



Nutzen Sie Astra Trident

Astra Trident

NetApp
January 14, 2026

Inhalt

Nutzen Sie Astra Trident	1
Bereiten Sie den Knoten „Worker“ vor	1
Auswahl der richtigen Werkzeuge	1
Ermittlung des Node-Service	1
NFS Volumes	2
ISCSI-Volumes	2
NVMe/TCP-Volumes	6
Konfiguration und Management von Back-Ends	7
Back-Ends konfigurieren	7
Azure NetApp Dateien	8
Cloud Volumes Service für Google Cloud-Back-End konfigurieren	24
Konfigurieren Sie ein NetApp HCI- oder SolidFire-Backend	40
ONTAP SAN-Treiber	46
ONTAP-NAS-Treiber	69
Amazon FSX für NetApp ONTAP	98
Back-Ends mit kubectl erstellen	116
Back-Ends managen	123
Erstellen und Managen von Storage-Klassen	132
Erstellen Sie eine Speicherklasse	132
Management von Storage-Klassen	135
Provisionierung und Management von Volumes	137
Bereitstellen eines Volumes	137
Erweitern Sie Volumes	142
Volumes importieren	150
Ein NFS-Volume kann über Namespaces hinweg genutzt werden	157
Replizieren Sie Volumes mit SnapMirror	161
Verwenden Sie die CSI-Topologie	177
Arbeiten Sie mit Snapshots	185

Nutzen Sie Astra Trident

Bereiten Sie den Knoten „Worker“ vor

Alle Worker-Nodes im Kubernetes-Cluster müssen in der Lage sein, die Volumes, die Sie für Ihre Pods bereitgestellt haben, zu mounten. Um die Worker-Nodes vorzubereiten, müssen Sie auf der Grundlage Ihrer Treiberauswahl NFS-, iSCSI- oder NVMe/TCP-Tools installieren.

Auswahl der richtigen Werkzeuge

Wenn Sie eine Kombination von Treibern verwenden, sollten Sie alle erforderlichen Tools für Ihre Treiber installieren. Bei aktuellen Versionen von RedHat CoreOS sind die Tools standardmäßig installiert.

NFS Tools

["Installieren Sie die NFS Tools"](#) Wenn Sie Folgendes verwenden: `ontap-nas`, `ontap-nas-economy`, `ontap-nas-flexgroup`, `azure-netapp-files`, `gcp-cvs`.

iSCSI-Tools

["Installieren Sie die iSCSI-Tools"](#) Wenn Sie Folgendes verwenden: `ontap-san`, `ontap-san-economy`, `solidfire-san`.

NVMe-Tools

["Installation der NVMe Tools"](#) Wenn Sie verwenden `ontap-san` Für das NVMe-over-TCP-Protokoll (Nonvolatile Memory Express).



Wir empfehlen ONTAP 9.12 oder höher für NVMe/TCP.

Ermittlung des Node-Service

Astra Trident versucht automatisch zu erkennen, ob der Node iSCSI- oder NFS-Services ausführen kann.



Die Ermittlung des Node-Service erkennt erkannte Services, gewährleistet jedoch nicht, dass Services ordnungsgemäß konfiguriert wurden. Umgekehrt kann das Fehlen eines entdeckten Service nicht garantieren, dass die Volume-Bereitstellung fehlschlägt.

Überprüfen Sie Ereignisse

Astra Trident erstellt Ereignisse für den Node zur Identifizierung der erkannten Services. Um diese Ereignisse zu überprüfen, führen Sie folgende Schritte aus:

```
kubectl get event -A --field-selector involvedObject.name=<Kubernetes node name>
```

Überprüfen Sie erkannte Services

Astra Trident erkennt aktivierte Services für jeden Knoten auf der Trident Node CR. Um die ermittelten Dienste anzuzeigen, führen Sie folgende Schritte aus:

```
tridentctl get node -o wide -n <Trident namespace>
```

NFS Volumes

Installieren Sie die NFS-Tools unter Verwendung der Befehle für Ihr Betriebssystem. Stellen Sie sicher, dass der NFS-Dienst während des Bootens gestartet wird.

