Hinzufügen eines selbstverwalteten Clusters

Astra Control Service

NetApp
April 24, 2024

This PDF was generated from https://docs.netapp.com/de-de/astra-control-service/get-started/add-public-self-managed-cluster.html on April 24, 2024. Always check docs.netapp.com for the latest.

Inhalt

Hinzufügen eines selbstverwalteten Clusters

Fügen Sie Astra Control Service einen öffentlichen, selbst gemanagten Cluster hinzu

Nach der Einrichtung der Umgebung erstellen Sie sofort einen Kubernetes Cluster und fügen ihn dann dem Astra Control Service hinzu.

Ein selbstverwalteter Cluster ist ein Cluster, den Sie direkt bereitstellen und managen können. Astra Control Service unterstützt selbst gemanagte Cluster, die in einer Public-Cloud-Umgebung ausgeführt werden. Sie können einen selbstverwalteten Cluster zum Astra Control Service hinzufügen, indem Sie ein hochladen `kubeconfig.yaml` Datei: Sie müssen sicherstellen, dass das Cluster die hier aufgeführten Anforderungen erfüllt.

Unterstützte Kubernetes-Distributionen

Mit Astra Control Service können Sie folgende Arten von öffentlichen, selbst gemanagten Clustern managen:

Kubernetes-Distribution	Unterstützte Versionen
Kubernetes (Vorgelagert)	1.27 bis 1.29
Rancher Kubernetes Engine (RKE)	RKE 1: Versionen 1.24.17, 1.25.13, 1.26.8 mit Rancher Manager 2.7.9 RKE 2: Versionen 1.23.16 und 1.24.13 mit Rancher Manager 2.6.13 RKE 2: Versionen 1.24.17, 1.25.14, 1.26.9 mit Rancher Manager 2.7.9
Red hat OpenShift Container Platform	4.12 bis 4.14

Bei diesen Anweisungen wird davon ausgegangen, dass Sie bereits einen selbstverwalteten Cluster erstellt haben.

- Fügen Sie das Cluster zu Astra Control Service hinzu
- Ändern der Standard-Storage-Klasse

Fügen Sie das Cluster zu Astra Control Service hinzu

Nachdem Sie sich beim Astra Control Service angemeldet haben, beginnen Sie zunächst mit dem Verwalten Ihrer Cluster. Bevor Sie Astra Control Service ein Cluster hinzufügen, müssen Sie bestimmte Aufgaben ausführen und sicherstellen, dass das Cluster bestimmte Anforderungen erfüllt.

Bevor Sie beginnen

Ein selbstverwalteter Cluster ist ein Cluster, den Sie direkt bereitstellen und managen können. Astra Control Service unterstützt selbst gemanagte Cluster, die in einer Public-Cloud-Umgebung ausgeführt werden. Die selbstverwalteten Cluster können über Astra Control Provisioner eine Schnittstelle zu NetApp Storage-Services aufbauen. Alternativ können sie über CSI-Treiber (Container Storage Interface) eine Schnittstelle zu Amazon Elastic Block Store (EBS), Azure Managed Disks und Google Persistent Disk erstellen.

Astra Control Service unterstützt selbst gemanagte Cluster, die die folgenden Kubernetes-Distributionen verwenden:

- Red hat OpenShift Container Platform
- Rancher Kubernetes Engine
- Vorgelagerte Kubernetes-Systeme

Ihr Self-Managed-Cluster muss folgende Anforderungen erfüllen:

- Der Cluster muss über das Internet zugänglich sein.
- Wenn Sie Speicher mit CSI-Treibern verwenden oder planen, diese zu verwenden, müssen auf dem Cluster die entsprechenden CSI-Treiber installiert sein. Weitere Informationen zur Verwendung von CSI-Treibern zur Integration von Speicher finden Sie in der Dokumentation Ihres Speicherservices.
- Sie haben Zugriff auf die Cluster-Datei kubeconfig, die nur ein Kontextelement enthält. Folgen "Diese Anweisungen" Um eine kubeconfig-Datei zu erzeugen.
- Wenn Sie den Cluster mit einer kubeconfig-Datei hinzufügen, die auf eine private Zertifizierungsstelle verweist, fügen Sie der folgende Zeile hinzu `cluster` Abschnitt der Datei kubeconfig. So kann Astra Control das Cluster hinzufügen:

  ```
  insecure-skip-tls-verify: true
  ```

- **Rancher only**: Ändern Sie beim Verwalten von Anwendungsclustern in einer Rancher-Umgebung den Standardkontext des Anwendungsclusters in der von Rancher bereitgestellten kubeconfig-Datei, um einen Steuerebenen-Kontext anstelle des Rancher API-Serverkontexts zu verwenden. So wird die Last auf dem Rancher API Server reduziert und die Performance verbessert.
- **Anforderungen für die Astra Control-Bereitstellung**: Sie sollten einen ordnungsgemäß konfigurierten Astra Control Provisioner einschließlich der Astra Trident-Komponenten verwenden, um Cluster zu managen.
  - **Umgebungsanforderungen für Astra Trident prüfen**: Lesen Sie vor der Installation oder dem Upgrade von Astra Control Provisioner die "Unterstützte Frontends, Back-Ends und Host-Konfigurationen".
  - **Astra Control-Provisioner aktivieren**: Es wird dringend empfohlen, Astra Trident 23.10 oder höher zu installieren und zu aktivieren "Astra Control bietet erweiterte Storage-Funktionen zur Bereitstellung". In den kommenden Versionen unterstützt Astra Control nicht Astra Trident, wenn der Astra Control Provisioner nicht ebenfalls aktiviert ist.
  - **Konfiguration eines Speicher-Backends**: Mindestens ein Speicher-Backend muss sein "In Astra Trident konfiguriert" Auf dem Cluster.
  - **Konfiguration einer Storage-Klasse**: Mindestens eine Storage-Klasse muss sein "In Astra Trident konfiguriert" Auf dem Cluster. Wenn eine Standardspeicherklasse konfiguriert ist, stellen

Sie sicher, dass sie die **einzige** Speicherklasse ist, die die Standardanmerkung hat.

- **Konfigurieren Sie einen Volume-Snapshot-Controller und installieren Sie eine Volume-Snapshot-Klasse**: "Installieren Sie einen Volume-Snapshot-Controller" Damit Snapshots in Astra Control erstellt werden können. "Erstellen" Mindestens eine `VolumeSnapshotClass` Einsatz von Astra Trident:

Schritte

1. Wählen Sie im Dashboard **Kubernetes Cluster managen** aus.

   Befolgen Sie die Aufforderungen zum Hinzufügen des Clusters.

