Kubernetes

Data Infrastructure Insights

NetApp
December 19, 2024

Inhalt

Kubernetes . . . 1
Kubernetes-Cluster – Übersicht . . . 1
Bevor Sie den NetApp Kubernetes Monitoring Operator installieren oder aktualisieren . . . 2
Installation und Konfiguration des Kubernetes Monitoring Operator . . . 6
Konfigurationsoptionen Für Kubernetes Monitoring Operator . . . 23
Detailseite Zu Kubernetes Cluster . . . 36
Performance-Monitoring und -Zuordnung des Kubernetes-Netzwerks . . . 40
Kubernetes Change Analytics . . . 48

Kubernetes

Kubernetes-Cluster – Übersicht

Der Data Infrastructure Insights Kubernetes Explorer ist ein leistungsstarkes Tool zum Anzeigen des Gesamtzustands und der Auslastung Ihrer Kubernetes-Cluster. Hier können Sie ganz einfach detaillierte Untersuchungsbereiche aufschlüsseln.

Durch Klicken auf **Dashboards > Kubernetes Explorer** wird die Listenseite für Kubernetes-Cluster geöffnet. Diese Übersichtsseite enthält eine Tabelle der Kubernetes-Cluster auf Ihrem Mandanten.

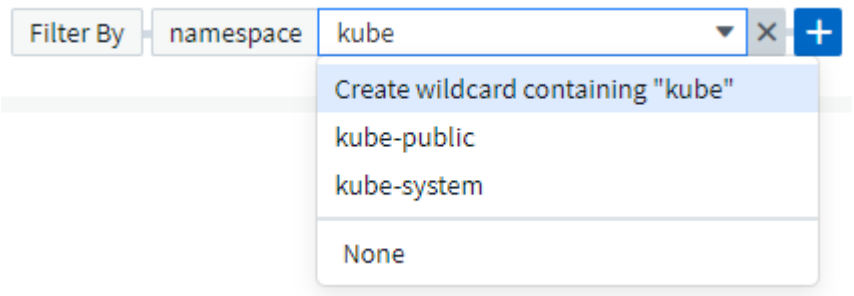
Cluster-Liste

In der Cluster-Liste werden die folgenden Informationen zu jedem Cluster des Mandanten angezeigt:

- Cluster **Name**. Wenn Sie auf einen Cluster-Namen klicken, wird das für dieses Cluster geöffnet" Detailseite".
- **Sättigung** Prozentsätze. „Gesamteinlagerung“ entspricht dem höchsten Wert für CPU, Speicher oder Speichersättigung.
- Anzahl **Nodes** im Cluster. Wenn Sie auf diese Nummer klicken, wird die Seite Knotenliste geöffnet.
- Anzahl **Pods** im Cluster. Wenn Sie auf diese Nummer klicken, wird die Pod-Listenseite geöffnet.
- Anzahl **Namesaces** im Cluster. Wenn Sie auf diese Nummer klicken, wird die Namespace-Listenseite geöffnet.
- Anzahl **Workloads** im Cluster. Wenn Sie auf diese Nummer klicken, wird die Listenseite Workload geöffnet.

Verfeinern des Filters

Wenn Sie filtern, werden Sie beim Eingeben mit der Option angezeigt, basierend auf dem aktuellen Text einen **Platzhalterfilter** zu erstellen. Wenn Sie diese Option auswählen, werden alle Ergebnisse angezeigt, die dem Platzhalterausdruck entsprechen. Sie können auch **Expressions** mit NOT oder UND erstellen, oder Sie können die Option "Keine" auswählen, um nach Null-Werten im Feld zu filtern.



Filter basierend auf Platzhalter oder Ausdrücken (z. B. NOT, AND, „None“ etc.) wird im Filterfeld dunkelblau angezeigt. Elemente, die Sie direkt aus der Liste auswählen, werden hellblau angezeigt.

Kubernetes-Filter sind kontextbezogen, d. h., wenn Sie sich beispielsweise auf einer bestimmten Knotenseite befinden, listet der Pod_Name-Filter nur die Pods auf, die mit diesem Node zusammenhängen. Wenn Sie darüber hinaus einen Filter für einen bestimmten Namespace anwenden, werden im Pod_Name-Filter nur Pods auf diesem Node *und* in diesem Namespace aufgelistet.

Beachten Sie, dass die Platzhalter- und Ausdrucksfilterung mit Text oder Listen funktioniert, jedoch nicht mit numerischen Werten, Daten oder Booleanen.

Bevor Sie den NetApp Kubernetes Monitoring Operator installieren oder aktualisieren

Lesen Sie diese Informationen, bevor Sie das installieren oder aktualisieren"Kubernetes Monitoring Operator".

Komponente	Anforderungen
Kubernetes-Version	Kubernetes v1.20 und höher
Kubernetes Distributionen	AWS Elastic Kubernetes Service (EKS) Azure Kubernetes Service (AKS) Google Kubernetes Engine (GKE) Red hat OpenShift Rancher Kubernetes Engine (RKE) VMware Tanzu
Linux BS	Data Infrastructure Insights unterstützt keine Nodes, die mit einer Arm64-Architektur ausgeführt werden. Netzwerküberwachung: Muss Linux Kernel Version 4.18.0 oder höher ausführen. Photon OS wird nicht unterstützt.
Etiketten	Data Infrastructure Insights unterstützt das Monitoring von Kubernetes-Nodes, auf denen Linux ausgeführt wird, indem eine Kubernetes-Node-Auswahl angegeben wird, die auf diesen Plattformen nach den folgenden Kubernetes-Labels sucht: Kubernetes v1.20 und höher: Kubernetes.io/os = linux Rancher + Cattle.io als Orchestrierungs-/Kubernetes-Plattform: Cattle.io/os = linux
Befehle	Die Befehle Curl und kubectl müssen verfügbar sein.; für optimale Ergebnisse fügen Sie diese Befehle dem PFAD hinzu.

Komponente	Anforderungen
Konnektivität	Kubectl cli ist für die Kommunikation mit dem Ziel-K8s-Cluster konfiguriert und verfügt über eine Internetverbindung zur Data Infrastructure Insights Umgebung. Wenn Sie während der Installation hinter einem Proxy stehen, befolgen Sie die Anweisungen im "Proxy-Unterstützung Wird Konfiguriert"Abschnitt der Installation des Bedieners. Für genaue Audit- und Datenberichte synchronisieren Sie die Zeit auf dem Agent-Computer mit Network Time Protocol (NTP) oder Simple Network Time Protocol (SNTP).
Sonstiges	Wenn Sie OpenShift 4.6 oder höher ausführen, müssen Sie zusätzlich zur Erfüllung dieser Voraussetzungen die folgenden Schritte ausführen"OpenShift-Anweisungen".
API-Token	Wenn Sie den Operator neu bereitstellen (d. h. aktualisieren oder ersetzen), müssen Sie kein neues API-Token erstellen; Sie können das vorherige Token erneut verwenden.

Wichtige Dinge, die Sie beachten sollten, bevor Sie beginnen

Wenn Sie mit einem laufen Proxy, haben ein Benutzerdefiniertes Repository, oder verwenden OpenShift, lesen Sie die folgenden Abschnitte sorgfältig.

Lesen Sie auch über Berechtigungen.

Proxy-Unterstützung Wird Konfiguriert

An zwei Stellen können Sie einen Proxy für Ihren Mandanten verwenden, um den NetApp Kubernetes Monitoring Operator zu installieren. Es kann sich um dieselben oder separate Proxy-Systeme handelt:

- Proxy wird während der Ausführung des Installationscode-Snippets (mit „Curl“) benötigt, um das System zu verbinden, auf dem das Snippet ausgeführt wird, mit Ihrer Data Infrastructure Insights-Umgebung
- Der vom Kubernetes Ziel-Cluster benötigte Proxy für die Kommunikation mit der Insights Umgebung Ihrer Dateninfrastruktur ist erforderlich

Wenn Sie einen Proxy für eine oder beide dieser Optionen verwenden, müssen Sie zuerst sicherstellen, dass Ihr Proxy für eine gute Kommunikation mit Ihrer Data Infrastructure Insights-Umgebung konfiguriert ist, um den NetApp Kubernetes Operating Monitor zu installieren. Beispielsweise müssen Sie auf den Servern/VMs, von denen Sie den Operator installieren möchten, auf Data Infrastructure Insights zugreifen und Binärdateien von Data Infrastructure Insights herunterladen können.

Legen Sie für den Proxy, der zur Installation des NetApp Kubernetes Operating Monitor verwendet wurde, vor der Installation des Operators die Umgebungsvariablen *http_Proxy/https_Proxy* fest. In einigen Proxy-Umgebungen müssen Sie möglicherweise auch die Variable *no_Proxy Environment* festlegen.

Um die Variable(en) festzulegen, führen Sie auf Ihrem System **vor** der Installation des NetApp Kubernetes Monitoring Operators folgende Schritte aus:

1. Legen Sie die Umgebungsvariable *https_Proxy* und/oder *http_Proxy* für den aktuellen Benutzer fest:

a. Wenn der Proxy, der eingerichtet wird, keine Authentifizierung (Benutzername/Passwort) aufweist, führen Sie den folgenden Befehl aus:

```
 export https_proxy=<proxy_server>:<proxy_port>
.. Wenn der Proxy, der eingerichtet wird, über Authentifizierung
(Benutzername/Passwort) verfügt, führen Sie folgenden Befehl aus:
```

```
export
http_proxy=<proxy_username>:<proxy_password>@<proxy_server>:<proxy_po
rt>
```

Wenn der Proxy, der für das Kubernetes-Cluster zur Kommunikation mit der Insights Umgebung für die Dateninfrastruktur verwendet wird, verwendet wird, installieren Sie den NetApp Kubernetes Monitoring Operator, nachdem Sie alle diese Anweisungen gelesen haben.

Konfigurieren Sie den Proxy-Abschnitt von AgentConfiguration in Operator-config.yaml, bevor Sie den NetApp Kubernetes Monitoring Operator bereitstellen.

```
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
  ...
...
```

Verwenden eines benutzerdefinierten oder privaten Docker Repositorys

Standardmäßig zieht der NetApp-Kubernetes-Überwachungsoperator Container-Images aus dem Repository „Einblicke in die Dateninfrastruktur“. Wenn Sie ein Kubernetes-Cluster als Ziel für das Monitoring verwenden und der Cluster so konfiguriert ist, dass er nur Container-Images aus einem benutzerdefinierten oder privaten Docker-Repository oder der Container-Registrierung zieht, müssen Sie den Zugriff auf die Container

konfigurieren, die vom NetApp Kubernetes Monitoring Operator benötigt werden.

Führen Sie das „Image Pull Snippet“ aus der NetApp Monitoring Operator Installationskachel aus. Dieser Befehl meldet sich beim Repository Data Infrastructure Insights an, zieht alle Image-Abhängigkeiten für den Operator ab und meldet sich vom Repository Data Infrastructure Insights ab. Wenn Sie dazu aufgefordert werden, geben Sie das angegebene temporäre Repository-Passwort ein. Mit diesem Befehl werden alle vom Bediener verwendeten Bilder heruntergeladen, einschließlich optionaler Funktionen. Nachfolgend sehen Sie, für welche Funktionen diese Bilder verwendet werden.

Core Operator-Funktionalität und Kubernetes Monitoring

- netapp Monitoring
- kube-rbac-Proxy
- status-Kennzahlen von kube
- telegraf
- Distroless-root-user

Ereignisprotokoll

- Fluent-Bit
- kubernetes Event Exporter

Netzwerkleistung und -Zuordnung

- ci-Netz-Beobachter

Übertragen Sie das Operator-Docker-Image gemäß Ihren Unternehmensrichtlinien in das private/lokale/unternehmenseigene Docker-Repository. Stellen Sie sicher, dass die Bild-Tags und Verzeichnispfade zu diesen Images in Ihrem Repository mit denen im Data Infrastructure Insights Repository übereinstimmen.

Bearbeiten Sie die Bereitstellung des Monitoring-Operators in Operator-Deployment.yaml, und ändern Sie alle Bildverweise, um Ihr privates Docker-Repository zu verwenden.

```
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>
```

Bearbeiten Sie die AgentConfiguration in Operator-config.yaml, um die neue Position des Docker-Repo zu berücksichtigen. Erstellen Sie ein neues imagePullSecret für Ihr privates Repository. Weitere Informationen finden Sie unter *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 for
xref:{relative_path}task_config_telegraf_agent_k8s.html#using-a-custom-or-
private-docker-repository[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-Anweisungen

Wenn Sie OpenShift 4.6 oder höher ausführen, müssen Sie die AgentConfiguration in *Operator-config.yaml* bearbeiten, um die Einstellung *runPrivileged* zu aktivieren:

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

OpenShift kann zusätzliche Sicherheitsstufen implementieren, die den Zugriff auf einige Kubernetes-Komponenten blockieren könnten.

Berechtigungen

Wenn das zu überwachende Cluster benutzerdefinierte Ressourcen enthält, die nicht über einen ClusterRole verfügen"AnzuzeiEinblick in Aggregate", müssen Sie dem Bediener manuell Zugriff auf diese Ressourcen gewähren, um sie mit Ereignisprotokollen zu überwachen.

1. Bearbeiten Sie *Operator-additional-permissions.yaml* vor der Installation oder nach der Installation bearbeiten Sie die Ressource *ClusterRole/<namespace>-additional-permissions*
2. Erstellen Sie eine neue Regel für die gewünschten apiGroups und Ressourcen mit den Verben ["get", "watch", "list"]. Siehe https://kubernetes.io/docs/reference/access-authn-authz/rbac/
3. Übernehmen Sie die Änderungen auf das Cluster

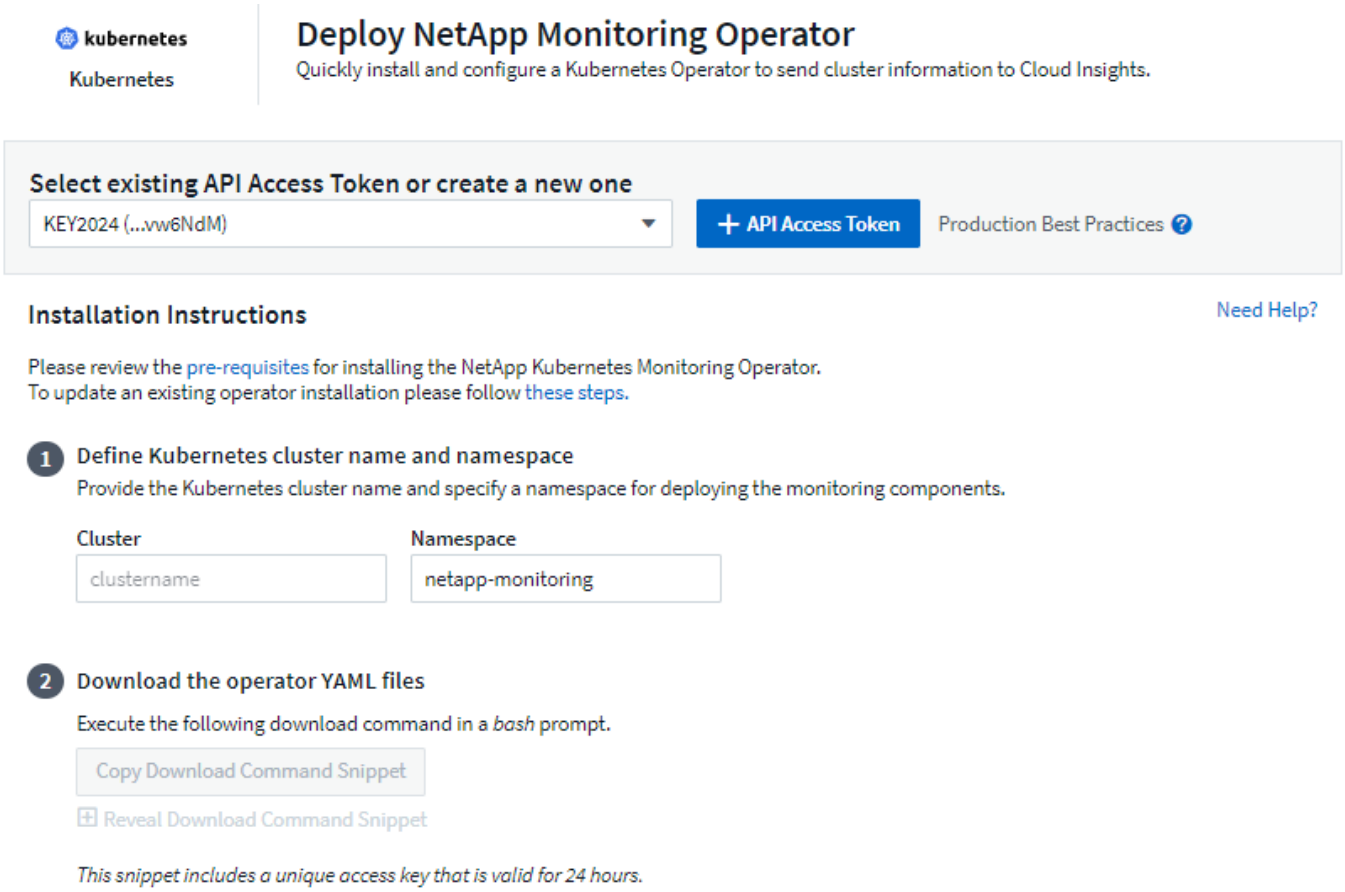
Installation und Konfiguration des Kubernetes Monitoring Operator