RHEL 8 ODER HÖHER

```
sudo yum install -y nfs-utils
```

Ubuntu

```
sudo apt-get install -y nfs-common
```



Starten Sie die Worker-Nodes nach der Installation der NFS-Tools neu, um einen Fehler beim Anschließen von Volumes an Container zu vermeiden.

ISCSI-Volumes

Astra Trident kann automatisch eine iSCSI-Sitzung einrichten, LUNs scannen und Multipath-Geräte erkennen, sie formatieren und auf einem Pod mounten.

ISCSI-Funktionen zur Selbstreparatur

Bei ONTAP Systemen führt Astra Trident alle fünf Minuten iSCSI-Selbstheilung aus und bietet folgende Vorteile:

1. * Identifizieren Sie den gewünschten iSCSI-Sitzungsstatus und den aktuellen iSCSI-Sitzungsstatus.
2. **Vergleichen** der gewünschte Zustand mit dem aktuellen Zustand, um notwendige Reparaturen zu identifizieren. Astra Trident ermittelt Reparaturprioritäten und wann Maßnahmen ergriffen werden müssen.
3. **Durchführung von Reparaturen** erforderlich, um den aktuellen iSCSI-Sitzungsstatus auf den gewünschten iSCSI-Sitzungsstatus zurückzusetzen.



Protokolle der Selbstheilungsaktivität befinden sich im `trident-main` Behälter auf dem jeweiligen Demonset Pod. Um Protokolle anzuzeigen, müssen Sie festgelegt haben `debug` Auf „true“ bei der Installation von Astra Trident zu setzen.

Astra Trident iSCSI-Funktionen zur Selbstheilung verhindern:

- Veraltete oder ungesunde iSCSI-Sitzungen, die nach einem Problem mit der Netzwerkverbindung auftreten können. Im Falle einer veralteten Sitzung wartet Astra Trident sieben Minuten vor der Anmeldung, um die Verbindung mit einem Portal wiederherzustellen.



Wenn beispielsweise CHAP-Schlüssel auf dem Speicher-Controller gedreht wurden und die Verbindung zum Netzwerk unterbrochen wird, können die alten (*Inated*) CHAP-Schlüssel bestehen bleiben. Selbstheilung kann dies erkennen und die Sitzung automatisch wiederherstellen, um die aktualisierten CHAP-Schlüssel anzuwenden.

- iSCSI-Sitzungen fehlen
- LUNs sind nicht vorhanden

Installieren Sie die iSCSI-Tools

Installieren Sie die iSCSI-Tools mit den Befehlen für Ihr Betriebssystem.

Bevor Sie beginnen

- Jeder Node im Kubernetes-Cluster muss über einen eindeutigen IQN verfügen. **Dies ist eine notwendige Voraussetzung.**
- Bei Verwendung von RHCOS Version 4.5 oder höher oder einer anderen RHEL-kompatiblen Linux-Distribution mit dem `solidfire-san` Treiber und Element OS 12.5 oder früher: Stellen Sie sicher, dass der CHAP-Authentifizierungsalgorithmus auf MD5 in eingestellt ist `/etc/iscsi/iscsid.conf`. Sichere, FIPS-konforme CHAP-Algorithmen SHA1, SHA-256 und SHA3-256 sind mit Element 12.7 erhältlich.

```
sudo sed -i 's/^\(node.session.auth.chap_algs\) .*/\1 = MD5/'  
/etc/iscsi/iscsid.conf
```

- Geben Sie bei Verwendung von Worker-Nodes, die RHEL/RedHat CoreOS mit iSCSI PVS ausführen, die `discard` MountOption in StorageClass für die Inline-Speicherplatzrückgewinnung. Siehe ["Red hat-Dokumentation"](#).