2. **Provider**: Wählen Sie den Reiter **andere**, um Details zu Ihrem selbst verwalteten Cluster hinzuzufügen.
   a. **Other**: Geben Sie Details über Ihren selbstverwalteten Cluster durch das Hochladen eines `kubeconfig.yaml` Datei oder durch Einfügen des Inhalts des `kubeconfig.yaml` Datei aus der Zwischenablage.

   > Wenn Sie Ihre eigenen erstellen `kubeconfig` Datei, Sie sollten nur ein **ein**-Kontext -Element darin definieren. Siehe "Kubernetes-Dokumentation" Weitere Informationen zum Erstellen `kubeconfig` Dateien:

3. **Credential Name**: Geben Sie einen Namen für die selbstverwalteten Cluster-Zugangsdaten ein, die Sie auf Astra Control hochladen. Standardmäßig wird der Name der Anmeldeinformationen automatisch als Name des Clusters ausgefüllt.
4. **Private Route Identifier**: Dieses Feld ist nur für private Cluster bestimmt.
5. Wählen Sie **Weiter**.
6. (Optional) **Speicher**: Wählen Sie optional die Storage-Klasse aus, die Kubernetes-Anwendungen, die auf diesem Cluster bereitgestellt werden sollen, standardmäßig verwenden sollen.
   a. Um eine neue Standard-Storage-Klasse für den Cluster auszuwählen, aktivieren Sie das Kontrollkästchen **Neue Standard-Storage-Klasse zuweisen**.
   b. Wählen Sie eine neue Standard-Storage-Klasse aus der Liste aus.

Jeder Storage-Service eines Cloud-Providers enthält die folgenden Informationen zu Preis, Performance und Ausfallsicherheit:

- Cloud Volumes Service für Google Cloud: Informationen zu Preis, Performance und Ausfallsicherheit
- Google Persistent Disk: Keine Informationen über Preis, Performance oder Ausfallsicherheit verfügbar
- Azure NetApp Files: Informationen zu Performance und Ausfallsicherheit
- Azure Managed Disks: Es sind weder Preis-, Performance- oder Resilience-Informationen verfügbar
- Amazon Elastic Block Store: Keine Informationen zu Preis, Performance oder Ausfallsicherheit verfügbar
- Amazon FSX für NetApp ONTAP: Keine Informationen zu Preis, Performance und Ausfallsicherheit verfügbar
- NetApp Cloud Volumes ONTAP: Keine Informationen zu Preis, Performance oder Ausfallsicherheit verfügbar

Jede Storage-Klasse kann einen der folgenden Services nutzen:

- "Cloud Volumes Service für Google Cloud"
- "Google Persistent Disk"
  - "Azure NetApp Dateien"
  - "Von Azure gemanagte Festplatten"
  - "Amazon Elastic Block Store"
  - "Amazon FSX für NetApp ONTAP"
  - "NetApp Cloud Volumes ONTAP"

  Weitere Informationen zu "Storage-Klassen für Amazon Web Services Cluster". Weitere Informationen zu "Speicherklassen für AKS-Cluster". Weitere Informationen zu "Speicherklassen für GKE-Cluster".

c. Wählen Sie **Weiter**.

d. **Überprüfen und genehmigen**: Überprüfen Sie die Konfigurationsdetails.

e. Wählen Sie **Add**, um den Cluster zu Astra Control Service hinzuzufügen.

Ändern der Standard-Storage-Klasse

Sie können die Standard-Storage-Klasse für ein Cluster ändern.

Ändern Sie die Standard-Storage-Klasse mit Astra Control

Sie können die Standard-Storage-Klasse für ein Cluster aus Astra Control ändern. Wenn Ihr Cluster einen zuvor installierten Speicher-Backend-Service verwendet, können Sie diese Methode möglicherweise nicht verwenden, um die Standard-Speicherklasse zu ändern (die Aktion **default** ist nicht wählbar). In diesem Fall können Sie Ändern Sie die Standard-Storage-Klasse über die Befehlszeile.

Schritte

1. Wählen Sie in der Astra Control Service-UI **Cluster** aus.
2. Wählen Sie auf der Seite **Cluster** den Cluster aus, den Sie ändern möchten.
3. Wählen Sie die Registerkarte **Storage** aus.
4. Wählen Sie die Kategorie **Speicherklassen** aus.
5. Wählen Sie das Menü **Aktionen** für die Speicherklasse aus, die Sie als Standard festlegen möchten.
6. Wählen Sie **als Standard**.

Ändern Sie die Standard-Storage-Klasse über die Befehlszeile

Sie können die Standard-Storage-Klasse für ein Cluster mit Kubernetes-Befehlen ändern. Diese Methode funktioniert unabhängig von der Konfiguration Ihres Clusters.

Schritte

1. Melden Sie sich bei Ihrem Kubernetes Cluster an.
2. Listen Sie die Storage-Klassen in Ihrem Cluster auf:

```
kubectl get storageclass
```

3. Entfernen Sie die Standardbezeichnung aus der Standardspeicherklasse. Ersetzen Sie <SC_NAME> durch den Namen der Speicherklasse:

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-
class":"false"}}}'
```

4. Markieren Sie standardmäßig eine andere Storage-Klasse. Ersetzen Sie <SC_NAME> durch den Namen der Speicherklasse:

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

5. Bestätigen Sie die neue Standard-Speicherklasse:

