Kubernetes

Data Infrastructure Insights

NetApp
February 03, 2026

Inhalt

Kubernetes

Übersicht über Kubernetes-Cluster

Der Data Infrastructure Insights Kubernetes Explorer ist ein leistungsstarkes Tool zur Anzeige des Gesamtzustands und der Nutzung Ihrer Kubernetes-Cluster und ermöglicht Ihnen eine einfache Detailansicht der Untersuchungsbereiche.

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

[Kubernetes-Listenseite]

Clusterliste

Die Clusterliste zeigt die folgenden Informationen für jeden Cluster auf Ihrem Mandanten an:

- Cluster **Name**. Durch Klicken auf einen Clusternamen wird das"**Detailseite**" für diesen Cluster.
- *Sättigungs*prozentsätze. Die Gesamtsättigung ist die höchste CPU-, Speicher- oder Speichersättigung.
- Anzahl der **Knoten** im Cluster. Durch Klicken auf diese Nummer wird die Seite mit der Knotenliste geöffnet.
- Anzahl der **Pods** im Cluster. Durch Klicken auf diese Nummer wird die Pod-Listenseite geöffnet.
- Anzahl der **Namespaces** im Cluster. Durch Klicken auf diese Nummer wird die Seite mit der Namespace-Liste geöffnet.
- Anzahl der **Workloads** im Cluster. Durch Klicken auf diese Nummer wird die Seite mit der Arbeitslastliste geöffnet.

Verfeinern des Filters

Wenn Sie beim Filtern mit der Eingabe beginnen, wird Ihnen die Option angezeigt, einen **Platzhalterfilter** basierend auf dem aktuellen Text zu erstellen. Wenn Sie diese Option auswählen, werden alle Ergebnisse zurückgegeben, die mit dem Platzhalterausdruck übereinstimmen. Sie können **Ausdrücke** auch mit NOT oder AND erstellen oder die Option „Keine“ auswählen, um nach Nullwerten im Feld zu filtern.

[Filtern mit Platzhaltern im K8S Explorer]

Filter, die auf Platzhaltern oder Ausdrücken basieren (z. B. NICHT, UND, „Keine“ usw.), werden im Filterfeld dunkelblau angezeigt. Elemente, die Sie direkt aus der Liste auswählen, werden hellblau angezeigt.

[Filter mit Platzhaltern und ausgewählten Elementen]

Kubernetes-Filter sind kontextbezogen. Das bedeutet beispielsweise, dass, wenn Sie sich auf einer bestimmten Knotenseite befinden, der Filter „pod_name“ nur Pods auflistet, die mit diesem Knoten in Zusammenhang stehen. Wenn Sie außerdem einen Filter für einen bestimmten Namespace anwenden, listet der Pod_Name-Filter nur Pods auf diesem Knoten *und* in diesem Namespace auf.

Beachten Sie, dass die Platzhalter- und Ausdrucksfilterung mit Text oder Listen funktioniert, jedoch nicht mit Zahlen, Datumsangaben oder Booleschen Werten.

Vor der Installation oder Aktualisierung des NetApp Kubernetes Monitoring Operator

Lesen Sie diese Informationen, bevor Sie das"Kubernetes-Überwachungsoperator" .

Komponente	Erfordernis
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-Betriebssystem	Data Infrastructure Insights unterstützt keine Knoten, die mit Arm64-Architektur ausgeführt werden. Netzwerküberwachung: Es muss Linux-Kernel Version 4.18.0 oder höher ausgeführt werden. Photon OS wird nicht unterstützt.
Labels	Data Infrastructure Insights unterstützt die Überwachung von Kubernetes-Knoten, auf denen Linux ausgeführt wird, indem ein Kubernetes-Knotenselektor angegeben wird, der 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 PATH hinzu.
Konnektivität	kubectl cli ist für die Kommunikation mit dem Ziel-K8s-Cluster konfiguriert und verfügt über eine Internetverbindung zu Ihrer Data Infrastructure Insights -Umgebung. Wenn Sie sich während der Installation hinter einem Proxy befinden, folgen Sie den Anweisungen im"Konfigurieren der Proxy-Unterstützung" Abschnitt der Operator-Installation. Für eine genaue Prüfung und Datenberichterstattung synchronisieren Sie die Zeit auf dem Agent-Computer mithilfe des Network Time Protocol (NTP) oder Simple Network Time Protocol (SNTP).
Sonstige	Wenn Sie OpenShift 4.6 oder höher verwenden, müssen Sie die folgenden Schritte ausführen:"OpenShift-Anweisungen" zusätzlich zur Sicherstellung, dass diese Voraussetzungen erfüllt sind.
API-Token	Wenn Sie den Operator erneut bereitstellen (d. h. ihn aktualisieren oder ersetzen), müssen Sie kein neues API-Token erstellen. Sie können das vorherige Token wiederverwenden.

Wichtige Hinweise vor dem Start

Wenn Sie mit einemProxy , haben einebenutzerdefiniertes Repository oder verwendenOpenShift , lesen Sie die folgenden Abschnitte sorgfältig durch.

Lesen Sie auch überBerechtigungen .

Konfigurieren der Proxy-Unterstützung

Es gibt zwei Stellen, an denen Sie einen Proxy auf Ihrem Mandanten verwenden können, um den NetApp Kubernetes Monitoring Operator zu installieren. Dabei kann es sich um dasselbe oder um separate Proxy-Systeme handeln:

- Proxy, der während der Ausführung des Installationscode-Snippets (mit „curl“) benötigt wird, um das System, auf dem das Snippet ausgeführt wird, mit Ihrer Data Infrastructure Insights -Umgebung zu verbinden
- Proxy, der vom Ziel-Kubernetes-Cluster zur Kommunikation mit Ihrer Data Infrastructure Insights -Umgebung benötigt wird

Wenn Sie für eines oder beides einen Proxy verwenden, müssen Sie zur Installation des NetApp Kubernetes Operating Monitor zunächst sicherstellen, dass Ihr Proxy so konfiguriert ist, dass eine gute Kommunikation mit Ihrer Data Infrastructure Insights Umgebung möglich ist. Beispielsweise müssen Sie von 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 wird, vor der Installation des Operators die Umgebungsvariablen *http_proxy/https_proxy* fest. Für einige Proxy-Umgebungen müssen Sie möglicherweise auch die Umgebungsvariable *no_proxy* festlegen.

Um die Variable(n) festzulegen, führen Sie **vor** der Installation des NetApp Kubernetes Monitoring Operator die folgenden Schritte auf Ihrem System aus:

1. Legen Sie die Umgebungsvariable(n) *https_proxy* und/oder *http_proxy* für den aktuellen Benutzer fest:
   a. Wenn der einzurichtende Proxy keine Authentifizierung (Benutzername/Passwort) hat, führen Sie den folgenden Befehl aus:

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

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

Damit der für Ihren Kubernetes-Cluster verwendete Proxy mit Ihrer Data Infrastructure Insights -Umgebung kommunizieren kann, installieren Sie nach dem Lesen aller dieser Anweisungen den NetApp Kubernetes Monitoring Operator.

Konfigurieren Sie den Proxy-Abschnitt der 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 ruft der NetApp Kubernetes Monitoring Operator Container-Images aus dem Data Infrastructure Insights -Repository ab. Wenn Sie einen Kubernetes-Cluster als Ziel für die Überwachung verwenden und dieser Cluster so konfiguriert ist, dass er nur Container-Images aus einem benutzerdefinierten oder privaten Docker-Repository oder einer Container-Registrierung abruft, müssen Sie den Zugriff auf die vom NetApp Kubernetes Monitoring Operator benötigten Container konfigurieren.

Führen Sie das „Image Pull Snippet“ aus der Installationskachel des NetApp Monitoring Operator aus. Mit diesem Befehl melden Sie sich beim Data Infrastructure Insights -Repository an, rufen alle Bildabhängigkeiten für den Operator ab und melden sich vom Data Infrastructure Insights -Repository ab. Geben Sie bei der entsprechenden Aufforderung das bereitgestellte temporäre Repository-Passwort ein. Dieser Befehl lädt alle vom Bediener verwendeten Bilder herunter, auch für optionale Funktionen. Unten sehen Sie, für welche Funktionen diese Bilder verwendet werden.

Kernoperator-Funktionalität und Kubernetes-Überwachung

- NetApp-Überwachung
- Kube-RBAC-Proxy
- Kube-State-Metriken
- Telegraf
- Distroless-Root-Benutzer

Ereignisprotokoll

- fließendes Bit
- Kubernetes-Ereignis-Exporteur

Netzwerkleistung und Karte

- ci-net-observer

Übertragen Sie das Operator-Docker-Image gemäß Ihren Unternehmensrichtlinien in Ihr privates/lokales/Unternehmens-Docker-Repository. Stellen Sie sicher, dass die Bild-Tags und Verzeichnispfade zu diesen Bildern in Ihrem Repository mit denen im Data Infrastructure Insights -Repository übereinstimmen.

Bearbeiten Sie die Bereitstellung des Überwachungsoperators in operator-deployment.yaml und ändern Sie alle Bildreferenzen, 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 den neuen Speicherort des Docker-Repositorys widerzuspiegeln. 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 verwenden, 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 implementiert möglicherweise eine zusätzliche Sicherheitsebene, die den Zugriff auf einige Kubernetes-Komponenten blockieren kann.

Berechtigungen

Wenn der Cluster, den Sie überwachen, benutzerdefinierte Ressourcen enthält, die keine ClusterRole haben, die"Aggregate zum Anzeigen" , müssen Sie dem Operator manuell Zugriff auf diese Ressourcen gewähren, um sie mit Ereignisprotokollen zu überwachen.

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

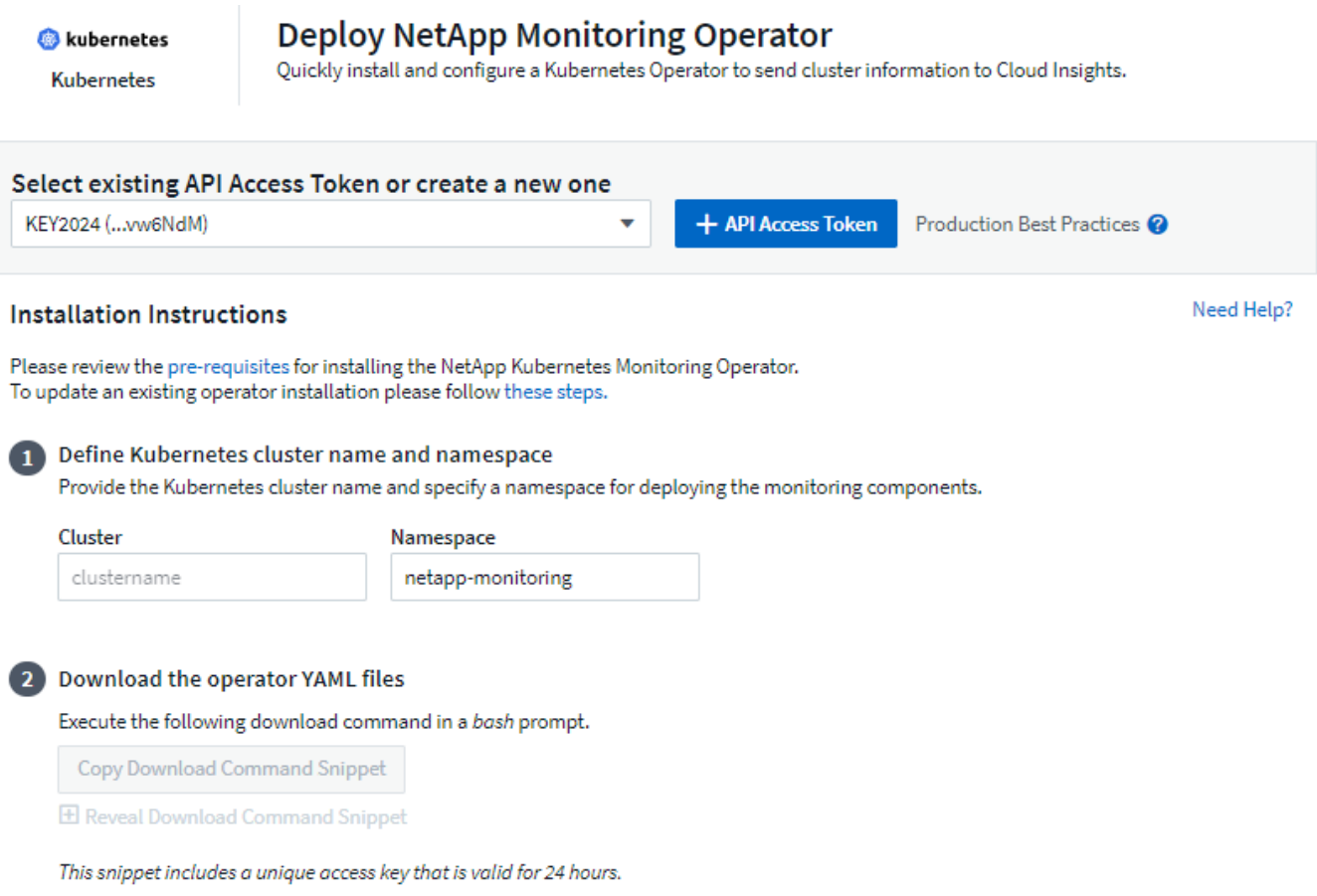
Installation und Konfiguration des Kubernetes Monitoring Operators

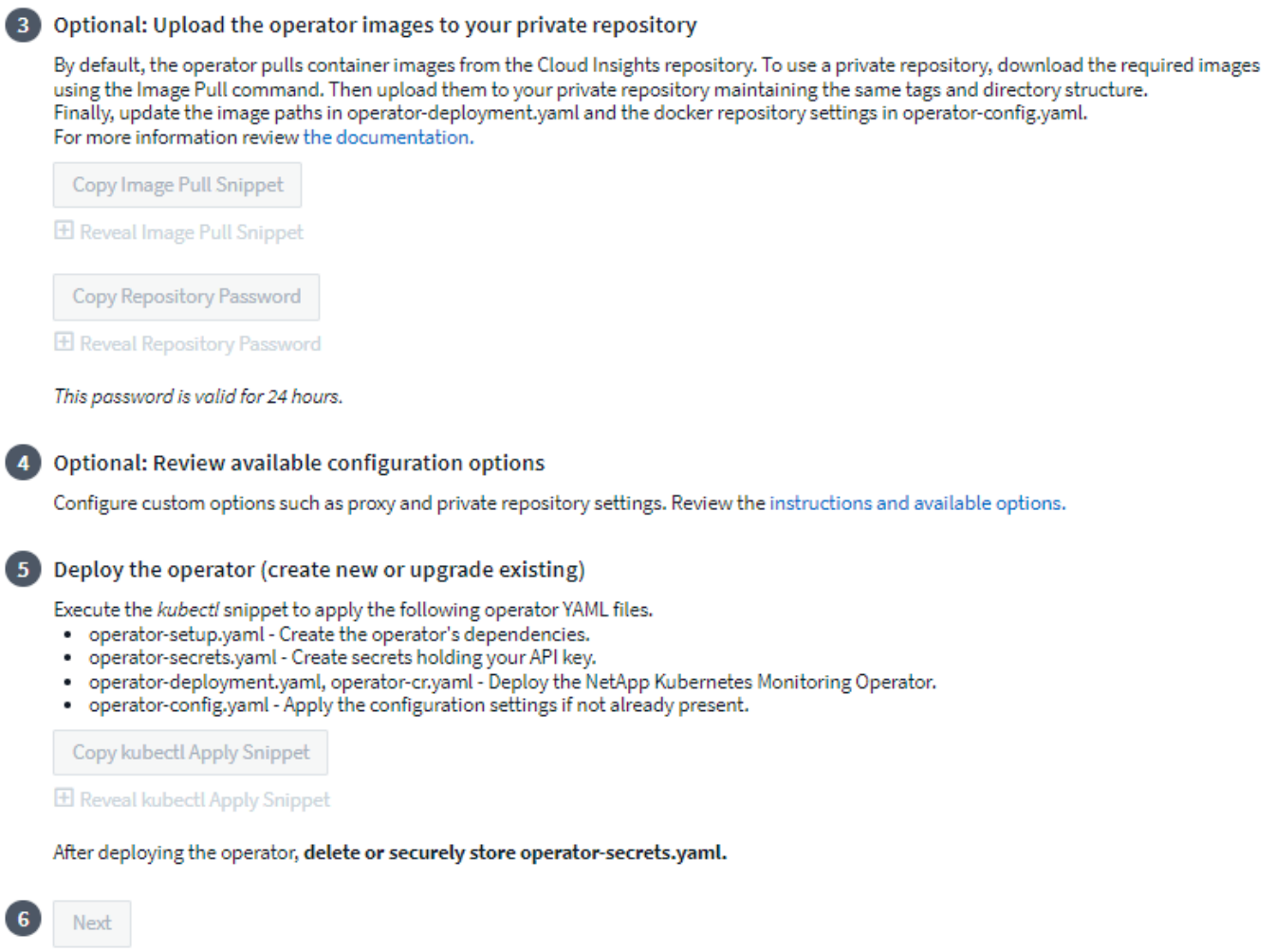
Data Infrastructure Insights bietet den **Kubernetes Monitoring Operator** für die Kubernetes-Sammlung. Navigieren Sie zu **Kubernetes > Collectors > +Kubernetes Collector**, um einen neuen Operator bereitzustellen.

Vor der Installation des Kubernetes Monitoring Operator

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

Installieren des Kubernetes Monitoring Operators


kubernetes
Kubernetes
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
clustername
Namespace
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.