RHEL 8 ODER HÖHER

1. Installieren Sie die folgenden Systempakete:

```
sudo yum install -y lsscsi iscsi-initiator-utils sg3_utils device-  
mapper-multipath
```

2. Überprüfen Sie, ob die Version von iscsi-Initiator-utils 6.2.0.874-2.el7 oder höher ist:

```
rpm -q iscsi-initiator-utils
```

3. Scannen auf manuell einstellen:

```
sudo sed -i 's/^\(node.session.scan\).*$/\1 = manual/'  
/etc/iscsi/iscsid.conf
```

4. Multipathing aktivieren:

```
sudo mpathconf --enable --with_multipathd y --find_multipaths n
```



Unbedingt `etc/multipath.conf` Enthält `find_multipaths no` Unter defaults.

5. Stellen Sie das sicher iscsid Und multipathd Laufen:

```
sudo systemctl enable --now iscsid multipathd
```

6. Aktivieren und starten iscsi:

```
sudo systemctl enable --now iscsi
```

Ubuntu

1. Installieren Sie die folgenden Systempakete:

```
sudo apt-get install -y open-iscsi lsscsi sg3-utils multipath-tools  
scsitools
```

2. Stellen Sie sicher, dass Open-iscsi-Version 2.0.874-5ubuntu2.10 oder höher (für bionic) oder 2.0.874-7.1ubuntu6.1 oder höher (für Brennweite) ist:

```
dpkg -l open-iscsi
```

3. Scannen auf manuell einstellen:

```
sudo sed -i 's/^\(node.session.scan\).*\/\1 = manual/'  
/etc/iscsi/iscsid.conf
```

4. Multipathing aktivieren:

```
sudo tee /etc/multipath.conf <<-'EOF'  
defaults {  
    user_friendly_names yes  
    find_multipaths no  
}  
EOF  
sudo systemctl enable --now multipath-tools.service  
sudo service multipath-tools restart
```



Unbedingt `etc/multipath.conf` Enthält `find_multipaths no` Unter `defaults`.

5. Stellen Sie das sicher `open-iscsi` Und `multipath-tools` Sind aktiviert und läuft:

```
sudo systemctl status multipath-tools  
sudo systemctl enable --now open-iscsi.service  
sudo systemctl status open-iscsi
```



Für Ubuntu 18.04, müssen Sie Ziel-Ports mit erkennen `iscsiadm` Vor dem Start `open-iscsi` Damit der iSCSI-Daemon gestartet werden kann. Alternativ können Sie den ändern `iscsi` Dienst zu starten `iscsid` Automatisch

Konfigurieren oder deaktivieren Sie die iSCSI-Selbstheilung

Sie können die folgenden Selbstreparatureinstellungen von Astra Trident iSCSI konfigurieren, um veraltete Sitzungen zu beheben:

- **iSCSI-Selbstheilungsintervall:** Bestimmt die Häufigkeit, mit der iSCSI-Selbstheilung aufgerufen wird (Standard: 5 Minuten). Sie können ihn so konfigurieren, dass er häufiger ausgeführt wird, indem Sie eine kleinere Zahl oder weniger häufig einstellen, indem Sie eine größere Zahl einstellen.



Wenn Sie das iSCSI-Selbstreparaturintervall auf 0 setzen, wird die iSCSI-Selbstheilung vollständig beendet. Wir empfehlen keine Deaktivierung der iSCSI-Selbstheilung. Sie sollte nur in bestimmten Szenarien deaktiviert werden, wenn die iSCSI-Selbstheilung nicht wie vorgesehen funktioniert oder zu Debugging-Zwecken verwendet wird.

- **iSCSI Self-Healing-Wartezeit:** Bestimmt die Dauer, die iSCSI Self-Healing wartet, bevor Sie sich von einer ungesunden Sitzung abmelden und erneut anmelden (Standard: 7 Minuten). Sie können sie für eine größere Anzahl konfigurieren, sodass Sitzungen, die als „fehlerhaft“ identifiziert werden, länger warten müssen, bevor sie abgemeldet werden. Anschließend wird versucht, sich erneut anzumelden, oder eine kleinere Zahl, um sich früher abzumelden und anzumelden.