```
kubectl get storageclass
```

Fügen Sie Astra Control Service einen privaten, selbst gemanagten Cluster hinzu

Nach der Einrichtung der Umgebung erstellen Sie sofort einen Kubernetes Cluster und fügen ihn dann dem Astra Control Service hinzu.

Ein selbstverwalteter Cluster ist ein Cluster, den Sie direkt bereitstellen und managen können. Astra Control

Service unterstützt selbst gemanagte Cluster, die in einer Public-Cloud-Umgebung ausgeführt werden. Sie können einen selbstverwalteten Cluster zum Astra Control Service hinzufügen, indem Sie ein hochladen `kubeconfig.yaml` Datei: Sie müssen sicherstellen, dass das Cluster die hier aufgeführten Anforderungen erfüllt.

Unterstützte Kubernetes-Distributionen

Mit Astra Control Service können Sie folgende Arten von privaten, selbst gemanagten Clustern managen:

Kubernetes-Distribution	Unterstützte Versionen
Kubernetes (Vorgelagert)	1.27 bis 1.29
Rancher Kubernetes Engine (RKE)	RKE 1: Versionen 1.24.17, 1.25.13, 1.26.8 mit Rancher Manager 2.7.9 RKE 2: Versionen 1.23.16 und 1.24.13 mit Rancher Manager 2.6.13 RKE 2: Versionen 1.24.17, 1.25.14, 1.26.9 mit Rancher Manager 2.7.9
Red hat OpenShift Container Platform	4.12 bis 4.14

In diesen Anweisungen wird davon ausgegangen, dass Sie bereits einen privaten Cluster erstellt und eine sichere Methode für den Remote-Zugriff darauf vorbereitet haben.

Führen Sie die folgenden Aufgaben aus, um Ihren privaten Cluster zum Astra Control Service hinzuzufügen:

1. Astra Connector Installieren
2. Einrichtung von persistentem Storage
3. selbst gemanagten Cluster zum Astra Control Service hinzu

Astra Connector Installieren

Bevor Sie einen privaten Cluster hinzufügen, müssen Sie Astra Connector im Cluster installieren, damit Astra Control damit kommunizieren kann. Siehe "Installieren Sie die vorherige Version von Astra Connector für private Cluster, die mit nicht-Kubernetes-nativen Workflows gemanagt werden" Weitere Anweisungen.

Einrichtung von persistentem Storage

Konfigurieren Sie persistenten Storage für das Cluster. In der Dokumentation „erste Schritte“ finden Sie weitere Informationen zum Konfigurieren von persistentem Storage:

- "Microsoft Azure mit Azure NetApp Files einrichten"
- "Richten Sie Microsoft Azure mit von Azure gemanagten Festplatten ein"
- "Einrichten von Amazon Web Services"
- "Google Cloud einrichten"

Fügen Sie den privaten, selbst gemanagten Cluster zum Astra Control Service hinzu

Sie können den privaten Cluster jetzt dem Astra Control Service hinzufügen.

Bevor Sie beginnen

Ein selbstverwalteter Cluster ist ein Cluster, den Sie direkt bereitstellen und managen können. Astra Control Service unterstützt selbst gemanagte Cluster, die in einer Public-Cloud-Umgebung ausgeführt werden. Die selbstverwalteten Cluster können über Astra Control Provisioner eine Schnittstelle zu NetApp Storage-Services aufbauen. Alternativ können sie über CSI-Treiber (Container Storage Interface) eine Schnittstelle zu Amazon Elastic Block Store (EBS), Azure Managed Disks und Google Persistent Disk erstellen.

Astra Control Service unterstützt selbst gemanagte Cluster, die die folgenden Kubernetes-Distributionen verwenden:

- Red hat OpenShift Container Platform
- Rancher Kubernetes Engine
- Vorgelagerte Kubernetes-Systeme

Ihr Self-Managed-Cluster muss folgende Anforderungen erfüllen:

- Der Cluster muss über das Internet zugänglich sein.
- Wenn Sie Speicher mit CSI-Treibern verwenden oder planen, diese zu verwenden, müssen auf dem Cluster die entsprechenden CSI-Treiber installiert sein. Weitere Informationen zur Verwendung von CSI-Treibern zur Integration von Speicher finden Sie in der Dokumentation Ihres Speicherservices.
- Sie haben Zugriff auf die Cluster-Datei kubeconfig, die nur ein Kontextelement enthält. Folgen "Diese Anweisungen" Um eine kubeconfig-Datei zu erzeugen.
- Wenn Sie den Cluster mit einer kubeconfig-Datei hinzufügen, die auf eine private Zertifizierungsstelle verweist, fügen Sie der folgende Zeile hinzu `cluster` Abschnitt der Datei kubeconfig. So kann Astra Control das Cluster hinzufügen:

  ```
  insecure-skip-tls-verify: true
  ```

- **Rancher only**: Ändern Sie beim Verwalten von Anwendungsclustern in einer Rancher-Umgebung den Standardkontext des Anwendungsclusters in der von Rancher bereitgestellten kubeconfig-Datei, um einen Steuerebenen-Kontext anstelle des Rancher API-Serverkontexts zu verwenden. So wird die Last auf dem Rancher API Server reduziert und die Performance verbessert.
- **Anforderungen für die Astra Control-Bereitstellung**: Sie sollten einen ordnungsgemäß konfigurierten Astra Control Provisioner einschließlich der Astra Trident-Komponenten verwenden, um Cluster zu managen.
  - **Umgebungsanforderungen für Astra Trident prüfen**: Lesen Sie vor der Installation oder dem Upgrade von Astra Control Provisioner die "Unterstützte Frontends, Back-Ends und Host-Konfigurationen".
  - **Astra Control-Provisioner aktivieren**: Es wird dringend empfohlen, Astra Trident 23.10 oder höher zu installieren und zu aktivieren "Astra Control bietet erweiterte Storage-Funktionen zur Bereitstellung". In den kommenden Versionen unterstützt Astra Control nicht Astra Trident, wenn der Astra Control Provisioner nicht ebenfalls aktiviert ist.
  - **Konfiguration eines Speicher-Backends**: Mindestens ein Speicher-Backend muss sein "In Astra Trident konfiguriert" Auf dem Cluster.
  - **Konfiguration einer Storage-Klasse**: Mindestens eine Storage-Klasse muss sein "In Astra Trident konfiguriert" Auf dem Cluster. Wenn eine Standardspeicherklasse konfiguriert ist, stellen

Sie sicher, dass sie die **einzige** Speicherklasse ist, die die Standardanmerkung hat.

- **Konfigurieren Sie einen Volume-Snapshot-Controller und installieren Sie eine Volume-Snapshot-Klasse**: "Installieren Sie einen Volume-Snapshot-Controller" Damit Snapshots in Astra Control erstellt werden können. "Erstellen" Mindestens eine `VolumeSnapshotClass` Einsatz von Astra Trident:

Schritte

1. Wählen Sie im Dashboard **Kubernetes Cluster managen** aus.

   Befolgen Sie die Aufforderungen zum Hinzufügen des Clusters.

2. **Provider**: Wählen Sie den Reiter **andere**, um Details zu Ihrem selbst verwalteten Cluster hinzuzufügen.
3. **Other**: Geben Sie Details über Ihren selbstverwalteten Cluster durch das Hochladen eines `kubeconfig.yaml` Datei oder durch Einfügen des Inhalts des `kubeconfig.yaml` Datei aus der Zwischenablage.

   > Wenn Sie Ihre eigenen erstellen `kubeconfig` Datei, Sie sollten nur ein **ein**-Kontext -Element darin definieren. Siehe "Diese Anweisungen" Weitere Informationen zum Erstellen `kubeconfig` Dateien:

4. **Credential Name**: Geben Sie einen Namen für die selbstverwalteten Cluster-Zugangsdaten ein, die Sie auf Astra Control hochladen. Standardmäßig wird der Name der Anmeldeinformationen automatisch als Name des Clusters ausgefüllt.
5. **Private Route Identifier**: Geben Sie die private Route Identifier ein, die Sie vom Astra Connector erhalten können. Wenn Sie den Astra Connector über die abfragen `kubectl get astraconnector -n astra-connector` Die Kennung der privaten Route wird als bezeichnet `ASTRACONNECTORID`.

   > Die Private-Route-ID ist der Name, der dem Astra Connector zugeordnet ist. Damit kann ein privates Kubernetes-Cluster von Astra gemanagt werden. In diesem Kontext ist ein privates Cluster ein Kubernetes-Cluster, das seinen API-Server nicht zum Internet bereitstellt.

6. Wählen Sie **Weiter**.
7. (Optional) **Speicher**: Wählen Sie optional die Storage-Klasse aus, die Kubernetes-Anwendungen, die auf diesem Cluster bereitgestellt werden sollen, standardmäßig verwenden sollen.
   a. Um eine neue Standard-Storage-Klasse für den Cluster auszuwählen, aktivieren Sie das Kontrollkästchen **Neue Standard-Storage-Klasse zuweisen**.
   b. Wählen Sie eine neue Standard-Storage-Klasse aus der Liste aus.

Jeder Storage-Service eines Cloud-Providers enthält die folgenden Informationen zu Preis, Performance und Ausfallsicherheit:

- Cloud Volumes Service für Google Cloud: Informationen zu Preis, Performance und Ausfallsicherheit
- Google Persistent Disk: Keine Informationen über Preis, Performance oder Ausfallsicherheit verfügbar
- Azure NetApp Files: Informationen zu Performance und Ausfallsicherheit
- Azure Managed Disks: Es sind weder Preis-, Performance- oder Resilience-Informationen verfügbar
- Amazon Elastic Block Store: Keine Informationen zu Preis, Performance oder Ausfallsicherheit verfügbar
- Amazon FSX für NetApp ONTAP: Keine Informationen zu Preis, Performance und Ausfallsicherheit verfügbar
- NetApp Cloud Volumes ONTAP: Keine Informationen zu Preis, Performance oder Ausfallsicherheit verfügbar

Jede Storage-Klasse kann einen der folgenden Services nutzen:

- "Cloud Volumes Service für Google Cloud"
- "Google Persistent Disk"
- "Azure NetApp Dateien"
- "Von Azure gemanagte Festplatten"
- "Amazon Elastic Block Store"
- "Amazon FSX für NetApp ONTAP"
- "NetApp Cloud Volumes ONTAP"

Weitere Informationen zu "Storage-Klassen für Amazon Web Services Cluster". Weitere Informationen zu "Speicherklassen für AKS-Cluster". Weitere Informationen zu "Speicherklassen für GKE-Cluster".

c. Wählen Sie **Weiter**.

d. **Überprüfen und genehmigen**: Überprüfen Sie die Konfigurationsdetails.

e. Wählen Sie **Add**, um den Cluster zu Astra Control Service hinzuzufügen.

Ändern der Standard-Storage-Klasse

Sie können die Standard-Storage-Klasse für ein Cluster ändern.

Ändern Sie die Standard-Storage-Klasse mit Astra Control

Sie können die Standard-Storage-Klasse für ein Cluster aus Astra Control ändern. Wenn Ihr Cluster einen zuvor installierten Speicher-Backend-Service verwendet, können Sie diese Methode möglicherweise nicht verwenden, um die Standard-Speicherklasse zu ändern (die Aktion **default** ist nicht wählbar). In diesem Fall können Sie Ändern Sie die Standard-Storage-Klasse über die Befehlszeile.

Schritte

1. Wählen Sie in der Astra Control Service-UI **Cluster** aus.
2. Wählen Sie auf der Seite **Cluster** den Cluster aus, den Sie ändern möchten.
3. Wählen Sie die Registerkarte **Storage** aus.
4. Wählen Sie die Kategorie **Speicherklassen** aus.
5. Wählen Sie das Menü **Aktionen** für die Speicherklasse aus, die Sie als Standard festlegen möchten.
6. Wählen Sie **als Standard**.

Ändern Sie die Standard-Storage-Klasse über die Befehlszeile

Sie können die Standard-Storage-Klasse für ein Cluster mit Kubernetes-Befehlen ändern. Diese Methode funktioniert unabhängig von der Konfiguration Ihres Clusters.

Schritte

1. Melden Sie sich bei Ihrem Kubernetes Cluster an.
2. Listen Sie die Storage-Klassen in Ihrem Cluster auf:

```
kubectl get storageclass
```

3. Entfernen Sie die Standardbezeichnung aus der Standardspeicherklasse. Ersetzen Sie <SC_NAME> durch den Namen der Speicherklasse:

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-
class":"false"}}}'
```

4. Markieren Sie standardmäßig eine andere Storage-Klasse. Ersetzen Sie <SC_NAME> durch den Namen der Speicherklasse:

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

5. Bestätigen Sie die neue Standard-Speicherklasse:

```
kubectl get storageclass
```

Prüfen Sie die Astra Trident Version

Wenn Sie einen selbst gemanagten Cluster hinzufügen möchten, der Astra Control Provisioner oder Astra Trident für Storage-Services verwendet, müssen Sie sicherstellen, dass die installierte Version von Astra Trident 23.10 oder aktuell ist.

Schritte

1. Bestimmen Sie die Astra Trident-Version, die Sie ausführen:

```
kubectl get tridentversions -n trident
```

Wenn Astra Trident installiert ist, wird die Ausgabe wie folgt ausgegeben:

```
NAME      VERSION
trident   24.02.0
```

Wenn Astra Trident nicht installiert ist, wird die Ausgabe wie folgt angezeigt:

```
error: the server doesn't have a resource type "tridentversions"
```

2. Führen Sie einen der folgenden Schritte aus:
   - Wenn Sie Astra Trident 23.01 oder eine frühere Version verwenden, verwenden Sie diese "Anweisungen" Das Upgrade auf eine neuere Version von Astra Trident erfolgt vor dem Upgrade auf Astra Control Provisioner. Das können Sie "Führen Sie ein direktes Upgrade durch" Astra Control Provisioner 24.02, wenn Ihr Astra Trident in einem Fenster mit vier Versionen von Version 24.02 angezeigt wird. Sie können beispielsweise direkt von Astra Trident 23.04 auf Astra Control Provisioner 24.02 aktualisieren.
   - Wenn Sie Astra Trident 23.10 oder höher verwenden, stellen Sie sicher, dass es für Astra Control Provisioner verwendet wurde "Aktiviert". Astra Control Provisioner kann nicht mit Versionen von Astra Control Center vor 23.10 verwendet werden. "Upgrade für die Astra Control Provisioner" Da es nun dieselbe Version wie das Astra Control Center hat, stellen Sie ein Upgrade auf die neuesten Funktionen bereit.
3. Stellen Sie sicher, dass die Pods ausgeführt werden:

```
kubectl get pods -n trident
```

4. Prüfen Sie, ob die Storage-Klassen die unterstützten Astra Trident Treiber verwenden. Der bereitstellungsname sollte lauten `csi.trident.netapp.io`. Im folgenden Beispiel finden Sie weitere Informationen:

```
kubectl get sc
```

Beispielantwort:

```
NAME                    PROVISIONER                    RECLAIMPOLICY
VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
ontap-gold (default)    csi.trident.netapp.io          Delete
Immediate           true                   5d23h
```

Erstellen Sie eine kubeconfig-Datei

Sie können dem Astra Control Service ein Cluster mithilfe einer kubeconfig-Datei hinzufügen. Je nach dem Typ des Clusters, den Sie hinzufügen möchten, müssen Sie möglicherweise manuell eine kubeconfig-Datei für Ihr Cluster mithilfe bestimmter Schritte erstellen.