Schritte zum Installieren des Kubernetes Monitoring Operator-Agenten auf Kubernetes:

1. Geben Sie einen eindeutigen Clusternamen und Namespace ein. Wenn SieUpgrade von einem vorherigen Kubernetes-Operator, verwenden Sie denselben Clusternamen und Namespace.
2. Sobald diese eingegeben sind, können Sie den Download-Befehlsausschnitt in die Zwischenablage kopieren.
3. Fügen Sie den Snippet in ein *Bash*-Fenster ein und führen Sie ihn aus. Die Operator-Installationsdateien werden heruntergeladen. Beachten Sie, dass das Snippet einen eindeutigen Schlüssel hat und 24 Stunden gültig ist.
4. Wenn Sie ein benutzerdefiniertes oder privates Repository haben, kopieren Sie den optionalen Image Pull-Ausschnitt, fügen Sie ihn in eine *Bash*-Shell ein und führen Sie ihn aus. Sobald die Bilder abgerufen wurden, kopieren Sie sie in Ihr privates Repository. Achten Sie darauf, dieselben Tags und dieselbe Ordnerstruktur beizubehalten. Aktualisieren Sie die Pfade in *operator-deployment.yaml* sowie die Docker-Repository-Einstellungen in *operator-config.yaml*.
5. Überprüfen Sie bei Bedarf die verfügbaren Konfigurationsoptionen wie Proxy- oder private Repository-Einstellungen. Weitere Informationen finden Sie unter"Konfigurationsoptionen" .
6. Wenn Sie bereit sind, stellen Sie den Operator bereit, indem Sie das kubectl Apply-Snippet kopieren, herunterladen und ausführen.
7. Die Installation erfolgt automatisch. Wenn der Vorgang abgeschlossen ist, klicken Sie auf die Schaltfläche *Weiter*.

8. Wenn die Installation abgeschlossen ist, klicken Sie auf die Schaltfläche *Weiter*. Denken Sie daran, auch die Datei *operator-secrets.yaml* zu löschen oder sicher zu speichern.

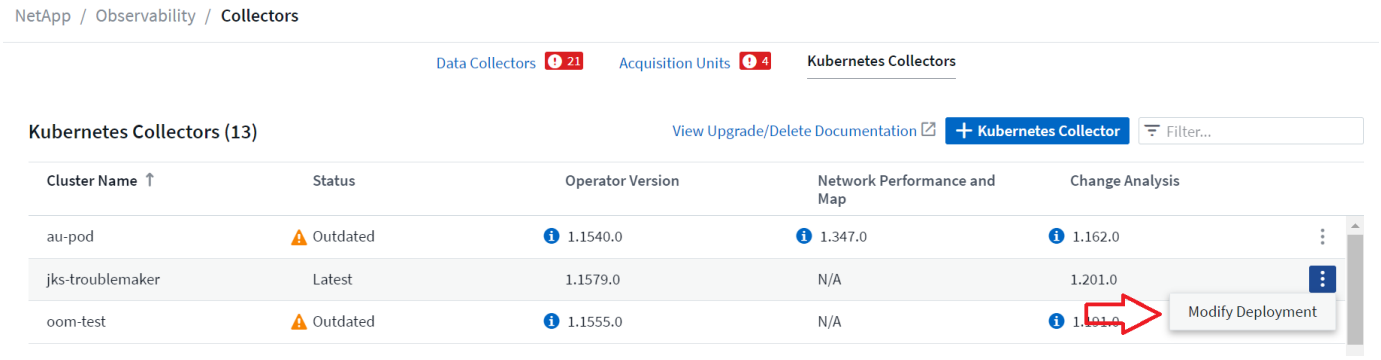
Wenn Sie ein benutzerdefiniertes Repository haben, lesen Sie überVerwenden eines benutzerdefinierten/privaten Docker-Repositorys .

Kubernetes-Überwachungskomponenten

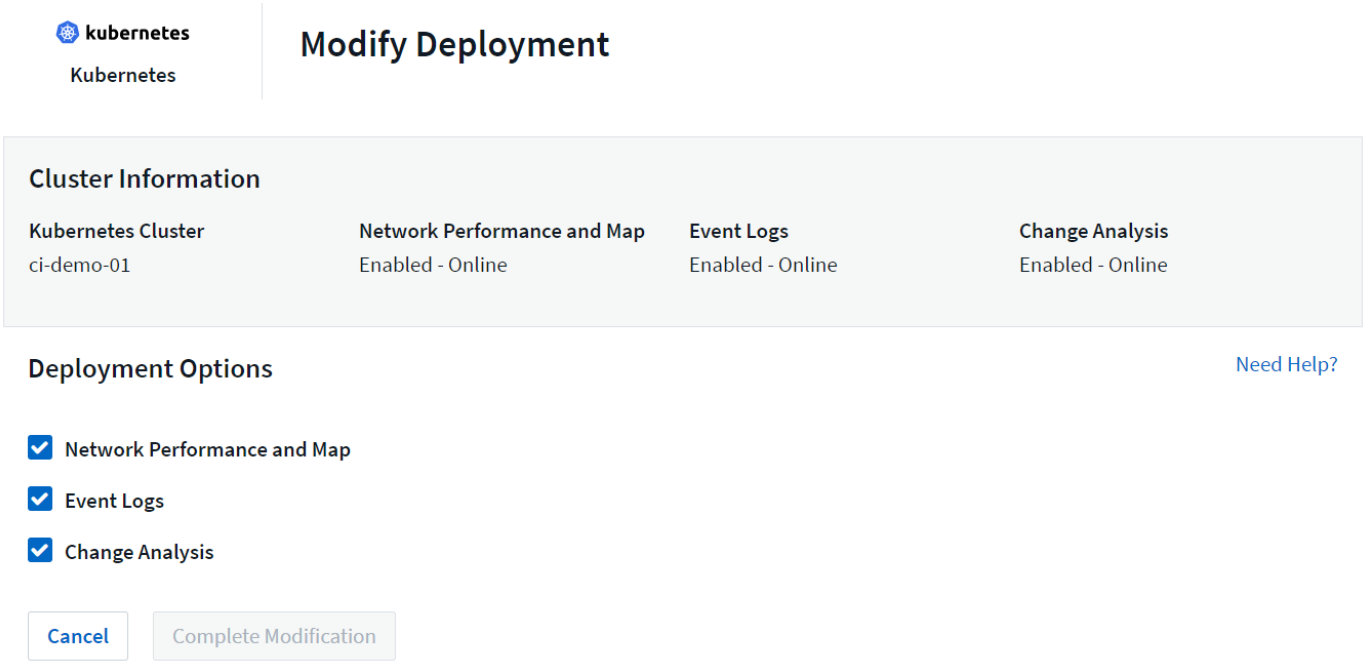
Data Infrastructure Insights Kubernetes Monitoring besteht aus vier Überwachungskomponenten:

- Clustermetriken
- Netzwerkleistung und Karte (optional)
- Ereignisprotokolle (optional)
- Änderungsanalyse (optional)

Die oben genannten optionalen Komponenten sind standardmäßig für jeden Kubernetes-Collector aktiviert. Wenn Sie entscheiden, dass Sie eine Komponente für einen bestimmten Collector nicht benötigen, können Sie sie deaktivieren, indem Sie zu **Kubernetes > Collectors** navigieren und im Drei-Punkte-Menü des Collectors auf der rechten Bildschirmseite *Bereitstellung ändern* auswählen.



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



Upgrade auf den neuesten Kubernetes Monitoring Operator

DII-Druckknopf-Upgrades

Sie können den Kubernetes Monitoring Operator über die DII Kubernetes Collectors-Seite aktualisieren. Klicken Sie auf das Menü neben dem Cluster, den Sie aktualisieren möchten, und wählen Sie *Upgrade*. Der Betreiber überprüft die Bildsignaturen, erstellt einen Snapshot Ihrer aktuellen Installation und führt das Upgrade durch. Innerhalb weniger Minuten sollte der Status des Operators von „Upgrade läuft“ bis „Neueste“ fortschreiten. Wenn ein Fehler auftritt, können Sie für weitere Einzelheiten den Fehlerstatus auswählen und die Tabelle zur Fehlerbehebung bei Push-Button-Upgrades weiter unten zu Rate ziehen.

Push-Button-Upgrades mit privaten Repositories

Wenn Ihr Operator für die Verwendung eines privaten Repositorys konfiguriert ist, stellen Sie bitte sicher, dass alle zum Ausführen des Operators erforderlichen Bilder und deren Signaturen in Ihrem Repository verfügbar sind. Wenn während des Upgrade-Vorgangs ein Fehler aufgrund fehlender Bilder auftritt, fügen Sie diese einfach zu Ihrem Repository hinzu und versuchen Sie das Upgrade erneut. Um die Bildsignaturen in Ihr Repository hochzuladen, verwenden Sie bitte das Cosign-Tool wie folgt und stellen Sie sicher, dass Sie Signaturen für alle unter 3 angegebenen Bilder hochladen. Optional: Laden Sie die Operatorbilder in Ihr privates Repository hoch > Image Pull Snippet

```
cosign copy example.com/src:v1 example.com/dest:v1
#Example
cosign copy <DII container registry>/netapp-monitoring:<image version>
<private repository>/netapp-monitoring:<image version>
```

Rollback auf eine zuvor ausgeführte Version

Wenn Sie das Upgrade mithilfe der Funktion „Upgrade per Knopfdruck“ durchgeführt haben und innerhalb von sieben Tagen nach dem Upgrade Probleme mit der aktuellen Version des Operators auftreten, können Sie mithilfe des während des Upgrade-Vorgangs erstellten Snapshots ein Downgrade auf die zuvor ausgeführte

Version durchführen. Klicken Sie auf das Menü neben dem Cluster, für den Sie ein Rollback durchführen möchten, und wählen Sie *Rollback* aus.

Manuelle Upgrades

Stellen Sie fest, ob eine Agentenkonfiguration mit dem vorhandenen Operator vorhanden ist (wenn Ihr Namespace nicht der Standardnamespace *netapp-monitoring* ist, ersetzen Sie ihn durch den entsprechenden Namespace):

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

- Installierender neueste Operator über den vorhandenen Operator.
  - Stellen Sie sicher, dass SieAbrufen der neuesten Container-Images wenn Sie ein benutzerdefiniertes Repository verwenden.

Wenn die Agentenkonfiguration nicht vorhanden ist:

- Notieren Sie sich den von Data Infrastructure Insights erkannten Clusternamen (wenn Ihr Namespace nicht der Standardnamespace „netapp-monitoring“ ist, ersetzen Sie ihn durch den entsprechenden Namespace):

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

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

  - Verwenden Sie denselben Clusternamen.
  - Nachdem Sie die neuesten Operator-YAML-Dateien heruntergeladen haben, portieren Sie vor der Bereitstellung alle in agent_backup.yaml gefundenen Anpassungen in die heruntergeladene operator-config.yaml.
  - Stellen Sie sicher, dass SieAbrufen der neuesten Container-Images wenn Sie ein benutzerdefiniertes Repository verwenden.

Stoppen und Starten des Kubernetes-Überwachungsoperators

So stoppen 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

So entfernen Sie den Kubernetes Monitoring Operator

Beachten Sie, dass der Standardnamespace 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 Monitoring-Operators 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>
Hinweis: Wenn der erste Befehl „Keine Ressourcen gefunden“ zurückgibt,
befolgen Sie die folgenden Anweisungen, um ältere Versionen des
Überwachungsoperators zu deinstallieren.
```

Führen Sie die folgenden Befehle der Reihe nach aus. Abhängig von Ihrer aktuellen Installation können einige dieser Befehle die Meldung „Objekt nicht gefunden“ zurückgeben. Diese Nachrichten können bedenkenlos 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 zu Kube-State-Metrics finden Sie unter"diese Seite" .

Konfigurieren/Anpassen des Operators

Diese Abschnitte enthalten Informationen zum Anpassen Ihrer Operatorkonfiguration, zum Arbeiten mit Proxy, zum Verwenden eines benutzerdefinierten oder privaten Docker-Repositorys oder zum Arbeiten mit OpenShift.

Konfigurationsoptionen

Die am häufigsten geänderten Einstellungen können in der benutzerdefinierten Ressource *AgentConfiguration* konfiguriert werden. Sie können diese Ressource vor der Bereitstellung des Operators bearbeiten, indem Sie die Datei *operator-config.yaml* bearbeiten. Diese Datei enthält auskommentierte Beispiele für Einstellungen. Siehe die Liste der"Verfügbare Einstellungen" für die neueste 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 Ihre 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 (Nicht gefunden)“ angezeigt wird, muss
Ihr Operator aktualisiert werden, bevor Sie die Agentenkonfiguration
verwenden können.
```

Konfigurieren der Proxy-Unterstützung

Es gibt zwei Stellen, an denen Sie einen Proxy auf Ihrem Mandanten verwenden können, um den Kubernetes Monitoring Operator zu installieren. Dabei kann es sich um dasselbe oder um separate Proxy-Systeme handeln:

- Proxy, der während der Ausführung des Installationscode-Snippets (mit „curl“) benötigt wird, um das System, auf dem das Snippet ausgeführt wird, mit Ihrer Data Infrastructure Insights -Umgebung zu verbinden
- Proxy, der vom Ziel-Kubernetes-Cluster zur Kommunikation mit Ihrer Data Infrastructure Insights -Umgebung benötigt wird

Wenn Sie für einen oder beide einen Proxy 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/der Sie den Operator installieren möchten, auf Data Infrastructure Insights zugreifen können, ist Ihr Proxy wahrscheinlich richtig konfiguriert.

Legen Sie für den Proxy, der zur Installation des Kubernetes Operating Monitor verwendet wird, vor der Installation des Operators die Umgebungsvariablen *http_proxy/https_proxy* fest. Für einige Proxy-Umgebungen müssen Sie möglicherweise auch die Umgebungsvariable *no_proxy* festlegen.

Um die Variable(n) festzulegen, führen Sie **vor** der Installation des Kubernetes Monitoring Operator die folgenden Schritte auf Ihrem System aus:

1. Legen Sie die Umgebungsvariable(n) *https_proxy* und/oder *http_proxy* für den aktuellen Benutzer fest:
   a. Wenn der einzurichtende Proxy keine Authentifizierung (Benutzername/Passwort) hat, führen Sie den folgenden Befehl aus:

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

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

Damit der für Ihren Kubernetes-Cluster verwendete Proxy mit Ihrer Data Infrastructure Insights -Umgebung kommunizieren kann, installieren Sie nach dem Lesen aller dieser Anweisungen den Kubernetes Monitoring Operator.

Konfigurieren Sie den Proxy-Abschnitt der 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 ruft der Kubernetes Monitoring Operator Container-Images aus dem Data Infrastructure Insights Repository ab. Wenn Sie einen Kubernetes-Cluster als Ziel für die Überwachung verwenden und dieser Cluster so konfiguriert ist, dass er nur Container-Images aus einem benutzerdefinierten oder privaten Docker-Repository oder Container-Register abruft, müssen Sie den Zugriff auf die vom Kubernetes Monitoring Operator benötigten Container konfigurieren.

Führen Sie das „Image Pull Snippet“ aus der Installationskachel des NetApp Monitoring Operator aus. Mit diesem Befehl melden Sie sich beim Data Infrastructure Insights -Repository an, rufen alle Bildabhängigkeiten für den Operator ab und melden sich vom Data Infrastructure Insights -Repository ab. Geben Sie bei der entsprechenden Aufforderung das bereitgestellte temporäre Repository-Passwort ein. Dieser Befehl lädt alle vom Bediener verwendeten Bilder herunter, auch für optionale Funktionen. Unten sehen Sie, für welche Funktionen diese Bilder verwendet werden.

Kernoperator-Funktionalität und Kubernetes-Überwachung

- NetApp-Überwachung
- ci-kube-rbac-proxy
- ci-ksm
- ci-telegraf
- Distroless-Root-Benutzer

Ereignisprotokoll

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

Netzwerkleistung und Karte

- ci-net-observer

Übertragen Sie das Operator-Docker-Image gemäß Ihren Unternehmensrichtlinien in Ihr privates/lokales/Unternehmens-Docker-Repository. Stellen Sie sicher, dass die Bild-Tags und Verzeichnispfade zu diesen Bildern in Ihrem Repository mit denen im Data Infrastructure Insights -Repository übereinstimmen.

Bearbeiten Sie die Bereitstellung des Überwachungsoperators in operator-deployment.yaml und ändern Sie alle Bildreferenzen, 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 den neuen Speicherort des Docker-Repositorys widerzuspiegeln. 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 verwenden, 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 implementiert möglicherweise eine zusätzliche Sicherheitsebene, die den Zugriff auf einige Kubernetes-Komponenten blockieren kann.

Toleranzen und Makel

Die DaemonSets *netapp-ci-telegraf-ds*, *netapp-ci-fluent-bit-ds* und *netapp-ci-net-observer-l4-ds* müssen auf jedem Knoten in Ihrem Cluster einen Pod planen, um Daten auf allen Knoten korrekt zu erfassen. Der Operator wurde so konfiguriert, dass er einige bekannte **Verunreinigungen** toleriert. Wenn Sie benutzerdefinierte Taints auf Ihren Knoten konfiguriert haben und dadurch verhindern, dass Pods auf jedem Knoten ausgeführt werden, können Sie eine **Toleranz** für diese Taints erstellen."in der *AgentConfiguration*" . Wenn Sie benutzerdefinierte Taints auf alle Knoten in Ihrem Cluster angewendet haben, müssen Sie der Operatorbereitstellung auch die erforderlichen Toleranzen hinzufügen, damit der Operator-Pod geplant und ausgeführt werden kann.

Mehr über Kubernetes erfahren"Makel und Duldungen" .