Helm

Um iSCSI-Selbstreparatureinstellungen zu konfigurieren oder zu ändern, übergeben Sie den `iscsiSelfHealingInterval` Und `iscsiSelfHealingWaitTime` Parameter während der Ruderinstallation oder der Ruderaktualisierung.

Im folgenden Beispiel wird das iSCSI-Intervall für die Selbstheilung auf 3 Minuten und die Wartezeit für die Selbstheilung auf 6 Minuten eingestellt:

```
helm install trident trident-operator-100.2402.0.tgz --set
iscsiSelfHealingInterval=3m0s --set iscsiSelfHealingWaitTime=6m0s -n
trident
```

Tridentctl

Um iSCSI-Selbstreparatureinstellungen zu konfigurieren oder zu ändern, übergeben Sie den `iscsi-self-healing-interval` Und `iscsi-self-healing-wait-time` Parameter während der `tridentctl`-Installation oder -Aktualisierung.

Im folgenden Beispiel wird das iSCSI-Intervall für die Selbstheilung auf 3 Minuten und die Wartezeit für die Selbstheilung auf 6 Minuten eingestellt:

```
tridentctl install --iscsi-self-healing-interval=3m0s --iscsi-self
-healing-wait-time=6m0s -n trident
```

NVMe/TCP-Volumes

Installieren Sie die NVMe Tools mithilfe der Befehle für Ihr Betriebssystem.



- Für NVMe ist RHEL 9 oder höher erforderlich.
- Wenn die Kernel-Version Ihres Kubernetes Node zu alt ist oder das NVMe-Paket für Ihre Kernel-Version nicht verfügbar ist, müssen Sie möglicherweise die Kernel-Version Ihres Node mit dem NVMe-Paket auf eine aktualisieren.

RHEL 9

```
sudo yum install nvme-cli
sudo yum install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

Ubuntu

```
sudo apt install nvme-cli
sudo apt -y install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

Überprüfen Sie die Installation

Überprüfen Sie nach der Installation mit dem Befehl, ob für jeden Node im Kubernetes-Cluster ein eindeutiges NQN verwendet wird:

```
cat /etc/nvme/hostnqn
```



Astra Trident ändert die `ctrl_device_tmo` Nutzen, um zu gewährleisten, dass NVMe bei einem Ausfall nicht auf dem Weg bleibt. Ändern Sie diese Einstellung nicht.

Konfiguration und Management von Back-Ends

Back-Ends konfigurieren

Ein Backend definiert die Beziehung zwischen Astra Trident und einem Storage-System. Er erzählt Astra Trident, wie man mit diesem Storage-System kommuniziert und wie Astra Trident Volumes darauf bereitstellen sollte.

Astra Trident stellt automatisch Storage-Pools aus Back-Ends bereit, die den von einer Storage-Klasse definierten Anforderungen entsprechen. Erfahren Sie, wie Sie das Backend für Ihr Storage-System konfigurieren.

- ["Konfigurieren Sie ein Azure NetApp Files-Backend"](#)
- ["Konfigurieren Sie ein Back-End für Cloud Volumes Service für Google Cloud Platform"](#)
- ["Konfigurieren Sie ein NetApp HCI- oder SolidFire-Backend"](#)
- ["Konfigurieren Sie ein Backend mit ONTAP- oder Cloud Volumes ONTAP-NAS-Treibern"](#)
- ["Konfigurieren Sie ein Backend mit ONTAP- oder Cloud Volumes ONTAP-SAN-Treibern"](#)
- ["Setzen Sie Astra Trident mit Amazon FSX für NetApp ONTAP ein"](#)

Azure NetApp Dateien

Konfigurieren Sie ein Azure NetApp Files-Backend

Sie können Azure NetApp Files als Backend für Astra Trident konfigurieren. Sie können NFS- und SMB-Volumes über ein Azure NetApp Files-Back-End einbinden. Astra Trident unterstützt auch das Anmeldeinformationsmanagement mithilfe von Managed Identities für AKS-Cluster (Azure Kubernetes Services).