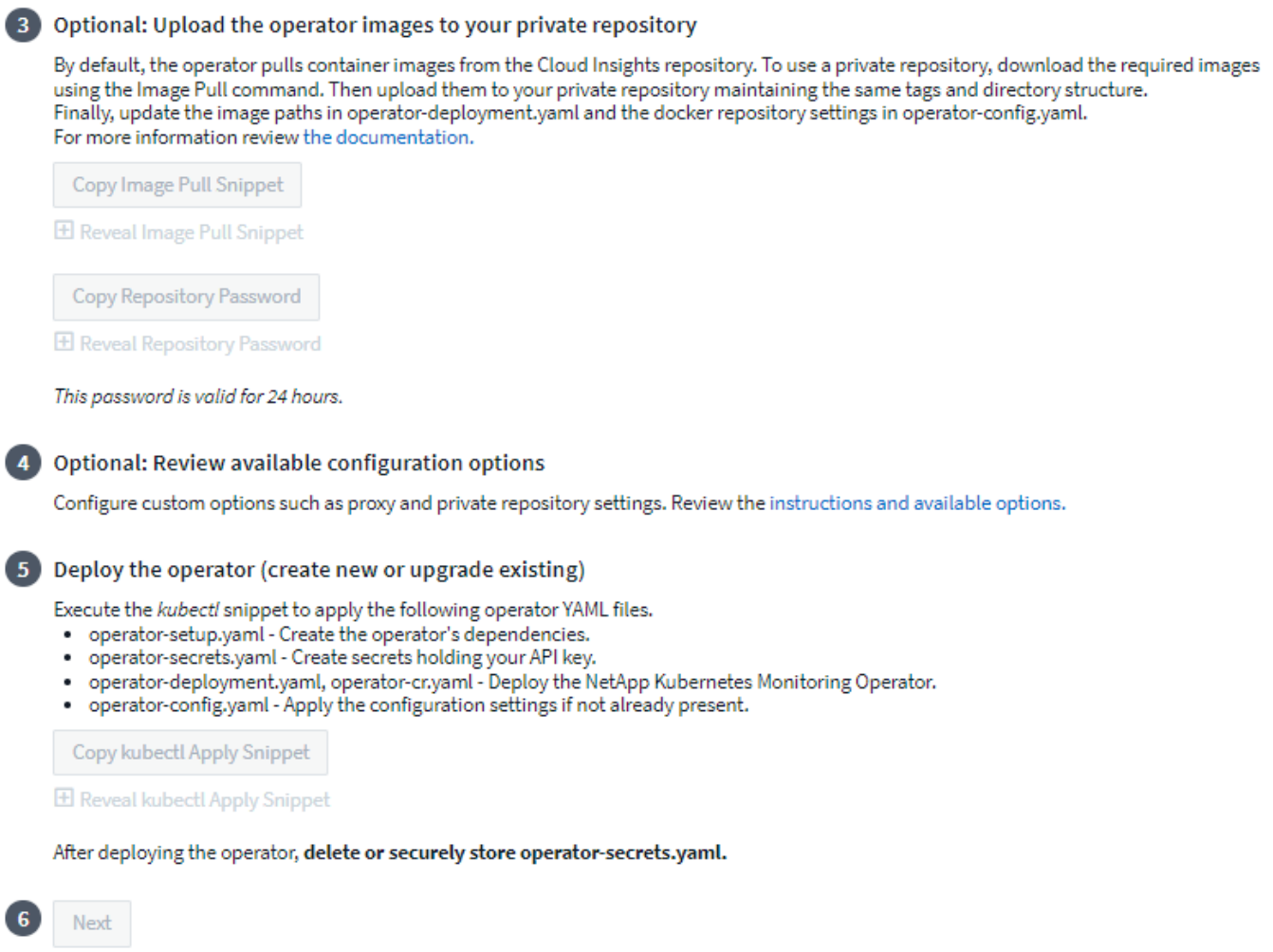
Data Infrastructure Insights bietet den **Kubernetes Monitoring Operator** für die Kubernetes-Sammlung an. Navigieren Sie zu **Kubernetes > Collectors > +Kubernetes Collector**, um einen neuen Operator bereitzustellen.

Bevor Sie den Kubernetes Monitoring Operator installieren

Lesen Sie die "Voraussetzungen" Dokumentation, bevor Sie den Kubernetes Monitoring Operator installieren oder aktualisieren.

Installieren des Kubernetes Monitoring Operator





Schritte zum Installieren des Kubernetes Monitoring Operator Agent auf Kubernetes:

1. Geben Sie einen eindeutigen Cluster-Namen und einen eindeutigen Namespace ein. Wenn Sie von einem früheren Kubernetes-Operator stammenAktualisierung, verwenden Sie den gleichen Cluster-Namen und den gleichen Namespace.
2. Sobald diese eingegeben wurden, können Sie den Download-Befehl-Snippet in die Zwischenablage kopieren.
3. Fügen Sie das Snippet in ein *bash* Fenster ein und führen Sie es aus. Die Installationsdateien des Bedieners werden heruntergeladen. Beachten Sie, dass das Snippet einen eindeutigen Schlüssel hat und für 24 Stunden gültig ist.
4. Wenn Sie ein benutzerdefiniertes oder privates Repository haben, kopieren Sie das optionale Bild-Pull-Snippet, fügen Sie es in eine *bash*-Shell ein und führen Sie es aus. Nachdem die Bilder gezogen wurden, kopieren Sie sie in Ihr privates Repository. Stellen Sie sicher, dass Sie dieselben Tags und Ordnerstrukturen beibehalten. Aktualisieren Sie die Pfade in *Operator-Deployment.yaml* sowie die Einstellungen des Docker-Repository in *Operator-config.yaml*.
5. Prüfen Sie bei Bedarf die verfügbaren Konfigurationsoptionen, z. B. Proxy- oder private Repository-Einstellungen. Lesen Sie mehr über "Konfigurationsoptionen".
6. Wenn Sie bereit sind, stellen Sie den Operator bereit, indem Sie den kubectl Apply-Snippet kopieren, herunterladen und ausführen.
7. Die Installation wird automatisch ausgeführt. Klicken Sie anschließend auf die Schaltfläche „*Next*“.

8. Wenn die Installation abgeschlossen ist, klicken Sie auf die Schaltfläche „*Next*“. Achten Sie darauf, auch die Datei *Operator-Secrets.yaml* zu löschen oder sicher zu speichern.

Wenn Sie einen Proxy verwenden, lesen Sie über Proxy wird konfiguriert.

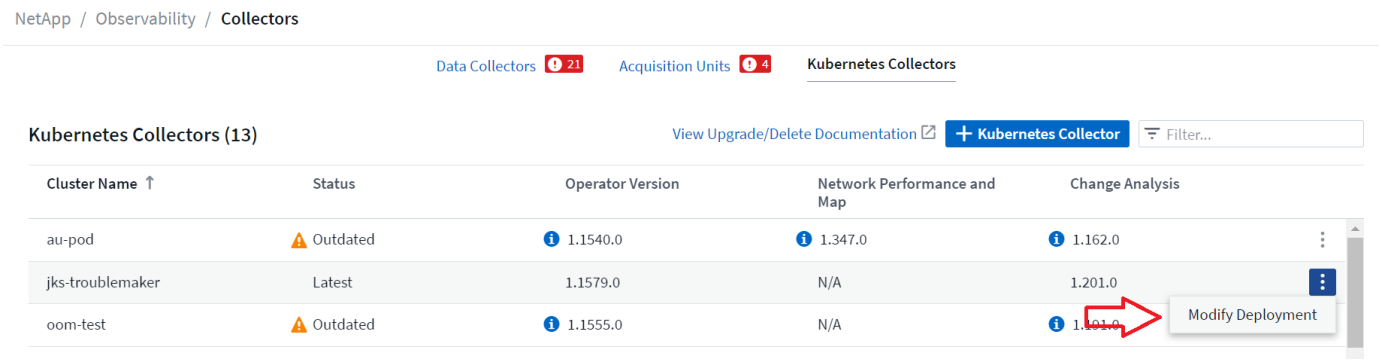
Wenn Sie ein benutzerdefiniertes Repository haben, lesen Sie über Ein benutzerdefiniertes/privates Docker-Repository verwenden.

Kubernetes-Monitoring-Komponenten

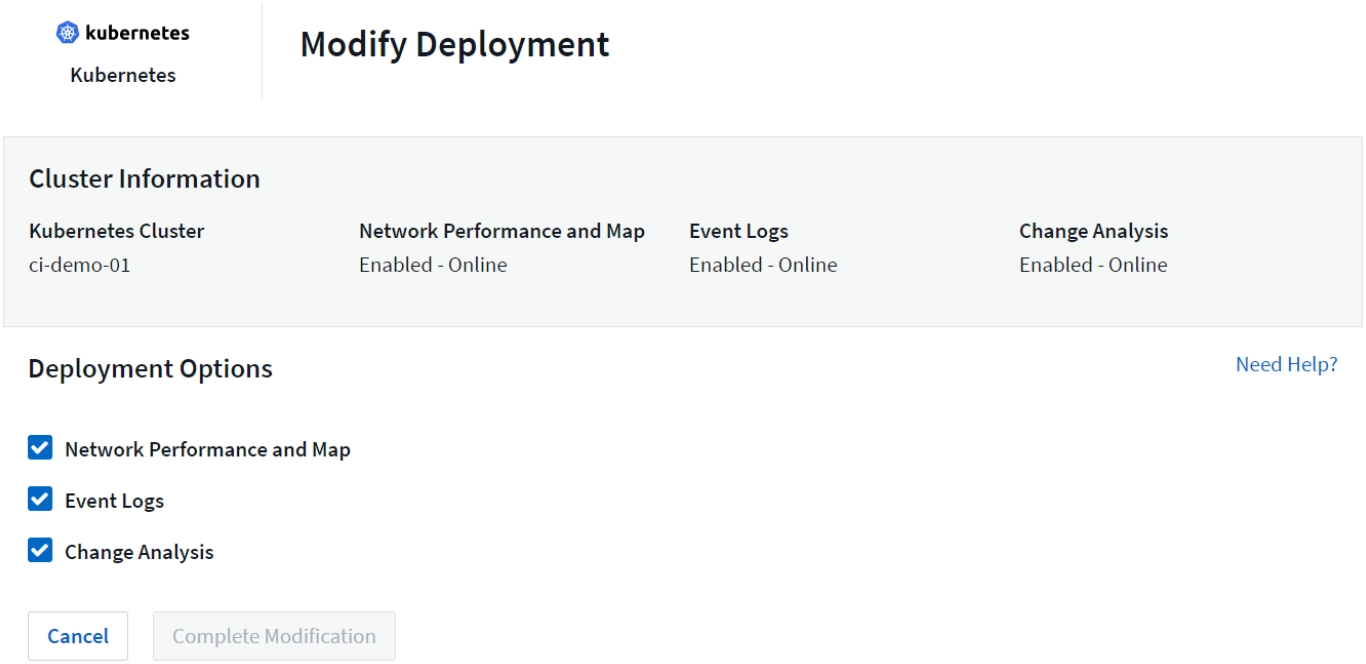
Data Infrastructure Insights Kubernetes Monitoring besteht aus vier Monitoring-Komponenten:

- Cluster-Kennzahlen
- Netzwerkleistung und -Zuordnung (optional)
- Ereignisprotokolle (optional)
- Änderungsanalyse (optional)

Die oben aufgeführten optionalen Komponenten sind standardmäßig für jeden Kubernetes-Collector aktiviert. Wenn Sie sich entscheiden, keine Komponente für einen bestimmten Collector zu benötigen, können Sie sie deaktivieren, indem Sie zu **Kubernetes > Collectors** navigieren und im Collector-Menü „drei Punkte“ rechts auf dem Bildschirm *Modify Deployment* auswählen.



Der Bildschirm zeigt den aktuellen Status jeder Komponente an und ermöglicht es Ihnen, Komponenten für diesen Collector nach Bedarf zu deaktivieren oder zu aktivieren.



Upgrade auf den neuesten Kubernetes Monitoring Operator

Ermitteln Sie, ob eine AgentConfiguration bei dem vorhandenen Operator vorhanden ist (wenn Ihr Namespace nicht der Standardwert *netapp-Monitoring* ist, ersetzen Sie den entsprechenden Namespace):

```
 kubectl -n netapp-monitoring get agentconfiguration netapp-monitoring-
configuration
Wenn eine AgentConfiguration vorhanden ist:
```

- Installieren Der letzte Operator über den vorhandenen Operator.
  - Stellen Sie sicher, dass Die neuesten Container-Bilder werden angezeigtSie ein benutzerdefiniertes Repository verwenden.

Wenn AgentConfiguration nicht vorhanden ist:

- Notieren Sie sich den von Data Infrastructure Insights erkannten Cluster-Namen (wenn Ihr Namespace nicht das standardmäßige NetApp-Monitoring ist, ersetzen Sie den entsprechenden Namespace):

```
 kubectl -n netapp-monitoring get agent -o
jsonpath='{.items[0].spec.cluster-name}'
* Erstellen Sie eine Sicherung des bestehenden Operators (wenn Ihr
Namespace nicht der Standard-netapp-Überwachung ist, ersetzen Sie den
entsprechenden Namespace):
```

```
 kubectl -n netapp-monitoring get agent -o yaml > agent_backup.yaml
* <<to-remove-the-kubernetes-monitoring-operator,Deinstallieren>> Der
vorhandene Operator.
* <<installing-the-kubernetes-monitoring-operator,Installieren>> Der
neueste Bediener.
```

- Verwenden Sie denselben Cluster-Namen.
- Nachdem Sie die neuesten Operator YAML-Dateien heruntergeladen haben, können Sie alle in Agent_Backup.yaml gefundenen Anpassungen vor der Bereitstellung an den heruntergeladenen Operator-config.yaml übertragen.
- Stellen Sie sicher, dass Die neuesten Container-Bilder werden angezeigtSie ein benutzerdefiniertes Repository verwenden.

Anhalten und Starten des Kubernetes Monitoring Operator

So beenden Sie den Kubernetes Monitoring Operator:

```
 kubectl -n netapp-monitoring scale deploy monitoring-operator
--replicas=0
So starten Sie den Kubernetes Monitoring Operator:
```

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

Deinstallation

Um den Kubernetes Monitoring Operator zu entfernen

Beachten Sie, dass der Standard-Namespace für den Kubernetes Monitoring Operator „netapp-Monitoring“ ist. Wenn Sie Ihren eigenen Namespace festgelegt haben, ersetzen Sie diesen Namespace in diesen und allen nachfolgenden Befehlen und Dateien.

Neuere Versionen des Überwachungsoperators können mit den folgenden Befehlen deinstalliert werden:

```
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>
```

Wenn der Überwachungsoperator in seinem eigenen dedizierten Namespace bereitgestellt wurde, löschen Sie den Namespace:

```
 kubectl delete ns <NAMESPACE>
Wenn der erste Befehl „Keine Ressourcen gefunden“ zurückgibt, verwenden
Sie die folgenden Anweisungen, um ältere Versionen des
Überwachungsoperators zu deinstallieren.
```

Führen Sie jeden der folgenden Befehle in der Reihenfolge aus. Abhängig von Ihrer aktuellen Installation können einige dieser Befehle Nachrichten ‘object not found’ zurückgeben. Diese Meldungen können sicher ignoriert werden.

```
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>
```

Wenn zuvor eine Sicherheitskontextbeschränkung erstellt wurde:

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

Über Kube-State-Metrics

Der NetApp Kubernetes Monitoring Operator installiert seine eigenen kube-State-Metriken, um Konflikte mit anderen Instanzen zu vermeiden.

Informationen über Kube-State-Metrics finden Sie unter "Auf dieser Seite".

Konfigurieren/Anpassen des Bedieners

Diese Abschnitte enthalten Informationen zur Anpassung Ihrer Bedienerkonfiguration, zur Arbeit mit Proxy, zur Verwendung eines benutzerdefinierten oder privaten Docker-Repositorys oder zur Arbeit mit OpenShift.

Konfigurationsoptionen

Die am häufigsten geänderten Einstellungen können in der benutzerdefinierten Ressource *AgentConfiguration* konfiguriert werden. Sie können diese Ressource bearbeiten, bevor Sie den Operator bereitstellen, indem Sie die Datei *Operator-config.yaml* bearbeiten. Diese Datei enthält kommentierte Beispiele für Einstellungen. In der Liste "Verfügbare Einstellungen"finden Sie die aktuellste Version des Operators.

Sie können diese Ressource auch bearbeiten, nachdem der Operator bereitgestellt wurde, indem Sie den

folgenden Befehl verwenden:

```
 kubectl -n netapp-monitoring edit AgentConfiguration
Um festzustellen, ob die bereitgestellte Version des Operators
AgentConfiguration unterstützt, führen Sie den folgenden Befehl aus:
```

```
 kubectl get crd agentconfigurations.monitoring.netapp.com
Wenn die Meldung „Fehler vom Server (notfound)“ angezeigt wird, muss Ihr
Bediener aktualisiert werden, bevor Sie die AgentConfiguration verwenden
können.
```

Proxy-Unterstützung Wird Konfiguriert

An zwei Stellen können Sie einen Proxy für Ihren Mandanten verwenden, um den Kubernetes Monitoring Operator zu installieren. Es kann sich um dieselben oder separate Proxy-Systeme handelt:

- Proxy wird während der Ausführung des Installationscode-Snippets (mit „Curl“) benötigt, um das System zu verbinden, auf dem das Snippet ausgeführt wird, mit Ihrer Data Infrastructure Insights-Umgebung
- Der vom Kubernetes Ziel-Cluster benötigte Proxy für die Kommunikation mit der Insights Umgebung Ihrer Dateninfrastruktur ist erforderlich

Wenn Sie einen Proxy für eine oder beide dieser Optionen verwenden, müssen Sie zur Installation des Kubernetes Operating Monitor zunächst sicherstellen, dass Ihr Proxy so konfiguriert ist, dass eine gute Kommunikation mit Ihrer Data Infrastructure Insights-Umgebung möglich ist. Wenn Sie über einen Proxy verfügen und von dem Server/der VM, von dem aus Sie den Operator installieren möchten, auf Data Infrastructure Insights zugreifen können, ist Ihr Proxy wahrscheinlich richtig konfiguriert.

Für den Proxy, der zur Installation des Kubernetes Operating Monitor verwendet wird, legen Sie vor der Installation des Operators die Umgebungsvariablen *http_Proxy/https_Proxy* fest. In einigen Proxy-Umgebungen müssen Sie möglicherweise auch die Variable *no_Proxy Environment* festlegen.

Um die Variablen festzulegen, führen Sie die folgenden Schritte auf Ihrem System aus * bevor* den Kubernetes Monitoring Operator installiert:

1. Legen Sie die Umgebungsvariable *https_Proxy* und/oder *http_Proxy* für den aktuellen Benutzer fest:
   a. Wenn der Proxy, der eingerichtet wird, keine Authentifizierung (Benutzername/Passwort) aufweist, führen Sie den folgenden Befehl aus:

```
 export https_proxy=<proxy_server>:<proxy_port>
.. Wenn der Proxy, der eingerichtet wird, über Authentifizierung
(Benutzername/Passwort) verfügt, führen Sie folgenden Befehl aus:
```

```
export
http_proxy=<proxy_username>:<proxy_password>@<proxy_server>:<proxy_po
rt>
```

Wenn der Proxy, der für das Kubernetes-Cluster zur Kommunikation mit der Insights Umgebung Ihrer Dateninfrastruktur verwendet wird, verwendet wird, installieren Sie den Kubernetes Monitoring Operator, nachdem Sie alle diese Anweisungen gelesen haben.