Zurück zum"* Seite „NetApp Kubernetes Monitoring Operator Installation“*"

Eine Anmerkung zu Geheimnissen

Um dem Kubernetes Monitoring Operator die Berechtigung zum Anzeigen von Geheimnissen im gesamten Cluster zu entziehen, löschen Sie vor der Installation die folgenden Ressourcen aus der Datei *operator-setup.yaml*:

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

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 Datei *AgentConfiguration* oder *operator-config.yaml*, um den Abschnitt zur Änderungsverwaltung zu kommentieren und *kindsToIgnoreFromWatch: '"secrets"'* in den Abschnitt zur Änderungsverwaltung aufzunehmen. Beachten Sie 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 Bildsignaturen des Kubernetes-Überwachungsoperators

Das Image für den Operator und alle zugehörigen Images, die er bereitstellt, sind von NetApp signiert. Sie können die Images vor der Installation manuell mit dem Cosign-Tool überprüfen oder einen Kubernetes-Zulassungscontroller konfigurieren. Weitere Einzelheiten finden Sie in der"Kubernetes-Dokumentation" .

Der öffentliche Schlüssel, der zum Überprüfen der Bildsignaturen verwendet wird, ist in der Installationskachel des Überwachungsoperators unter *Optional: Laden Sie die Operatorbilder in Ihr privates Repository hoch > Öffentlicher Schlüssel der Bildsignatur* verfügbar.

Um eine Bildsignatur manuell zu überprüfen, führen Sie die folgenden Schritte aus:

1. Kopieren und führen Sie das Image Pull Snippet aus
2. Kopieren Sie das Repository-Passwort und geben Sie es ein, wenn Sie dazu aufgefordert werden.
3. Speichern Sie den öffentlichen Schlüssel der Bildsignatur (dii-image-signing.pub im Beispiel).
4. Überprüfen Sie die Bilder mit Cosign. Siehe 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

Wenn beim Einrichten des Kubernetes Monitoring Operators Probleme auftreten, können Sie Folgendes versuchen:

Problem:	Versuchen Sie Folgendes:
Ich sehe keinen Hyperlink/keine Verbindung zwischen meinem Kubernetes Persistent Volume und dem entsprechenden Back-End-Speichergerät. Mein Kubernetes Persistent Volume wird mit dem Hostnamen des Speicherservers konfiguriert.	Befolgen Sie die Schritte zum Deinstallieren des vorhandenen Telegraf-Agenten und installieren Sie anschließend den neuesten Telegraf-Agenten neu. Sie müssen Telegraf Version 2.0 oder höher verwenden und Ihr Kubernetes-Clusterspeicher muss aktiv von Data Infrastructure Insights überwacht werden.

Problem:	Versuchen Sie Folgendes:
Ich sehe in den Protokollen Meldungen, die den folgenden ähneln: E0901 15:21:39.962145 1 reflector.go:178] k8s.io/kube-state-metrics/internal/store/builder.go:352: *v1.MutatingWebhookConfiguration konnte nicht aufgelistet werden: Der Server konnte die angeforderte Ressource nicht finden. E0901 15:21:43.168161 1 reflector.go:178] k8s.io/kube-state-metrics/internal/store/builder.go:352: *v1.Lease konnte nicht aufgelistet werden: Der Server konnte die angeforderte Ressource nicht finden (get leases.coordination.k8s.io) usw.	Diese Meldungen können auftreten, wenn Sie kube-state-metrics Version 2.0.0 oder höher mit Kubernetes-Versionen unter 1.20 ausführen. So erhalten Sie die Kubernetes-Version: *kubectl version* So erhalten Sie die kube-state-metrics-Version: *kubectl get deploy/kube-state-metrics -o jsonpath='{..image}'* Um diese Meldungen zu verhindern, können Benutzer ihre kube-state-metrics-Bereitstellung ändern, um die folgenden Leases zu deaktivieren: *mutatingwebhookconfigurations validatingwebhookconfigurations volumeattachments resources* Genauer gesagt können sie das folgende CLI-Argument verwenden: resources=certificatesigningrequests,configmaps,cronjobs,daemonsets, deployments,endpoints,horizontalpodautoscalers,ingresses,jobs,limitranges, namespaces,networkpolicies,nodes,persistentvolumeclaims,persistentvolumes, poddisruptionbudgets,pods,replicasets,replicationcontrollers,resourcequotas, secrets,services,statefulsets,storageclasses Die Standardressourcenliste ist: „Zertifikatsignaturanforderungen, Konfigurationszuordnungen, Cronjobs, Daemonsets, Bereitstellungen, Endpunkte, horizontale Pod-Autoskalierer, Ingresses, Jobs, Leases, Grenzwertbereiche, mutierende Webhookkonfigurationen, Namespaces, Netzwerkrichtlinien, Knoten, persistente Volumeansprüche, persistente Volumes, Pod-Unterbrechungsbudgets, Pods, Replikatsets, Replikationscontroller, Ressourcenkontingente, Geheimnisse, Dienste, Statefulsets, Speicherklassen, validierende Webhookkonfigurationen, Volumeanhänge“

Problem:	Versuchen Sie Folgendes:
Ich sehe Fehlermeldungen von Telegraf, die den folgenden ähneln, aber Telegraf wird gestartet und ausgeführt: 11. Okt. 14:23:41 ip-172-31-39-47 systemd[1]: Der Plugin-gesteuerte Server-Agent zum Melden von Metriken in InfluxDB wurde gestartet. 11. Okt. 14:23:41 ip-172-31-39-47 telegraf[1827]: time="2021-10-11T14:23:41Z" level=error msg="Cache-Verzeichnis konnte nicht erstellt werden. /etc/telegraf/.cache/snowflake, err: mkdir /etc/telegraf/.ca che: Zugriff verweigert. Ignoriert\n" func="gosnowflake.(*defaultLogger).Errorf" file="log.go:120" 11. Okt. 14:23:41 ip-172-31-39-47 telegraf[1827]: time="2021-10-11T14:23:41Z" level=error msg="Öffnen fehlgeschlagen. Ignoriert. Öffnen Sie /etc/telegraf/.cache/snowflake/ocsp_response_cache.json: keine solche Datei oder kein solches Verzeichnis\n" func="gosnowflake.(*defaultLogger).Errorf" file="log.go:120" 11. Okt. 14:23:41 ip-172-31-39-47 telegraf[1827]: 2021-10-11T14:23:41Z I! Telegraf 1.19.3 wird gestartet	Dies ist ein bekanntes Problem. Siehe"Dieser GitHub-Artikel" für weitere Details. Solange Telegraf läuft, können Benutzer diese Fehlermeldungen ignorieren.
Auf Kubernetes melden meine Telegraf-Pods den folgenden Fehler: „Fehler beim Verarbeiten der Mountstats-Informationen: Mountstats-Datei konnte nicht geöffnet werden: /hostfs/proc/1/mountstats, Fehler: Öffnen von /hostfs/proc/1/mountstats: Berechtigung verweigert“	Wenn SELinux aktiviert ist und erzwungen wird, verhindert es wahrscheinlich, dass die Telegraf-Pods auf die Datei /proc/1/mountstats auf dem Kubernetes-Knoten zugreifen können. Um diese Einschränkung zu umgehen, bearbeiten Sie die Agentenkonfiguration und aktivieren Sie die Einstellung „runPrivileged“. Weitere Einzelheiten finden Sie in den OpenShift-Anweisungen.
Auf Kubernetes meldet mein Telegraf ReplicaSet-Pod den folgenden Fehler: [inputs.prometheus] Fehler im Plugin: Schlüsselpaar /etc/kubernetes/pki/etcd/server.crt konnte nicht geladen werden:/etc/kubernetes/pki/etcd/server.key: öffne /etc/kubernetes/pki/etcd/server.crt: keine solche Datei oder kein solches Verzeichnis	Der Telegraf ReplicaSet-Pod soll auf einem Knoten ausgeführt werden, der als Master oder für etcd bestimmt ist. Wenn der ReplicaSet-Pod auf einem dieser Knoten nicht ausgeführt wird, werden diese Fehler angezeigt. Überprüfen Sie, ob Ihre Master-/etcd-Knoten Verunreinigungen aufweisen. Wenn dies der Fall ist, fügen Sie die erforderlichen Toleranzen zum Telegraf ReplicaSet, telegraf-rs, hinzu. Bearbeiten Sie beispielsweise das ReplicaSet … kubectl edit rs telegraf-rs … und fügen Sie der Spezifikation die entsprechenden Toleranzen hinzu. Starten Sie dann den ReplicaSet-Pod neu.

Problem:	Versuchen Sie Folgendes:
Ich habe eine PSP/PSA-Umgebung. Betrifft dies meinen Überwachungsbetreiber?	Wenn Ihr Kubernetes-Cluster mit Pod Security Policy (PSP) oder Pod Security Admission (PSA) ausgeführt wird, müssen Sie auf den neuesten Kubernetes Monitoring Operator aktualisieren. Befolgen Sie diese Schritte, um auf den aktuellen Operator mit Unterstützung für PSP/PSA zu aktualisieren: 1. Deinstallieren der vorherige Überwachungsoperator: kubectl delete agent 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-cluster-admin-rolebinding 2. Installieren die neueste Version des Überwachungsoperators.
Beim Versuch, den Operator bereitzustellen, sind mir Probleme begegnet, und ich verwende PSP/PSA.	1. Bearbeiten Sie den Agenten mit dem folgenden Befehl: kubectl -n <name-space> edit agent 2. Markieren Sie „security-policy-enabled“ als „false“. Dadurch werden die Pod-Sicherheitsrichtlinien und die Pod-Sicherheitszulassung deaktiviert und dem Operator die Bereitstellung ermöglicht. Bestätigen Sie mit den folgenden Befehlen: kubectl get psp (sollte anzeigen, dass die Pod-Sicherheitsrichtlinie entfernt wurde) kubectl get all -n <namespace>
grep -i psp (sollte anzeigen, dass nichts gefunden wurde)	„ImagePullBackoff“-Fehler aufgetreten
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 es ordnungsgemäß erkannt wird. Mehr lesen Informationen zur Konfiguration für benutzerdefinierte/private Repos.	Ich habe ein Problem mit der Bereitstellung meines Überwachungsoperators und die aktuelle Dokumentation hilft mir nicht bei der Lösung.

Problem:	Versuchen Sie Folgendes:

Erfassen oder notieren Sie die Ausgabe der folgenden Befehle und wenden Sie sich an das technische Supportteam.
 	
`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-Pods (Workload Map) im Operator-Namespace befinden sich in CrashLoopBackOff

Diese Pods entsprechen dem Workload Map-Datensammler für die Netzwerkbeobachtung. Versuchen Sie Folgendes: • Überprüfen Sie die Protokolle eines der Pods, um die Mindestkernelversion zu bestätigen. Beispiel: ---- {"ci-tenant-id":"Ihre Mandanten-ID","collector-cluster":"Ihr K8S-Clustername","environment":"prod","level":"error","msg":"Validierung fehlgeschlagen. Grund: Kernelversion 3.10.0 ist niedriger als die Mindestkernelversion 4.18.0","time":"2022-11-09T08:23:08Z"} ---- • Net-Observer-Pods erfordern mindestens die Linux-Kernelversion 4.18.0. Überprüfen Sie die Kernelversion mit dem Befehl „uname -r“ und stellen Sie sicher, dass sie >= 4.18.0 ist	Pods werden im Operator-Namespace ausgeführt (Standard: Netapp-Monitoring), aber in der Benutzeroberfläche werden keine Daten für die Workload-Map 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-Computer mithilfe des Network Time Protocol (NTP) oder Simple Network Time Protocol (SNTP) zu synchronisieren.	Einige der Net-Observer-Pods im Operator-Namespace befinden sich im Status „Ausstehend“
Net-Observer ist ein DaemonSet 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 die CPU oder den Speicher vorliegt. Stellen Sie sicher, dass im Knoten genügend Speicher und CPU verfügbar sind.	Unmittelbar nach der Installation des Kubernetes Monitoring Operator wird mir in meinen Protokollen Folgendes angezeigt: [inputs.prometheus] Fehler im Plug-In: Fehler beim Senden der 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: no such host

Problem:	Versuchen Sie Folgendes:
Diese Meldung wird normalerweise nur angezeigt, wenn ein neuer Operator installiert wird und der *telegraf-rs*-Pod vor dem *ksm*-Pod aktiv ist. Diese Nachrichten sollten aufhören, sobald alle Pods ausgeführt werden.	Ich sehe keine erfassten Metriken für die in meinem Cluster vorhandenen Kubernetes-CronJobs.
Überprüfen Sie Ihre Kubernetes-Version (d. h. `kubectl version`). Wenn es sich um v1.20.x oder niedriger handelt, ist dies eine erwartete Einschränkung. Die mit dem Kubernetes Monitoring Operator bereitgestellte Version von kube-state-metrics unterstützt nur v1.CronJob. Bei Kubernetes 1.20.x und darunter befindet sich die CronJob-Ressource unter v1beta.CronJob. Aus diesem Grund kann kube-state-metrics die CronJob-Ressource nicht finden.	Nach der Installation des Operators treten die Telegraf-DS-Pods in CrashLoopBackOff ein und die Pod-Protokolle zeigen „su: Authentifizierungsfehler“ an.
Bearbeiten Sie den Telegraf-Abschnitt in *AgentConfiguration* und setzen Sie *dockerMetricCollectionEnabled* auf „false“. Weitere Einzelheiten finden Sie in der Betriebsanleitung des Betreibers."Konfigurationsoptionen" . … Spezifikation: … Telegraf: … - Name: Docker-Ausführungsmodus: - DaemonSet-Ersetzungen: - Schlüssel: DOCKER_UNIX_SOCK_PLACEHOLDER-Wert: unix:///run/docker.sock … …	In meinen Telegraf-Protokollen werden immer wieder Fehlermeldungen angezeigt, die den folgenden ähneln: E! [Agent] Fehler beim Schreiben in outputs.http: Post "https://<tenant_url>/rest/v1/lake/ingest/influxdb": Kontextfrist überschritten (Client.Timeout beim Warten auf Header überschritten)
Bearbeiten Sie den Telegraf-Abschnitt in *AgentConfiguration* und erhöhen Sie *outputTimeout* auf 10 s. Weitere Einzelheiten finden Sie in der Betriebsanleitung des Betreibers."Konfigurationsoptionen" .	Mir fehlen *involvedobject*-Daten für einige Ereignisprotokolle.
Stellen Sie sicher, dass Sie die Schritte in der"Berechtigungen" Abschnitt oben.	Warum werden zwei Überwachungsoperator-Pods ausgeführt, einer mit dem Namen netapp-ci-monitoring-operator-<pod> und der andere mit dem Namen monitoring-operator-<pod>?
Ab dem 12. Oktober 2023 hat Data Infrastructure Insights den Operator überarbeitet, um unseren Benutzern einen besseren Service zu bieten. Damit diese Änderungen vollständig übernommen werden können, müssen SieEntfernen Sie den alten Operator Undinstallieren Sie die neue .	Meine Kubernetes-Ereignisse wurden unerwartet nicht mehr an Data Infrastructure Insights gemeldet.
Rufen Sie den Namen des Event-Exporter-Pods ab: `kubectl -n netapp-monitoring get pods`	grep event-exporter

Problem:	Versuchen Sie Folgendes:
awk '{print $1}'	sed 's/event-exporter./event-exporter/'` Es sollte entweder „netapp-ci-event-exporter“ oder „event-exporter“ sein. Bearbeiten Sie als Nächstes den Überwachungsagenten `kubectl -n netapp-monitoring edit agent` und legen Sie den Wert für LOG_FILE so fest, dass er den entsprechenden Event-Exporter-Pod-Namen widerspiegelt, der im vorherigen Schritt gefunden wurde. Genauer gesagt sollte LOG_FILE entweder auf "/var/log/containers/netapp-ci-event-exporter.log" oder "/var/log/containers/event-exporter*.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 auchdeinstallieren UndNeuinstallation der Agent.
Ich sehe, dass vom Kubernetes Monitoring Operator bereitgestellte Pods aufgrund unzureichender Ressourcen abstürzen.	Siehe den Kubernetes Monitoring Operator"Konfigurationsoptionen" um die CPU- und/oder Speichergrenzen nach Bedarf zu erhöhen.
Ein fehlendes Image oder eine ungültige Konfiguration führte dazu, dass die netapp-ci-kube-state-metrics-Pods nicht gestartet werden konnten oder nicht bereit waren. Jetzt steckt das StatefulSet fest und Konfigurationsänderungen werden nicht auf die Netapp-CI-Kube-State-Metrics-Pods angewendet.	Das StatefulSet ist in einem"gebrochen" Zustand. Nachdem Sie alle Konfigurationsprobleme behoben haben, führen Sie einen Bounce der Netapp-CI-Kube-State-Metrics-Pods durch.
netapp-ci-kube-state-metrics-Pods können nach der Ausführung eines Kubernetes Operator-Upgrades nicht gestartet werden und lösen ErrImagePull aus (das Abrufen des Images schlägt fehl).	Versuchen Sie, die Pods manuell zurückzusetzen.
Bei der Protokollanalyse werden für meinen Kubernetes-Cluster Meldungen vom Typ „Ereignis verworfen, da es älter ist als maxEventAgeSeconds“ beobachtet.	Ändern Sie die Operator-Agentenkonfiguration und erhöhen Sie *event-exporter-maxEventAgeSeconds* (z. B. auf 60 s), *event-exporter-kubeQPS* (z. B. auf 100) und *event-exporter-kubeBurst* (z. B. auf 500). Weitere Einzelheiten zu diesen Konfigurationsoptionen finden Sie im"Konfigurationsoptionen" Seite.