Azure NetApp Files-Treiberdetails

Astra Trident bietet die folgenden Azure NetApp Files Storage-Treiber für die Kommunikation mit dem Cluster. Unterstützte Zugriffsmodi sind: *ReadWriteOnce* (RWO), *ReadOnly Many* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Treiber	Protokoll	VolumeModus	Unterstützte Zugriffsmodi	Unterstützte Filesysteme
azure-netapp-files	NFS SMB	Dateisystem	RWO, ROX, RWX, RWOP	nfs, smb

Überlegungen

- Der Azure NetApp Files-Service unterstützt keine Volumes mit einer Größe von weniger als 100 GB. Astra Trident erstellt automatisch 100-gib-Volumes, wenn ein kleineres Volume angefordert wird.
- Astra Trident unterstützt SMB Volumes, die nur auf Windows Nodes laufenden Pods gemountet werden.

Verwaltete Identitäten für AKS

Astra Trident unterstützt "[Verwaltete Identitäten](#)" Für Cluster von Azure Kubernetes Services. Um die Vorteile einer optimierten Verwaltung von Anmeldeinformationen zu nutzen, die von verwalteten Identitäten angeboten wird, müssen Sie über Folgendes verfügen:

- Ein mit AKS implementierter Kubernetes-Cluster
- Verwaltete Identitäten, die auf dem AKS kubernetes-Cluster konfiguriert sind
- Astra Trident installiert, einschließlich `cloudProvider` Angabe "Azure".

Betreiber von Trident

Um Astra Trident mit dem Trident-Operator zu installieren, bearbeiten Sie

`tridentorchestrator_cr.yaml` Einstellen `cloudProvider` Bis "Azure". Beispiel:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "Azure"
```

Helm

Im folgenden Beispiel werden die Astra Trident Sets installiert `cloudProvider` Zu Azure unter Verwendung der Umgebungsvariable `$CP`:

```
helm install trident trident-operator-100.2402.0.tgz --create
--namespace --namespace <trident-namespace> --set cloudProvider=$CP
```

`tridentctl`

Im folgenden Beispiel wird Astra Trident installiert und legt den fest `cloudProvider` Flag an Azure:

```
tridentctl install --cloud-provider="Azure" -n trident
```

Cloud-Identität für AKS

Die Cloud-Identität ermöglicht Kubernetes-Pods den Zugriff auf Azure-Ressourcen durch Authentifizierung als Workload-Identität anstatt durch Angabe explizite Azure-Anmeldedaten.

Um die Vorteile der Cloud-Identität in Azure zu nutzen, müssen Sie über folgende Voraussetzungen verfügen:

- Ein mit AKS implementierter Kubernetes-Cluster
- Workload-Identität und `oidc-issuer`, die auf dem AKS Kubernetes-Cluster konfiguriert sind
- Astra Trident installiert, einschließlich `cloudProvider` Angabe "Azure" Und `cloudIdentity` Angabe der Workload-Identität

Betreiber von Trident

Um Astra Trident mit dem Trident-Operator zu installieren, bearbeiten Sie

`tridentorchestrator_cr.yaml` Einstellen `cloudProvider` Bis "Azure" Und gesetzt `cloudIdentity` Bis `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx`.

Beispiel:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "Azure"
  *cloudIdentity: 'azure.workload.identity/client-id: xxxxxxxx-xxxx-
xxxx-xxxx-xxxxxxxxxxxx' *
```

Helm

Legen Sie die Werte für **Cloud-Provider (CP)** und **Cloud-Identity (CI)** unter Verwendung der folgenden Umgebungsvariablen fest:

```
export CP="Azure"
export CI="azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxxx"
```

Im folgenden Beispiel werden Astra Trident und Sätze installiert `cloudProvider` Zu Azure unter Verwendung der Umgebungsvariable `$CP` Und legt die fest `cloudIdentity` Verwenden der Umgebungsvariable `$CI`:

```
helm install trident trident-operator-100.2402.0.tgz --set
cloudProvider=$CP --set cloudIdentity=$CI
```