Konfigurieren Sie den Proxy-Abschnitt von AgentConfiguration in Operator-config.yaml, bevor Sie den Kubernetes Monitoring Operator bereitstellen.

```
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
  ...
...
```

Verwenden eines benutzerdefinierten oder privaten Docker Repositorys

Standardmäßig zieht der Kubernetes Monitoring Operator Container-Images aus dem Repository Data Infrastructure Insights. Wenn Sie ein Kubernetes-Cluster als Ziel für das Monitoring verwenden und der Cluster so konfiguriert ist, dass er nur Container-Images aus einem benutzerdefinierten oder privaten Docker-Repository oder der Container-Registrierung zieht, müssen Sie den Zugriff auf die Container konfigurieren, die vom Kubernetes Monitoring Operator benötigt werden.

Führen Sie das „Image Pull Snippet“ aus der NetApp Monitoring Operator Installationskachel aus. Dieser Befehl meldet sich beim Repository Data Infrastructure Insights an, zieht alle Image-Abhängigkeiten für den Operator ab und meldet sich vom Repository Data Infrastructure Insights ab. Wenn Sie dazu aufgefordert werden, geben Sie das angegebene temporäre Repository-Passwort ein. Mit diesem Befehl werden alle vom Bediener verwendeten Bilder heruntergeladen, einschließlich optionaler Funktionen. Nachfolgend sehen Sie, für welche Funktionen diese Bilder verwendet werden.

Core Operator-Funktionalität und Kubernetes Monitoring

- netapp Monitoring
- ci-kube-rbac-Proxy
- ci-ksm

- ci-telegraf
- Distroless-root-user

Ereignisprotokoll

- ci-Fluent-Bit
- ci-kubernetes-Event-Exporteur

Netzwerkleistung und -Zuordnung

- ci-Netz-Beobachter

Übertragen Sie das Operator-Docker-Image gemäß Ihren Unternehmensrichtlinien in das private/lokale/unternehmenseigene Docker-Repository. Stellen Sie sicher, dass die Bild-Tags und Verzeichnispfade zu diesen Images in Ihrem Repository mit denen im Data Infrastructure Insights Repository übereinstimmen.

Bearbeiten Sie die Bereitstellung des Monitoring-Operators in Operator-Deployment.yaml, und ändern Sie alle Bildverweise, um Ihr privates Docker-Repository zu verwenden.

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

Bearbeiten Sie die AgentConfiguration in Operator-config.yaml, um die neue Position des Docker-Repo zu berücksichtigen. Erstellen Sie ein neues imagePullSecret für Ihr privates Repository. Weitere Informationen finden Sie unter *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:
xref:{relative_path}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-Anweisungen

Wenn Sie OpenShift 4.6 oder höher ausführen, müssen Sie die AgentConfiguration in *Operator-config.yaml* bearbeiten, um die Einstellung *runPrivileged* zu aktivieren:

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

OpenShift kann zusätzliche Sicherheitsstufen implementieren, die den Zugriff auf einige Kubernetes-Komponenten blockieren könnten.

Toleranzen und Verfleckungen

Die DemonSets *netapp-CI-telegraf-ds*, *netapp-CI-Fluent-Bit-ds* und *netapp-CI-net-Observer-l4-ds* müssen für jeden Node im Cluster einen Pod planen, damit Daten auf allen Nodes korrekt erfasst werden. Der Operator wurde so konfiguriert, dass er einige bekannte **Fehler** toleriert. Wenn Sie auf Ihren Knoten benutzerdefinierte Taints konfiguriert haben und damit verhindern, dass Pods auf jedem Knoten ausgeführt werden, können Sie für diese Taints eine **Toleration** erstellen"In der *AgentConfiguration*". Wenn Sie auf alle Nodes im Cluster benutzerdefinierte Taints angewendet haben, müssen Sie der Operator-Bereitstellung auch die erforderlichen Toleranzen hinzufügen, damit der Operator-Pod geplant und ausgeführt werden kann.

Erfahren Sie mehr über Kubernetes "Tönungen und Tolerationen".

Kehren Sie zum zurück **"NetApp Kubernetes Monitoring Operator Installation** Seite"

Ein Hinweis über Geheimnisse

Um die Berechtigung für den Kubernetes Monitoring Operator zum Anzeigen der geheimen Daten im gesamten Cluster zu entfernen, löschen Sie vor der Installation die folgenden Ressourcen aus der Datei *Operator-Setup.yaml*:

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

Wenn es sich um ein Upgrade handelt, löschen Sie auch die Ressourcen aus Ihrem Cluster:

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

Wenn die Änderungsanalyse aktiviert ist, ändern Sie die Optionen *AgentConfiguration* oder *Operator-config.yaml*, um den Änderungsmanagementabschnitt zu entkommentieren und *kindsToIgnoreFromWatch: '"Secrets"* im Bereich Change-Management aufzunehmen. Notieren Sie sich das Vorhandensein und die Position von einfachen und doppelten Anführungszeichen in dieser Zeile.

```
# 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"'
  ...
```

Überprüfen Der Signaturen Der Kubernetes Monitoring Operator Images

Das Bild für den Betreiber und alle damit verbundenen Bilder werden von NetApp signiert. Sie können die Images vor der Installation mit dem cosign-Tool manuell überprüfen oder einen Kubernetes-Aufnahme-Controller konfigurieren. Weitere Informationen finden Sie im "Kubernetes-Dokumentation".

Der öffentliche Schlüssel, der zur Überprüfung der Bildsignaturen verwendet wird, ist in der Kachel Monitoring Operator install unter *Optional: Laden Sie die Operatorbilder in Ihr privates Repository > Image Signature Public Key*

So überprüfen Sie eine Bildsignatur manuell:

1. Kopieren Sie das Bild-Pull-Snippet, und führen Sie es aus
2. Kopieren Sie das Repository-Kennwort, und geben Sie es ein, wenn Sie dazu aufgefordert werden
3. Speichern Sie den Public Key der Bildsignatur (im Beispiel dii-image-signing.Pub).
4. Überprüfen Sie die Bilder mit cosign. Beachten Sie das folgende Beispiel für die Verwendung von Cosign

```
$ cosign verify --key dii-image-signing.pub --insecure-ignore-sct
--insecure-ignore-tlog <repository>/<image>:<tag>
Verification for <repository>/<image>:<tag> --
The following checks were performed on each of these signatures:
  - The cosign claims were validated
  - The signatures were verified against the specified public key
[{"critical":{"identity":{"docker-
reference":"<repository>/<image>"},"image":{"docker-manifest-
digest":"sha256:<hash>"},"type":"cosign container image
signature"},"optional":null}]
```

Fehlerbehebung

Bei Problemen beim Einrichten des Kubernetes Monitoring Operator sollten Sie Folgendes versuchen:

Problem:	Versuchen Sie dies:
Ich sehe keinen Hyperlink/Verbindung zwischen meinem Kubernetes Persistent Volume und dem entsprechenden Back-End Storage-Gerät. Mein Kubernetes Persistent Volume wird mit dem Hostnamen des Storage-Servers konfiguriert.	Befolgen Sie die Schritte, um den bestehenden Telegraf-Agent zu deinstallieren, und installieren Sie dann den neuesten Telegraf-Agent erneut. Sie müssen Telegraf Version 2.0 oder höher verwenden. Der Kubernetes-Cluster-Storage muss aktiv durch Data Infrastructure Insights überwacht werden.
Ich sehe Meldungen in den Protokollen, die folgende ähneln: E0901 15 352:21:39.962145 1 Reflektor.go:178] k8s.io/kube-State-metrics/internal/Store/Builder.go:352: Fehler beim Auflisten *v1.MutatingWebhookKonfiguration: Der Server konnte die angeforderte Ressource E0901 15:21:43.168161 1 Reflector.go:178] k8s.io/kube-Builder nicht finden	Diese Nachrichten können auftreten, wenn Sie kube-State-Metrics Version 2.0.0 oder höher mit Kubernetes-Versionen unter 1.20 ausführen. Um die Kubernetes-Version zu erhalten: *Kubectl Version* um die kube-State-metrics-Version zu erhalten: *Kubectl get Deploy/kube-State-metrics -o jsonpath='{..image}'* um zu verhindern, dass diese Nachrichten passieren, können Benutzer ihre kube-State-Metrics-Implementierung ändern, um die folgenden Elemente zu deaktivieren: _Mutingwebhookkonfigurationen___volumehaWeitere Resources=certificationesigningrequests,configmaps, cronjobs,dämsets, Bereitstellungen,Endpunkte,HorizontalpodAutoscaler, nesresses,Jobs,Begrenzungsbereiche,Namensräume, Netzwerkrichtlinien,Knoten,Persistenz,stagemasnesmases,nesmasnesmases,nesmasnesmasnesmasnesnesmasnesequets,ndecoses,nescontascrises,nesequequequesefises,nesequequesequesefiscones,mases,nesequidatequesequesefiscones,nesequesequesefiscrises,nesequesequesefiscones,nesefisconesefisconmases,mases,nesequesequesefiscones,necequesequeseques Validatingwebhookkonfigurationen, Volumeanhänge“
Ich sehe Fehlermeldungen von Telegraf ähnlich wie die folgenden, aber Telegraf startet und läuft: Okt 11 14:23:41 ip-172-31-39-47 systemd[1]: Startete den Plugin-getriebenen Server Agent für das Reporting von Metriken in InfluxDB. Okt 11 14:23:41 ip-172-31-39-47 telegraf[1827]: Time=„2021-10-11T14:23:41Z“ Level=error msg=„konnte kein Cache-Verzeichnis erstellen. /Etc/telegraf/.Cache/snowflake, err: Mkdir /etc/telegraf/.ca che: Berechtigung verweigert. Ignored\n" func=„gosnowflake.(*defaultLogger).Errorf“ file=„log.go:1827 23“ Okt 31 2021:39-47 10 ip-172-11 14-23:41 telegraf[120]: Time=„41-11TZ Fehler“:41T14=. Ignored. Open /etc/telegraf/.Cache/snowflake/ocsp_response_Cache .json: No such file or Directory\n" func=„gosnowflake.(*defaultLogger).Errorf“ file=„log.go:23“ Okt 2021:10 ip-1827-31-39-47 telegraf[172]: 11 14-23:41-11T11T14:120:41Z I! Telegraf 1.19.3 Starten	Dies ist ein bekanntes Problem. "Dieser GitHub-Artikel"Weitere Informationen finden Sie unter. Solange Telegraf läuft, können Benutzer diese Fehlermeldungen ignorieren.

Problem:	Versuchen Sie dies:
Auf Kubernetes meldet mein Telegraf pod(s) den folgenden Fehler: „Fehler in der Verarbeitung von mountstats-Infos: Habe mountstats-Datei nicht geöffnet: /Hostfs/proc/1/mountstats, Fehler: Open /hostfs/proc/1/mountstats: Permission dementied“	Wenn SELinux aktiviert und durchgesetzt wird, wird wahrscheinlich verhindert, dass die Telegraf PODs auf die Datei /proc/1/mountstats auf dem Kubernetes-Knoten zugreifen. Um diese Einschränkung zu überwinden, bearbeiten Sie die Agentkonfiguration und aktivieren Sie die runPrivileged-Einstellung. Weitere Informationen finden Sie im "OpenShift-Anweisungen".
Auf Kubernetes meldet mein Telegraf ReplicaSet POD den folgenden Fehler: [inputs.prometheus] Fehler im Plugin: Konnte keine keypair /etc/kubernetes/pki/etcd/Server.crt:/etc/kubernetes/pki /etcd/Server.key: Öffnen /etc/kubernetes/pki/etcd/Server.crt: Keine solche Datei oder Verzeichnis	Der Pod Telegraf ReplicaSet soll auf einem Knoten ausgeführt werden, der als Master oder für etc bestimmt ist. Wenn der ReplicaSet-Pod auf einem dieser Knoten nicht ausgeführt wird, werden diese Fehler angezeigt. Überprüfen Sie, ob Ihre Master/etcd-Knoten eine Tönungswalle haben. Fügen Sie in diesem Fall die erforderlichen Verträgungen in das Telegraf ReplicaSet, telegraf-rs ein. Bearbeiten Sie zum Beispiel die Datei ReplicaSet… kubectl edit rs telegraf-rs …und fügen Sie die entsprechenden Verträgungen der Spezifikation hinzu. Starten Sie anschließend den Pod ReplicaSet neu.
Ich habe eine PSP/PSA Umgebung. Hat dies Auswirkungen auf meinen Überwachungsperator?	Wenn Ihr Kubernetes-Cluster mit Pod-Sicherheitsrichtlinie (PSP) oder Pod Security Admission (PSA) ausgeführt wird, müssen Sie ein Upgrade auf den aktuellen Kubernetes Monitoring Operator durchführen. Gehen Sie wie folgt vor, um auf den aktuellen Operator mit Unterstützung für PSP/PSA zu aktualisieren: 1. Deinstallieren Der bisherige Monitoring-Operator: Kubectl delete Agent-Monitoring-NetApp -n NetApp-Monitoring kubectl delete ns NetApp-Monitoring kubectl delete crd Agents.Monitoring.NetApp.com kubectl delete clusterrole Agent-Manager-role Agent-Proxy-role Agent-metrics-reader kubectl delete clusterrolebinding Agent-Manager-rolebinding Agent-Proxy-rolebinding Agent-rolebinding Agent-Cluster-admin-rolebinding 2. Installieren Die neueste Version des Überwachungsbedieners.
Ich habe Probleme beim Versuch, den Operator bereitzustellen, und ich habe PSP/PSA in Gebrauch.	1. Bearbeiten Sie den Agenten mit folgendem Befehl: Kubectl -n <name-space> edit Agent 2. Markieren Sie „Sicherheitspolitik aktiviert“ als „falsch“. Dadurch werden Pod-Sicherheitsrichtlinien und Pod-Sicherheitszulassung deaktiviert und der Bediener kann die Bereitstellung durchführen. Bestätigung mit den folgenden Befehlen: Kubectl get psp (sollte Pod Security Policy entfernt zeigen) kubectl get all -n <Namespace> grep -i psp (sollte zeigen, dass nichts gefunden wird)

Problem:	Versuchen Sie dies:
„ImagePullBackoff“-Fehler erkannt	Diese Fehler können auftreten, wenn Sie über ein benutzerdefiniertes oder privates Docker-Repository verfügen und den Kubernetes Monitoring Operator noch nicht so konfiguriert haben, dass er es richtig erkennt. Weitere Informationen Info über die Konfiguration für benutzerdefinierte/private Repo.

Ich habe ein Problem mit der Installation meines Monitoring-Bedieners, und die aktuelle Dokumentation hilft mir nicht, es zu lösen.	Erfassen oder notieren Sie die Ausgabe der folgenden Befehle, und wenden Sie sich an den technischen Support.
	`kubectl -n netapp-monitoring get all`
	`kubectl -n netapp-monitoring describe all`
	`kubectl -n netapp-monitoring logs <monitoring-operator-pod> --all-containers=true`
	`kubectl -n netapp-monitoring logs <telegraf-pod> --all-containers=true`

NET-Observer (Workload Map)-Pods im Operator Namespace befinden sich in CrashLoopBackOff	Diese Pods entsprechen dem Workload Map-Datensammler für Network Observability. Versuchen Sie Folgendes: • Überprüfen Sie die Protokolle eines der Pods, um die minimale Kernel-Version zu bestätigen. Beispiel: --- {"CI-Tenant-id":"your-Tenant-id","Collector-Cluster":"your-k8s-Cluster-Name","Environment":"prod","Level":"error","msg":"failed in validation. Grund: Kernel-Version 3.10.0 ist kleiner als die minimale Kernel-Version von 4.18.0","Time":"2022-11-09T08:23:08Z"} ---- • Net-Observer-Pods erfordern die Linux-Kernel-Version mindestens 4.18.0. Überprüfen Sie die Kernel-Version mit dem Befehl „uname -r“ und stellen Sie sicher, dass sie >= 4.18.0 sind
Pods werden im Operator Namespace ausgeführt (Standard: netapp-Monitoring), es werden jedoch keine Daten in der UI für die Workload-Zuordnung oder Kubernetes-Metriken in Abfragen angezeigt	Überprüfen Sie die Zeiteinstellung auf den Knoten des K8S-Clusters. Für eine genaue Prüfung und Datenberichterstattung wird dringend empfohlen, die Zeit auf dem Agent-Rechner mit Network Time Protocol (NTP) oder Simple Network Time Protocol (SNTP) zu synchronisieren.

