.netapp.com/us-en/cloudinsights/task\_config\_telegraf\_agent\_k8s.html#using-a-custom-or-private-docker-repository
  dockerRepo: 'DOCKER_REPO_PLACEHOLDER'
  # # [Required Field] The name of the imagePullSecret for dockerRepo.
  # # If you are using a private repository, change this field from
'docker' to the name of your secret.
  {{ if not (contains .Values.config.cloudType "aws") }}# {{ end -}}
  dockerImagePullSecret: 'docker'

  # # Allow the operator to automatically rotate its ApiKey before
expiration.
  # tokenRotationEnabled: '{{
.Values.telegraf_installer.kubernetes.rs.shim_token_rotation  }}'
  # # Number of days before expiration that the ApiKey should be
rotated. This must be less than the total ApiKey duration.
  # tokenRotationThresholdDays: '{{
.Values.telegraf_installer.kubernetes.rs.shim_token_rotation_threshold_days  }}'

```

```

telegraf:
  # # Settings to fine-tune metrics data collection. Telegraf config
names are included in parenthesis.
  # # See
https://github.com/influxdata/telegraf/blob/master/docs/CONFIGURATION.md#agent

  # # The default time telegraf will wait between inputs for all plugins
(interval). Max=60
  # collectionInterval: '{{
.Values.telegraf_installer.agent_resources.collection_interval }}'
  # # Maximum number of records per output that telegraf will write in
one batch (metric_batch_size).
  # batchSize: '{{
.Values.telegraf_installer.agent_resources.metric_batch_size }}'
  # # Maximum number of records per output that telegraf will cache
pending a successful write (metric_buffer_limit).
  # bufferLimit: '{{
.Values.telegraf_installer.agent_resources.metric_buffer_limit }}'
  # # Collect metrics on multiples of interval (round_interval).
  # roundInterval: '{{
.Values.telegraf_installer.agent_resources.round_interval }}'
  # # Each plugin waits a random amount of time between the scheduled
collection time and that time + collection_jitter before collecting inputs
(collection_jitter).
  # collectionJitter: '{{
.Values.telegraf_installer.agent_resources.collection_jitter }}'
  # # Collected metrics are rounded to the precision specified. When set
to "0s" precision will be set by the units specified by interval
(precision).
  # precision: '{{ .Values.telegraf_installer.agent_resources.precision
}}'
  # # Time telegraf will wait between writing outputs (flush_interval).
Max=collectionInterval
  # flushInterval: '{{
.Values.telegraf_installer.agent_resources.flush_interval }}'
  # # Each output waits a random amount of time between the scheduled
write time and that time + flush_jitter before writing outputs
(flush_jitter).
  # flushJitter: '{{
.Values.telegraf_installer.agent_resources.flush_jitter }}'
  # # Timeout for writing to outputs (timeout).
  # outputTimeout: '{{
.Values.telegraf_installer.http_output_plugin.timeout }}'

```



```

# # telegraf-ds CPU/Mem limits and requests.
# # See https://kubernetes.io/docs/concepts/configuration/manage-
resources-containers/
dsCpuLimit: '{{
.Values.telegraf_installer.telegraf_resources.ds_cpu_limits  }}'
dsMemLimit: '{{
.Values.telegraf_installer.telegraf_resources.ds_mem_limits  }}'
dsCpuRequest: '{{
.Values.telegraf_installer.telegraf_resources.ds_cpu_request  }}'
dsMemRequest: '{{
.Values.telegraf_installer.telegraf_resources.ds_mem_request  }}'

# # telegraf-rs CPU/Mem limits and requests.
rsCpuLimit: '{{
.Values.telegraf_installer.telegraf_resources.rs_cpu_limits  }}'
rsMemLimit: '{{
.Values.telegraf_installer.telegraf_resources.rs_mem_limits  }}'
rsCpuRequest: '{{
.Values.telegraf_installer.telegraf_resources.rs_cpu_request  }}'
rsMemRequest: '{{
.Values.telegraf_installer.telegraf_resources.rs_mem_request  }}'

# # telegraf additional tolerations. Use the following abbreviated
single line format only.
# # Inspect telegraf-rs/-ds to view tolerations which are always
present.
# # Example: '{key: taint1, operator: Exists, effect:
NoSchedule},{key: taint2, operator: Exists, effect: NoExecute}'
# dsTolerations: ''
# rsTolerations: ''