`tridentctl`

Legen Sie die Werte für **Cloud Provider** und **Cloud Identity** unter Verwendung der folgenden Umgebungsvariablen fest:

```
export CP="Azure"
export CI="azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxxx"
```

Im folgenden Beispiel wird Astra Trident installiert und legt den fest `cloud-provider` Flag an `$CP`, und `cloud-identity` Bis `$CI`:

```
tridentctl install --cloud-provider=$CP --cloud-identity="$CI" -n
trident
```

Konfiguration eines Azure NetApp Files-Backends wird vorbereitet

Bevor Sie Ihr Azure NetApp Files-Backend konfigurieren können, müssen Sie sicherstellen, dass die folgenden Anforderungen erfüllt sind.

Voraussetzungen für NFS und SMB Volumes

Wenn Sie Azure NetApp Files zum ersten Mal oder an einem neuen Standort verwenden, ist eine Erstkonfiguration erforderlich, um Azure NetApp Files einzurichten und ein NFS-Volume zu erstellen. Siehe ["Azure: Azure NetApp Files einrichten und ein NFS Volume erstellen"](#).

Um ein zu konfigurieren und zu verwenden ["Azure NetApp Dateien"](#) Back-End, Sie benötigen Folgendes:



- `subscriptionID`, `tenantID`, `clientID`, `location`, und `clientSecret` Sind optional, wenn verwaltete Identitäten auf einem AKS-Cluster verwendet werden.
- `tenantID`, `clientID`, und `clientSecret` Sind optional, wenn eine Cloud-Identität auf einem AKS-Cluster verwendet wird.

- Ein Kapazitäts-Pool. Siehe ["Microsoft: Erstellen Sie einen Kapazitäts-Pool für Azure NetApp Files"](#).
- Ein an Azure NetApp Files delegiertes Subnetz. Siehe ["Microsoft: Delegieren Sie ein Subnetz an Azure NetApp Files"](#).
- `subscriptionID` Über ein Azure Abonnement mit aktiviertem Azure NetApp Files.
- `tenantID`, `clientID`, und `clientSecret` Von einem ["App-Registrierung"](#) In Azure Active Directory mit ausreichenden Berechtigungen für den Azure NetApp Files-Service. Die App-Registrierung sollte Folgendes verwenden:
 - Der Eigentümer oder die Rolle des Mitarbeiters ["Vordefiniert von Azure"](#).
 - A ["Benutzerdefinierte Beitragsrolle"](#) Auf Abonnementebene (`assignableScopes`) Mit den folgenden Berechtigungen, die auf nur das beschränkt sind, was Astra Trident erfordert. Nach dem Erstellen der benutzerdefinierten Rolle ["Weisen Sie die Rolle über das Azure-Portal zu"](#).

```
{
  "id": "/subscriptions/<subscription-id>/providers/Microsoft.Authorization/roleDefinitions/<role-definition-id>",
  "properties": {
    "roleName": "custom-role-with-limited-perms",
    "description": "custom role providing limited permissions",
    "assignableScopes": [
      "/subscriptions/<subscription-id>"
    ],
    "permissions": [
      {
        "actions": [

"Microsoft.NetApp/netAppAccounts/capacityPools/read",

"Microsoft.NetApp/netAppAccounts/capacityPools/write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/read",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/delete",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/read",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/delete",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/MountTargets/read",

          "Microsoft.Network/virtualNetworks/read",

"Microsoft.Network/virtualNetworks/subnets/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrations/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
```

```

ions/write",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/delete",

        "Microsoft.Features/features/read",
        "Microsoft.Features/operations/read",
        "Microsoft.Features/providers/features/read",

"Microsoft.Features/providers/features/register/action",

"Microsoft.Features/providers/features/unregister/action",

"Microsoft.Features/subscriptionFeatureRegistrations/read"
    ],
    "notActions": [],
    "dataActions": [],
    "notDataActions": []
  }
]
}
}

```

- Im Azure location Das enthält mindestens eine ["Delegiertes Subnetz"](#). Ab Trident 22.01 finden Sie das location Parameter ist ein erforderliches Feld auf der obersten Ebene der Backend-Konfigurationsdatei. In virtuellen Pools angegebene Standortwerte werden ignoriert.
- Zu verwenden Cloud Identity, Holen Sie sich die client ID Von A ["Vom Benutzer zugewiesene verwaltete Identität"](#) Und geben Sie diese ID in an azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.