Problem:	Versuchen Sie dies:
Einige der Net-Observer-Pods im Namespace Operator befinden sich im Status „Ausstehend“	NET-Observer ist ein DemonSet und führt in jedem Knoten des K8s-Clusters einen Pod aus. • Beachten Sie den Pod, der sich im Status „Ausstehend“ befindet, und prüfen Sie, ob ein Ressourcenproblem für CPU oder Speicher vorliegt. Stellen Sie sicher, dass der erforderliche Arbeitsspeicher und die erforderliche CPU im Knoten verfügbar sind.
Ich sehe Folgendes in meinen Protokollen sofort nach der Installation des Kubernetes Monitoring Operators: [inputs.prometheus] Fehler im Plugin: Fehler beim Erstellen einer HTTP-Anforderung an 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: Kein solcher Host	Diese Meldung wird normalerweise nur angezeigt, wenn ein neuer Operator installiert ist und der Pod „*telegraf-rs*“ vor dem Einschalten des Pod „*ksm*“ steht. Diese Meldungen sollten beendet werden, sobald alle Pods ausgeführt werden.
Ich sehe keine Kennzahlen für die Kubernetes-Kronjobs, die in meinem Cluster vorhanden sind, erfasst.	Überprüfen Sie Ihre Kubernetes-Version (d. h. `kubectl version`). Wenn es v1.20.x oder niedriger ist, ist dies eine erwartete Einschränkung. Die mit dem Kubernetes Monitoring Operator implementierte Version von kube-State-Metrics unterstützt nur v1.cronjob. Bei Kubernetes 1.20.x und niedriger befindet sich die Ressource cronjob unter v1beta.cronjob. Daher können kube-State-Metriken die Ressource cronjob nicht finden.
Nach der Installation des Bedieners geben die telegraf-ds-Pods CrashLoopBackOff ein und die POD-Protokolle zeigen „su: Authentication failure“ an.	Bearbeiten Sie den Abschnitt telegraf in *AgentConfiguration*, und setzen Sie *dockerMetricCollectionEnabled* auf false. Weitere Informationen finden Sie im "Konfigurationsoptionen". … Spec: … telegraf: … - Name: docker Run-Mode: - DemonSet Ersetzungen: - Schlüssel: DOCKER_UNIX_SOCK_PLACEHOLDER Wert: unix:///run/Docker.Sock … …
Ich sehe wiederholte Fehlermeldungen wie die folgenden in meinen Telegraf-Logs: E! [Agent] Fehler beim Schreiben in Outputs.http: Post "https://<tenant_url>/Rest/v1/Lake/ingest/influxdb": Kontext-Deadline überschritten (Client. Zeitüberschreitung beim Warten auf Header überschritten)	Bearbeiten Sie den Abschnitt telegraf in *AgentConfiguration*, und erhöhen Sie *outputTimeout* auf 10s. Weitere Informationen finden Sie im "Konfigurationsoptionen".
Ich vermisse *involvedobject* Daten für einige Event Logs.	Stellen Sie sicher, dass Sie die Schritte im Abschnitt oben befolgt haben"Berechtigungen".

Problem:	Versuchen Sie dies:
Wieso werden zwei Monitoring Operator Pods ausgeführt, einer mit dem Namen netapp-CI-Monitoring-Operator-<pod> und der andere mit dem Namen Monitoring-Operator-<pod>?	Seit dem 12. Oktober 2023 hat Data Infrastructure Insights den Betreiber refaktorisiert, um unseren Benutzern besser dienen zu können. Damit diese Änderungen vollständig umgesetzt werden, müssen Sie Entfernen Sie den alten Bedienerund Installieren Sie den neuen.
Meine kubernetes-Ereignisse haben unerwartet aufgehört, Daten bei Infrastruktur-Insights zu melden.	Rufen Sie den Namen des POD für den Event-Exporter ab: `kubectl -n netapp-monitoring get pods`
grep event-exporter	awk '{print $1}'
sed 's/event-exporter./event-exporter/'` Es sollte entweder „netapp-CI-Event-Exporteur“ oder „Event-Exporteur“ sein. Bearbeiten Sie anschließend den Überwachungsagenten `kubectl -n netapp-monitoring edit agent`und legen Sie den Wert für LOG_FILE so fest, dass der entsprechende POD-Name des Ereignisexporteurs im vorherigen Schritt angezeigt wird. Genauer gesagt sollte LOG_FILE auf "/var/log/Containers/netapp-CI-Event-exporteur.log" oder "/var/log/Containers/Event-exporteur*.log" gesetzt werden …. fluent-bit: … - name: event-exporter-ci substitutions: - key: LOG_FILE values: - /var/log/containers/netapp-ci-event-exporter*.log … …. Alternativ kann man auch Deinstallieren und Neu installieren den Agenten.	Ich sehe POD(s), die vom Kubernetes-Monitoring-Operator bereitgestellt werden, aufgrund unzureichender Ressourcen.
Informationen zum Erhöhen der CPU- und/oder Speichergrenzen finden Sie im Kubernetes Monitoring Operator"Konfigurationsoptionen".	Durch ein fehlendes Image oder eine ungültige Konfiguration wurden die netapp-CI-kube-State-metrics Pods nicht gestartet oder nicht einsatzbereit gemacht. Jetzt bleibt StatefulSet stecken und Konfigurationsänderungen werden nicht auf die Pods mit den netapp-CI-kube-State-Metriken angewendet.

Problem:	Versuchen Sie dies:
StatefulSet befindet sich in einem "Defekt" Status. Nachdem Sie Konfigurationsprobleme behoben haben, springen die netapp-CI-kube-State-metrics-Pods an.	Pods mit netapp-CI-kube-Status-Metriken können nicht gestartet werden, nachdem ein Kubernetes Operator Upgrade ausgeführt wurde. Es wird ErrImagePull geworfen (es konnte nicht das Image entfernt werden).
Versuchen Sie, die Pods manuell zurückzusetzen.	„Event disorded as being alder then maxEventAgeSeconds“ Meldungen werden für meinen Kubernetes Cluster unter Log Analysis beobachtet.
Ändern Sie den Operator *agentkonfiguration*, und erhöhen Sie die Erweiterung *Event-exporteur-maxEventAgeSeconds* (d. h. auf 60s), *Event-exporteur-kubeQPS* (d. h. auf 100) und *Event-exporteur-kubeBurst* (d. h. auf 500). Weitere Informationen zu diesen Konfigurationsoptionen finden Sie auf der "Konfigurationsoptionen" Seite.	Telegraf warnt vor unzureichenden, abschließbaren Speichern oder stürzt ab.
Versuchen Sie, die Grenze des abschließbaren Speichers für Telegraf im zugrunde liegenden Betriebssystem/Knoten zu erhöhen. Wenn eine Erhöhung des Limits keine Option ist, ändern Sie die NKMO-Agentkonfiguration und setzen Sie *Unprotected* auf *true*. Dadurch wird Telegraf angewiesen, keine gesperrten Speicherseiten zu reservieren. Dies kann zwar ein Sicherheitsrisiko darstellen, da entschlüsselte Geheimnisse möglicherweise auf die Festplatte ausgetauscht werden, ermöglicht aber die Ausführung in Umgebungen, in denen das Reservieren von gesperrtem Speicher nicht möglich ist. Weitere Informationen zu den Konfigurationsoptionen *Unprotected* finden Sie auf der "Konfigurationsoptionen" Seite.	Ich sehe Warnhinweise von Telegraf wie folgt: *W! [Inputs.diskio] der Datenträgername für „vdc“ kann nicht erfasst werden: Fehler beim Lesen von /dev/vdc: Keine Datei oder Verzeichnis*
Für den Kubernetes Monitoring Operator sind diese Warnmeldungen gutartig und können sicher ignoriert werden. Alternativ können Sie den telegraf-Abschnitt in AgentConfiguration bearbeiten und *runDsPrivileged* auf true setzen. Weitere Informationen finden Sie im "Konfigurationsoptionen des Bedieners".	Mein Fluent-Bit-Pod schlägt mit den folgenden Fehlern fehl: [2024/10/16 14:16:23] [error] [/src/Fluent-Bit/Plugins/in_tail/tail_fs_inotify.c:360 errno=10/16 14] zu viele geöffnete Dateien [16/23:16:23] [error] initialisieren des Input tail.0 [2024/24:2024:10/16 14] [error] die Eingabe-Initialisierung ist fehlgeschlagen

Weitere Informationen finden Sie auf der "Support" Seite oder im "Data Collector Supportmatrix".

Konfigurationsoptionen Für Kubernetes Monitoring Operator

Die "Kubernetes Monitoring Operator" Konfiguration kann angepasst werden.

In der folgenden Tabelle sind die möglichen Optionen für die *AgentConfiguration*-Datei aufgeführt:

Komponente	Option	Beschreibung
Agent		Konfigurationsoptionen, die allen Komponenten gemeinsam sind, die der Bediener installieren kann. Diese können als "globale" Optionen betrachtet werden.
	DockerRepo	Ein ockerRepo-Überschreiben, um Bilder von privaten Docker-Repos des Kunden im Vergleich zu Data Infrastructure Insights Docker Repo zu beziehen. Der Standardwert ist Data Infrastructure Insights Docker Repo
	DockerImagePullSecret	Optional: Ein Geheimnis für den Kunden private repo
	ClusterName	Freitextfeld, das einen Cluster über alle Kundencluster eindeutig identifiziert. Diese sollte bei einem Mandanten von Dateninfrastruktur Insights eindeutig sein. Der Standardwert ist das, was der Kunde in die Benutzeroberfläche für das Feld „Cluster Name“ eingibt
	Proxy Format: Proxy: Server: Port: Benutzername: Passwort: Noproxy: IsTelegrafProxyEnabled: IsAuProxyEnabled: IsFluentbitProxyEnabled: IsCollectorProxyEnabled:	Optional zum Festlegen des Proxys. Dies ist in der Regel der Unternehmensvertreter des Kunden.
telegraf		Konfigurationsoptionen, mit denen die telegraf-Installation des Bedieners angepasst werden kann
	Erfassungsintervall	Messgrößen-Erfassungsintervall, in Sekunden (max. = 60 s)
	DsCpuLimit	CPU-Limit für telegraf ds
	DsMemLimit	Speicherlimit für telegraf ds
	DsCpuRequest	CPU-Anforderung für telegraf ds
	DsMemRequest	Speicheranforderung für telegraf ds
	RsCpuLimit	CPU-Limit für telegraf rs
	RsMemLimit	Speichergrenze für telegraf rs
	RsCpuRequest	CPU-Anforderung für telegraf rs
	RsMemRequest	Speicheranforderung für telegraf rs
	RunPrivileged	Führen Sie den Container *telegraf-mountstats-Poller* des telegraf DemonSet im privilegierten Modus aus. Setzen Sie dies auf „true“, wenn SELinux auf Ihren Kubernetes-Nodes aktiviert ist.
	RunDsPrivileged	Stellen Sie runDsPrivileged auf true, um den telegraf DemonSet-Container im privilegierten Modus auszuführen.

Komponente	Option	Beschreibung
	Stapelgröße	Siehe "Telegraf-Konfigurationsdokumentation"
	BufferLimit	Siehe "Telegraf-Konfigurationsdokumentation"
	Rundintervall	Siehe "Telegraf-Konfigurationsdokumentation"
	SammlungJitter	Siehe "Telegraf-Konfigurationsdokumentation"
	Präzision	Siehe "Telegraf-Konfigurationsdokumentation"
	Flushintervall	Siehe "Telegraf-Konfigurationsdokumentation"
	FlushJitter	Siehe "Telegraf-Konfigurationsdokumentation"
	AusgabeTimeout	Siehe "Telegraf-Konfigurationsdokumentation"
	DsToleranzen	telegraf-ds zusätzliche Toleranzen.
	RsToleranzen	telegraf-rs zusätzliche Toleranzen.
	SkipProcessorsAfterAggregatoren	Siehe "Telegraf-Konfigurationsdokumentation"
	Ungeschützt	Siehe hier "Bekanntes Problem mit Telegraf". Durch die Einstellung *Unprotected* wird der Kubernetes Monitoring Operator angewiesen, Telegraf mit dem Flag auszuführen `--unprotected`.
status-Kennzahlen von kube		Konfigurationsoptionen, mit denen die installation von kube-Statusmetriken des Operators angepasst werden kann
	CpuLimit	CPU-Limit für die bereitstellung von kube-State-Metriken
	MemLimit	MEM-Limit für die implementierung von kube-State-Metriken
	CpuRequest	CPU-Anforderung für die Bereitstellung von kube-Statusmetriken
	MemRequest	MEM-Anforderung für die Bereitstellung von kube-Statuskennzahlen
	Ressourcen	Eine kommagetrennte Liste der Ressourcen, die erfasst werden sollen. Beispiel: Cronjobs,demonsets,Bereitstellungen,ingresses,Jobs, Namespaces,Nodes,persistent Volumeclaims, persistent Volumes,Pods,Replikasets,resourcequotas,Services,statefulsets
	Toleranzen	zusätzliche Toleranzen für kube-State-Metriken.
	Etiketten	Eine durch Kommas getrennte Liste von Ressourcen, die kube-State-metrics erfassen sollte + Beispiel: Cronjobs=**[],demonsets=[]**,Deployments=**[],ingresses=[]**,Jobs=**[],namepaces=[]**,Nodes=**[], persistent volumeclaims=[]=[][[**+

Komponente	Option	Beschreibung
Protokolle		Konfigurationsoptionen, mit denen die Protokollsammlung und die Installation des Bedieners angepasst werden können
	Wieder FromHead	Wahr/falsch, sollte fließendes Bit das Protokoll vom Kopf lesen
	Zeitüberschreitung	Timeout in Sekunden
	DnsMode	TCP/UDP, Modus für DNS
	Fluent-Bit-Tolerationen	Fluent-Bit-ds zusätzliche Toleranzen.
	Ereignis-Exporteur-Tolerationen	Ereignis-Exporteur zusätzliche Toleranzen.
	Event-Exporteur-maxEventAgeSeconds	Ereignis-Exporteur max. Ereignisalter. Siehe https://github.com/jkroepke/resmoio-kubernetes-event-exporter
Workload-Zuordnung		Konfigurationsoptionen, mit denen die Erfassung der Workload-Zuordnung und die Installation des Operators angepasst werden können.
	CpuLimit	CPU-Limit für Netto-Observer ds
	MemLimit	MEM-Grenze für Netto-Beobachter ds
	CpuRequest	CPU-Anforderung für Netto-Observer-ds
	MemRequest	MEM-Anforderung für Netto-Beobachter ds
	MetricAggregationInterval	Intervall für die metrische Aggregation in Sekunden
	BpfPollInterval	BPF-Abfrageintervall in Sekunden
	EnableDNSLookup	True/false, DNS-Suche aktivieren
	l4-Tolerationen	NET-Observer-l4-ds zusätzliche Toleranzen.
	RunPrivileged	True/false - Setzen Sie runPrivileged auf true, wenn SELinux auf Ihren Kubernetes-Knoten aktiviert ist.
Änderungsmanagement		Konfigurationsoptionen für das Kubernetes Change Management und die Analyse
	CpuLimit	CPU-Limit für Change-Observer-watch-rs
	MemLimit	MEM Limit für Change-Observer-Watch-rs
	CpuRequest	CPU-Anforderung für Change-Observer-watch-rs
	MemRequest	MEM-Anforderung für Change-Observer-Watch-rs
	AusfallerklärungIntervalMins	Intervall in Minuten, nach dem eine nicht erfolgreiche Bereitstellung eines Workloads als fehlgeschlagen markiert wird
	EinsatzAggrIntervalSekunden	Häufigkeit, mit der Ereignisse zur laufenden Workload-Bereitstellung gesendet werden

Komponente	Option	Beschreibung
	Nicht-WorkloadAggrIntervalSekunden	Häufigkeit der Kombination und des Sendeens von nicht-Workload-Implementierungen
	TermsToAkt	Ein Satz von regulären Ausdrücken, die in Env-Namen und Datenkarten verwendet werden, deren Wert bearbeitet wird Beispielbegriffe:"pwd", "password", "Token", "apikey", "API-key", "jwt"
	Zusätzlich KindsToWatch	Eine kommagetrennte Liste mit weiteren Arten, die von den vom Sammler überwachten Standardtypen überwacht werden sollen
	KindsToIgnoreFromWatch	Eine kommagetrennte Liste von Arten, die ignoriert werden sollen, wenn sie von den vom Sammler überwachten Standardtypen überwacht werden
	LogRecordAggrIntervalSekunden	Häufigkeit, mit der Protokolldatensätze vom Collector an CI gesendet werden
	Überwachen von Toleranzen	Change-Observer-watch-ds zusätzliche Toleranzen. Nur abgekürztes Einzelzeilenformat. Beispiel: '{key: Taint1, Operator: Existiert, Effekt: NoSchedule},{key: Taint2, Operator: Existiert, Effekt: NoExecute}'

Beispieldatei für AgentConfiguration

Unten finden Sie eine *AgentConfiguration*-Beispieldatei.