- Erstellen Sie eine kubeconfig-Datei für Amazon EKS-Cluster
- Erstellen Sie eine kubeconfig-Datei für Red hat OpenShift Service on AWS (ROSA) Cluster
- Erstellen Sie eine kubeconfig-Datei für andere Cluster-Typen

Erstellen Sie eine kubeconfig-Datei für Amazon EKS-Cluster

Befolgen Sie diese Anweisungen, um eine kubeconfig-Datei und ein permanentes Token-Geheimnis für Amazon EKS-Cluster zu erstellen. Für Cluster, die in EKS gehostet werden, ist ein permanenter Tokenschlüssel erforderlich.

Schritte

1. Befolgen Sie die Anweisungen in der Amazon-Dokumentation, um eine kubeconfig-Datei zu erstellen:

   "Erstellen oder Aktualisieren einer kubeconfig-Datei für einen Amazon EKS-Cluster"

2. Erstellen Sie ein Service-Konto wie folgt:

   a. Erstellen Sie eine Dienstkontendatei mit dem Namen `astracontrol-service-account.yaml`.

      Passen Sie den Namen des Servicekontos nach Bedarf an. Der Namespace `kube-system` Ist für diese Schritte erforderlich. Wenn Sie hier den Namen des Servicekontos ändern, sollten Sie die gleichen Änderungen in den folgenden Schritten anwenden.

```
<strong>astracontrol-service-account.yaml</strong>
```

+

```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: astra-admin-account
  namespace: kube-system
```

3. Wenden Sie das Servicekonto an:

```
kubectl apply -f astracontrol-service-account.yaml
```

4. Erstellen Sie ein `ClusterRoleBinding` Datei aufgerufen `astracontrol-clusterrolebinding.yaml`.

```
<strong>astracontrol-clusterrolebinding.yaml</strong>
```

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: astra-admin-binding
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: astra-admin-account
  namespace: kube-system
```

5. Wenden Sie die Bindung der Cluster-Rolle an:

```
kubectl apply -f astracontrol-clusterrolebinding.yaml
```

6. Erstellen Sie eine Geheimdatei für das Dienstkonto-Token mit dem Namen `astracontrol-secret.yaml`.

```
<strong>astracontrol-secret.yaml</strong>
```

```yaml
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: astra-admin-account
  name: astra-admin-account
  namespace: kube-system
type: kubernetes.io/service-account-token
```

7. Wenden Sie den Token-Schlüssel an:

```
kubectl apply -f astracontrol-secret.yaml
```

8. Rufen Sie den Token-Schlüssel ab:

```
kubectl get secret astra-admin-account -n kube-system -o
jsonpath='{.data.token}' | base64 -d
```

9. Ersetzen Sie den `user` Abschnitt der AWS EKS kubeconfig-Datei mit dem Token, wie im folgenden Beispiel gezeigt:

```
user:
    token: k8s-aws-
v1.aHR0cHM6Ly9zdHMudXMtd2VzdC0yLmFtYXpvbmF3cy5jb20vP0FjdGlvbj1HZXRDYWxsZ
XJJZGVudGl0eSZWZXJzaW9uPTIwMTEtMDYtMTUmWC1BbXotQWxnb3JpdGhtPUFXUzQtSE1BQ
y1TSEEyNTYmWC1BbXotQ3JlZGVudGlhbD1BS0lBM1JEWDdKU0haWU9LSEQ2SyUyRjIwMjMwN
DAzJTJGdXMtd2VzdC0yJTJGc3RzJTJGYXdzNF9yZXF1ZXN0JlgtQW16LURhdGU9MjAyMzA0M
DNUMjA0MzQwWiZYLUFtei1FeHBpcmVzPTYwJlgtQW16LVNpZ25lZEhlYWRlcnM9aG9zdCUzQ
ngtazhzLWF3cy1pZCZYLUFtei1TaWduYXR1cmU9YjU4ZWM0NzdiM2NkZGYxNGRhNzU4MGI2Z
WQ2zY2NzI2YWIwM2UyNThjMjRhNTJjNmVhNjc4MTRlNjJkOTg2Mg
```

Erstellen Sie eine kubeconfig-Datei für Red hat OpenShift Service on AWS (ROSA) Cluster

Befolgen Sie diese Anweisungen, um eine kubeconfig-Datei für Red hat OpenShift Service on AWS (ROSA)-Cluster zu erstellen.

Schritte

1. Melden Sie sich beim ROSA-Cluster an.
2. Service-Konto erstellen:

```
oc create sa astracontrol-service-account
```

3. Cluster-Rolle hinzufügen:

```
oc adm policy add-cluster-role-to-user cluster-admin -z astracontrol-
service-account
```

4. Erstellen Sie mithilfe des folgenden Beispiels eine geheime Konfigurationsdatei für das Dienstkonto:

```
<strong>secret-astra-sa.yaml</strong>
```

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: secret-astracontrol-service-account
  annotations:
    kubernetes.io/service-account.name: "astracontrol-service-account"
type: kubernetes.io/service-account-token
```

5. Erstellen Sie das Geheimnis:

```
oc create -f secret-astra-sa.yaml
```

6. Bearbeiten Sie das von Ihnen erstellte Dienstkonto, und fügen Sie dem den geheimen Namen des Astra Control-Dienstkontos hinzu `secrets` Abschnitt:

```
oc edit sa astracontrol-service-account
```

```yaml
apiVersion: v1
imagePullSecrets:
- name: astracontrol-service-account-dockercfg-dvfcd
kind: ServiceAccount
metadata:
  creationTimestamp: "2023-08-04T04:18:30Z"
  name: astracontrol-service-account
  namespace: default
  resourceVersion: "169770"
  uid: 965fa151-923f-4fbd-9289-30cad15998ac
secrets:
- name: astracontrol-service-account-dockercfg-dvfcd
- name: secret-astracontrol-service-account ####ADD THIS ONLY####
```

7. Listen Sie die Geheimnisse des Dienstkontos auf, ersetzen Sie `<CONTEXT>` Mit dem richtigen Kontext für Ihre Installation:

```
kubectl get serviceaccount astracontrol-service-account --context
<CONTEXT> --namespace default -o json
```

Das Ende der Ausgabe sollte wie folgt aussehen:

```
"secrets": [
{ "name": "astracontrol-service-account-dockercfg-dvfcd"},
{ "name": "secret-astracontrol-service-account"}
]
```

Die Indizes für jedes Element im `secrets` Array beginnt mit 0. Im obigen Beispiel der Index für `astracontrol-service-account-dockercfg-dvfcd` Wäre 0 und der Index für `secret-astracontrol-service-account` Sind es 1. Notieren Sie sich in Ihrer Ausgabe die Indexnummer für den Geheimschlüssel des Dienstkontos. Diese Indexnummer benötigen Sie im nächsten Schritt.