Zusätzliche Anforderungen für SMB Volumes

Zur Erstellung eines SMB-Volumes müssen folgende Voraussetzungen erfüllt sein:

- Active Directory konfiguriert und mit Azure NetApp Files verbunden. Siehe ["Microsoft: Erstellen und Verwalten von Active Directory-Verbindungen für Azure NetApp Files"](#).
- Kubernetes-Cluster mit einem Linux-Controller-Knoten und mindestens einem Windows-Worker-Node, auf dem Windows Server 2019 ausgeführt wird. Astra Trident unterstützt SMB Volumes, die nur auf Windows Nodes laufenden Pods gemountet werden.
- Mindestens ein Astra Trident-Schlüssel mit Ihren Active Directory-Anmeldeinformationen, damit Azure NetApp Files sich bei Active Directory authentifizieren kann. Um Geheimnis zu erzeugen smbcreds:

```

kubectl create secret generic smbcreds --from-literal username=user
--from-literal password='password'

```

- Ein CSI-Proxy, der als Windows-Dienst konfiguriert ist. Zum Konfigurieren von A `csi-proxy` Weitere

Informationen finden Sie unter ["GitHub: CSI-Proxy"](#) Oder ["GitHub: CSI Proxy für Windows"](#) Für Kubernetes-Knoten, die auf Windows ausgeführt werden.

Azure NetApp Files Back-End-Konfigurationsoptionen und -Beispiele

Informieren Sie sich über die Backend-Konfigurationsoptionen NFS und SMB für Azure NetApp Files und sehen Sie sich Konfigurationsbeispiele an.

Back-End-Konfigurationsoptionen

Astra Trident erstellt mithilfe Ihrer Backend-Konfiguration (Subnetz, virtuelles Netzwerk, Service-Level und Standort) Azure NetApp Files Volumes in Kapazitätspools, die am angeforderten Speicherort verfügbar sind und mit dem angeforderten Service-Level und Subnetz übereinstimmen.



Astra Trident unterstützt keine manuellen QoS-Kapazitäts-Pools.

Azure NetApp Files Back-Ends bieten diese Konfigurationsoptionen.

Parameter	Beschreibung	Standard
version		Immer 1
storageDriverName	Name des Speichertreibers	„azure-netapp-files“
backendName	Benutzerdefinierter Name oder das Storage-Backend	Treibername + „_“ + zufällige Zeichen
subscriptionID	Die Abonnement-ID Ihres Azure Abonnements Optional, wenn verwaltete Identitäten auf einem AKS-Cluster aktiviert sind.	
tenantID	Die Mandanten-ID aus einer App-Registrierung Optional, wenn verwaltete Identitäten oder Cloud-Identität auf einem AKS-Cluster verwendet wird.	
clientID	Die Client-ID aus einer App-Registrierung Optional, wenn verwaltete Identitäten oder Cloud-Identität auf einem AKS-Cluster verwendet wird.	
clientSecret	Der Client-Schlüssel aus einer App-Registrierung Optional, wenn verwaltete Identitäten oder Cloud-Identität auf einem AKS-Cluster verwendet wird.	

Parameter	Beschreibung	Standard
serviceLevel	Einer von Standard, Premium, Oder Ultra	„“ (zufällig)
location	Name des Azure Speicherorts, an dem die neuen Volumes erstellt werden Optional, wenn verwaltete Identitäten auf einem AKS-Cluster aktiviert sind.	
resourceGroups	Liste der Ressourcengruppen zum Filtern ermittelter Ressourcen	„[]“ (kein Filter)
netappAccounts	Liste von NetApp Accounts zur Filterung erkannter Ressourcen	„[]“ (kein Filter)
capacityPools	Liste der Kapazitäts-Pools zur Filterung erkannter Ressourcen	„[]“ (kein Filter, zufällig)
virtualNetwork	Name eines virtuellen Netzwerks mit einem delegierten Subnetz	“
subnet	Name eines an delegierten Subnetzes Microsoft.Netapp/volumes	“
networkFeatures	Eventuell Set von vnet-Funktionen für ein Volumen Basic Oder Standard. Netzwerkfunktionen sind nicht in allen Regionen verfügbar und müssen möglicherweise in einem Abonnement aktiviert werden. Angabe networkFeatures Wenn die Funktion nicht aktiviert ist, schlägt die Volume-Bereitstellung fehl.	“