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

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 Data
Infrastructure Insights environment.
    clusterName: "my_cluster"
```

```
    # # Proxy settings. The proxy that the operator should use to send
metrics to Data Infrastructure 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.c01.cloudinsights.netapp.com'
    # # [Required Field] The name of the imagePullSecret for dockerRepo.
    # # If you are using a private repository, change this field from
'netapp-ci-docker' to the name of your secret.
    dockerImagePullSecret: 'netapp-ci-docker'

    # # Allow the operator to automatically rotate its ApiKey before
expiration.
    # tokenRotationEnabled: 'true'
    # # Number of days before expiration that the ApiKey should be
rotated. This must be less than the total ApiKey duration.
    # tokenRotationThresholdDays: '30'

  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#a
gent

    # # The default time telegraf will wait between inputs for all plugins
(interval). Max=60
    # collectionInterval: '60s'
    # # Maximum number of records per output that telegraf will write in
one batch (metric_batch_size).
```

```
    # batchSize: '10000'
    # # Maximum number of records per output that telegraf will cache
pending a successful write (metric_buffer_limit).
    # bufferLimit: '150000'
    # # Collect metrics on multiples of interval (round_interval).
    # roundInterval: 'true'
    # # Each plugin waits a random amount of time between the scheduled
collection time and that time + collection_jitter before collecting inputs
(collection_jitter).
    # collectionJitter: '0s'
    # # Collected metrics are rounded to the precision specified. When set
to "0s" precision will be set by the units specified by interval
(precision).
    # precision: '0s'
    # # Time telegraf will wait between writing outputs (flush_interval).
Max=collectionInterval
    # flushInterval: '60s'
    # # Each output waits a random amount of time between the scheduled
write time and that time + flush_jitter before writing outputs
(flush_jitter).
    # flushJitter: '0s'
    # # Timeout for writing to outputs (timeout).
    # outputTimeout: '5s'

    # # telegraf-ds CPU/Mem limits and requests.
    # # See https://kubernetes.io/docs/concepts/configuration/manage-
resources-containers/
    # dsCpuLimit: '750m'
    # dsMemLimit: '800Mi'
    # dsCpuRequest: '100m'
    # dsMemRequest: '500Mi'

    # # telegraf-rs CPU/Mem limits and requests.
    # rsCpuLimit: '3'
    # rsMemLimit: '4Gi'
    # rsCpuRequest: '100m'
    # rsMemRequest: '500Mi'

    # # Skip second run of processors after aggregators
    # skipProcessorsAfterAggregators: 'true'

    # # 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: ''


    # If telegraf warns of insufficient lockable memory, try increasing
the limit of lockable memory for Telegraf in the underlying operating
system/node.  If increasing the limit is not an option, set this to true
to instruct Telegraf to not attempt to reserve locked memory pages.  While
this might pose a security risk as decrypted secrets might be swapped out
to disk, it allows for execution in environments where reserving locked
memory is not possible.
    # unprotected: 'false'

    # # Run the telegraf DaemonSet's telegraf-mountstats-poller container
in privileged mode.  Set runPrivileged to true if SELinux is enabled on
your Kubernetes nodes.
    # runPrivileged: '{{
.Values.telegraf_installer.kubernetes.privileged_mode }}'

    # # Set runDsPrivileged to true to run the telegraf DaemonSet's
telegraf container in privileged mode
    # runDsPrivileged: '{{
.Values.telegraf_installer.kubernetes.ds.privileged_mode }}'

    # # Collect container Block IO metrics.
    # dsBlockIOEnabled: 'true'

    # # Collect NFS IO metrics.
    # dsNfsIOEnabled: 'true'

    # # 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: 'false'

    # # Collect kubernetes.pod_volume (pod ephemeral storage) metrics.
Set this to true if you want to collect these metrics.
    # podVolumeMetricCollectionEnabled: 'false'

    # # Declare Rancher cluster as managed.  Set this to true if your
Rancher cluster is managed as opposed to on-premise.
    # isManagedRancher: 'false'

    # # If telegraf-rs fails to start due to being unable to find the etcd
crt and key, manually specify the appropriate path here.
```

```
    # rsHostEtcdCrt: ''
    # rsHostEtcdKey: ''

  # kube-state-metrics:
    # # kube-state-metrics CPU/Mem limits and requests.
    # cpuLimit: '500m'
    # memLimit: '1Gi'
    # cpuRequest: '100m'
    # memRequest: '500Mi'

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

    # # Comma-separated list of metrics to enable.
    # # See metric-allowlist in https://github.com/kubernetes/kube-state-
metrics/blob/main/docs/cli-arguments.md
    # metrics:
'kube_cronjob_created,kube_cronjob_status_active,kube_cronjob_labels,kube_
daemonset_created,kube_daemonset_status_current_number_scheduled,kube_daem
onset_status_desired_number_scheduled,kube_daemonset_status_number_availab
le,kube_daemonset_status_number_misscheduled,kube_daemonset_status_number_
ready,kube_daemonset_status_number_unavailable,kube_daemonset_status_obser
ved_generation,kube_daemonset_status_updated_number_scheduled,kube_daemons
et_metadata_generation,kube_daemonset_labels,kube_deployment_status_replic
as,kube_deployment_status_replicas_available,kube_deployment_status_replic
as_unavailable,kube_deployment_status_replicas_updated,kube_deployment_sta
tus_observed_generation,kube_deployment_spec_replicas,kube_deployment_spec
_paused,kube_deployment_spec_strategy_rollingupdate_max_unavailable,kube_d
eployment_spec_strategy_rollingupdate_max_surge,kube_deployment_metadata_g
eneration,kube_deployment_labels,kube_deployment_created,kube_job_created,
kube_job_owner,kube_job_status_active,kube_job_status_succeeded,kube_job_s
tatus_failed,kube_job_labels,kube_job_status_start_time,kube_job_status_co
mpletion_time,kube_namespace_created,kube_namespace_labels,kube_namespace_
status_phase,kube_node_info,kube_node_labels,kube_node_role,kube_node_spec
_unschedulable,kube_node_created,kube_persistentvolume_capacity_bytes,kube
_persistentvolume_status_phase,kube_persistentvolume_labels,kube_persisten
tvolume_info,kube_persistentvolume_claim_ref,kube_persistentvolumeclaim_ac
cess_mode,kube_persistentvolumeclaim_info,kube_persistentvolumeclaim_label
s,kube_persistentvolumeclaim_resource_requests_storage_bytes,kube_persiste
ntvolumeclaim_status_phase,kube_pod_info,kube_pod_start_time,kube_pod_comp
letion_time,kube_pod_owner,kube_pod_labels,kube_pod_status_phase,kube_pod_
```

```
status_ready,kube_pod_status_scheduled,kube_pod_container_info,kube_pod_co
ntainer_status_waiting,kube_pod_container_status_waiting_reason,kube_pod_c
ontainer_status_running,kube_pod_container_state_started,kube_pod_containe
r_status_terminated,kube_pod_container_status_terminated_reason,kube_pod_c
ontainer_status_last_terminated_reason,kube_pod_container_status_ready,kub
e_pod_container_status_restarts_total,kube_pod_overhead_cpu_cores,kube_pod
_overhead_memory_bytes,kube_pod_created,kube_pod_deletion_timestamp,kube_p
od_init_container_info,kube_pod_init_container_status_waiting,kube_pod_ini
t_container_status_waiting_reason,kube_pod_init_container_status_running,k
ube_pod_init_container_status_terminated,kube_pod_init_container_status_te
rminated_reason,kube_pod_init_container_status_last_terminated_reason,kube
_pod_init_container_status_ready,kube_pod_init_container_status_restarts_t
otal,kube_pod_status_scheduled_time,kube_pod_status_unschedulable,kube_pod
_spec_volumes_persistentvolumeclaims_readonly,kube_pod_container_resource_
requests_cpu_cores,kube_pod_container_resource_requests_memory_bytes,kube_
pod_container_resource_requests_storage_bytes,kube_pod_container_resource_
requests_ephemeral_storage_bytes,kube_pod_container_resource_limits_cpu_co
res,kube_pod_container_resource_limits_memory_bytes,kube_pod_container_res
ource_limits_storage_bytes,kube_pod_container_resource_limits_ephemeral_st
orage_bytes,kube_pod_init_container_resource_limits_cpu_cores,kube_pod_ini
t_container_resource_limits_memory_bytes,kube_pod_init_container_resource_
limits_storage_bytes,kube_pod_init_container_resource_limits_ephemeral_sto
rage_bytes,kube_pod_init_container_resource_requests_cpu_cores,kube_pod_in
it_container_resource_requests_memory_bytes,kube_pod_init_container_resour
ce_requests_storage_bytes,kube_pod_init_container_resource_requests_epheme
ral_storage_bytes,kube_replicaset_status_replicas,kube_replicaset_status_r
eady_replicas,kube_replicaset_status_observed_generation,kube_replicaset_s
pec_replicas,kube_replicaset_metadata_generation,kube_replicaset_labels,ku
be_replicaset_created,kube_replicaset_owner,kube_resourcequota,kube_resour
cequota_created,kube_service_info,kube_service_labels,kube_service_created
,kube_service_spec_type,kube_statefulset_status_replicas,kube_statefulset_
status_replicas_current,kube_statefulset_status_replicas_ready,kube_statef
ulset_status_replicas_updated,kube_statefulset_status_observed_generation,
kube_statefulset_replicas,kube_statefulset_metadata_generation,kube_statef
ulset_created,kube_statefulset_labels,kube_statefulset_status_current_revi
sion,kube_statefulset_status_update_revision,kube_node_status_capacity,kub
e_node_status_allocatable,kube_node_status_condition,kube_pod_container_re
source_requests,kube_pod_container_resource_limits,kube_pod_init_container
_resource_limits,kube_pod_init_container_resource_requests'

    # # 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=[*],namesp
```

```
aces=[*],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: ''

    # # kube-state-metrics shards.  Increase the number of shards for
larger clusters if telegraf RS pod(s) experience collection timeouts
    # shards: '2'

  # # Settings for the Events Log feature.
  # logs:
    # # Set runPrivileged to true if Fluent Bit fails to start, trying to
open/create its database.
    # runPrivileged: 'false'

    # # 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"

    # # DNS resolver that Fluent Bit should use: "LEGACY" or "ASYNC"
    # fluentBitDNSResolver: "LEGACY"

    # # 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: ''

    # # event-exporter CPU/Mem limits and requests.
    # # See https://kubernetes.io/docs/concepts/configuration/manage-
resources-containers/
    # event-exporter-cpuLimit: '500m'
    # event-exporter-memLimit: '1Gi'
```

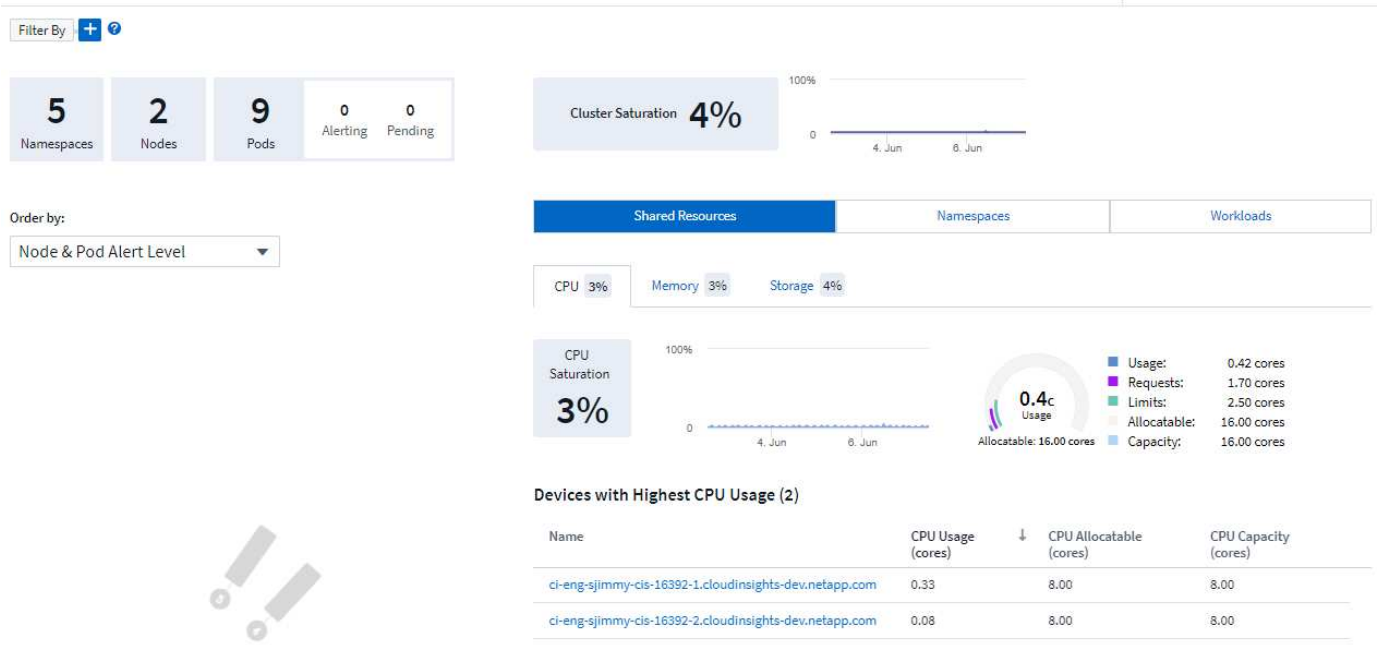
```
    # event-exporter-cpuRequest: '50m'
    # event-exporter-memRequest: '100Mi'

    # # event-exporter max event age.
    # # See https://github.com/jkroepke/resmoio-kubernetes-event-exporter
    # event-exporter-maxEventAgeSeconds: '10'

    # # event-exporter client-side throttling
    # # Set kubeBurst to roughly match your events per minute and
kubeQPS=kubeBurst/5
    # # See https://github.com/resmoio/kubernetes-event-
exporter#troubleshoot-events-discarded-warning
    # event-exporter-kubeQPS: 20
    # event-exporter-kubeBurst: 100

    # # fluent-bit CPU/Mem limits and requests.
    # # See https://kubernetes.io/docs/concepts/configuration/manage-
resources-containers/
    # fluent-bit-cpuLimit: '500m'
    # fluent-bit-memLimit: '1Gi'
    # fluent-bit-cpuRequest: '50m'
    # fluent-bit-memRequest: '100Mi'

  # # Settings for the Network Performance and Map feature.
  # workload-map:
    # # netapp-ci-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.
    # enableDNSLookup: 'true'

    # # netapp-ci-net-observer-l4-ds additional tolerations. Use the
following abbreviated single line format only.
    # # Inspect netapp-ci-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: '1'
    # memLimit: '1Gi'
    # cpuRequest: '500m'
    # 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",
"api_key", "jwt", "accesskey", "access_key", "access-key", "ca-file",
"key-file", "cert", "cafile", "keyfile", "tls", "crt", "salt",
".dockerconfigjson", "auth", "secret"'

    # # 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 additional field paths whose diff is
ignored as part of change analytics. This list in addition to the default
set of field paths ignored by the collector.
    # # Example: '"metadata.specTime", "data.status"'
```

```
    # additionalFieldsDiffToIgnore: ''