Problem:	Versuchen Sie Folgendes:
Telegraf warnt vor unzureichendem sperrbaren Speicher oder stürzt ab.	Versuchen Sie, das Limit des sperrbaren Speichers für Telegraf im zugrunde liegenden Betriebssystem/Knoten zu erhöhen. Wenn eine Erhöhung des Limits keine Option ist, ändern Sie die NKMO-Agentenkonfiguration und setzen Sie *unprotected* auf *true*. Dadurch wird Telegraf angewiesen, keinen Versuch zu unternehmen, gesperrte Speicherseiten zu reservieren. Dies kann zwar ein Sicherheitsrisiko darstellen, da entschlüsselte Geheimnisse möglicherweise auf die Festplatte ausgelagert werden, ermöglicht jedoch die Ausführung in Umgebungen, in denen die Reservierung gesperrten Speichers nicht möglich ist. Weitere Informationen zu den *ungeschützten* Konfigurationsoptionen finden Sie im"Konfigurationsoptionen" Seite.
Ich sehe Warnmeldungen von Telegraf, die etwa wie folgt aussehen: *W! [inputs.diskio] Der Datenträgername für „vdc“ konnte nicht ermittelt werden: Fehler beim Lesen von /dev/vdc: keine solche Datei oder kein solches Verzeichnis*	Für den Kubernetes Monitoring Operator sind diese Warnmeldungen harmlos und können ignoriert werden. Alternativ können Sie den Abschnitt „Telegraf“ in der AgentConfiguration bearbeiten und *runDsPrivileged* auf „true“ setzen. Weitere Einzelheiten finden Sie im"Konfigurationsoptionen des Betreibers" .

Problem:	Versuchen Sie Folgendes:

Mein Fluent-Bit-Pod schlägt mit den folgenden Fehlern fehl: [2024/10/16 14:16:23] [Fehler] [/src/fluent-bit/plugins/in_tail/tail_fs_inotify.c:360 errno=24] Zu viele offene Dateien [2024/10/16 14:16:23] [Fehler] Initialisierung der Eingabe tail.0 fehlgeschlagen [2024/10/16 14:16:23] [Fehler] [Engine] Initialisierung der Eingabe fehlgeschlagen	Versuchen Sie, Ihre *fsnotify*-Einstellungen in Ihrem Cluster zu ändern:
	`sudo sysctl fs.inotify.max_user_instances (take note of setting)`
	`sudo sysctl fs.inotify.max_user_instances=<something larger than current setting>`
	`sudo sysctl fs.inotify.max_user_watches (take note of setting)`
	`sudo sysctl fs.inotify.max_user_watches=<something larger than current setting>`
	Starten Sie Fluent-bit neu.
	Hinweis: Um diese Einstellungen auch nach einem Neustart des Knotens dauerhaft zu halten, müssen Sie die folgenden Zeilen in */etc/sysctl.conf* einfügen.
	`fs.inotify.max_user_instances=<something larger than current setting>`
	`fs.inotify.max_user_watches=<something larger than current setting>`

Problem:	Versuchen Sie Folgendes:
Die Telegraf DS-Pods melden Fehler im Zusammenhang mit dem Kubernetes-Eingabe-Plugin, das keine HTTP-Anfragen stellen kann, da das TLS-Zertifikat nicht validiert werden kann. Zum Beispiel: E! [inputs.kubernetes] Fehler im Plugin: Fehler beim Senden einer HTTP-Anfrage an"<a href="https://<kubelet_IP>:10250/stats/summary": " class="bare">https://<kubelet_IP>:10250/stats/summary": Erhalten"<a href="https://<kubelet_IP>:10250/stats/summary": " class="bare">https://<kubelet_IP>:10250/stats/summary": tls: Zertifikat konnte nicht überprüft werden: x509: Zertifikat für <kubelet_IP> kann nicht validiert werden, da es keine IP-SANs enthält	Dies tritt auf, wenn das Kubelet selbstsignierte Zertifikate verwendet und/oder das angegebene Zertifikat die <kubelet_IP> nicht in der Liste „Subject Alternative Name“ des Zertifikats enthält. Um dieses Problem zu lösen, kann der Benutzer die"Agentenkonfiguration" , und setzen Sie *telegraf:insecureK8sSkipVerify* auf *true*. Dadurch wird das Telegraf-Eingabe-Plugin so konfiguriert, dass die Überprüfung übersprungen wird. Alternativ kann der Benutzer das Kubelet konfigurieren für"serverTLSBootstrap" , wodurch eine Zertifikatsanforderung von der API „certificates.k8s.io“ ausgelöst wird.

Weitere Informationen finden Sie in der"Support" Seite oder in der"Datensammler-Supportmatrix" .

Konfigurationsoptionen für den Kubernetes-Überwachungsoperator

Der "Kubernetes-Überwachungsoperator" bietet umfangreiche Anpassungsmöglichkeiten über die AgentConfiguration-Datei. Sie können Ressourcenlimits, Erfassungsintervalle, Proxy-Einstellungen, Toleranzen und komponentenspezifische Einstellungen konfigurieren, um die Überwachung Ihrer Kubernetes-Umgebung zu optimieren. Nutzen Sie diese Optionen, um Telegraf, Kube-State-Metrics, die Protokollerfassung, die Workload-Zuordnung, das Änderungsmanagement und andere Überwachungskomponenten anzupassen.

Beispieldatei für AgentConfiguration

Nachfolgend finden Sie eine Beispiel-AgentConfiguration-Datei mit Beschreibungen für jede Option.

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

spec:
  ##
  ## One can modify the following settings to configure and customize the
operator.
  ## Optional settings are commented out with their 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 cluster name
    ## The cluster name must be unique across all clusters in your Data
Infrastructure Insights (DII) environment.
    ##
    clusterName: "my_cluster"

    ##
    ## Proxy settings
    ## If applicable, specify the proxy through which the operator should
communicate with DII.
    ## Refer to additional 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]
    ## Repository from which the operator pulls the required images
    ## By default, the operator pulls from the DII repository. To use a
private repository, set this field to the
    ## applicable repository name. Refer to additional 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]
    ## Name of the imagePullSecret required for dockerRepo
    ## When using a private repository, set this field to the applicable
```

```
secret name.
    ##
    dockerImagePullSecret: 'netapp-ci-docker'

    ##
    ## Automatic expiring API key rotation settings
    ## Allow the operator to automatically rotate its expiring API key,
generating a new API key and
    ## using it to replace the expiring one. The expiring API key itself
must support auto rotation.
    ##
    # tokenRotationEnabled: 'true'
    ##
    ## Threshold (number of days before expiration) at which the operator
should trigger rotation.
    ## The threshold must be less than the total duration of the API key.
    ##
    # tokenRotationThresholdDays: '30'

  push-button-upgrades:
    ##
    ## Allow the operator to be upgraded using the Data Infrastructure
Insights (DII) UI
    ##
    # enabled: 'true'

    ##
    ## Frequency at which the operator polls and checks for upgrade
requests from DII
    ##
    # polltimeSeconds: '60'

    ##
    ## Allow operator upgrade to proceed even if new images are not
present
    ##
    # ignoreImageNotPresent: 'false'

    ##
    ## Allow operator upgrade to proceed even if image signature
verification fails
    ## Warning: Enabling this setting is dangerous!
    ##
    # ignoreImageSignatureFailure: 'false'

    ##
```

```
    ## Allow operator upgrade to proceed even if image signature
verification fails
    ## Warning: Enabling this setting is dangerous!
    ##
    # ignoreYAMLSignatureFailure: 'false'

    ##
    ## Use dockerImagePullSecret to access the image repository and verify
the existence of the new images
    ##
    # imageValidationUseSecret: 'true'

    ##
    ## Time allowed for the old operator pod to shutdown before reporting
an upgrade failure to DII
    ##
    # upgradesShutdownTime: '240'

    ##
    ## Time allowed for the new operator pod to startup before reporting
an upgrade failure to DII
    ##
    # upgradesStartupTime: '600'

  telegraf:
    ##
    ## Frequency at which telegraf collects data
    ## The frequency should not exceed 60s.
    ##
    # collectionInterval: '60s'

    ##
    ## Maximum number of metrics per batch
    ## Telegraf sends metrics to outputs in batches. This controls the
size of those writes.
    ##
    # batchSize: '10000'

    ##
    ## Maximum number of unwritten metrics per output
    ## Telegraf caches metrics until they are successfully written by the
output. This controls how many metrics
    ## can be cached. Once the buffer is filled, the oldest metrics will
get dropped.
    ##
    # bufferLimit: '150000'
```

```
    ##
    ## Rounds collection interval to collectionInterval
    ## If collectionInterval is 60s, collection will occur on-the-minute
    ##
    # roundInterval: 'true'

    ##
    ## Jitter between plugins on collection
    ## Each input plugin sleeps a random amount of time within jitter
before collecting. This can be used to prevent
    ## multiple input plugins from querying the same resources at the same
time. The maximum collection interval would
    ## be collectionInterval + collectionJitter.
    ##
    # collectionJitter: '0s'

    ##
    ## Precision to which collected metrics are rounded
    ## When set to "0s", precision will be set by the units specified by
collectionInterval.
    ##
    # precision: '0s'

    ##
    ## Frequency at which telegraf flushes and writes data
    ## Frequency should not exceed collectionInterval.
    ##
    # flushInterval: '60s'

    ##
    ## Jitter between plugins on writes
    ## Each output plugin sleeps a random amount of time within jitter
before flushing. This can be used to prevent
    ## multiple output plugins from writing the same resources at the same
time, and causing large spikes. The maximum
    ## flush interval would be flushInterval + flushJitter.
    ##
    # flushJitter: '0s'

    ##
    ## Timeout for HTTP output plugins
    ## Time allowed for http output plugins to successfully writing before
failing.
    ##
    # outputTimeout: '5s'
```

```
    ##
    ## CPU/Mem limits and requests for netapp-ci-telegraf-ds DaemonSet
    ##
    # dsCpuLimit: '750m'
    # dsMemLimit: '800Mi'
    # dsCpuRequest: '100m'
    # dsMemRequest: '500Mi'

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

    ##
    ## telegraf runs through the processor plugins a second time after the
aggregators plugins, by default. Use this
    ## option to skip the second run.
    ##
    # skipProcessorsAfterAggregators: 'false'

    ##
    ## Additional tolerations for netapp-ci-telegraf-ds DaemonSet and
netapp-ci-telegraf-rs ReplicaSet
    ## Inspect the netapp-ci-telegraf-rs ReplicaSet and netapp-ci-
telegraf-ds DaemonSet to view the default tolerations.
    ## If additional tolerations are needed, specify them here using the
following abbreviated single line format:
    ##
    ## Example: '{key: taint1, operator: Exists, effect: NoSchedule},{key:
taint2, operator: Exists, effect: NoExecute}'
    ##
    # dsTolerations: ''
    # rsTolerations: ''

    ##
    ## Additional node selector terms for netapp-ci-telegraf-rs ReplicaSet
    ## Inspect the netapp-ci-telegraf-rs ReplicaSet to view the default
node selectors terms. If additional node
    ## selector terms are needed, specify them here using the following
abbreviated single line format:
    ##
    ## Example: '{"key": "myLabel1","operator": "In","values":
["myVal1"]},{"key": "myLabel2","operator": "In","values": ["myVal2"]}'
```

```
    ##
    ## These additional node selector terms will be AND'd with the default
ones via matchExpressions.
    ##
    # rsNodeSelectorTerms: ''

    ##
    ## telegraf uses lockable memory to protect secrets in memory. If
telegraf issues warnings about insufficient
    ## lockable memory, try increasing the limit of lockable memory on the
applicable nodes. If increasing this limit
    ## is not an option for the given environment, set unprotected to true
so telegraf does not attempt to use
    ## lockable memory.
    ##
    # unprotected: 'false'

    ##
    ## Run the netapp-ci-telegraf-ds DaemonSet's telegraf-mountstats-
poller container in privileged mode
    ## The telegraf-mountstats-poller container needs read-only access to
system files such as those in /proc/ (i.e. to
    ## monitor NFS IO metrics, etc.). Some environments impose restricts
that prevent the container from reading these
    ## system files. Unless those restrictions are lifted, users may need
to run this container in privileged mode.
    ##
    # runPrivileged: 'false'

    ##
    ## Run the netapp-ci-telegraf-ds DaemonSet's telegraf container in
privileged mode
    ## The telegraf container needs read-only access to system files such
as those in /dev/ (i.e. for the telegraf
    ## diskio input plugin to retrieve disk metrics). Some environments
impose restricts that prevent the container from
    ## accessing these system files. Unless those restrictions are lifted,
users may need to run this container in
    ## privileged mode.
    ##
    # runDsPrivileged: 'false'

    ##
    ## Allow the netapp-ci-telegraf-ds DaemonSet's telegraf-ds, telegraf-
init, and telegraf-mountstats-poller containers
    ## to run with escalation privilege. This is needed to access/read
```

```
root-protected files (node UUID,
    ## /proc/1/mountstats, etc.). Allowing escalation privilege should
negate the need to run these containers in
    ## privileged mode.
    ##
    # allowDsPrivilegeEscalation: 'true'

    ##
    ## Allow the netapp-ci-telegraf-rs DaemonSet's telegraf-rs and
telegraf-rs-init containers
    ## to run with escalation privilege. This is needed to access/read
root-protected files (node UUID,
    ## etcd credentials when applicable, etc.). Allowing escalation
privilege should negate the need to run these
    ## containers in privileged mode.
    ##
    # allowRsPrivilegeEscalation: 'true'

    ##
    ## Enable collection of block IO metrics (kubernetes.pod_to_storage)
    ##
    # dsBlockIOEnabled: 'true'

    ##
    ## Enable collection of NFS IO metrics (kubernetes.pod_to_storage)
    ##
    # dsNfsIOEnabled: 'true'

    ##
    ## Enable collection of system-specific objects/metrics for managed
k8s clusters
    ## This consists of k8s objects within the kube-system and cattle-
system namespaces for managed k8s clusters
    ## (i.e. EKS, AKS, GKE, managed Rancher, etc.).
    ##
    # managedK8sSystemMetricCollectionEnabled: 'false'

    ##
    ## Enable collection of pod ephemeral storage metrics
(kubernetes.pod_volume)
    ##
    # podVolumeMetricCollectionEnabled: 'false'

    ##
    ## Declare Rancher cluster is managed
    ## Rancher can be deployed in managed or on-premise environments. The
```

```
operator contains logic to try to determine
    ## which type of environment Rancher is running in (i.e. to factor
into managedK8sSystemMetricCollectionEnabled).
    ## If the operator logic misidentifies whether Rancher is running in a
managed environment or not, use this option
    ## to declare Rancher is managed.
    ##
    # isManagedRancher: 'false'

    ##
    ## Locations for the etcd certificate and key files
    ## The operator looks at well-known locations for the etcd certificate
and key files. If this cannot find these
    ## files, the applicable telegraf input plugin will fail. Use this
option to specify the complete filepath to these
    ## files on the nodes.
    ## Note that the well-known locations for these files are typically
root-protected. This is one of the reasons why
    ## the netapp-ci-telegraf-rs ReplicaSet's telegraf-rs-init container
needs to run with escalation privileges.
    ##
    # rsHostEtcdCrt: ''
    # rsHostEtcdKey: ''

    ##
    ## Allow operator/telegraf communications with k8s without TLS
verification
    ## In some environments, TLS verification will not succeed (i.e.
certificates lack IP SANs). To skip the
    ## verification, use this option.
    ##
    # insecureK8sSkipVerify: 'false'

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

    ##
    ## Comma-separated list of k8s resources for which to collect metrics
    ## Refer to the kube-state-metrics --resources CLI option
```

```
    ##
    # resources:
'cronjobs,daemonsets,deployments,horizontalpodautoscalers,ingresses,jobs,n
amespaces,nodes,persistentvolumeclaims,persistentvolumes,pods,replicasets,
resourcequotas,services,statefulsets'

    ##
    ## Comma-separated list of k8s metrics to collect
    ## Refer to the kube-state-metrics --metric-allowlist CLI option
    ##
    # 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,kube_horizontal
podautoscaler_spec_max_replicas,kube_horizontalpodautoscaler_spec_min_repl
icas,kube_horizontalpodautoscaler_status_condition,kube_horizontalpodautos
caler_status_current_replicas,kube_horizontalpodautoscaler_status_desired_
replicas'

    ##
    ## Comma-separated list of k8s label keys that will be used to
determine which labels to export/collect
    ## Refer to the kube-state-metrics --metric-labels-allowlist CLI
option
    ##
    # labels:
'cronjobs=[*],daemonsets=[*],deployments=[*],horizontalpodautoscalers=[*],
ingresses=[*],jobs=[*],namespaces=[*],nodes=[*],persistentvolumeclaims=[*]
,persistentvolumes=[*],pods=[*],replicasets=[*],resourcequotas=[*],service
s=[*],statefulsets=[*]'

    ##
```

```
    ## Additional tolerations for netapp-ci-kube-state-metrics StatefulSet
    ## Inspect the netapp-ci-kube-state-metrics StatefulSet to view the
default tolerations. If additional
    ## tolerations are needed, specify them here using the following
abbreviated single line format:
    ##
    ## Example: '{key: taint1, operator: Exists, effect: NoSchedule},{key:
taint2, operator: Exists, effect: NoExecute}'
    ##
    # tolerations: ''

    ##
    ## Additional node selector terms for netapp-ci-kube-state-metrics
StatefulSet
    ## Inspect the kube-state-metrics StatefulSet to view the default node
selectors terms. If additional node selector
    ## terms are needed, specify them here using the following abbreviated
single line format:
    ##
    ## Example: '{"key": "myLabel1","operator": "In","values":
["myVal1"]},{"key": "myLabel2","operator": "In","values": ["myVal2"]}'
    ##
    ## These additional node selector terms will be AND'd with the default
ones via matchExpressions.
    ##
    # nodeSelectorTerms: ''

    ##
    ## Number of kube-state-metrics shards
    ## For large clusters, kube-state-metrics may be overwhelmed with
collecting and exporting the amount of metrics
    ## generated. This can lead to collection timeouts for the netapp-ci-
telegraf-rs pod. If this is observed, use this
    ## option to increase the number of kube-state-metrics shards to
redistribute the workload.
    ##
    # shards: '2'

  logs:
    ##
    ## Allow the netapp-ci-fluent-bit-ds DaemonSet's fluent-bit container
to run with escalation privilege.
    ## This is needed to access/read root-protected files (event-exporter
pod log, fluent-bit DB file, etc.).
    ##
    # fluent-bit-allowPrivilegeEscalation: 'true'
```