# # Set runPrivileged to true if SELinux is enabled on your Kubernetes
nodes.
# runPrivileged: 'false'

# # Collect NFS IO metrics.
# dsNfsIOEnabled: '{{
.Values.telegraf_installer.kubernetes.ds.shim_nfs_io_processing  }}'

# # Collect kubernetes.system_container metrics and objects in the
kube-system|cattle-system namespaces for managed kubernetes clusters (EKS,
AKS, GKE, managed Rancher). Set this to true if you want collect these
metrics.
# managedK8sSystemMetricCollectionEnabled: '{{
.Values.telegraf_installer.kubernetes.shim_managed_k8s_system_metric_colle
ction  }}'

```

```

# # Collect kubernetes.pod_volume (pod ephemeral storage) metrics.
Set this to true if you want to collect these metrics.
# podVolumeMetricCollectionEnabled: '{{
.Values.telegraf_installer.kubernetes.shim_pod_volume_metric_collection
}}'

# # Declare Rancher cluster as managed. Set this to true if your
Rancher cluster is managed as opposed to on-premise.
# isManagedRancher: '{{
.Values.telegraf_installer.kubernetes.is_managed_rancher }}'

# kube-state-metrics:
# # kube-state-metrics CPU/Mem limits and requests. By default, when
unset, kube-state-metrics has no CPU/Mem limits nor request.
# cpuLimit:
# memLimit:
# cpuRequest:
# memRequest:

# # Comma-separated list of metrics to enable.
# # See metric-allowlist in https://github.com/kubernetes/kube-state-metrics/blob/main/docs/cli-arguments.md
# resources:
'cronjobs,daemonsets,deployments,ingresses,jobs,namespaces,nodes,persistentvolumes,persistentvolumeclaims,pods,replicasets,resourcequotas,services,statefulsets'

# # Comma-separated list of Kubernetes label keys that will be used in
the resources' labels metric.
# # See metric-labels-allowlist in https://github.com/kubernetes/kube-state-metrics/blob/main/docs/cli-arguments.md
# labels:
'cronjobs=[*],daemonsets=[*],deployments=[*],ingresses=[*],jobs=[*],namespaces=[*],nodes=[*],persistentvolumeclaims=[*],persistentvolumes=[*],pods=[*],replicasets=[*],resourcequotas=[*],services=[*],statefulsets=[*]'

# # kube-state-metrics additional tolerations. Use the following
abbreviated single line format only.
# # No tolerations are applied by default
# # Example: '{key: taint1, operator: Exists, effect: NoSchedule},{key: taint2, operator: Exists, effect: NoExecute}'
# tolerations: ''

# # Settings for the Events Log feature.
# logs:
# # If Fluent Bit should read new files from the head, not tail.

```

```

# # See Read_from_Head in
https://docs.fluentbit.io/manual/pipeline/inputs/tail
# readFromHead: "true"

# # Network protocol that Fluent Bit should use for DNS: "UDP" or
"TCP".
# dnsMode: "UDP"

# # Logs additional tolerations. Use the following abbreviated single
line format only.
# # Inspect fluent-bit-ds to view tolerations which are always
present. No tolerations are applied by default for event-exporter.
# # Example: '{key: taint1, operator: Exists, effect:
NoSchedule},{key: taint2, operator: Exists, effect: NoExecute}'
# fluent-bit-tolerations: ''
# event-exporter-tolerations: ''

# # Settings for the Network Performance and Map feature.
# workload-map:
# # net-observer-l4-ds CPU/Mem limits and requests.
# # See https://kubernetes.io/docs/concepts/configuration/manage-
resources-containers/
# cpuLimit: '500m'
# memLimit: '500Mi'
# cpuRequest: '100m'
# memRequest: '500Mi'

# # Metric aggregation interval in seconds. Min=30, Max=120
# metricAggregationInterval: '60'

# # Interval for bpf polling. Min=3, Max=15
# bpfPollInterval: '8'

# # Enable performing reverse DNS lookups on observed IPs.
# enabledDNSLookup: 'true'

# # net-observer-l4-ds additional tolerations. Use the following
abbreviated single line format only.
# # Inspect net-observer-l4-ds to view tolerations which are always
present.
# # Example: '{key: taint1, operator: Exists, effect:
NoSchedule},{key: taint2, operator: Exists, effect: NoExecute}'
# l4-tolerations: ''