    # # 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: ''

    # # 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: ''
```

Detailseite Zu Kubernetes Cluster

Auf der Kubernetes-Cluster-Detailseite wird eine detaillierte Übersicht über das Kubernetes-Cluster angezeigt.



Namespace, Node und Pod-Anzahl

Die Zählungen oben auf der Seite zeigen Ihnen die Gesamtzahl der Namespaces, Nodes und Pods im Cluster sowie die Anzahl der Pops, die derzeit Warnungen und ausstehend sind.

Shared Ressourcen und Sättigung

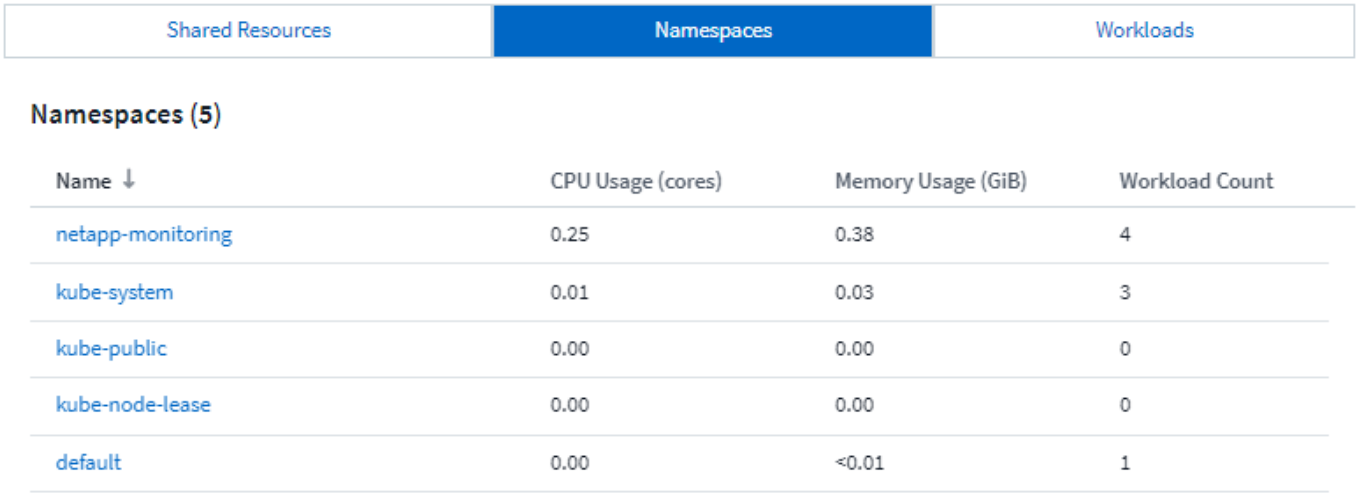
Oben rechts auf der Detailseite ist Ihre Cluster-Sättigung als aktueller Prozentsatz sowie ein Diagramm, das den letzten Trend im Laufe der Zeit zeigt. Cluster-Sättigung ist der höchste CPU-, Arbeitsspeicher- oder Storage-Sättigung bei jedem Zeitpunkt.

Im Folgenden wird die Seite standardmäßig **Nutzung von freigegebenen Ressourcen** mit Registerkarten für CPU, Speicher und Speicher angezeigt. Auf jeder Registerkarte werden der Sättigungspunkt und der Trend über die Zeit mit zusätzlichen Nutzungsdetails angezeigt. Für den Storage ist der angezeigte Wert der größere Backend- und Filesystem-Sättigung, die unabhängig voneinander berechnet wird.

Die Geräte mit der höchsten Nutzung werden in einer Tabelle unten angezeigt. Klicken Sie auf einen beliebigen Link, um diese Geräte zu durchsuchen.

Namespaces

Auf der Registerkarte Namespaces wird eine Liste aller Namespaces in der Kubernetes-Umgebung angezeigt. Die CPU- und Arbeitsspeicherauslastung sowie die Anzahl der Workloads in jedem Namespace werden angezeigt. Klicken Sie auf die Namenslinks, um die einzelnen Namespaces zu erkunden.



Workloads

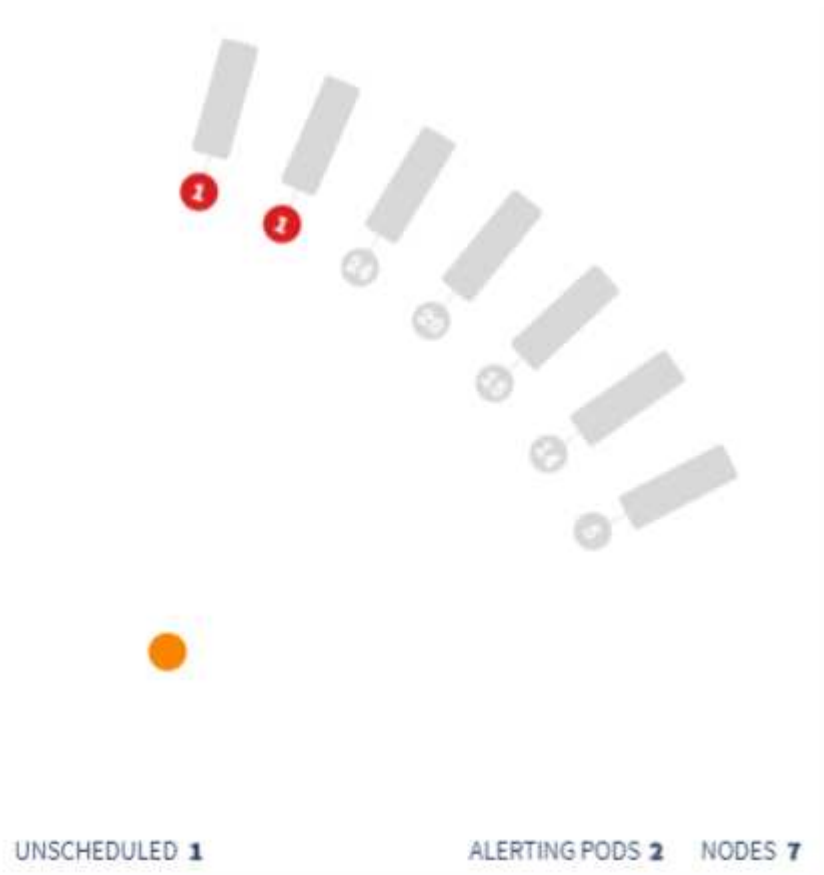
Auf der Registerkarte Workloads wird zudem eine Liste der Workloads in den einzelnen Namespace angezeigt. Auch hier wird die CPU- und Arbeitsspeicherauslastung angezeigt. Wenn Sie auf den Namespace-Links klicken, ist jeder Link bohrt.

Shared Resources | Namespaces | Workloads

Workloads (8)

Name ↓	CPU Usage (cores)	Memory Usage (GiB)	Namespace
telegraf-rs-lf9gg	0.24	0.24	netapp-monitoring
telegraf-ds-k957c	0.01	0.10	netapp-monitoring
nginx	0.00	<0.01	default
monitoring-operator-6fcf4755ff-p2cs6	<0.01	0.02	netapp-monitoring
metrics-server-7b4f8b595-f7j9f	<0.01	0.01	kube-system
local-path-provisioner-64d457c485-289gx	<0.01	0.01	kube-system
kube-state-metrics-7995866f8c-t8c49	<0.01	0.01	netapp-monitoring
coredns-5d69dc75db-nkw5p	<0.01	0.01	kube-system

Das Cluster „Wheel“

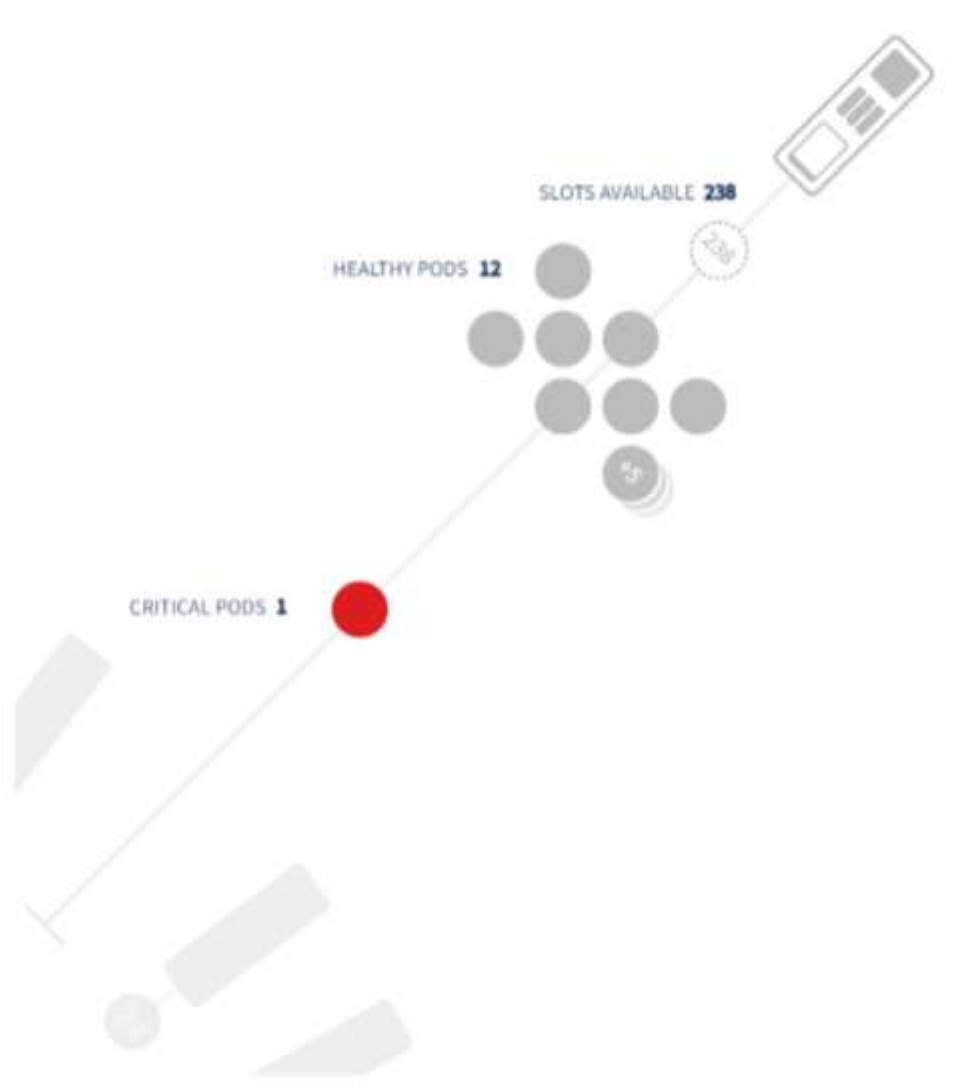


Im Abschnitt „Cluster „Wheel“ finden Sie auf einen Blick den Zustand der Nodes und des POD. Weitere Informationen hierzu finden Sie unter. Wenn Ihr Cluster mehr Nodes enthält, als in diesem Bereich der Seite angezeigt werden kann, können Sie das Rad mit den verfügbaren Schaltflächen drehen.

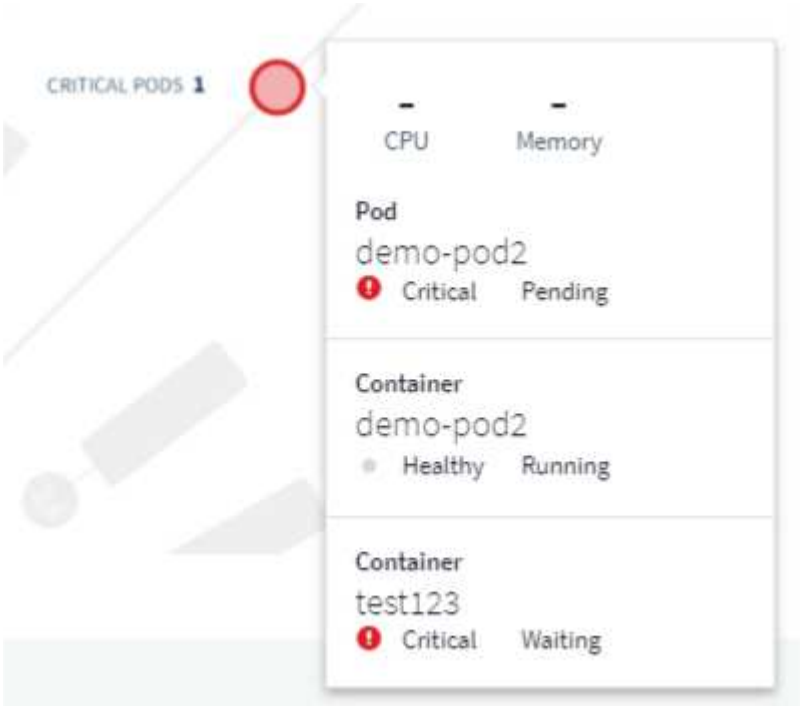
AlarmPods oder Nodes werden rot angezeigt. Die Bereiche „Warnung“ werden orange angezeigt. PODs, die nicht geplant sind (d.h. unangebracht), werden in der unteren Ecke des Cluster „Wheel“ angezeigt.

Wenn Sie sich über einen Pod (Kreis) oder Knoten (Balken) bewegen, wird die Ansicht des Knotens erweitert.

Wenn Sie in der Ansicht auf den Pod oder Node klicken, wird die Ansicht „erweiterter Node“ vergrößert.



Von hier aus können Sie mit dem Mauszeiger auf ein Element zeigen, um Details zu diesem Element anzuzeigen. Beispiel: Wenn Sie den Mauszeiger über den kritischen POD in diesem Beispiel halten, werden Details zu diesem POD angezeigt.



Sie können Filesystem-, Speicher- und CPU-Informationen anzeigen, indem Sie den Mauszeiger über die Knoten-Elemente bewegen.

Ein Hinweis zu den Messgeräten

Die Speicher- und CPU-Anzeigen zeigen drei Farben, da sie *used* in Bezug auf *zuteilbare Kapazität* und *Gesamtkapazität* zeigen.

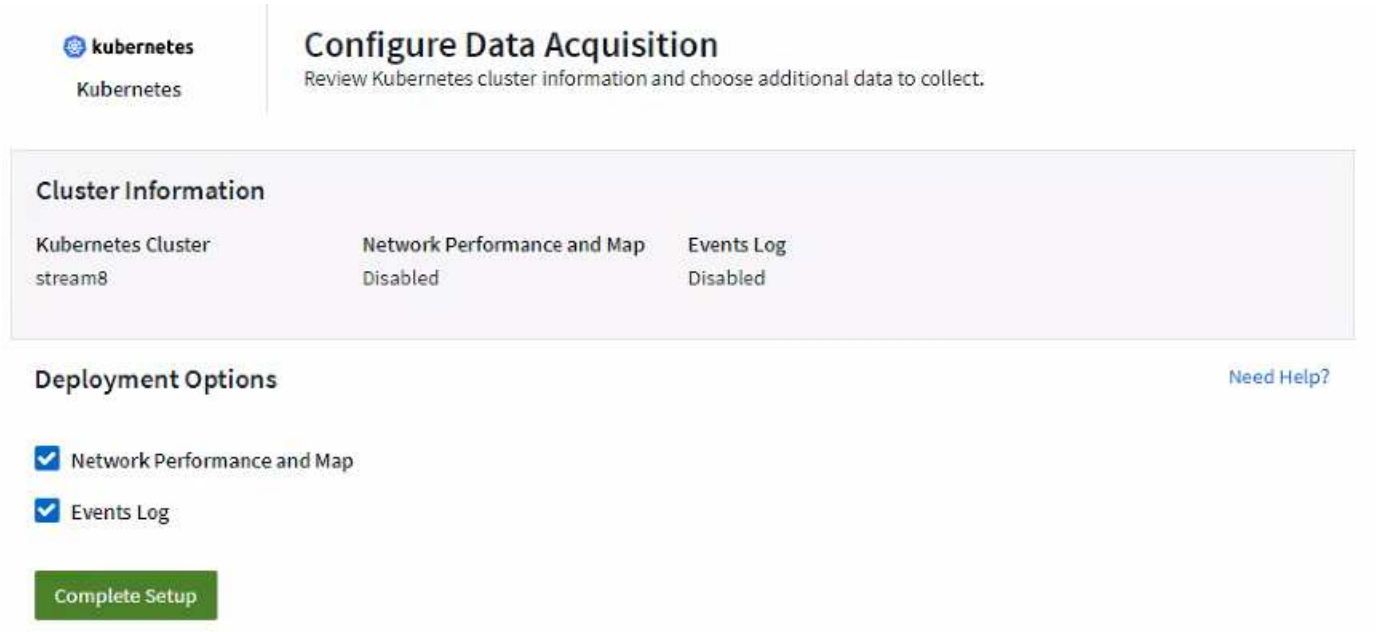
Performance-Monitoring und -Zuordnung des Kubernetes-Netzwerks

Die Kubernetes Network Performance Monitoring and Map Funktion vereinfacht die Fehlerbehebung durch die Zuordnung von Abhängigkeiten zwischen Services (auch Workloads genannt). Sie bietet Echtzeiteinblick in Latenzen und Anomalien bei der Netzwerk-Performance. So können Performance-Probleme erkannt werden, bevor sie sich auf die Benutzer auswirken. Diese Funktion hilft Unternehmen, durch Analyse und Prüfung des Kubernetes-Traffic-Flows die Gesamtkosten zu senken.