```
    ##
    ## Read content from the head of the file, not the tail
    ##
    # readFromHead: "true"

    ##
    ## Network protocol for DNS (i.e. UDP, TCP, etc.)
    ##
    # dnsMode: "UDP"

    ##
    ## DNS resolver (i.e. LEGACY, ASYNC, etc.)
    ##
    # fluentBitDNSResolver: "LEGACY"

    ##
    ## Additional tolerations for netapp-ci-fluent-bit-ds DaemonSet
    ## Inspect the netapp-ci-fluent-bit-ds DaemonSet to view the default
tolerations. If additional tolerations are
    ## needed, specify them here using the following abbreviated single
line format:
    ##
    ## Example: '{key: taint1, operator: Exists, effect: NoSchedule},{key:
taint2, operator: Exists, effect: NoExecute}'
    ##
    # fluent-bit-tolerations: ''

    ##
    ## CPU/Mem limits and requests for netapp-ci-fluent-bit-ds DaemonSet
    ##
    # fluent-bit-cpuLimit: '500m'
    # fluent-bit-memLimit: '1Gi'
    # fluent-bit-cpuRequest: '50m'
    # fluent-bit-memRequest: '100Mi'

    ##
    ## Top-level host path in which the kubernetes container logs reside,
including any symlinks from var/log/containers
    ## For example, if /var/log/containers/*.log is a symlink to
/kubernetes/log to
    ## /kubernetes/var/lib/docker/containers/*/*.log, fluent-bit-
containerLogPath should be set to '/kubernetes'.
    ##
    # fluent-bit-containerLogPath: '/var/lib/docker/containers'
```

```
    ## fluent-bit DB file path/location

    ##
    ## fluent-bit DB file path/location
    ## By default, fluent-bit is configured to use /var/log/netapp-
monitoring_flb_kube.db.  This path usually requires
    ## escalated privileges for read/write. Users who want to avoid
escalation privilege can use this option to specify
    ## a different DB file path/location. The custom path/location should
allow non-root users to read/write.
    ## Ideally, the path/location should be persistent.
    ##
    # fluent-bit-dbFile: '/var/log/netapp-monitoring_flb_kube.db'

    ##
    ## Additional tolerations for netapp-ci-event-exporter Deployment
    ## Inspect the netapp-ci-event-exporter Deployment to view the default
tolerations. If additional tolerations are
    ## needed, specify them here using the following abbreviated single
line format:
    ##
    ## Example: '{key: taint1, operator: Exists, effect: NoSchedule},{key:
taint2, operator: Exists, effect: NoExecute}'
    ##
    # event-exporter-tolerations: ''

    ##
    ## CPU/Mem limits and requests for netapp-ci-event-exporter Deployment
    ##
    # event-exporter-cpuLimit: '500m'
    # event-exporter-memLimit: '1Gi'
    # event-exporter-cpuRequest: '50m'
    # event-exporter-memRequest: '100Mi'

    ##
    ## Max age for events to be processed and exported; older events are
discarded
    ##
    # event-exporter-maxEventAgeSeconds: '10'

    ##
    ## Client-side throttling
    ## Set event-exporter-kubeBurst to roughly match event rate
    ## Set event-exporter-kubeQPS to approximately 1/5 of event-exporter-
kubeBurst
    ##
```

```
    # event-exporter-kubeQPS: 20
    # event-exporter-kubeBurst: 100

    ##
    ## Additional node selector terms for netapp-ci-event-exporter
Deployment
    ## Inspect the event-exporter Deployment to view the default node
selectors terms. If additional node selector terms
    ## are needed, specify them here using the following abbreviated
single line format:
    ##
    ## Example: '{"key": "myLabel1","operator": "In","values":
["myVal1"]},{"key": "myLabel2","operator": "In","values": ["myVal2"]}'
    ##
    ## These additional node selector terms will be AND'd with the default
ones via matchExpressions.
    ##
    # event-exporter-nodeSelectorTerms: ''

  workload-map:
    ## Run workload-map container with escalation privilege to coordinate
memlocks
    ##
    ## Allow the netapp-ci-net-observer-l4-ds DaemonSet's net-observer
container to run with escalation privilege.
    ## This is needed to coordinate memlocks.
    ##
    # allowPrivilegeEscalation: 'true'

    ##
    ## CPU/Mem limits and requests for netapp-ci-net-observer-l4-ds
DaemonSet
    ##
    # cpuLimit: '500m'
    # memLimit: '500Mi'
    # cpuRequest: '100m'
    # memRequest: '500Mi'

    ##
    ## Metric aggregation interval (in seconds)
    ## Set metricAggregationInterval between 30 and 120
    ##
    # metricAggregationInterval: '60'

    ##
    ## Interval for bpf polling
```

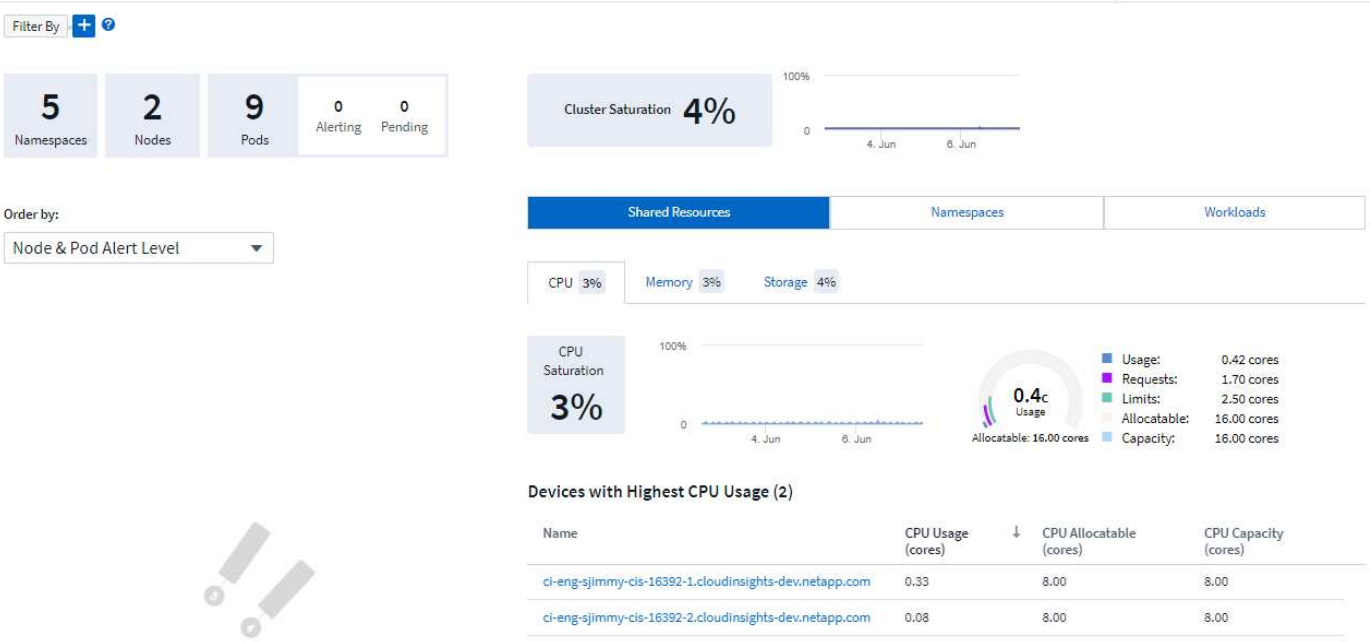
```
    ## Set bpfPollInterval between 3 and 15
    ##
    # bpfPollInterval: '8'

    ##
    ## Enable reverse DNS lookups on observed IPs
    ##
    # enableDNSLookup: 'true'

    ##
    ## Additional tolerations for netapp-ci-net-observer-l4-ds DaemonSet
    ## Inspect the netapp-ci-net-observer-l4-ds DaemonSet to view the
default tolerations. If additional tolerations
    ## are needed, specify them here using the following abbreviated
single line format:
    ##
    ## Example: '{key: taint1, operator: Exists, effect: NoSchedule},{key:
taint2, operator: Exists, effect: NoExecute}'
    ##
    # l4-tolerations: ''

    ##
    ## Run the netapp-ci-net-observer-l4-ds DaemonSet's net-observer
container in privileged mode
    ## Some environments impose restricts that prevent the net-observer
container from running.
    ## Unless those restrictions are lifted, users may need to run this
container in privileged mode.
    ##
    # runPrivileged: 'false'

  change-management:
    ##
    ## CPU/Mem limits and requests for netapp-ci-change-observer-watch-rs
ReplicaSet
    ##
    # cpuLimit: '1'
    # memLimit: '1Gi'
    # cpuRequest: '500m'
    # memRequest: '500Mi'

    ##
    ## Interval (in seconds) after which a non-successful deployment of a
workload will be marked as failed
    ##
    # workloadFailureDeclarationIntervalSeconds: '30'
```

```
    ##
    ## Frequency (in seconds) at which workload deployments are combined
and sent
    ##
    # workloadDeployAggrIntervalSeconds: '300'

    ##
    ## Frequency (in seconds) at which non-workload deployments are
combined and sent
    ##
    # nonWorkloadDeployAggrIntervalSeconds: '15'

    ##
    ## 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"'

    ##
    ## Additional node selector terms for netapp-ci-change-observer-watch-
rs ReplicaSet
    ## Inspect the netapp-ci-change-observer-watch-rs ReplicaSet to view
the default node selectors terms. If additional
    ## node selector terms are needed, specify them here using the
following abbreviated single line format:
    ##
    ## Example: '{"key": "myLabel1","operator": "In","values":
["myVal1"]},{"key": "myLabel2","operator": "In","values": ["myVal2"]}'
    ##
    ## These additional node selector terms will be AND'd with the default
ones via matchExpressions.
    ##
    # nodeSelectorTerms: ''

    ##
    ## Comma-separated list of additional kinds to watch
    ## Each kind should be prefixed by its API group. This list in
addition to the default set of kinds watched by the
    ## collector.
    ##
    ## Example: '"authorization.k8s.io.subjectaccessreviews"'
    ##
    # additionalKindsToWatch: ''
```

```
    ##
    ## 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: ''

    ##
    ## Comma-separated list of kinds to ignore from watching from the
default set of kinds watched by the collector
    ## Each kind should be prefixed by its API group.
    ##
    ## Example: '"networking.k8s.io.networkpolicies,batch.jobs",
"authorization.k8s.io.subjectaccessreviews"'
    ##
    # kindsToIgnoreFromWatch: ''