# # Set runPrivileged to true if SELinux is enabled on your Kubernetes
nodes.

```

```

# # Note: In OpenShift environments, this is set to true
automatically.
# runPrivileged: 'false'

# change-management:
# # change-observer-watch-rs CPU/Mem limits and requests.
# # See https://kubernetes.io/docs/concepts/configuration/manage-resources-containers/
# cpuLimit: '500m'
# memLimit: '500Mi'
# cpuRequest: '100m'
# memRequest: '500Mi'

# # Interval in minutes after which a non-successful deployment of a
workload will be marked as failed
# failureDeclarationIntervalMins: '30'

# # Frequency at which workload deployment in-progress events are sent
# deployAggrIntervalSeconds: '300'

# # Frequency at which non-workload deployments are combined and sent
# nonWorkloadAggrIntervalSeconds: '15'

# # A set of regular expressions used in env names and data maps whose
value will be redacted
# termsToRedact: '"pwd", "password", "token", "apikey", "api-key",
"jwt"'

# # A comma separated list of additional kinds to watch from the
default set of kinds watched by the collector
# # Each kind will have to be prefixed by its apigroup
# # Example: 'authorization.k8s.io.subjectaccessreviews'
# additionalKindsToWatch: ''

# # A comma separated list of kinds to ignore from watching from the
default set of kinds watched by the collector
# # Each kind will have to be prefixed by its apigroup
# # Example: 'networking.k8s.io.networkpolicies,batch.jobs'
# kindsToIgnoreFromWatch: ''

# # Frequency with which log records are sent to CI from the collector
# logRecordAggrIntervalSeconds: '20'

# # change-observer-watch-ds additional tolerations. Use the following
abbreviated single line format only.
# # Inspect change-observer-watch-ds to view tolerations which are
always present.

```

```
# # Example: '{key: taint1, operator: Exists, effect:
NoSchedule},{key: taint2, operator: Exists, effect: NoExecute}'
# watch-tolerations: ''----
```

```
[[IDc37a1a5fb354394fff3b96e51c90ced4]]
= Kubernetes Cluster Detail Page
:toc: macro
:hardbreaks:
:toclevels: 1
:icons: font
:linkattrs:
:relative_path: ./
:imagesdir: {root_path}{relative_path}./media/
```

```
[.lead]
The Kubernetes cluster detail page displays a detailed overview of your
Kubernetes cluster.
```

```
//The detail page is comprised of three distinct but linked landing pages
showing cluster, node, and pod information. The "Resource Usage" section
changes to show the details of the selected item (cluster, node, or pod).
You can see the current page type and name at the top of the screen. The
current page is shown in the following heirarchy: _Site Name / Kubernetes
/ Cluster / Node / Pod_. You can click any part of this "breadcrumb"
trail to go directly to that specific page.
```

```
//image:Kubernetes_Breadcrumb.png[Page Breadcrumb]
```

```
//== Cluster Overview
```

```
//The cluster overview page provides useful information at a glance:
```

```
image:Kubernetes_Detail_Page_new.png[Cluster detail page]
```

```
== Namespace, Node, and Pod Counts
```

```
The counts at the top of the page show you the total number of namespaces,
nodes, and pods in the cluster, as well as the number of popds that are
currently alerting and pending.
```