8. Erzeugen Sie den kubeconfig wie folgt:

   a. Erstellen Sie ein `create-kubeconfig.sh` Datei: Austausch `TOKEN_INDEX` Am Anfang des folgenden Skripts mit dem korrekten Wert.

```
<strong>create-kubeconfig.sh</strong>
```

```
# Update these to match your environment.
# Replace TOKEN_INDEX with the correct value
# from the output in the previous step. If you
# didn't change anything else above, don't change
# anything else here.

SERVICE_ACCOUNT_NAME=astracontrol-service-account
NAMESPACE=default
NEW_CONTEXT=astracontrol
KUBECONFIG_FILE='kubeconfig-sa'

CONTEXT=$(kubectl config current-context)

SECRET_NAME=$(kubectl get serviceaccount ${SERVICE_ACCOUNT_NAME} \
  --context ${CONTEXT} \
  --namespace ${NAMESPACE} \
  -o jsonpath='{.secrets[TOKEN_INDEX].name}')
TOKEN_DATA=$(kubectl get secret ${SECRET_NAME} \
  --context ${CONTEXT} \
  --namespace ${NAMESPACE} \
  -o jsonpath='{.data.token}')

TOKEN=$(echo ${TOKEN_DATA} | base64 -d)

# Create dedicated kubeconfig
# Create a full copy
kubectl config view --raw > ${KUBECONFIG_FILE}.full.tmp
```

```
# Switch working context to correct context
kubectl --kubeconfig ${KUBECONFIG_FILE}.full.tmp config use-context
${CONTEXT}

# Minify
kubectl --kubeconfig ${KUBECONFIG_FILE}.full.tmp \
  config view --flatten --minify > ${KUBECONFIG_FILE}.tmp

# Rename context
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  rename-context ${CONTEXT} ${NEW_CONTEXT}

# Create token user
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  set-credentials ${CONTEXT}-${NAMESPACE}-token-user \
  --token ${TOKEN}

# Set context to use token user
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  set-context ${NEW_CONTEXT} --user ${CONTEXT}-${NAMESPACE}-token
-user

# Set context to correct namespace
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  set-context ${NEW_CONTEXT} --namespace ${NAMESPACE}

# Flatten/minify kubeconfig
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  view --flatten --minify > ${KUBECONFIG_FILE}

# Remove tmp
rm ${KUBECONFIG_FILE}.full.tmp
rm ${KUBECONFIG_FILE}.tmp
```

b. Geben Sie die Befehle an, um sie auf Ihren Kubernetes-Cluster anzuwenden.

```
source create-kubeconfig.sh
```

9. (Optional) Umbenennen Sie die kubeconfig auf einen aussagekräftigen Namen für Ihr Cluster.

```
mv kubeconfig-sa YOUR_CLUSTER_NAME_kubeconfig
```

Erstellen Sie eine kubeconfig-Datei für andere Cluster-Typen

Befolgen Sie diese Anweisungen, um eine begrenzte oder erweiterte Kubeconfig-Datei für Rancher-, Upstream-Kubernetes- und Red hat OpenShift-Cluster zu erstellen.

Für Cluster, die mit kubeconfig gemanagt werden, können Sie optional eine Administratorrolle mit eingeschränkter Berechtigung oder erweiterten Berechtigungen für Astra Control Service erstellen.

Dieses Verfahren hilft Ihnen, ein separates kubeconfig zu erstellen, wenn eines der folgenden Szenarien auf Ihre Umgebung zutrifft:

- Sie möchten die Astra Control-Berechtigungen auf die Cluster beschränken, die sie verwaltet
- Sie verwenden mehrere Kontexte und können nicht den Standard Astra Control kubeconfig verwenden, der während der Installation konfiguriert wurde, oder eine eingeschränkte Rolle mit einem einzelnen Kontext funktioniert nicht in Ihrer Umgebung

Bevor Sie beginnen

Stellen Sie sicher, dass Sie für den Cluster, den Sie verwalten möchten, vor dem Ausführen der Schritte des Verfahrens Folgendes haben:

- A "Unterstützte Version" Von kubectl ist installiert.
- Kubectl Zugriff auf den Cluster, den Sie mit Astra Control Service hinzufügen und managen möchten

> Für dieses Verfahren benötigen Sie keinen kubectl-Zugriff auf den Cluster, auf dem Astra Control Service ausgeführt wird.

- Ein aktiver kubeconfig für den Cluster, den Sie mit Clusteradministratorrechten für den aktiven Kontext verwalten möchten

Schritte

1. Service-Konto erstellen:
   a. Erstellen Sie eine Dienstkontendatei mit dem Namen `astracontrol-service-account.yaml`.

```
<strong>astracontrol-service-account.yaml</strong>
```

```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: astracontrol-service-account
  namespace: default
```

   b. Wenden Sie das Servicekonto an:

```
kubectl apply -f astracontrol-service-account.yaml
```

2. Erstellen Sie eine der folgenden Clusterrollen mit ausreichenden Berechtigungen für ein Cluster, das von

Astra Control gemanagt werden kann:

Eingeschränkte Cluster-Rolle

Diese Rolle enthält die Mindestberechtigungen, die für das Management eines Clusters durch Astra Control erforderlich sind:

a. Erstellen Sie ein `ClusterRole` Datei mit dem Namen, z. B. `astra-admin-account.yaml`.

```
<strong>astra-admin-account.yaml</strong>
```

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: astra-admin-account
rules:

# Get, List, Create, and Update all resources
# Necessary to backup and restore all resources in an app
- apiGroups:
  - '*'
  resources:
  - '*'
  verbs:
  - get
  - list
  - create
  - patch

# Delete Resources
# Necessary for in-place restore and AppMirror failover
- apiGroups:
  - ""
  - apps
  - autoscaling
  - batch
  - crd.projectcalico.org
  - extensions
  - networking.k8s.io
  - policy
  - rbac.authorization.k8s.io
  - snapshot.storage.k8s.io
  - trident.netapp.io
  resources:
  - configmaps
  - cronjobs
  - daemonsets
```