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

    ##
    ## Additional tolerations for netapp-ci-change-observer-watch-rs
ReplicaSet
    ## Inspect the netapp-ci-change-observer-watch-rs ReplicaSet to view
the default tolerations. If additional
    ## tolerations are needed, specify them here using the following
abbreviated single line format:
    ##
    ## Example: '{key: taint1, operator: Exists, effect: NoSchedule},{key:
taint2, operator: Exists, effect: NoExecute}'
    ##
    # watch-tolerations: ''
```

Kubernetes-Cluster-Detailseite

Auf der Detailseite des Kubernetes-Clusters wird eine detaillierte Übersicht über Ihren Kubernetes-Cluster angezeigt.



Anzahl der Namespaces, Knoten und Pods

Die Zählungen oben auf der Seite zeigen Ihnen die Gesamtzahl der Namespaces, Knoten und Pods im Cluster sowie die Anzahl der Popds, die derzeit Alarm schlagen und ausstehen.

Gemeinsam genutzte Ressourcen und Sättigung

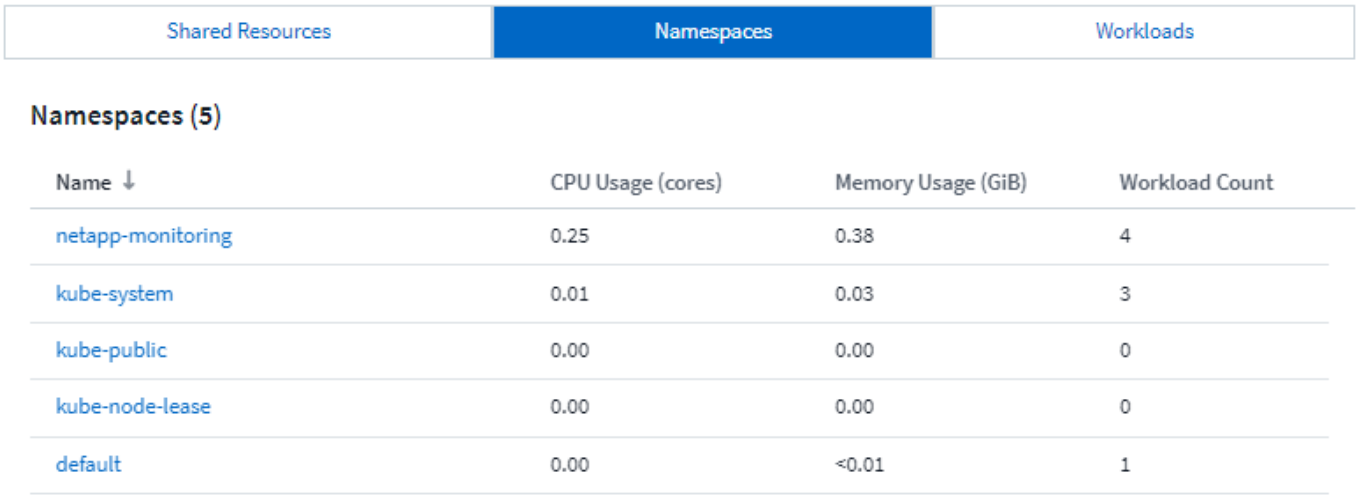
Oben rechts auf der Detailseite wird Ihre Clustersättigung als aktueller Prozentsatz sowie ein Diagramm angezeigt, das den aktuellen Trend im Zeitverlauf zeigt. Die Clustersättigung ist die höchste CPU-, Speicher- oder Speichersättigung zu jedem Zeitpunkt.

Darunter zeigt die Seite standardmäßig die Nutzung **gemeinsam genutzter Ressourcen** mit Registerkarten für CPU, Arbeitsspeicher und Speicherplatz. Jede Registerkarte zeigt den Sättigungsprozentsatz und den Trend im Zeitverlauf mit zusätzlichen Nutzungsdetails. Beim Speicher ist der angezeigte Wert der höhere der beiden Werte für Backend- und Dateisystemsättigung, die unabhängig voneinander berechnet werden.

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

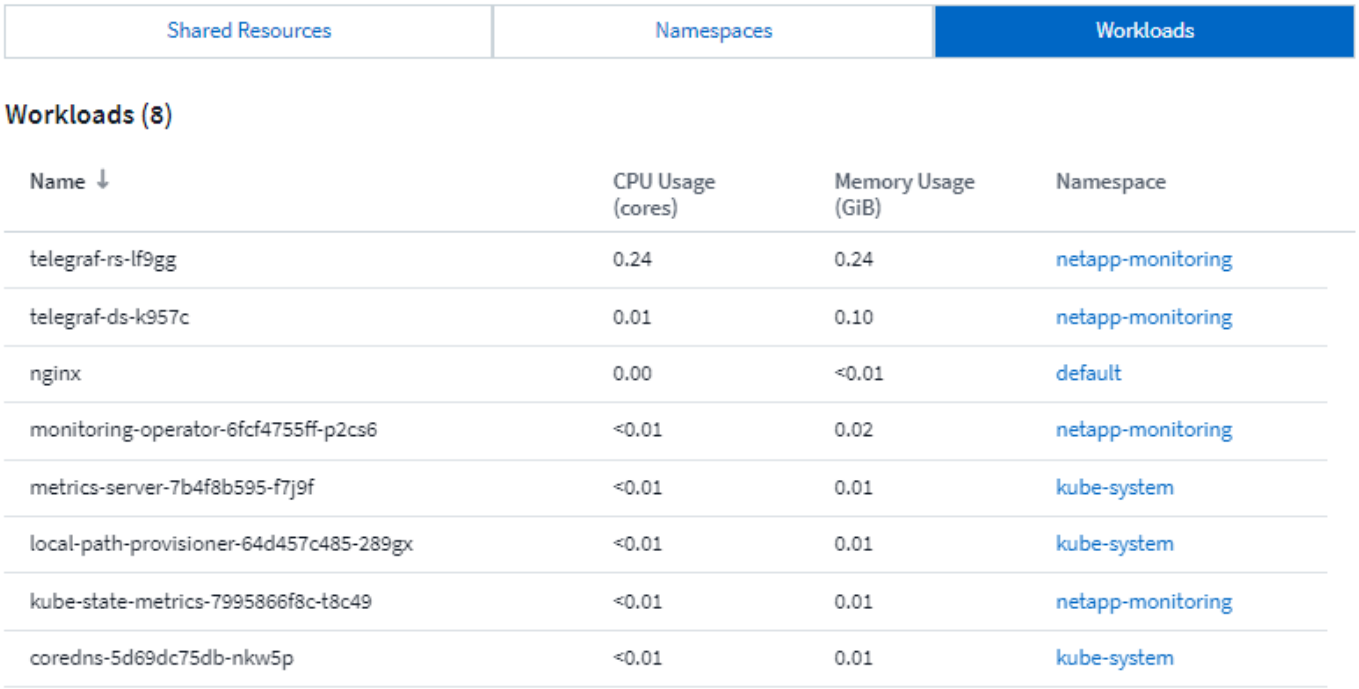
Namensräume

Auf der Registerkarte „Namespaces“ wird eine Liste aller Namespaces in Ihrer Kubernetes-Umgebung angezeigt, die die CPU- und Speicherauslastung sowie die Anzahl der Workloads in jedem Namespace anzeigt. Klicken Sie auf die Namenslinks, um die einzelnen Namespaces zu erkunden.



Arbeitslasten

Ebenso wird auf der Registerkarte „Workloads“ eine Liste der Workloads in jedem Namespace angezeigt, wobei auch hier die CPU- und Speicherauslastung angezeigt wird. Durch Klicken auf die Namespace-Links gelangen Sie zu den einzelnen Links.



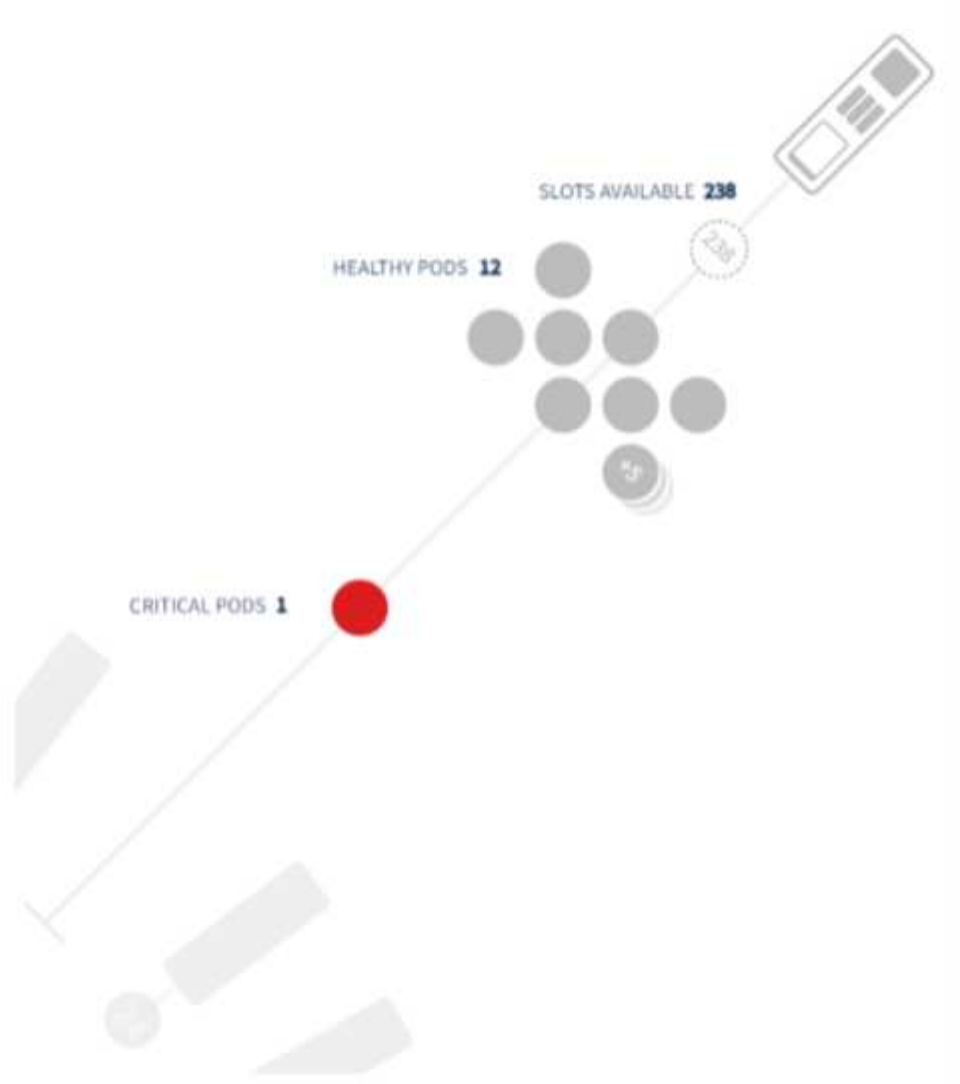
Das Cluster-„Rad“

Der Cluster-Abschnitt „Rad“ bietet auf einen Blick Informationen zum Zustand von Knoten und Pods, die Sie für weitere Informationen genauer untersuchen können. Wenn Ihr Cluster mehr Knoten enthält, als in diesem Bereich der Seite angezeigt werden können, können Sie das Rad mit den verfügbaren Schaltflächen drehen.

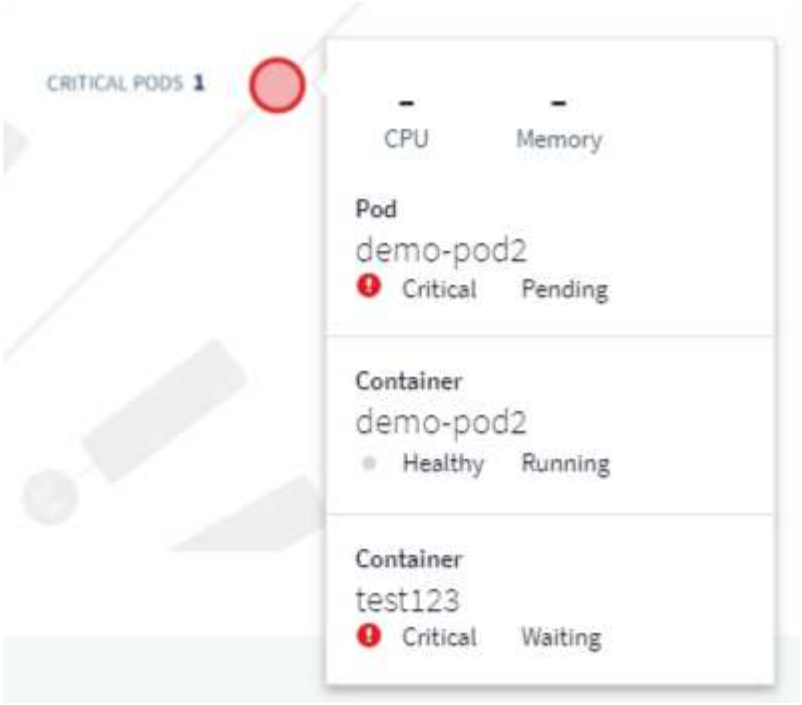
Alarmierende Pods oder Knoten werden rot angezeigt. „Warnbereiche“ werden orange angezeigt. Nicht geplante (d. h. nicht verbundene) Pods werden in der unteren Ecke des Cluster-„Rads“ angezeigt.

Wenn Sie mit der Maus über einen Pod (Kreis) oder Knoten (Balken) fahren, wird die Ansicht des Knotens erweitert.

Wenn Sie in dieser Ansicht auf den Pod oder Knoten klicken, wird die erweiterte Knotenansicht vergrößert.



Von hier aus können Sie mit der Maus über ein Element fahren, um Details zu diesem Element anzuzeigen. Wenn Sie in diesem Beispiel beispielsweise mit der Maus über den kritischen Pod fahren, werden Details zu diesem Pod angezeigt.



Sie können Informationen zu Dateisystem, Speicher und CPU anzeigen, indem Sie mit der Maus über die Knotenelemente fahren.

Ein Hinweis zu den Messgeräten

Die Speicher- und CPU-Anzeigen zeigen drei Farben an, da sie die *Nutzung* im Verhältnis zur *zuweisbaren Kapazität* und zur *Gesamtkapazität* anzeigen.

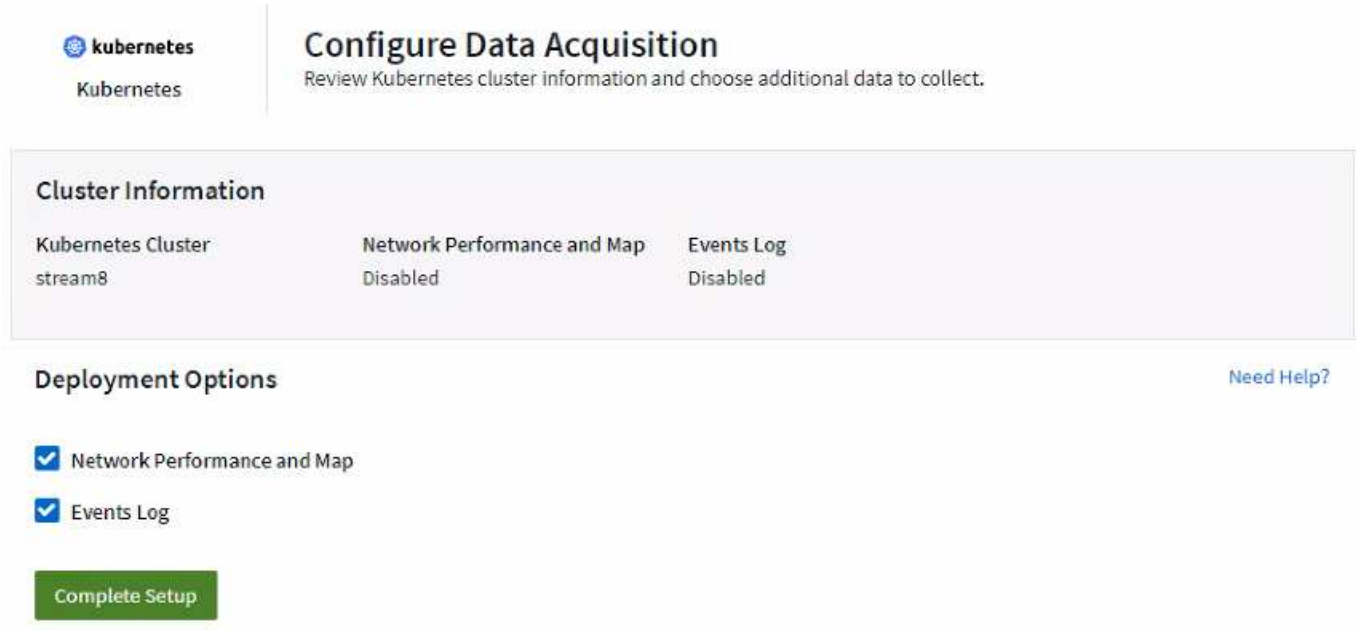
Überwachung und Zuordnung der Kubernetes-Netzwerkleistung

Die Funktion „Netzwerkleistungsüberwachung und -zuordnung“ von Kubernetes vereinfacht die Fehlerbehebung durch die Zuordnung von Abhängigkeiten zwischen Diensten (auch Workloads genannt) und bietet Echtzeit-Einblicke in Latenzen und Anomalien der Netzwerkleistung, um Leistungsprobleme zu erkennen, bevor sie sich auf Benutzer auswirken. Diese Funktion hilft Unternehmen, die Gesamtkosten durch die Analyse und Prüfung von Kubernetes-Verkehrsflüssen zu senken.