```
//NOTE: It is possible that the three pod sub-counts (healthy, alerting,
```

pending) can add up to more than the displayed total number of pods. This can happen because the `_pending_` count includes `_all_` pending pods, both unscheduled and scheduled (in other words, unattached and attached to nodes).

== Shared Resources and Saturation

On the top right of the detail page is your cluster saturation as a current percentage as well as a graph showing the recent trend over time. Cluster saturation is the highest of CPU, memory, or storage saturation at each point in time.

Below that, the page shows by default **Shared Resources** usage, with tabs for CPU, Memory, and Storage. Each tab shows the saturation percentage and trend over time, with additional usage details. For storage, the value shown is the greater of backend and filesystem saturation, which are calculated independently.

The devices with the highest usage are shown in a table at the bottom. Click any link to explore these devices.

== Namespaces

The Namespaces tab displays a list of all the namespaces in your Kubernetes environment, showing CPU and Memory usage as well as a count of workloads in each namespace. Click the Name links to explore each namespace.

image:Kubernetes_Namespace_tab_new.png[list of current namespaces in your K8s environment]

== Workloads

Similarly, the Workloads tab displays a list of the workloads in each namespace, again showing CPU and Memory usage. Clicking the Namespace links drills into each.

image:Kubernetes_Workloads_tab_new.png[list of current namespaces in your K8s environment]

== The Cluster "Wheel"

image:Kubernetes_Wheel_Section.png[Cluster Wheel]

The Cluster "Wheel" section provides node and pod health at a glance, which you can drill into for more information. If your cluster contains more nodes than can be displayed in this area of the page, you will be able to turn the wheel using the buttons available.

Alerting pods or nodes are displayed in red. "Warning" areas are displayed in orange. Pods that are unscheduled (that is, unattached) will display in the lower corner of the Cluster "Wheel".

Hovering over a pod (circle) or Node (bar) will extend the view of the node.

image:Kubernetes_Node_Expand.png[Expanded Node]

Clicking on the pod or node in that view will zoom in to the expanded Node view.

image:Kubernetes_Critical_Pod_Zoom.png[Expanded Node View]

From here, you can hover over an element to display details about that element. For example, hovering over the critical pod in this example displays details about that pod.

image:Kubernetes_Pod_Red.png[Critical Pod Information]

You can view Filesystem, Memory, and CPU information by hovering over the Node elements.

image:Kubernetes_Capacity_Info.png[Node Capacity and Allocation]

////

== Detail Section

Each page of the Kubernetes Explorer displays a set of usage graphs that may include CPU, Memory, and Storage. Below these graphs are summaries and lists of the top objects in each category, with links to underlying details. For example, `_Node_` shows pods and containers, `_Pod_` shows PVCs and related objects and containers, etc. The following illustration shows an example of the detailed information on a `_Node_` page:

image:Kubernetes_Node_Resource_Usage.png[Resource Usage Example]

Resources experiencing alerts will show at the top of the lists. Click on the affected resource to drill into it for more detail.

////

== A note about the gauges

The Memory and CPU gauges show three colors, since they show `_used_` in relation to both `_allocatable capacity_` and `_total capacity_`.

////

Keep the following in mind when reading the gauges.

The dark blue band shows the amount used.

* When viewed against the `_light blue band_`, the dark blue shows used as the % of allocatable amount. This matches the "% of allocatable" value shown (81 in the example below).

* When viewed against the `_white background_`, the dark blue shows used as the % of total capacity. This matches the "% of capacity" value shown (63 in this example).

image:Kubernetes_Gauge_Explained.png[Gauge Numbers Explained]

//The length of the light blue band against the white background shows the total allocatable amount vs the total capacity; that figure itself is not shown, but it's derived using the formula shown in the red text: $(\text{capacity} / \text{allocatable}) * 100$.

//Much of the time, our own environments show the same number for the 2 percent values in the gauge, and so you don't often see the white band because the light blue covers it completely (meaning 100% of the total capacity is allocatable).

////

[[ID2328f35c636fd679bd700a54fa36daed]]

= Kubernetes Network Performance Monitoring and Map