```
    - deployments
    - horizontalpodautoscalers
    - ingresses
    - jobs
    - namespaces
    - networkpolicies
    - persistentvolumeclaims
    - poddisruptionbudgets
    - pods
    - podtemplates
    - replicasets
    - replicationcontrollers
    - replicationcontrollers/scale
    - rolebindings
    - roles
    - secrets
    - serviceaccounts
    - services
    - statefulsets
    - tridentmirrorrelationships
    - tridentsnapshotinfos
    - volumesnapshots
    - volumesnapshotcontents
    verbs:
    - delete

  # Watch resources
  # Necessary to monitor progress
  - apiGroups:
    - ""
    resources:
    - pods
    - replicationcontrollers
    - replicationcontrollers/scale
    verbs:
    - watch

  # Update resources
  - apiGroups:
    - ""
    - build.openshift.io
    - image.openshift.io
    resources:
    - builds/details
    - replicationcontrollers
    - replicationcontrollers/scale
```

```
  - imagestreams/layers
  - imagestreamtags
  - imagetags
  verbs:
  - update
```

b. (Nur für OpenShift-Cluster) Anhängen Sie am Ende des an `astra-admin-account.yaml` Datei:

```
# OpenShift security
- apiGroups:
  - security.openshift.io
  resources:
  - securitycontextconstraints
  verbs:
  - use
  - update
```

c. Wenden Sie die Cluster-Rolle an:

```
kubectl apply -f astra-admin-account.yaml
```

Erweiterte Cluster-Rolle

Diese Rolle enthält erweiterte Berechtigungen für ein Cluster, das von Astra Control gemanagt werden kann. Sie können diese Rolle verwenden, wenn Sie mehrere Kontexte verwenden und nicht den während der Installation konfigurierten Astra Control kubeconfig verwenden können oder eine eingeschränkte Rolle mit einem einzelnen Kontext in Ihrer Umgebung nicht funktioniert:

> Im Folgenden `ClusterRole` Schritte sind ein allgemeines Kubernetes-Beispiel. Anweisungen zu Ihrer spezifischen Umgebung finden Sie in der Dokumentation zur Kubernetes-Distribution.

a. Erstellen Sie ein `ClusterRole` Datei mit dem Namen, z. B. `astra-admin-account.yaml`.

```
<strong>astra-admin-account.yaml</strong>
```

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: astra-admin-account
rules:
- apiGroups:
  - '*'
  resources:
  - '*'
  verbs:
  - '*'
- nonResourceURLs:
  - '*'
  verbs:
  - '*'
```

b. Wenden Sie die Cluster-Rolle an:

```
kubectl apply -f astra-admin-account.yaml
```

3. Erstellen Sie die Cluster-Rolle, die für die Cluster-Rolle an das Service-Konto gebunden ist:

a. Erstellen Sie ein `ClusterRoleBinding` Datei aufgerufen `astracontrol-clusterrolebinding.yaml`.

```
<strong>astracontrol-clusterrolebinding.yaml</strong>
```

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: astracontrol-admin
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: astra-admin-account
subjects:
- kind: ServiceAccount
  name: astracontrol-service-account
  namespace: default
```

b. Wenden Sie die Bindung der Cluster-Rolle an:

```
kubectl apply -f astracontrol-clusterrolebinding.yaml
```

4. Erstellen und Anwenden des Token-Geheimnisses:
   a. Erstellen Sie eine Geheimdatei mit dem Namen Token `secret-astracontrol-service-account.yaml`.

```
<strong>secret-astracontrol-service-account.yaml</strong>
```

```
apiVersion: v1
kind: Secret
metadata:
  name: secret-astracontrol-service-account
  namespace: default
  annotations:
    kubernetes.io/service-account.name: "astracontrol-service-
account"
type: kubernetes.io/service-account-token
```

   b. Wenden Sie den Token-Schlüssel an:

```
kubectl apply -f secret-astracontrol-service-account.yaml
```

5. Fügen Sie dem Dienstkonto den Token-Schlüssel hinzu, indem Sie den Namen dem hinzufügen `secrets` Array (die letzte Zeile im folgenden Beispiel):

```
kubectl edit sa astracontrol-service-account
```

```
apiVersion: v1
imagePullSecrets:
- name: astracontrol-service-account-dockercfg-48xhx
kind: ServiceAccount
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |

{"apiVersion":"v1","kind":"ServiceAccount","metadata":{"annotations":{},
"name":"astracontrol-service-account","namespace":"default"}}
  creationTimestamp: "2023-06-14T15:25:45Z"
  name: astracontrol-service-account
  namespace: default
  resourceVersion: "2767069"
  uid: 2ce068c4-810e-4a96-ada3-49cbf9ec3f89
secrets:
- name: astracontrol-service-account-dockercfg-48xhx
<strong>- name: secret-astracontrol-service-account</strong>
```

6. Listen Sie die Geheimnisse des Dienstkontos auf, ersetzen Sie `<context>` Mit dem richtigen Kontext für Ihre Installation:

```
kubectl get serviceaccount astracontrol-service-account --context
<context> --namespace default -o json
```

Das Ende der Ausgabe sollte wie folgt aussehen:

```
"secrets": [
{ "name": "astracontrol-service-account-dockercfg-48xhx"},
{ "name": "secret-astracontrol-service-account"}
]
```

Die Indizes für jedes Element im `secrets` Array beginnt mit 0. Im obigen Beispiel der Index für `astracontrol-service-account-dockercfg-48xhx` Wäre 0 und der Index für `secret-astracontrol-service-account` Sind es 1. Notieren Sie sich in Ihrer Ausgabe die Indexnummer für den Geheimschlüssel des Dienstkontos. Im nächsten Schritt benötigen Sie diese Indexnummer.

7. Erzeugen Sie den kubeconfig wie folgt:
   a. Erstellen Sie ein `create-kubeconfig.sh` Datei:
   b. Austausch `TOKEN_INDEX` Am Anfang des folgenden Skripts mit dem korrekten Wert.

```
<strong>create-kubeconfig.sh</strong>
```

```
# Update these to match your environment.
# Replace TOKEN_INDEX with the correct value
# from the output in the previous step. If you
# didn't change anything else above, don't change
# anything else here.