Hauptfunktionen: • Die Workload Map stellt Kubernetes-Workload-Abhängigkeiten und -Flows dar und hebt Netzwerk- und Leistungsprobleme hervor. • Überwachen Sie den Netzwerkverkehr zwischen Kubernetes-Pods, Workloads und Knoten; identifizieren Sie die Quelle von Verkehrs- und Latenzproblemen. • Reduzieren Sie die Gesamtkosten, indem Sie den eingehenden, ausgehenden, regions- und zonenübergreifenden Netzwerkverkehr analysieren.

Voraussetzungen

Bevor Sie die Kubernetes-Netzwerkleistungsüberwachung und -karte verwenden können, müssen Sie Folgendes konfiguriert haben:"NetApp Kubernetes Monitoring Operator" um diese Option zu aktivieren. Aktivieren Sie während der Bereitstellung des Operators das Kontrollkästchen „Netzwerkleistung und Karte“, um die Funktion zu aktivieren. Sie können diese Option auch aktivieren, indem Sie zu einer Kubernetes-Landingpage navigieren und „Bereitstellung ändern“ auswählen.



Monitore

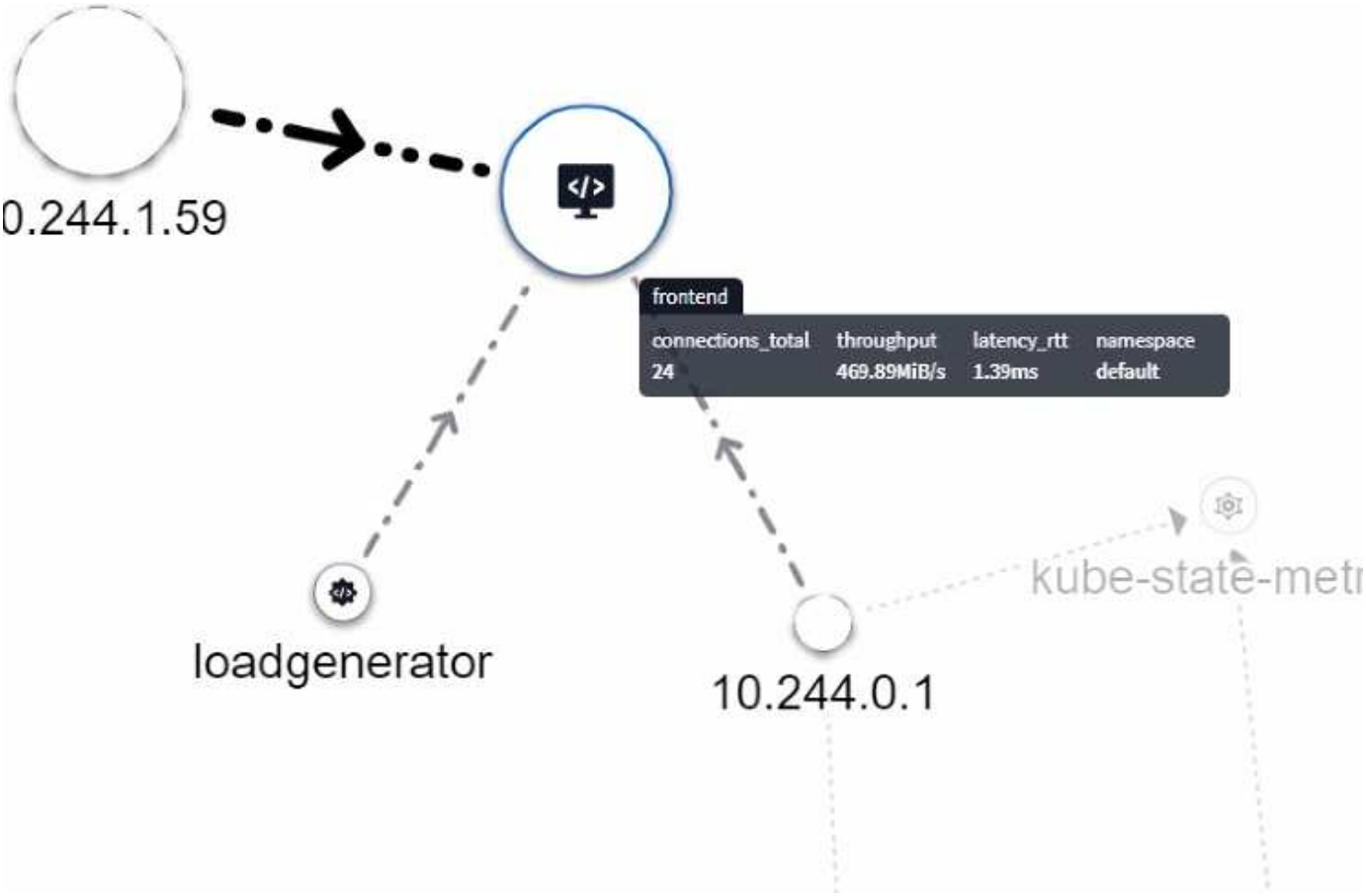
Die Workload Map verwendet"Monitore" um Informationen abzuleiten. Data Infrastructure Insights bietet eine Reihe von standardmäßigen Kubernetes-Monitoren (beachten Sie, dass diese standardmäßig *angehalten* sein können). Sie können die gewünschten Monitore *fortsetzen* (d. h. aktivieren) oder benutzerdefinierte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

Sie können Data Infrastructure Insights -Metrikwarnungen für alle der folgenden Objekttypen erstellen. Stellen Sie sicher, dass die Daten nach dem Standardobjekttyp gruppiert sind.

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

Die Karte

Die Karte zeigt Dienste/Workloads und ihre Beziehungen zueinander. Pfeile zeigen die Verkehrsrichtung an. Wenn Sie mit der Maus über eine Arbeitslast fahren, werden zusammenfassende Informationen zu dieser Arbeitslast angezeigt, wie Sie in diesem Beispiel sehen können:



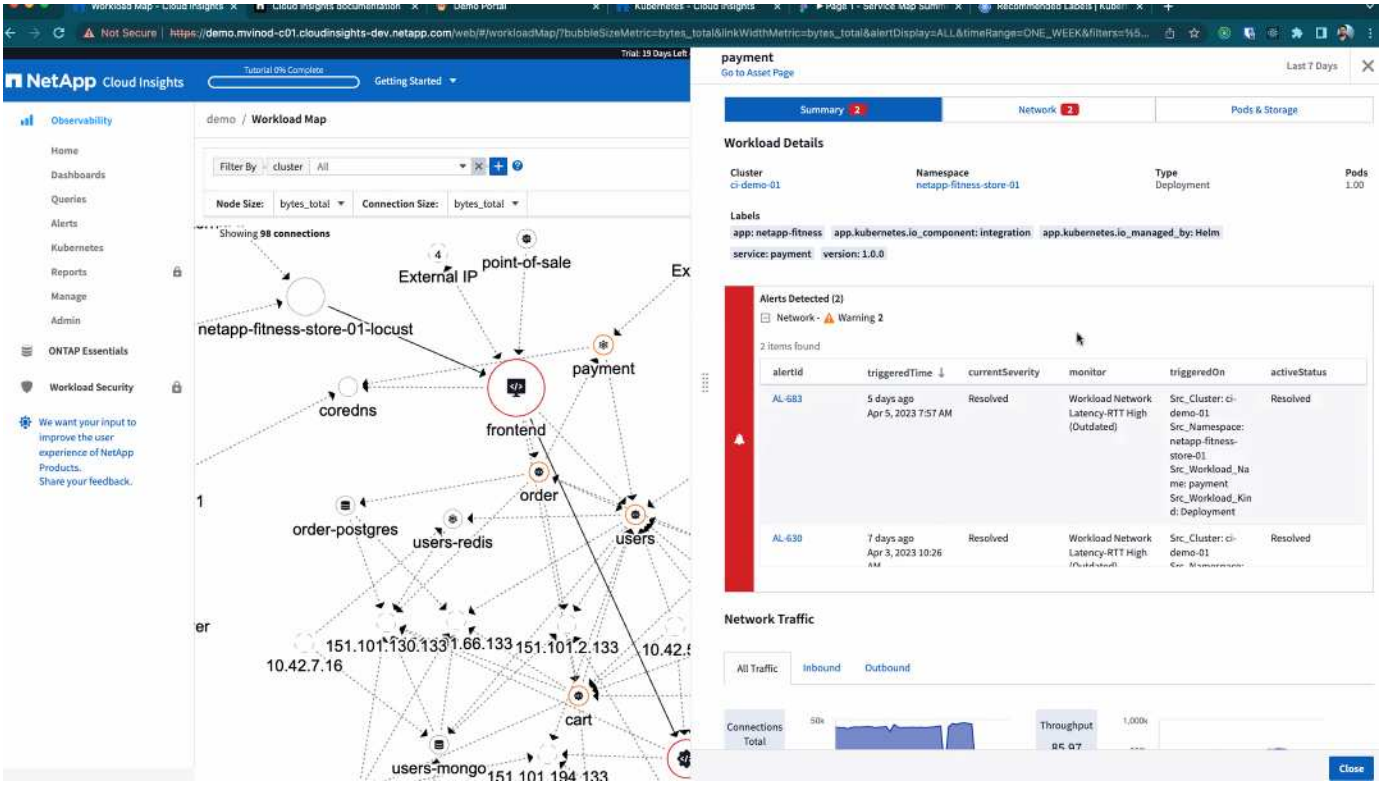
Die Symbole in den Kreisen stellen unterschiedliche Servicetypen dar. Beachten Sie, dass Symbole nur sichtbar sind, wenn die darunter liegenden ObjekteEtiketten .

Die Größe jedes Kreises gibt die Knotengröße an. Beachten Sie, dass diese Größen relativ sind. Die Zoomstufe Ihres Browsers oder die Bildschirmgröße können die tatsächliche Kreisgröße beeinflussen. Auf die gleiche Weise bietet Ihnen der Verkehrslinienstil einen Überblick über die Verbindungsgröße. Fette durchgezogene Linien stehen für hohen Verkehr, während hell gepunktete Linien für geringeren Verkehr stehen.

Die Zahlen in den Kreisen geben die Anzahl der externen Verbindungen an, die derzeit vom Dienst verarbeitet werden.

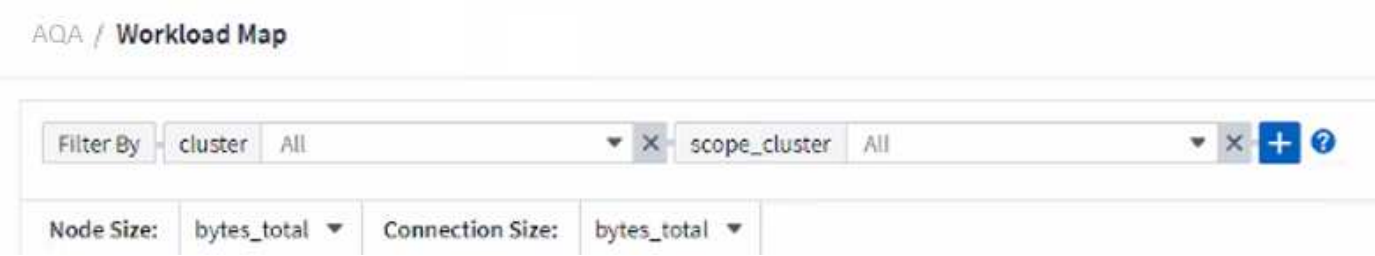
Arbeitslastdetails und Warnungen

Farbig angezeigte Kreise weisen auf eine Warnung oder einen Alarm der kritischen Stufe für die Arbeitslast hin. Bewegen Sie den Mauszeiger über den Kreis, um eine Zusammenfassung des Problems anzuzeigen, oder klicken Sie auf den Kreis, um ein Slideout-Fenster mit weiteren Details zu öffnen.

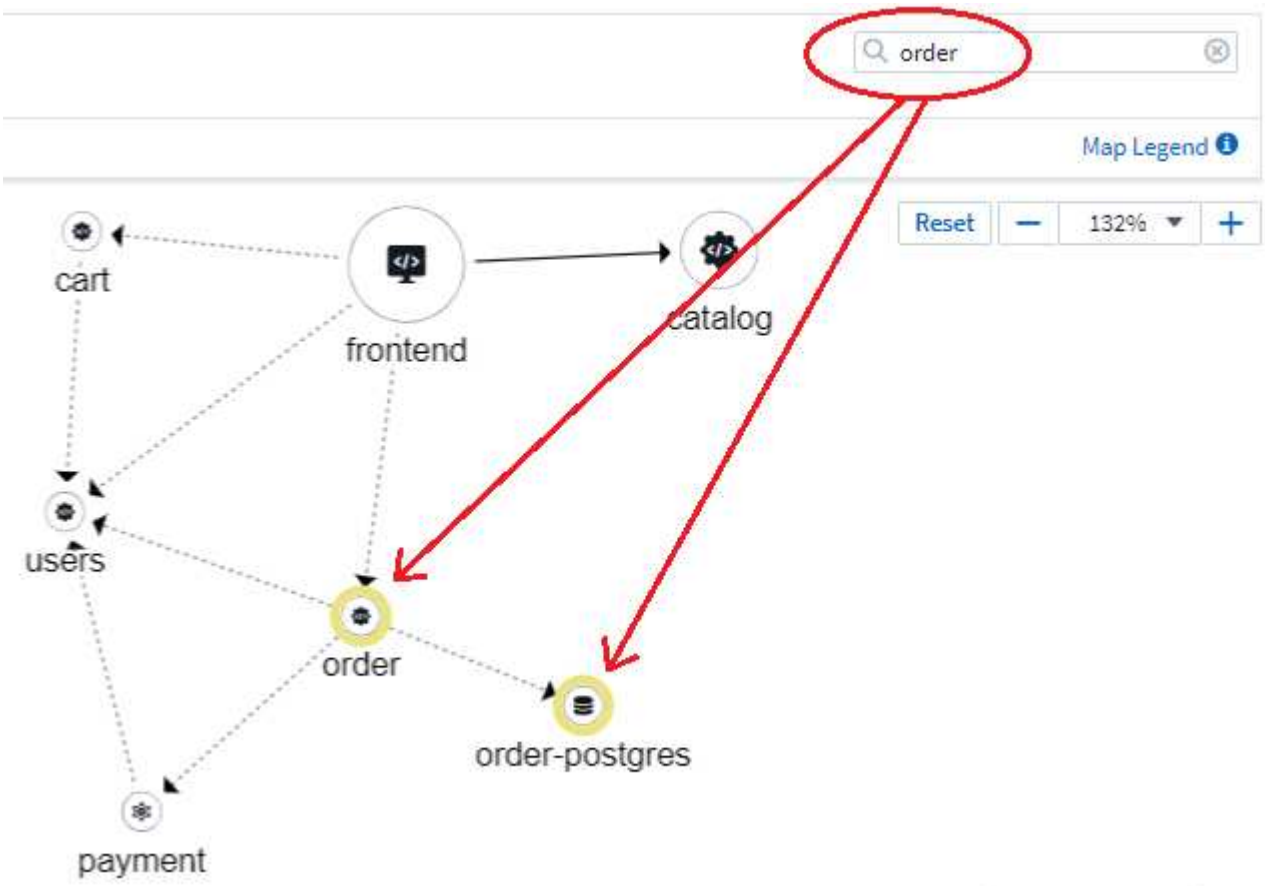


Suchen und Filtern

Wie bei anderen Data Infrastructure Insights Funktionen können Sie ganz einfach Filter festlegen, um sich auf die gewünschten spezifischen Objekte oder Workload-Attribute zu konzentrieren.



Ebenso werden durch die Eingabe einer Zeichenfolge in das Feld *Suchen* passende Workloads hervorgehoben.



Arbeitslastbezeichnungen

Arbeitslastbeschriftungen sind erforderlich, wenn die Karte die angezeigten Arbeitslasttypen identifizieren soll (d. h. die Kreissymbole). Die Bezeichnungen werden wie folgt abgeleitet:

- Name des ausgeführten Dienstes/der ausgeführten Anwendung in allgemeinen Begriffen
- Wenn die Quelle ein Pod ist:
  - Das Label wird vom Workload-Label des Pods abgeleitet
  - Erwartete Bezeichnung für die Arbeitslast: app.kubernetes.io/component
  - Referenz des Etikettennamens: https://kubernetes.io/docs/concepts/overview/working-with-objects/common-labels/
  - Empfohlene Etiketten:
    - Frontend

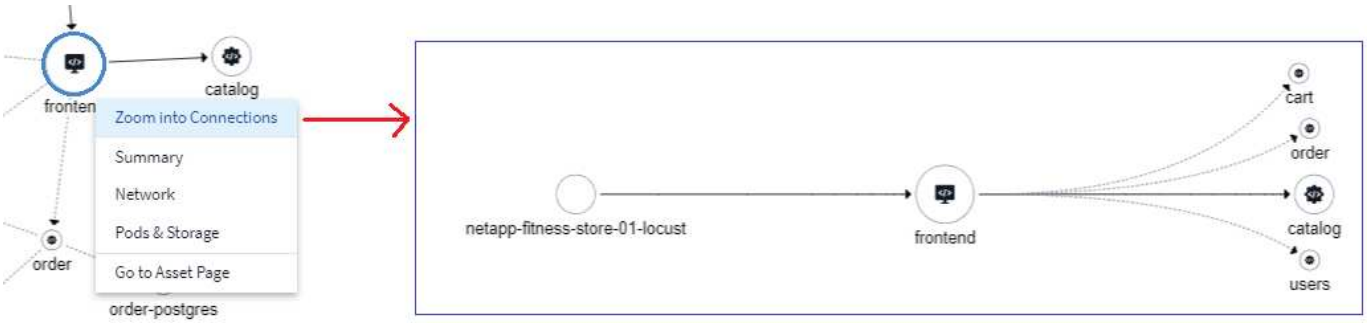
- Backend
- Datenbank
- Cache
- Warteschlange
- Kafka

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

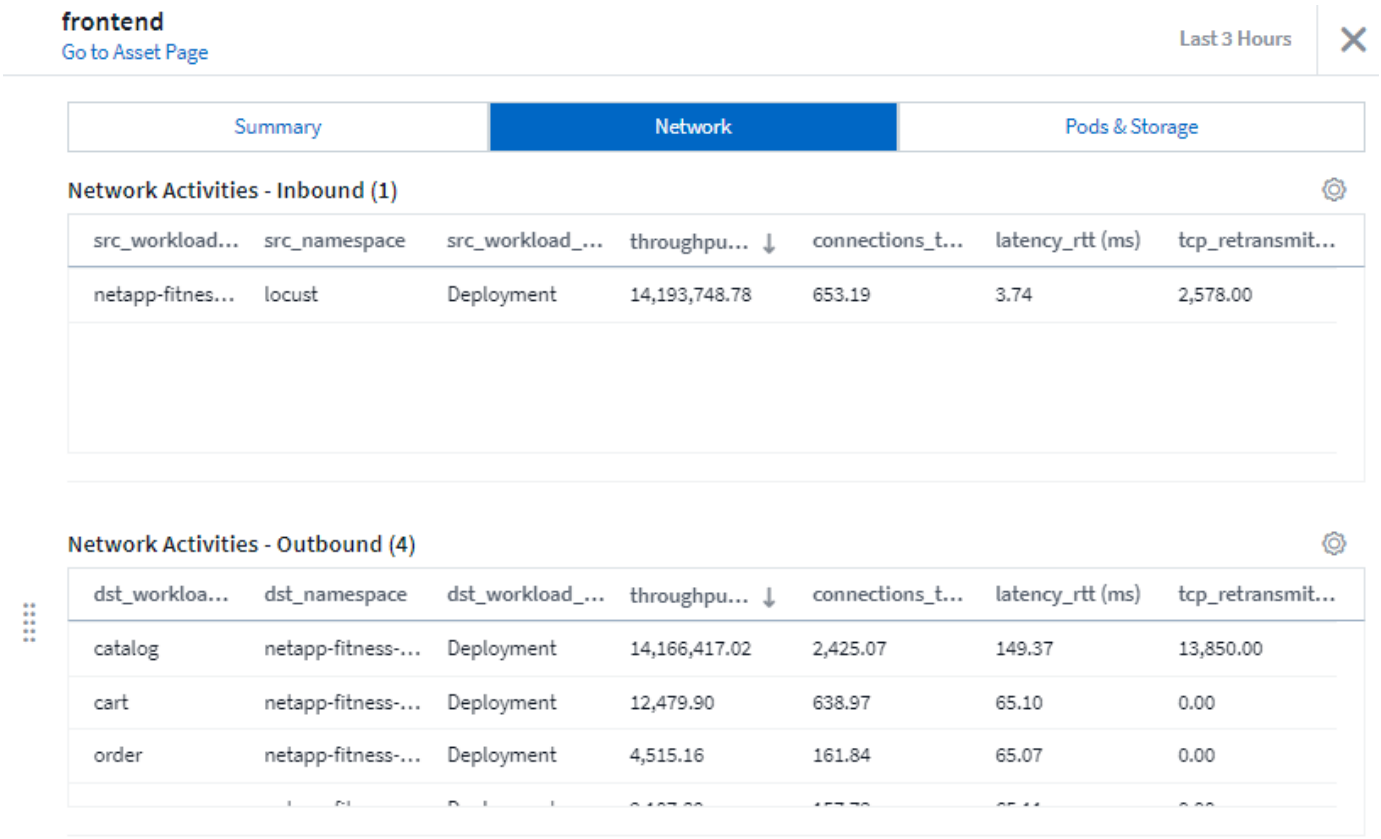
    Beispielsweise wird bei einem DNS-aufgelösten Namen von *s3.eu-north-1.amazonaws.com* der aufgelöste Name analysiert, um *s3* als Diensttyp zu erhalten.