```
:toc: macro
:hardbreaks:
:toclevels: 1
:icons: font
:linkattrs:
:relative_path: ./
:imagesdir: {root_path}{relative_path}./media/
```

[.lead]

The Kubernetes Network Performance Monitoring and Map feature simplifies troubleshooting by mapping dependencies between Kubernetes services (also called workloads). It provides real-time visibility into Kubernetes network performance latencies and anomalies to identify performance issues before they affect users.

This capability helps organizations reduce overall costs by analyzing and auditing Kubernetes traffic flows.

NOTE: The Kubernetes Network Performance Monitoring and Map is a `xref:{relative_path}concept_preview_features.html[Preview]` feature and is subject to change.

Key Features:

- The Workload Map presents Kubernetes workload dependencies and flows and highlights network and performance issues.
- Monitor network traffic between Kubernetes pods, workloads, and nodes; identifies the source of traffic and latency problems.
- Reduce overall costs by analyzing ingress, egress, cross-region, and cross-zone network traffic.

`//image:Workload Map Example_withSlideout.png[Workload Map example showing "Slideout" panel with details]`

`image:workload-map-animated.gif[Workload Map Example]`

== Pre-Requisites

Before you can use the Kubernetes Network Performance Monitoring and Map, you must have configured the

`xref:{relative_path}task_config_telegraf_agent_k8s.html[NetApp Kubernetes Monitoring Operator]` to enable this option. During deployment of the Operator, select the "Network Performance and Map" checkbox to enable. You can also enable this option by navigating to a Kubernetes landing page and selecting "Modify Deployment".

`image:ServiceMap_NKMO_Deployment_Options.png[selecting the Map option]`

during NKMO stup]

== Monitors

The Workload Map uses

`xref:{relative_path}task_create_monitor.html[monitors]` to derive information. Cloud Insights provides a number of default Kubernetes Monitors (note that these may be `_Paused_` by default. You can `_Resume_` (i.e. enable) the monitors you want), or you can create custom monitors for kubernetes objects, which the Workload Map will also use.

You can create Cloud insights metric alerts on any of the object types below. Make sure the data is grouped by the default object type.

- * `kubernetes.workload`
- * `kubernetes.daemonset`
- * `kubernetes.deployment`
- * `kubernetes.cronjob`
- * `kubernetes.job`
- * `kubernetes.replicaset`
- * `kubernetes.statefulset`
- * `kubernetes.pod`
- * `kubernetes.network_traffic_l4`

== The Map

The Map shows services/workloads and their relationships to each other. Arrows show directions of traffic. Hovering over a workload displays summary information for that workload, as you can see in this example:

image:ServiceMap_Simple_Example.png[Example of a Workload Map workload]

Icons within the circles represent different service types. Note that icons are only visible if the underlying objects have `<<workload-labels, labels>>`.

image:ServiceMap_Icons.png[Service Icons Explained]

The size of each circle indicates node size. Note that these sizes are relative, your browser zoom level or screen size may affect actual circle sizes. In the same way, the traffic line style gives you an at-a-glance view of the connection size; bold solid lines are high traffic, while light dotted lines are lower traffic.

Numbers inside the circles are the number of external connections currently being processed by the service.

image:ServiceMap_Node_and_Connection_Legend.png[legend showing relative circle (node) and connection sizes]

////

== Details

Hovering over a circle displays a summary of information for that service.

image:Workload_Map_Summary.png[Workload Hover Summary]

////

== Workload Details and Alerts

Circles displayed in color indicate a warning- or critical-level alert for the workload. Hover over the circle for a summary of the issue, or click on the circle to open a slideout panel with more detail.

image:Workload_Map_Slideout_with_Alert.png[Workload Slideout Details With Alerts]

== Finding and Filtering

As with other Cloud Insights features, you can easily set filters to focus on the specific objects or workload attributes you want.

image:Workload_Map_Filtering.png[Workload Map filtering]

Likewise, typing a string in the `_Find_` field will highlight matching workloads.

image:Workload_Map_Find_Highlighting.png[typing in find box highlights workloads]

== Workload Labels

Workload labels are necessary if you want the Map to identify the types of workloads displayed (i.e. the circle icons). Labels are derived as follows:

* Name of the service/application running in generic terms

```
* If the source is a pod:
** Label is derived from the workload label of the pod
** Expected label on the workload: app.kubernetes.io/component
** Label name reference:
https://kubernetes.io/docs/concepts/overview/working-with-objects/common-labels/
** Recommended labels:
*** frontend
*** backend
*** database
*** cache
*** queue
*** kafka

* If the source is external to the kubernetes cluster:
** Cloud Insights will attempt to parse the DNS resolved name to extract
the service type.
+
For example, with a DNS resolved name of _s3.eu-north-1.amazonaws.com_,
the resolved name is parsed to get _s3_ as the service type.
```

== Dive Deep

Right-clicking on a workload presents you with additional options to explore further. For example, from here you can zoom in to view the connections for that workload.

image:Workload_Map_Zoom_Into_Connections.png[Workload Map Right-Click Zoom to show the workload's connections]

Or you can open the detail slideout panel to directly view the _Summary_, _Network_, or _Pod & Storage_ tab.

image:Workload_Map_Detail_Network_Slideout.png[Detail Slideout Network Tab Example]

Finally, selecting _Go to Asset Page_ will open the detailed asset landing page for the workload.

image:Workload_Map_Asset_Page.png[Workload Asset Page]