SERVICE_ACCOUNT_NAME=astracontrol-service-account
NAMESPACE=default
NEW_CONTEXT=astracontrol
KUBECONFIG_FILE='kubeconfig-sa'

CONTEXT=$(kubectl config current-context)

SECRET_NAME=$(kubectl get serviceaccount ${SERVICE_ACCOUNT_NAME} \
  --context ${CONTEXT} \
  --namespace ${NAMESPACE} \
  *-o jsonpath='{.secrets[TOKEN_INDEX].name}')
TOKEN_DATA=$(kubectl get secret ${SECRET_NAME} \
  --context ${CONTEXT} \
  --namespace ${NAMESPACE} \
  -o jsonpath='{.data.token}')

TOKEN=$(echo ${TOKEN_DATA} | base64 -d)

# Create dedicated kubeconfig
# Create a full copy
kubectl config view --raw > ${KUBECONFIG_FILE}.full.tmp

# Switch working context to correct context
kubectl --kubeconfig ${KUBECONFIG_FILE}.full.tmp config use-context
${CONTEXT}

# Minify
kubectl --kubeconfig ${KUBECONFIG_FILE}.full.tmp \
  config view --flatten --minify > ${KUBECONFIG_FILE}.tmp

# Rename context
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  rename-context ${CONTEXT} ${NEW_CONTEXT}

# Create token user
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  set-credentials ${CONTEXT}-${NAMESPACE}-token-user \
  --token ${TOKEN}

# Set context to use token user
```

```
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  set-context ${NEW_CONTEXT} --user ${CONTEXT}-${NAMESPACE}-token-
user

# Set context to correct namespace
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  set-context ${NEW_CONTEXT} --namespace ${NAMESPACE}

# Flatten/minify kubeconfig
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  view --flatten --minify > ${KUBECONFIG_FILE}

# Remove tmp
rm ${KUBECONFIG_FILE}.full.tmp
rm ${KUBECONFIG_FILE}.tmp
```

c. Geben Sie die Befehle an, um sie auf Ihren Kubernetes-Cluster anzuwenden.

```
source create-kubeconfig.sh
```

8. (Optional) Umbenennen Sie die kubeconfig auf einen aussagekräftigen Namen für Ihr Cluster.

```
mv kubeconfig-sa YOUR_CLUSTER_NAME_kubeconfig
```

Copyright-Informationen

Copyright © 2024 NetApp. Alle Rechte vorbehalten. Gedruckt in den USA. Dieses urheberrechtlich geschützte Dokument darf ohne die vorherige schriftliche Genehmigung des Urheberrechtsinhabers in keiner Form und durch keine Mittel – weder grafische noch elektronische oder mechanische, einschließlich Fotokopieren, Aufnehmen oder Speichern in einem elektronischen Abrufsystem – auch nicht in Teilen, vervielfältigt werden.

Software, die von urheberrechtlich geschütztem NetApp Material abgeleitet wird, unterliegt der folgenden Lizenz und dem folgenden Haftungsausschluss:

DIE VORLIEGENDE SOFTWARE WIRD IN DER VORLIEGENDEN FORM VON NETAPP ZUR VERFÜGUNG GESTELLT, D. H. OHNE JEGLICHE EXPLIZITE ODER IMPLIZITE GEWÄHRLEISTUNG, EINSCHLIESSLICH, JEDOCH NICHT BESCHRÄNKT AUF DIE STILLSCHWEIGENDE GEWÄHRLEISTUNG DER MARKTGÄNGIGKEIT UND EIGNUNG FÜR EINEN BESTIMMTEN ZWECK, DIE HIERMIT AUSGESCHLOSSEN WERDEN. NETAPP ÜBERNIMMT KEINERLEI HAFTUNG FÜR DIREKTE, INDIREKTE, ZUFÄLLIGE, BESONDERE, BEISPIELHAFTE SCHÄDEN ODER FOLGESCHÄDEN (EINSCHLIESSLICH, JEDOCH NICHT BESCHRÄNKT AUF DIE BESCHAFFUNG VON ERSATZWAREN ODER -DIENSTLEISTUNGEN, NUTZUNGS-, DATEN- ODER GEWINNVERLUSTE ODER UNTERBRECHUNG DES GESCHÄFTSBETRIEBS), UNABHÄNGIG DAVON, WIE SIE VERURSACHT WURDEN UND AUF WELCHER HAFTUNGSTHEORIE SIE BERUHEN, OB AUS VERTRAGLICH FESTGELEGTER HAFTUNG, VERSCHULDENSUNABHÄNGIGER HAFTUNG ODER DELIKTSHAFTUNG (EINSCHLIESSLICH FAHRLÄSSIGKEIT ODER AUF ANDEREM WEGE), DIE IN IRGENDEINER WEISE AUS DER NUTZUNG DIESER SOFTWARE RESULTIEREN, SELBST WENN AUF DIE MÖGLICHKEIT DERARTIGER SCHÄDEN HINGEWIESEN WURDE.

NetApp behält sich das Recht vor, die hierin beschriebenen Produkte jederzeit und ohne Vorankündigung zu ändern. NetApp übernimmt keine Verantwortung oder Haftung, die sich aus der Verwendung der hier beschriebenen Produkte ergibt, es sei denn, NetApp hat dem ausdrücklich in schriftlicher Form zugestimmt. Die Verwendung oder der Erwerb dieses Produkts stellt keine Lizenzierung im Rahmen eines Patentrechts, Markenrechts oder eines anderen Rechts an geistigem Eigentum von NetApp dar.

Das in diesem Dokument beschriebene Produkt kann durch ein oder mehrere US-amerikanische Patente, ausländische Patente oder anhängige Patentanmeldungen geschützt sein.

ERLÄUTERUNG ZU „RESTRICTED RIGHTS“: Nutzung, Vervielfältigung oder Offenlegung durch die US-Regierung unterliegt den Einschränkungen gemäß Unterabschnitt (b)(3) der Klausel „Rights in Technical Data – Noncommercial Items“ in DFARS 252.227-7013 (Februar 2014) und FAR 52.227-19 (Dezember 2007).

Die hierin enthaltenen Daten beziehen sich auf ein kommerzielles Produkt und/oder einen kommerziellen Service (wie in FAR 2.101 definiert) und sind Eigentum von NetApp, Inc. Alle technischen Daten und die Computersoftware von NetApp, die unter diesem Vertrag bereitgestellt werden, sind gewerblicher Natur und wurden ausschließlich unter Verwendung privater Mittel entwickelt. Die US-Regierung besitzt eine nicht ausschließliche, nicht übertragbare, nicht unterlizenzierbare, weltweite, limitierte unwiderrufliche Lizenz zur Nutzung der Daten nur in Verbindung mit und zur Unterstützung des Vertrags der US-Regierung, unter dem die Daten bereitgestellt wurden. Sofern in den vorliegenden Bedingungen nicht anders angegeben, dürfen die Daten ohne vorherige schriftliche Genehmigung von NetApp, Inc. nicht verwendet, offengelegt, vervielfältigt, geändert, aufgeführt oder angezeigt werden. Die Lizenzrechte der US-Regierung für das US-Verteidigungsministerium sind auf die in DFARS-Klausel 252.227-7015(b) (Februar 2014) genannten Rechte beschränkt.

Markeninformationen

NETAPP, das NETAPP Logo und die unter http://www.netapp.com/TM aufgeführten Marken sind Marken von NetApp, Inc. Andere Firmen und Produktnamen können Marken der jeweiligen Eigentümer sein.