Tauchen Sie tief ein

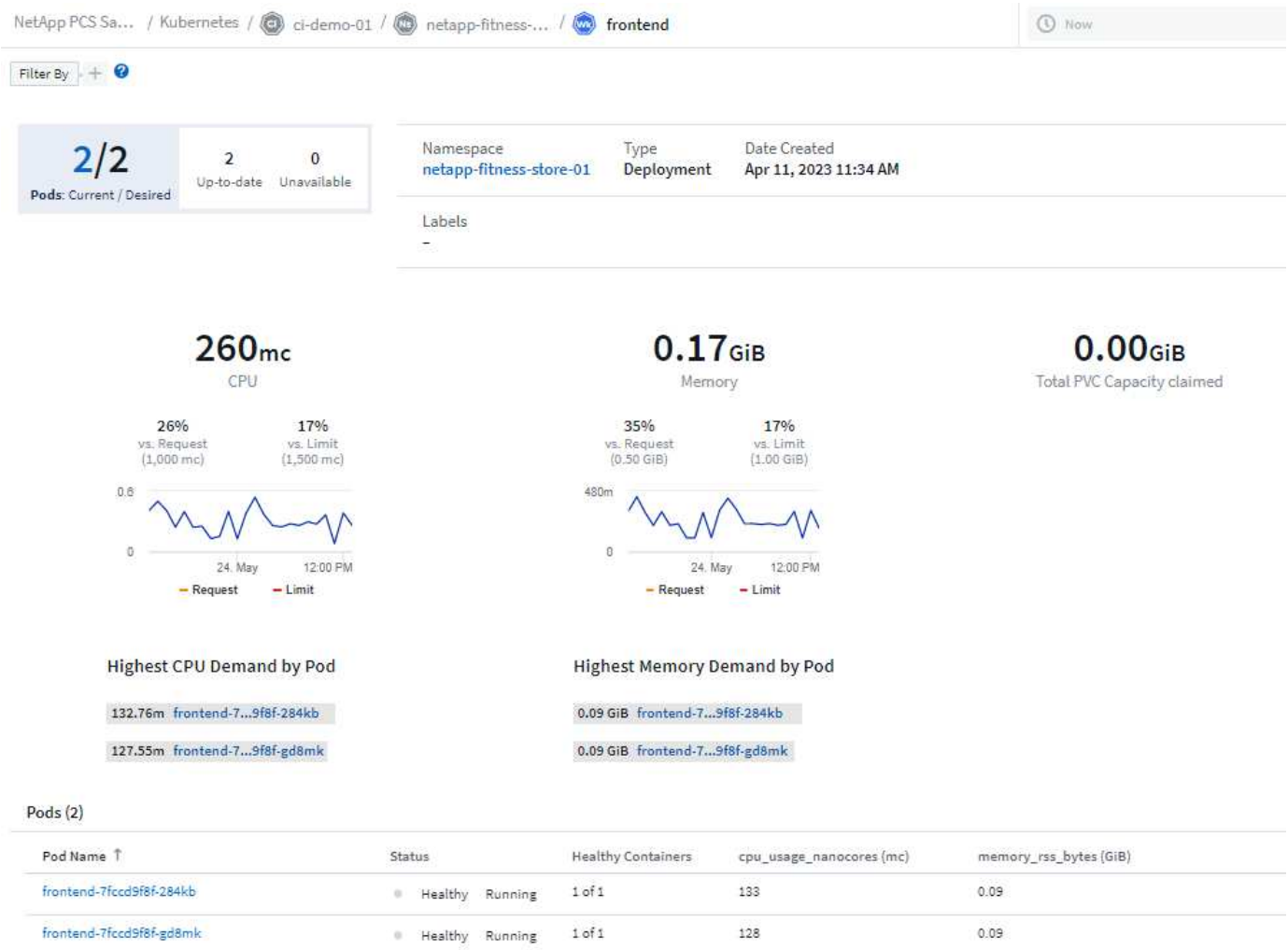
Wenn Sie mit der rechten Maustaste auf eine Arbeitslast klicken, werden Ihnen zusätzliche Optionen zur weiteren Erkundung angezeigt. Von hier aus können Sie beispielsweise hineinzoomen, um die Verbindungen für diese Arbeitslast anzuzeigen.



Oder Sie können das Detail-Slideout-Panel öffnen, um die Registerkarten „Zusammenfassung“, „Netzwerk“ oder „Pod & Speicher“ direkt anzuzeigen.



Wenn Sie abschließend „Zur Asset-Seite gehen“ auswählen, wird die detaillierte Asset-Landingpage für die Arbeitslast geöffnet.



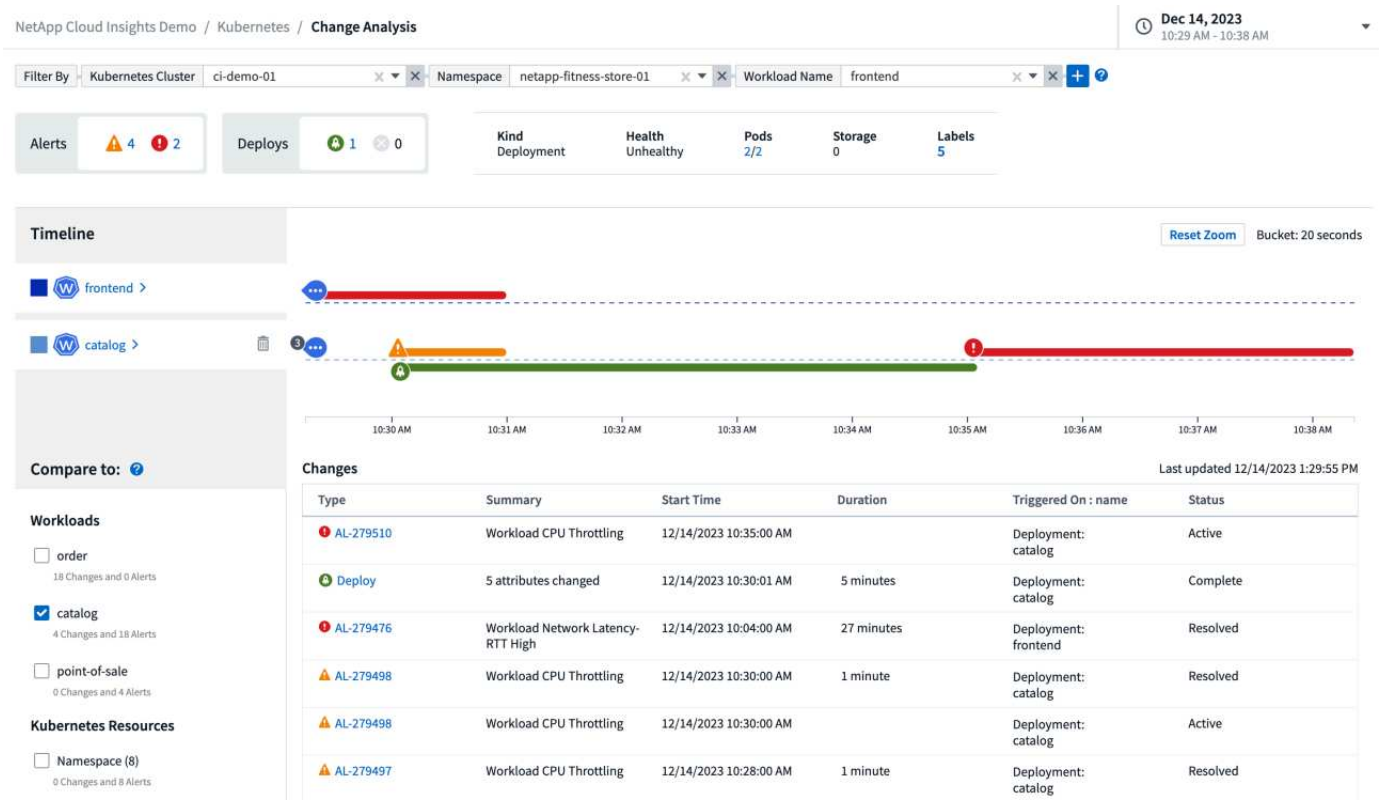
Kubernetes-Änderungsanalyse

Kubernetes Change Analytics bietet Ihnen eine umfassende Ansicht der jüngsten Änderungen an Ihrer K8s-Umgebung. Sie haben jederzeit Zugriff auf Warnmeldungen und den Bereitstellungsstatus. Mit Change Analytics können Sie jede Bereitstellungs- und Konfigurationsänderung verfolgen und mit dem Zustand und der Leistung der Dienste, Infrastruktur und Cluster von K8s korrelieren.

Wie hilft die Änderungsanalyse?

- In Kubernetes-Umgebungen mit mehreren Mandanten kann es aufgrund falsch konfigurierter Änderungen zu Ausfällen kommen. Change Analytics hilft dabei, indem es einen einzigen Bereich bereitstellt, in dem der Zustand von Workloads und Konfigurationsänderungen angezeigt und korreliert werden kann. Dies kann bei der Fehlerbehebung in dynamischen Kubernetes-Umgebungen hilfreich sein.

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



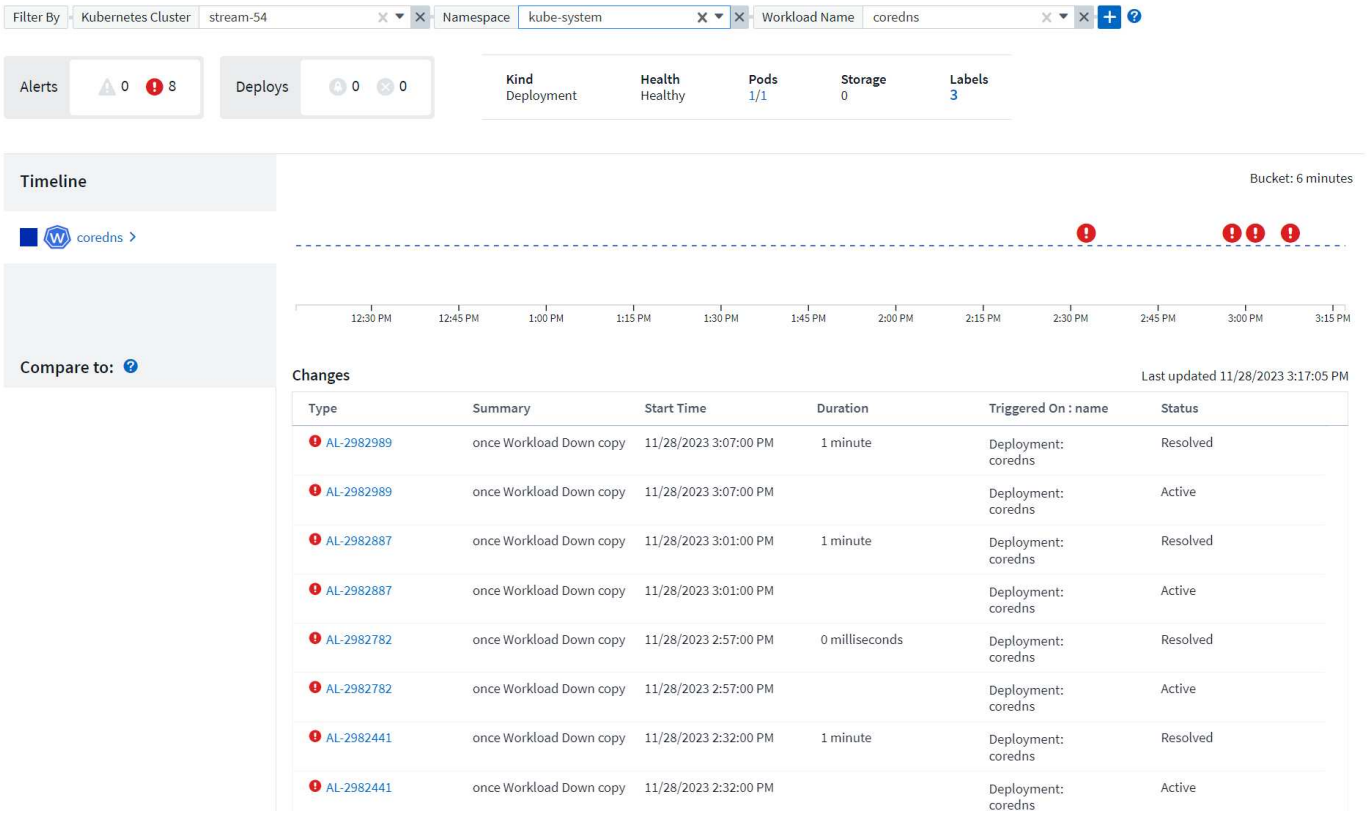
.

Die Seite wird automatisch basierend auf dem aktuell ausgewählten Data Infrastructure Insights -Zeitraum aktualisiert. Kleinere Zeiträume bedeuten eine häufigere Bildschirmaktualisierung.

Filtern

Wie bei allen Funktionen von Data Infrastructure Insights ist das Filtern der Änderungsliste intuitiv: Geben Sie oben auf der Seite Werte für Ihren Kubernetes-Cluster, Namespace oder Workload ein oder wählen Sie diese aus oder fügen Sie Ihre eigenen Filter hinzu, indem Sie die Schaltfläche {+] auswählen.

Wenn Sie auf einen bestimmten Cluster, Namespace und Workload (zusammen mit allen anderen von Ihnen festgelegten Filtern) filtern, wird Ihnen eine Zeitleiste mit Bereitstellungen und Warnungen für diesen Workload in diesem Namespace auf diesem Cluster angezeigt. Zoomen Sie weiter hinein, indem Sie im Diagramm klicken und ziehen, um sich auf einen spezifischeren Zeitbereich zu konzentrieren.

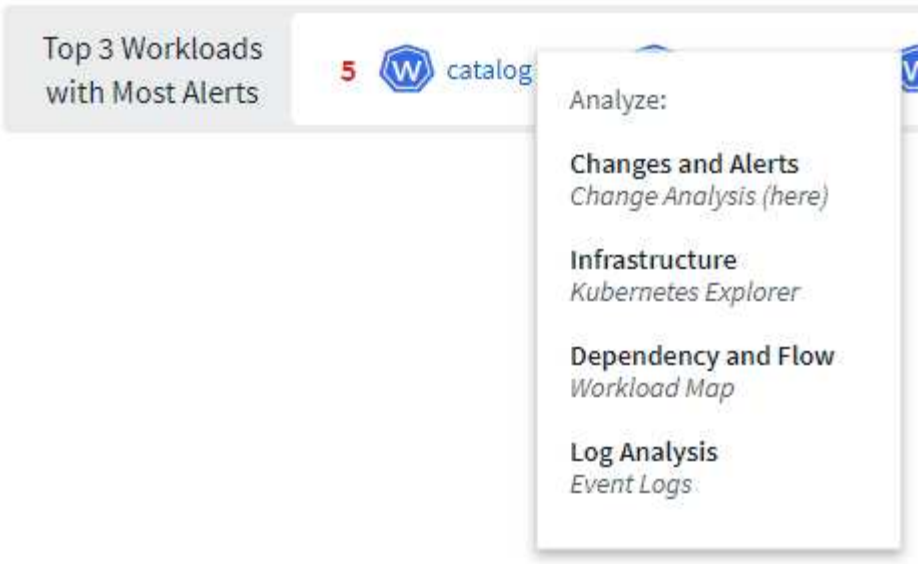


Schnellstatus

Unterhalb des Filterbereichs befinden sich eine Reihe von Indikatoren auf hoher Ebene. Links steht die Anzahl der Alarme (Warnung und Kritisch). Diese Zahl umfasst sowohl *aktive* als auch *gelöste* Warnungen. Um nur *aktive* Warnungen anzuzeigen, legen Sie einen Filter für „Status“ fest und wählen Sie „Aktiv“.

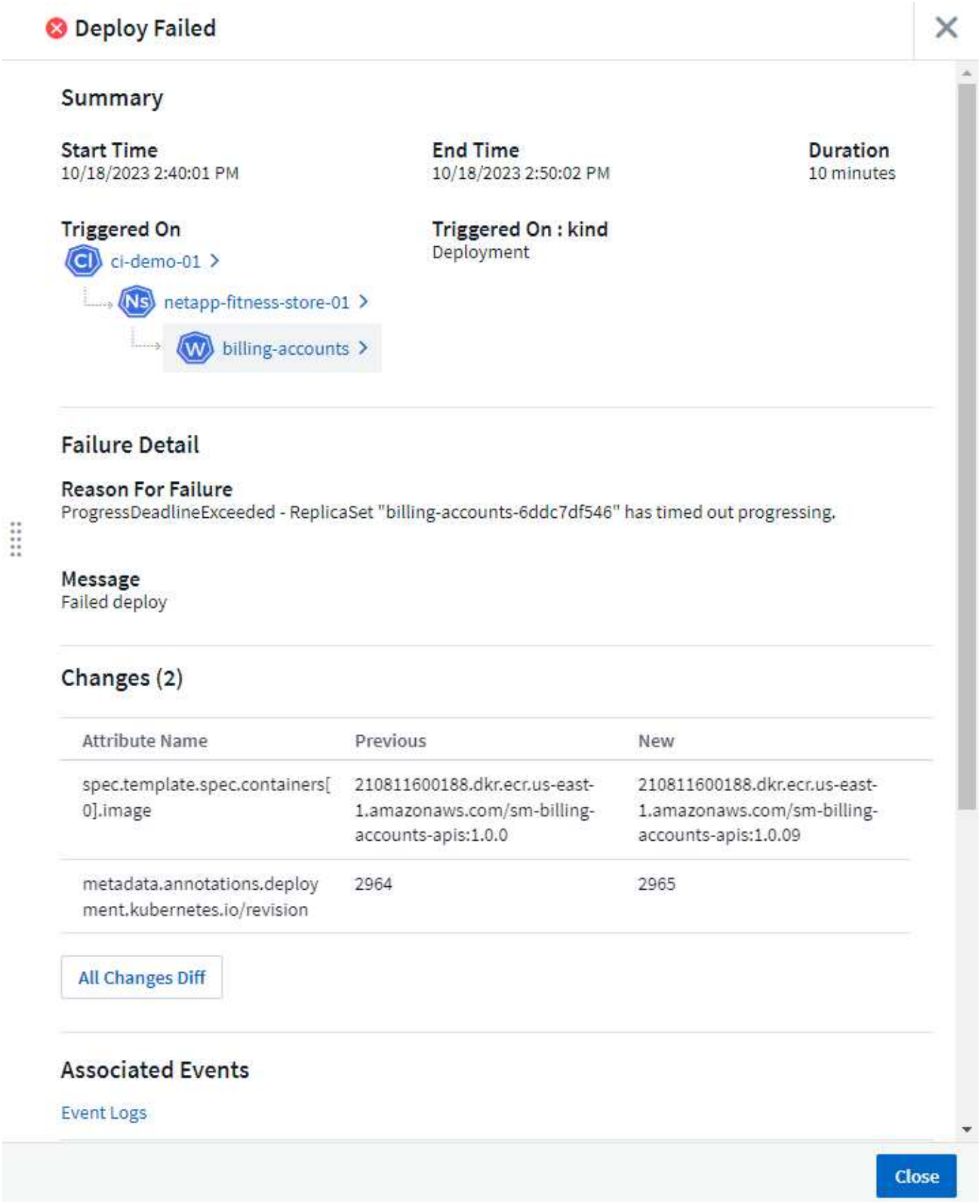
Der Bereitstellungsstatus wird hier ebenfalls angezeigt. Auch hier wird standardmäßig die Anzahl der gestarteten, abgeschlossenen und fehlgeschlagenen Bereitstellungen angezeigt. Um nur *fehlgeschlagene* Bereitstellungen anzuzeigen, legen Sie einen Filter für „Status“ fest und wählen Sie „Fehlgeschlagen“ aus.

Als nächstes folgen die drei Workloads mit den meisten Warnungen. Die rote Zahl neben jeder Arbeitslast gibt die Anzahl der Warnungen an, die mit dieser Arbeitslast in Zusammenhang stehen. Klicken Sie auf den Workload-Link, um Ihre Infrastruktur (Kubernetes Explorer), Abhängigkeiten (Workload Map) oder Protokollanalyse (Ereignisprotokolle) zu erkunden.



Detailbereich

Wenn Sie eine Änderung in der Liste auswählen, wird ein Fenster geöffnet, in dem die Änderung ausführlicher beschrieben wird. Wenn Sie beispielsweise eine fehlgeschlagene Bereitstellung auswählen, wird eine Zusammenfassung der Bereitstellung mit Start- und Endzeit, Dauer und Auslöseort der Bereitstellung sowie Links zum Durchsuchen dieser Ressourcen angezeigt. Außerdem werden der Grund für den Fehler, alle damit verbundenen Änderungen und alle zugehörigen Ereignisse angezeigt.



Durch Auswahl einer Warnung werden ebenfalls Details zur Warnung bereitgestellt, darunter der Monitor, der die Warnung ausgelöst hat, sowie ein Diagramm mit einer visuellen Zeitleiste für die Warnung.