Die wichtigsten Funktionen • die Workload-Map präsentiert Kubernetes-Workload-Abhängigkeiten und -Abläufe und hebt Netzwerk- und Performance-Probleme hervor. • Monitoring des Netzwerkverkehrs zwischen Kubernetes-Pods, Workloads und Nodes; Ermittlung der Quelle von Traffic- und Latenzproblemen • Senkung der Gesamtkosten durch Analyse des Ingress-, Egress-, Regions- und zonenübergreifenden Netzwerk-Traffics.

Voraussetzungen

Bevor Sie die Kubernetes-Netzwerk-Performance-Überwachung und -Zuordnung verwenden können, müssen Sie das für die Aktivierung dieser Option konfiguriert haben"NetApp Kubernetes Monitoring Operator". Aktivieren Sie während der Bereitstellung des Operators das Kontrollkästchen „Netzwerkleistung und Zuordnung“, um es zu aktivieren. Sie können diese Option auch aktivieren, indem Sie zu einer Kubernetes-Landing Page navigieren und „Implementierung ändern“ auswählen.



Monitore

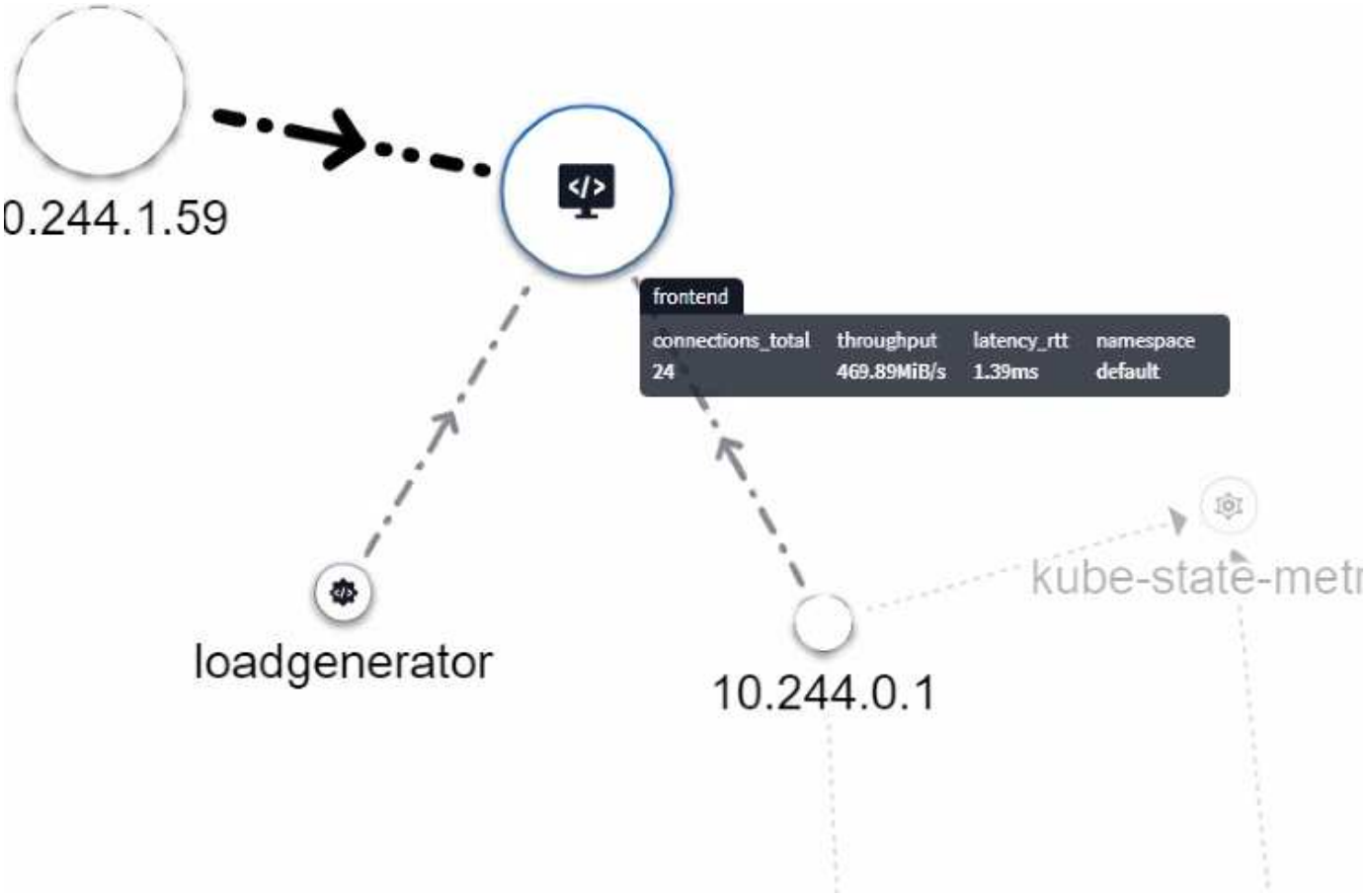
Die Workload Map verwendet"Monitore", um Informationen abzuleiten. Data Infrastructure Insights bietet eine Reihe von Kubernetes-Standardmonitoren (beachten Sie, dass diese standardmäßig „*Paused*“ sein können. Sie können die gewünschten Monitore *Resume* (d. h. aktivieren) oder benutzerdefinierte Monitore für kubernetes-Objekte erstellen, die auch von der Workload Map verwendet werden.

Sie können für jeden der unten aufgeführten Objekttypen metrische Warnmeldungen zu Data Infrastructure Insights erstellen. Stellen Sie sicher, dass die Daten nach dem Standardobjekttyp gruppiert sind.

- kubernetes.Workload
- kubernetes.demonset
- kubernetes.deployment
- kubernetes.cronjob
- kubernetes.Job
- kubernetes.Replicaset
- kubernetes.statefulset
- kubernetes.POD
- kubernetes.network_traffic_l4

Die Karte

Die Karte zeigt Services/Workloads und deren Beziehungen zueinander an. Pfeile zeigen die Verkehrsrichtung. Wenn Sie den Mauszeiger über einen Workload halten, werden zusammenfassende Informationen zu diesem Workload angezeigt, wie im folgenden Beispiel zu sehen ist:



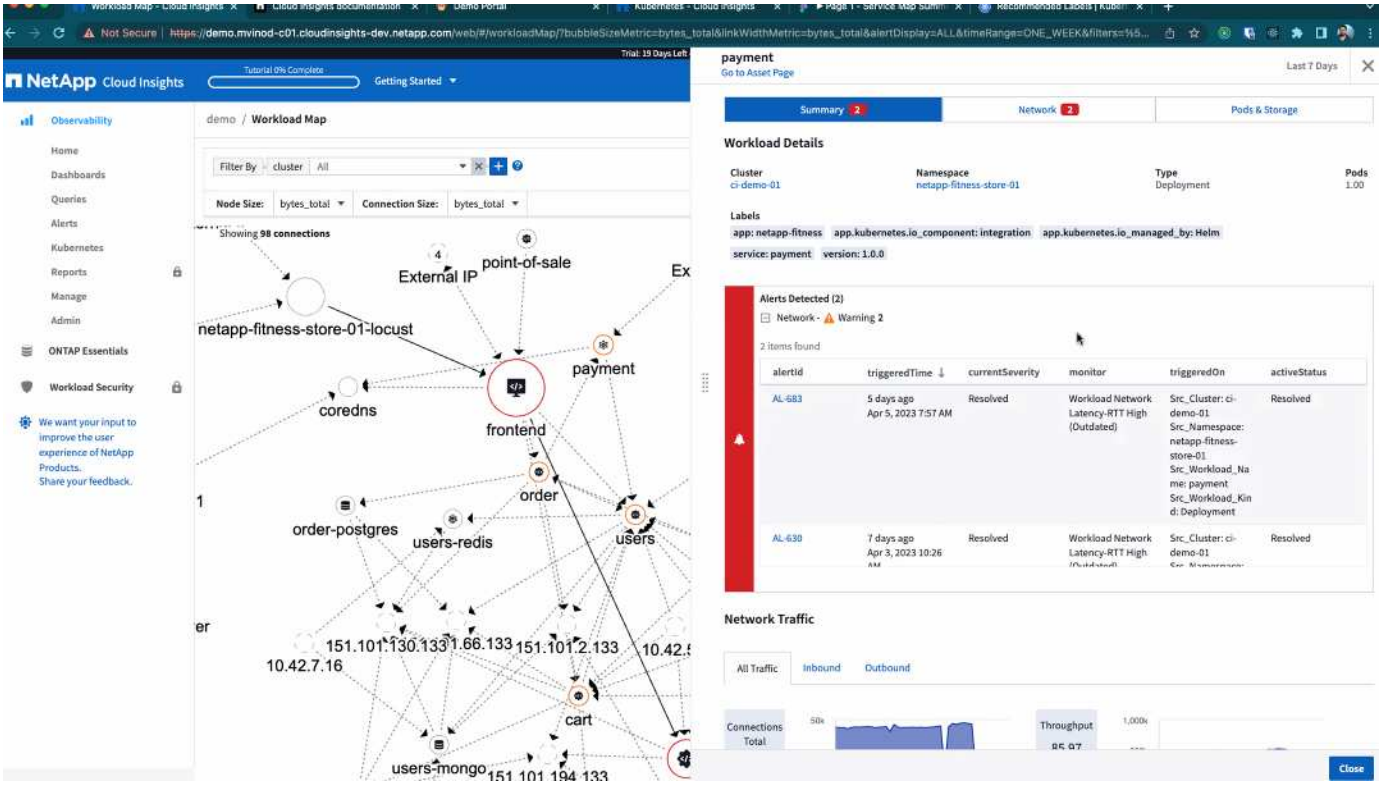
Symbole innerhalb der Kreise stellen verschiedene Diensttypen dar. Beachten Sie, dass Symbole nur sichtbar sind, wenn die zugrunde liegenden Objekte haben Etiketten.

Die Größe jedes Kreises gibt die Knotengröße an. Beachten Sie, dass diese Größen relativ sind. Ihr Browser-Zoom-Level oder die Bildschirmgröße kann sich auf die tatsächlichen Kreisgrößen auswirken. Auf die gleiche Weise gibt Ihnen der Linienstil einen schnellen Überblick über die Verbindungsgröße; fett leuchtete Linien sind stark frequentlicht, während die gestrichelten Linien weniger Verkehr aufweisen.

Zahlen innerhalb der Kreise sind die Anzahl der externen Verbindungen, die derzeit vom Dienst verarbeitet werden.

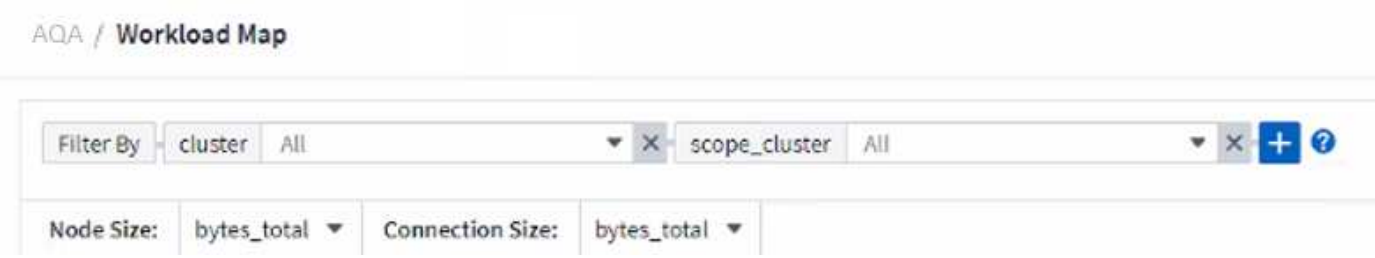
Workload-Details und -Alarme

Farbige Kreise weisen auf eine Warnung auf Warn- oder kritische Ebene für die Arbeitslast hin. Bewegen Sie den Mauszeiger über den Kreis, um eine Zusammenfassung des Problems zu erhalten, oder klicken Sie auf den Kreis, um ein Slideout-Fenster mit mehr Details zu öffnen.

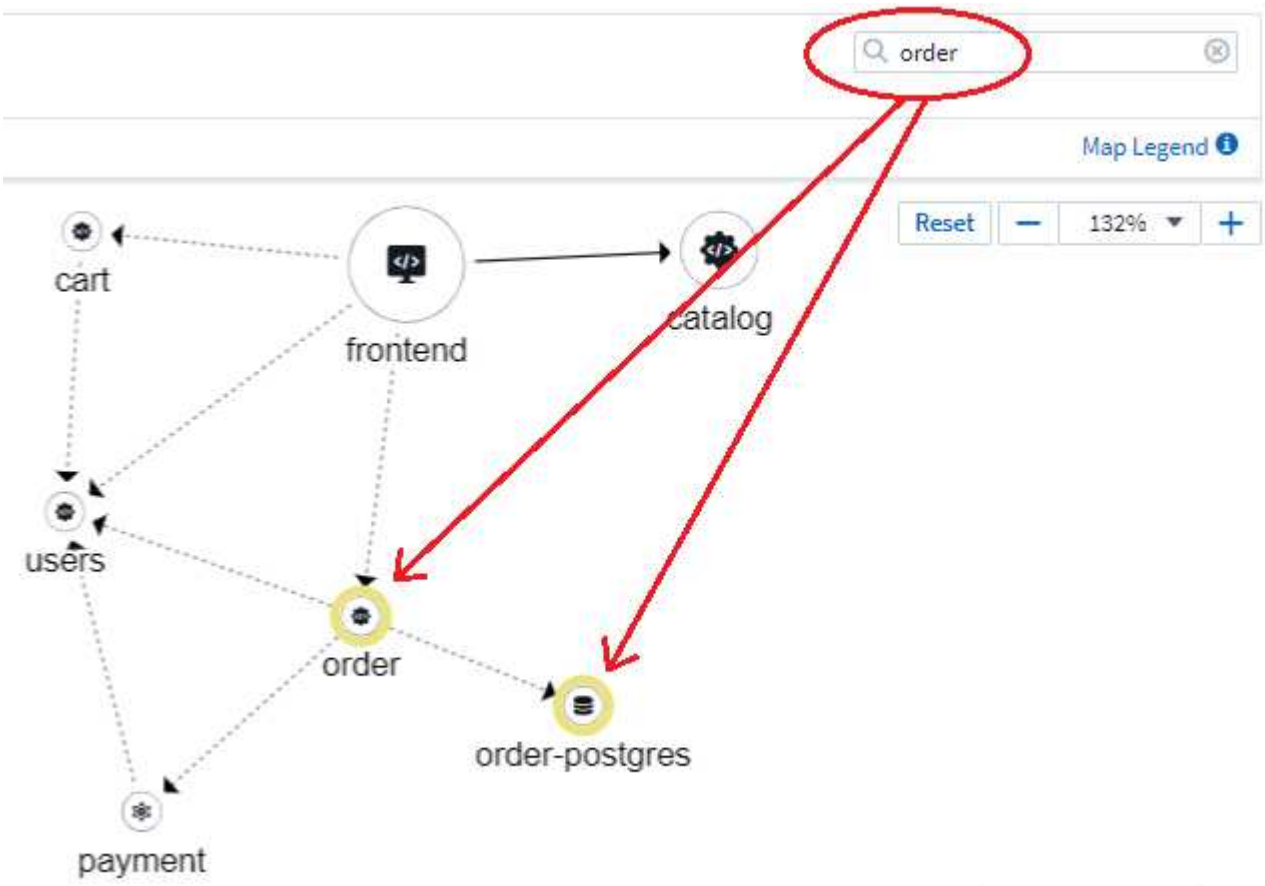


Suchen und Filtern

Wie andere Funktionen von Data Infrastructure Insights können Sie auch hier ganz einfach Filter festlegen, die sich ganz auf die gewünschten Objekte oder Workload-Attribute konzentrieren.



Ebenso wird durch Eingabe einer Zeichenfolge im Feld *Find* die übereinstimmenden Workloads hervorgehoben.



Workload-Etiketten

Workload-Bezeichnungen sind erforderlich, wenn die Zuordnung die angezeigten Workload-Typen (d. h. die Kreissymbole) identifizieren soll. Die Bezeichnungen werden wie folgt abgeleitet:

- Name des Dienstes/der Anwendung, der allgemein ausgeführt wird
- Wenn es sich bei der Quelle um einen Pod handelt:
  - Die Bezeichnung leitet sich vom Workload-Etikett des Pods ab
  - Erwartetes Label für den Workload: App.kubernetes.io/component
  - Bezeichnung Name Referenz: https://kubernetes.io/docs/concepts/overview/working-with-objects/common-labels/
  - Empfohlene Etiketten:
    - Frontend

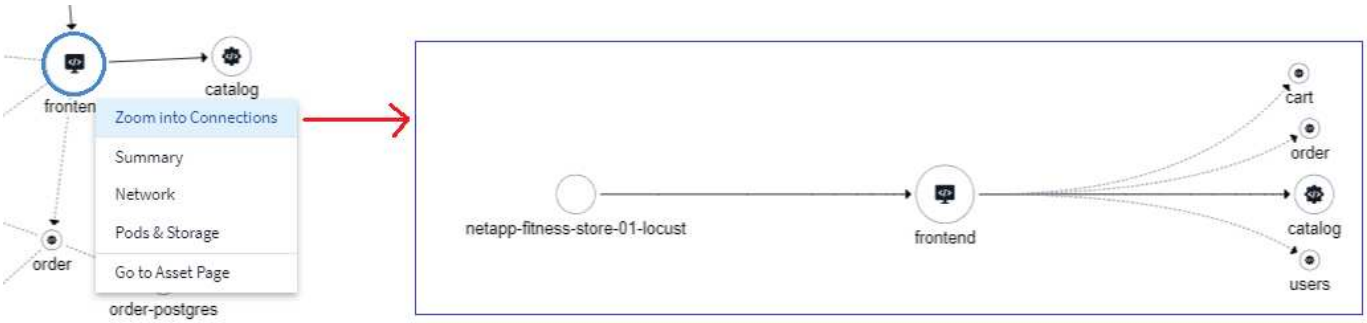
- Back-End
- Datenbank
- Cache
- Warteschlange
- kafka

- Wenn sich die Quelle außerhalb des kubernetes-Clusters befindet:
  - Data Infrastructure Insights versucht, den DNS-aufgelösten Namen zu analysieren, um den Diensttyp zu extrahieren.

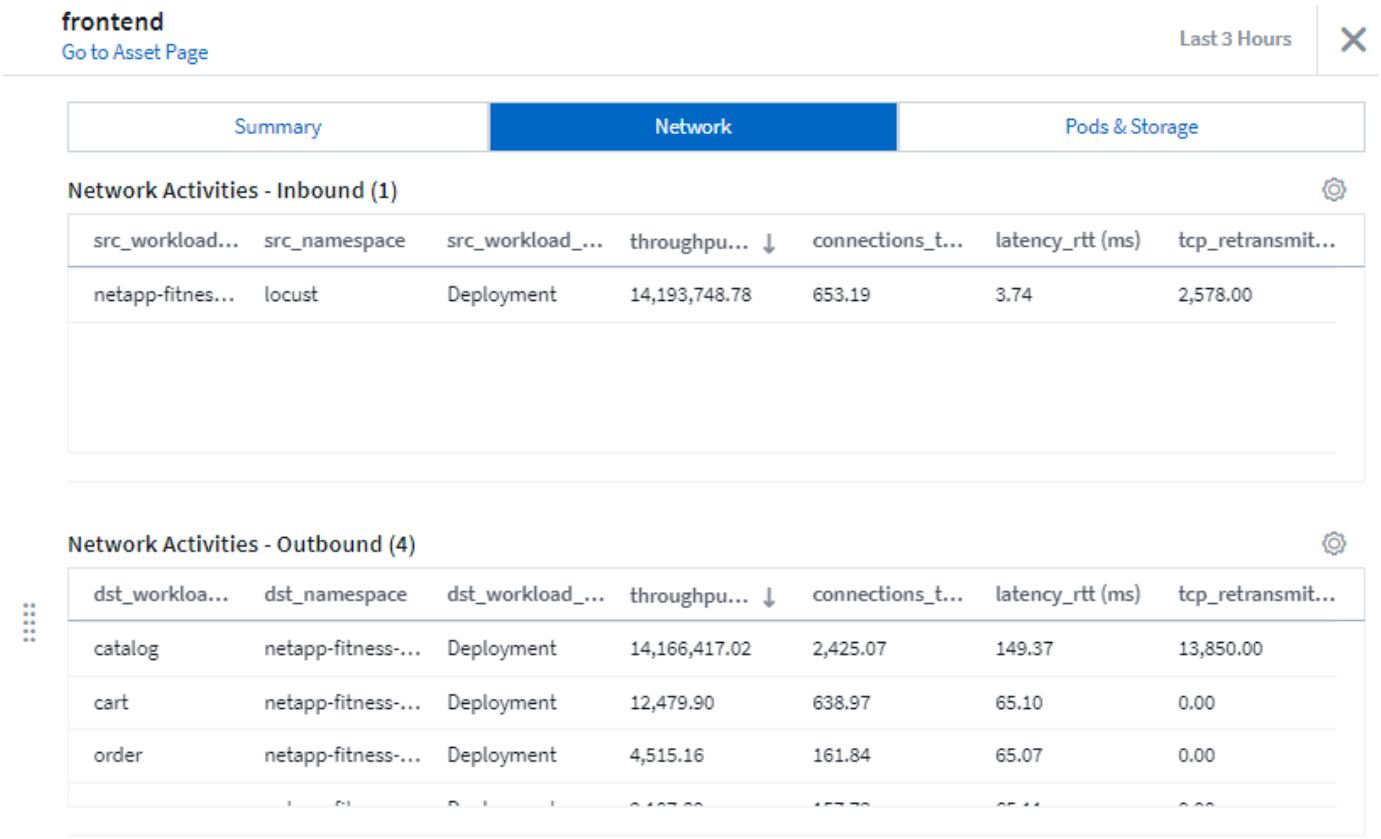