```
[[ID3f3bbefbb6d340b4d5d17c6417f8be4a]]
= Kubernetes Change Analytics
:toc: macro
:hardbreaks:
:toclevels: 1
:icons: font
:linkattrs:
:relative_path: ./
:imagesdir: {root_path}{relative_path}./media/
```

[.lead]

Kubernetes Change Analytics provide you with an all-in-one view of recent changes to your Kubernetes environment. Alerts and deployment status are at your fingertips. With Change Analytics, you can track every deployment and configuration change, and correlate it with the health and performance of K8s services, infrastructure, and clusters.

Keep the following in mind:

- * In multi-tenant Kubernetes environments, outages may happen because of mis-configured changes. In very dynamic environments, Cloud Insights may not be able to properly track all changes.
- * Change Analytics provides a single pane to view and correlate the health of workloads and configuration changes. This may help in troubleshooting dynamic Kubernetes environments.

To view Kubernetes Change Analytics, navigate to *Kubernetes > Change Analysis*.

image:ChangeAnalytitcs_Main_Screen.png[Kubernetes Change Analytics main screen, showing warning and critical alerts, successful and failed deployments, and the top 3 workloads triggering alerts].

The page automatically refreshes based on the currently-selected Cloud Insights time range. Smaller time ranges mean more frequent screen refreshing.

== Filtering

As with all features of Cloud Insights, filtering the change list is intuitive: at the top of the page, enter or select values for your Kubernetes Cluster, Namespace, or Workload, or add your own filters by selecting the `{+}` button.

When you filter down to a specific Cluster, Namespace, and Workload (along with any other filters you set), you are shown a timeline of deployments and alerts for that workload in that namespace on that cluster. Zoom in further by clicking and dragging in the graph to focus on a more specific time range.

image:ChangeAnalytics_Filtered_Timeline.png[Workload Timeline example]

== Quick Status

Below the filtering area are a number of high-level indicators. On the left is the number of alerts (Warning and Critical). This number includes `_Active_` as well as `_Resolved_` alerts. To see only `_Active_` alerts, set a filter for "Status" and choose "Active".

image:ChangeAnalytics_Alerts.png[Change Analytics Alerts]

Deployment status is also shown here. Again, the default is to show the count of `_Started_`, `_Complete_`, and `_Failed_` deployments. To see only `_Failed_` deployments, set a filter for "Status" and select "Failed".

image:ChangeAnalytics_Deploys.png[Change Analytics DSeloys]

The top 3 workloads with the most alerts are next. The number in red next to each workload indicates the number of alerts related to that workload. Click the workload link to explore through your Infrastructure (Kubernetes Explorer), Dependencies (Workload Map), or Log Analysis (Event Logs).

image:ChangeAnalytics_ExploreWorkloadAlerts.png[Change Analytics Workload Exploration Options]

== Detail Panel

Selecting a change in the list opens a panel describing the change in more detail. For example, selecting a failed Deploy shows a summary of the Deploy, with start and end times, duration, and where the deploy was triggered, with links to explore those resources. It also displays the reason for the failure, any related changes, and any associated events.

image:ChangeAnalytitscs_DeployDetailPanel.png[Deploy Failure Detail Panel]

Selecting an Alert similarly provides details about the alert, including the monitor that triggered the alert as well as a chart showing a visual timeline for the alert.

:leveloffset: -1

:leveloffset: -1

<<<

Copyright information

Copyright © 2024 NetApp, Inc. All Rights Reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government

is subject to restrictions as set forth in subparagraph (b) (3) of the Rights in Technical Data -Noncommercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, nonsublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

Trademark information

NETAPP, the NETAPP logo, and the marks listed at [link:http://www.netapp.com/TM](http://www.netapp.com/TM)^[http://www.netapp.com/TM^] are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.