    Beispiel: Mit einem DNS-aufgelösten Namen von *s3.eu-north-1.amazonaws.com* wird der aufgelöste Name analysiert, um *s3* als Diensttyp zu erhalten.

So Geht Es Richtig

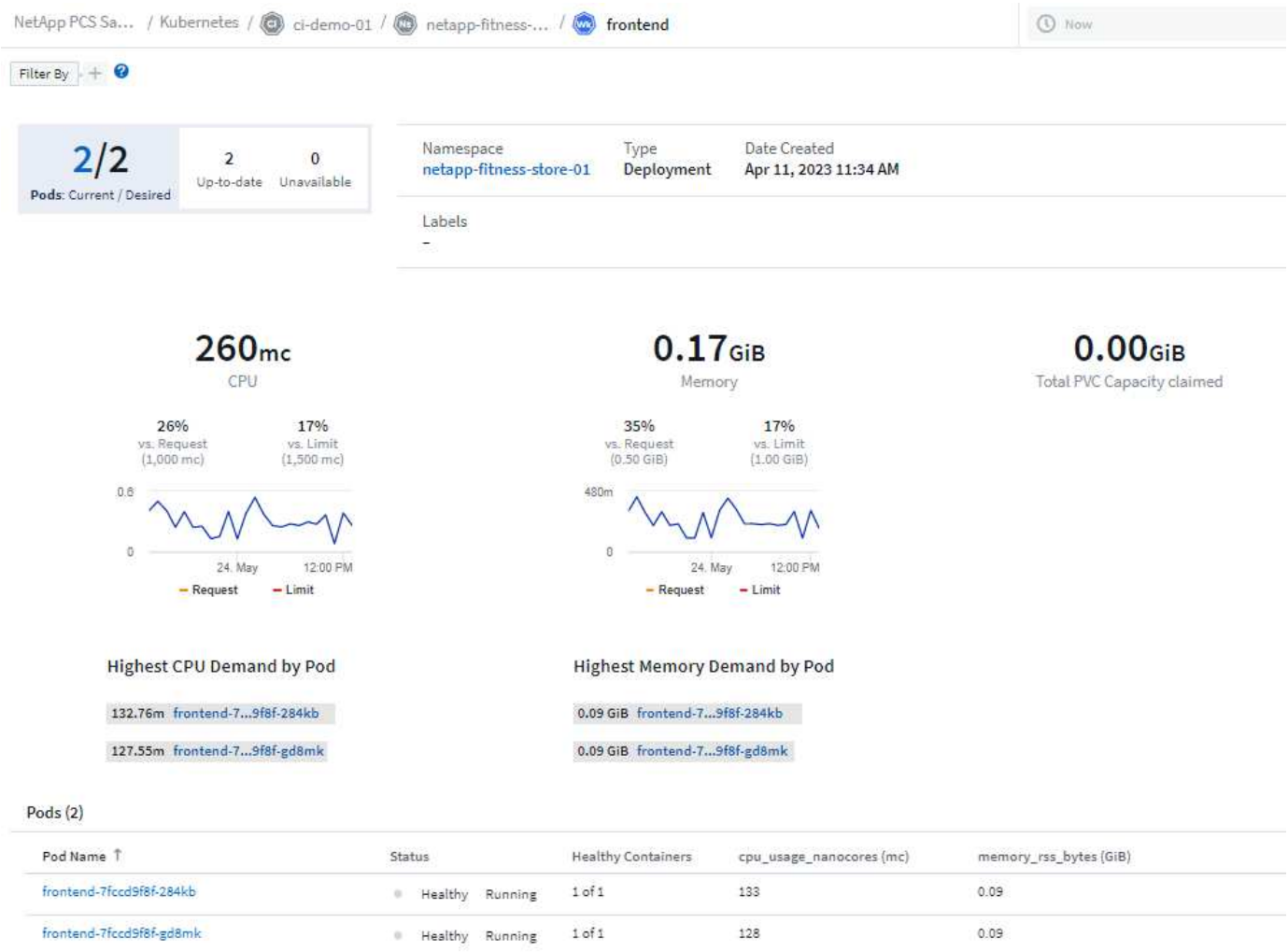
Mit einem Rechtsklick auf einen Workload erhalten Sie zusätzliche Optionen, um weitere Informationen zu erhalten. Von hier aus können Sie beispielsweise die Ansicht vergrößern, um die Verbindungen für diesen Workload anzuzeigen.



Alternativ können Sie das Detailslideout-Panel öffnen, um die Registerkarte *Summary*, *Network* oder *Pod & Storage* direkt anzuzeigen.



Durch Auswahl von *Gehe zu Anlagenseite* wird die detaillierte Zielseite für die Anlage für den Workload geöffnet.



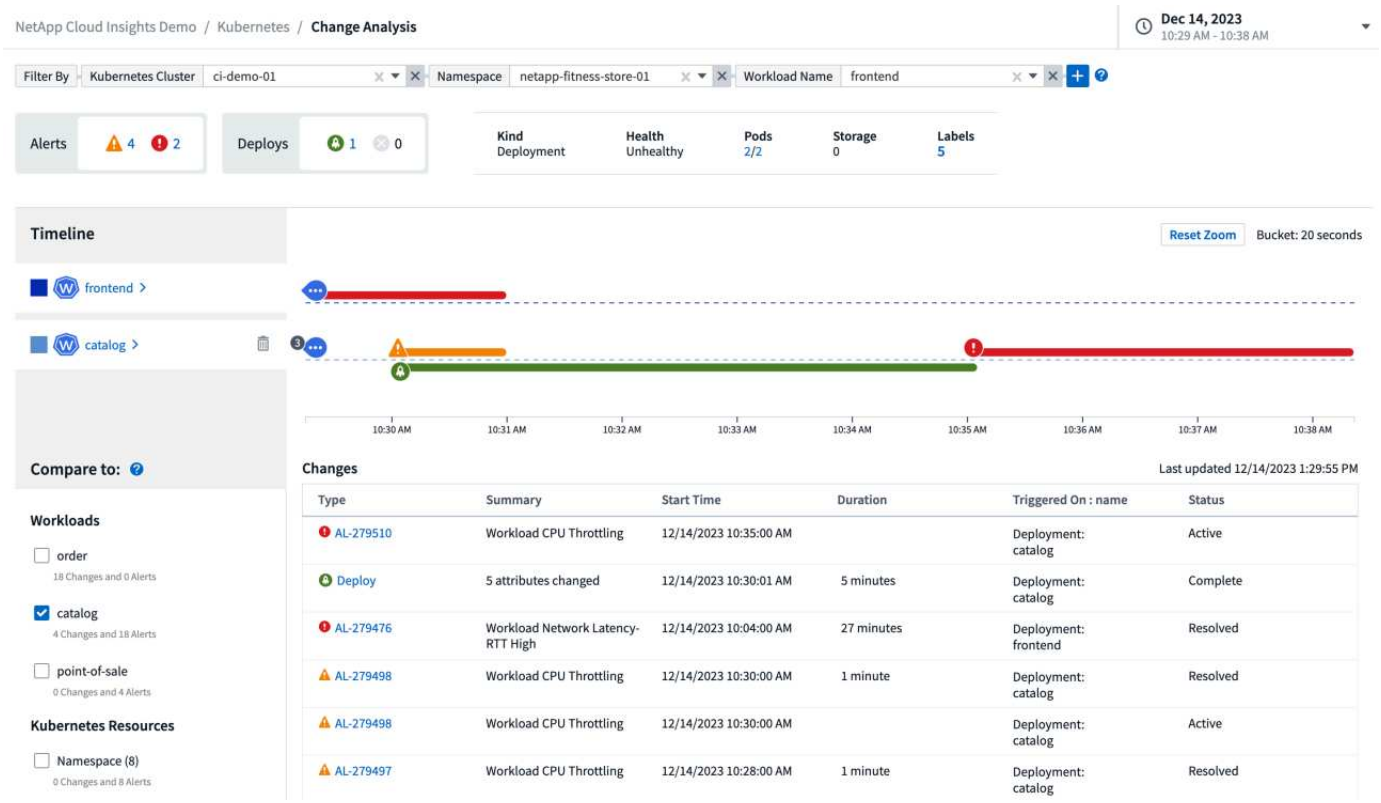
Kubernetes Change Analytics

Kubernetes Change Analytics bietet Ihnen einen All-in-One-Überblick über die letzten Änderungen an Ihrer K8s-Umgebung. Warnmeldungen und Bereitstellungsstatus stehen Ihnen jederzeit zur Verfügung. Mit Change Analytics lassen sich jede Implementierungs- und Konfigurationsänderung nachverfolgen und mit dem Zustand und der Performance von Kubernetes-Services, Infrastruktur und Clustern korrelieren.

Wie hilft die Änderungsanalyse?

- In mandantenfähigen Kubernetes-Umgebungen können Ausfälle aufgrund falsch konfigurierter Änderungen auftreten. Change Analytics unterstützt dies durch die Bereitstellung eines zentralen Fensters zur Ansicht und Korrelation des Systemzustands von Workloads und Konfigurationsänderungen. Dies kann bei der Fehlerbehebung in dynamischen Kubernetes-Umgebungen helfen.

Um Kubernetes Change Analytics anzuzeigen, navigieren Sie zu **Kubernetes > Change Analysis**.



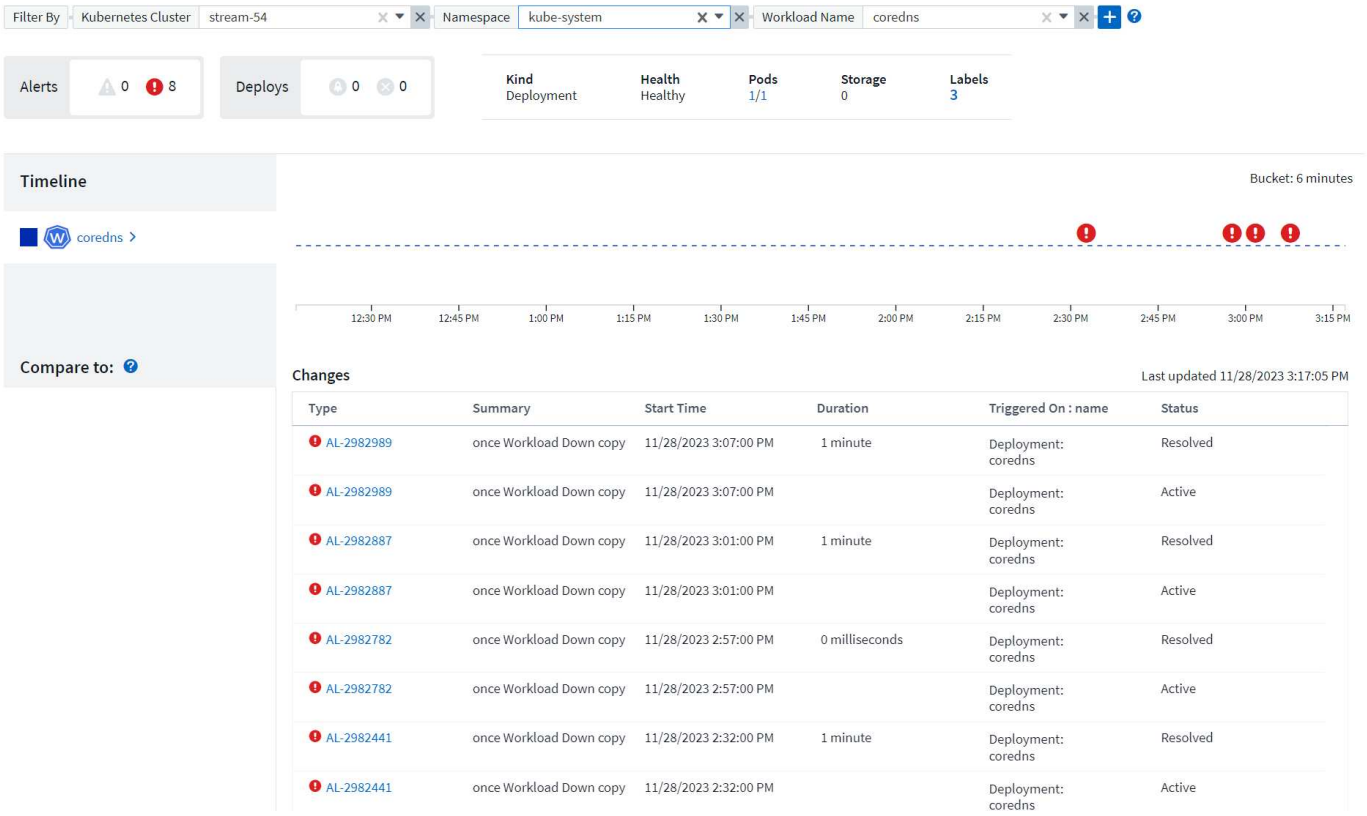
.

Die Seite wird automatisch aktualisiert, basierend auf dem aktuell ausgewählten Zeitbereich von Data Infrastructure Insights. Kleinere Zeitbereiche bedeuten eine häufigere Bildschirmerneuerung.

Filtern

Wie bei allen Funktionen von Data Infrastructure Insights ist auch das Filtern der Änderungsliste intuitiv: Ganz oben auf der Seite können Sie Werte für Ihren Kubernetes-Cluster, Namespace oder Workload eingeben oder auswählen oder mit der Schaltfläche {+] eigene Filter hinzufügen.

Wenn Sie nach unten zu einem bestimmten Cluster, Namespace und Workload filtern (zusammen mit allen anderen Filtern, die Sie festlegen), wird Ihnen ein Zeitplan für die Implementierungen und Warnungen für diesen Workload in diesem Namespace auf dem Cluster angezeigt. Vergrößern Sie die Ansicht weiter, indem Sie auf das Diagramm klicken und es ziehen, um einen bestimmten Zeitraum zu fokussieren.

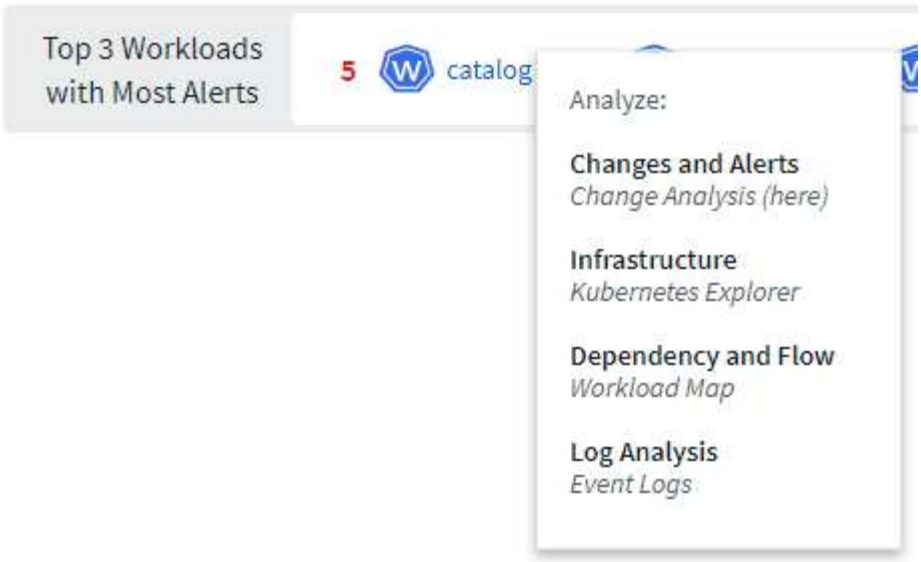


Schnellstatus

Unterhalb des Filterbereichs befinden sich eine Reihe von High-Level-Indikatoren. Auf der linken Seite ist die Anzahl der Warnungen (Warnung und kritisch). Diese Nummer enthält sowohl *Active* als auch *Resolved* Warnungen. Um nur *Active*-Warnungen anzuzeigen, setzen Sie einen Filter für „Status“ und wählen Sie „aktiv“.

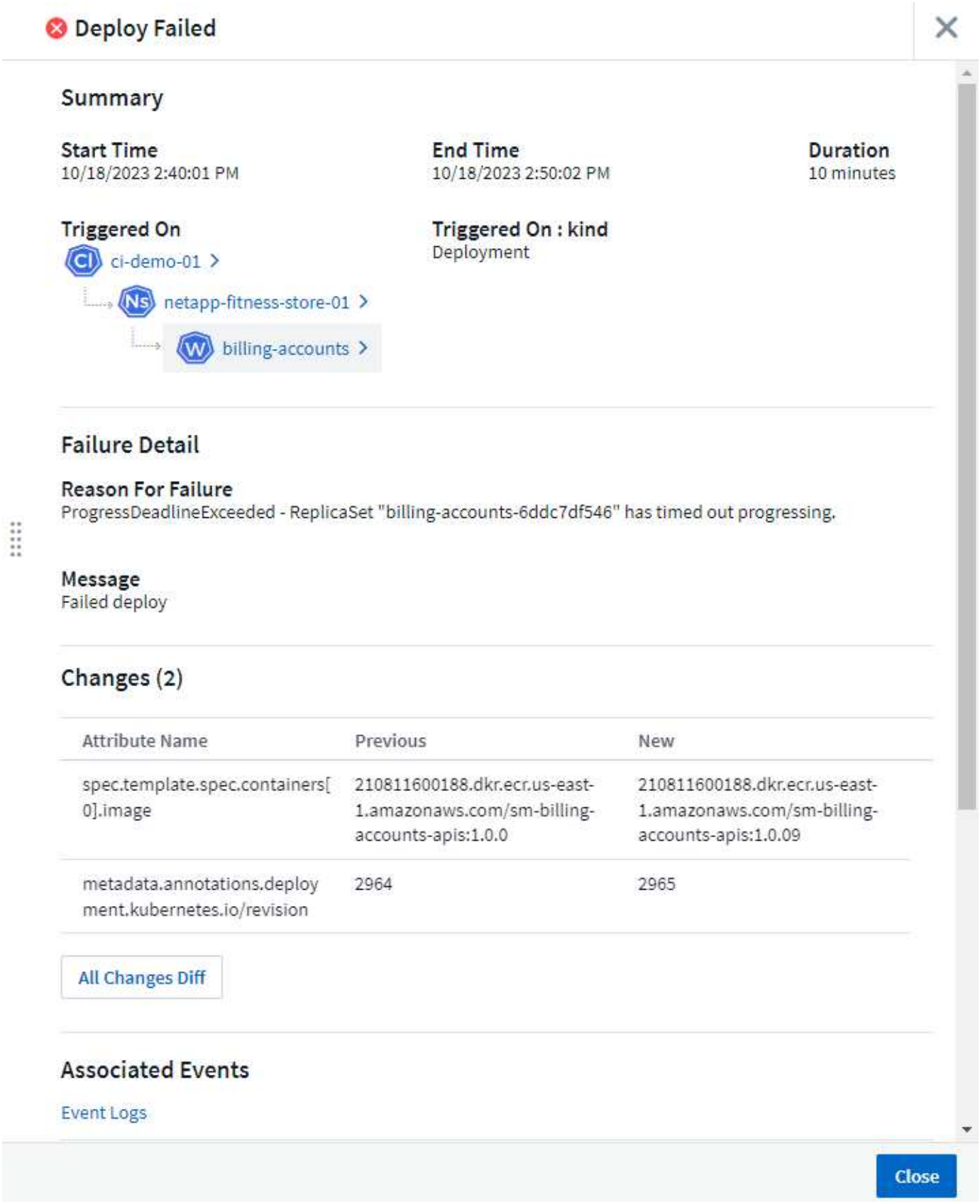
Hier wird auch der Bereitstellungsstatus angezeigt. Auch hier wird standardmäßig die Anzahl der Bereitstellungen *started*, *complete* und *failed* angezeigt. Um nur *failed*-Bereitstellungen anzuzeigen, setzen Sie einen Filter für „Status“ und wählen Sie „failed“ aus.

Als Nächstes kommen die 3 wichtigsten Workloads mit den meisten Warnmeldungen zum Einsatz. Die Zahl in rot neben jedem Workload gibt die Anzahl der Warnmeldungen in Bezug auf diesen Workload an. Klicken Sie auf den Workload-Link, um ihn in Ihre Infrastruktur (Kubernetes Explorer), Abhängigkeiten (Workload Map) oder Protokollanalyse (Event Logs) zu untersuchen.



Detailfenster

Durch Auswahl einer Änderung in der Liste wird ein Fenster geöffnet, in dem die Änderung näher beschrieben wird. Wenn Sie beispielsweise eine fehlgeschlagene Bereitstellung auswählen, wird eine Zusammenfassung der Bereitstellung mit Start- und Endzeiten, Dauer und dem Auslösungsort der Bereitstellung sowie Links zur Untersuchung dieser Ressourcen angezeigt. Außerdem werden der Grund für den Fehler, alle zugehörigen Änderungen und alle zugehörigen Ereignisse angezeigt.



Durch die Auswahl einer Warnmeldung erhalten Sie ebenfalls Details zur Warnmeldung, einschließlich des Monitors, der die Warnmeldung ausgelöst hat, sowie ein Diagramm mit einer visuellen Zeitleiste für die Warnmeldung.