Parameter	Beschreibung	Standard
nfsMountOptions	<p>Engmaschige Kontrolle der NFS-Mount-Optionen</p> <p>Für SMB Volumes ignoriert.</p> <p>Um Volumes mit NFS-Version 4.1 einzubinden, beinhalten <code>nfsvers=4</code> Wählen Sie in der Liste mit durch Komma getrennten Mount-Optionen NFS v4.1 aus.</p> <p>Mount-Optionen, die in einer Storage-Klassen-Definition festgelegt sind, überschreiben Mount-Optionen, die in der Backend-Konfiguration festgelegt sind.</p>	„Nfsvers=3“
limitVolumeSize	Bereitstellung fehlgeschlagen, wenn die angeforderte Volume-Größe über diesem Wert liegt	„“ (nicht standardmäßig durchgesetzt)
debugTraceFlags	<p>Fehler-Flags bei der Fehlerbehebung beheben. Beispiel: <code>\{"api": false, "method": true, "discovery": true\}</code>. Verwenden Sie dies nur, wenn Sie Fehler beheben und einen detaillierten Log Dump benötigen.</p>	Null
nasType	<p>Konfiguration der Erstellung von NFS- oder SMB-Volumes</p> <p>Die Optionen lauten <code>nfs</code>, <code>smb</code> Oder <code>null</code>. Einstellung auf <code>null</code> setzt standardmäßig auf NFS-Volumes.</p>	<code>nfs</code>



Weitere Informationen zu den Netzwerkfunktionen finden Sie unter ["Konfigurieren Sie Netzwerkfunktionen für ein Azure NetApp Files Volume"](#).

Erforderliche Berechtigungen und Ressourcen

Wenn Sie beim Erstellen einer PVC den Fehler „Keine Kapazitätspools gefunden“ erhalten, sind Ihre App-Registrierung wahrscheinlich nicht über die erforderlichen Berechtigungen und Ressourcen (Subnetz, virtuelles Netzwerk, Kapazitäts-Pool) verbunden. Wenn Debug aktiviert ist, protokolliert Astra Trident die Azure Ressourcen, die bei der Erstellung des Backend ermittelt wurden. Überprüfen Sie, ob eine geeignete Rolle verwendet wird.

Die Werte für `resourceGroups`, `netappAccounts`, `capacityPools`, `virtualNetwork`, und `subnet` Kann mit kurzen oder vollqualifizierten Namen angegeben werden. In den meisten Fällen werden vollqualifizierte Namen empfohlen, da kurze Namen mehrere Ressourcen mit demselben Namen entsprechen können.

Der `resourceGroups`, `netappAccounts`, und `capacityPools` Werte sind Filter, die die ermittelten Ressourcen auf die in diesem Storage-Back-End verfügbaren Personen beschränken und in beliebiger Kombination angegeben werden können. Vollqualifizierte Namen folgen diesem Format:

Typ	Formatieren
Ressourcengruppe	<Ressourcengruppe>
NetApp Konto	<Resource Group>/<netapp Account>
Kapazitäts-Pool	<Resource Group>/<netapp Account>/<Capacity Pool>
Virtuelles Netzwerk	<Ressourcengruppe>/<virtuelles Netzwerk>
Subnetz	<Ressourcengruppe>/<virtuelles Netzwerk>/<Subnetz>

Volume-Provisionierung

Sie können die standardmäßige Volume-Bereitstellung steuern, indem Sie die folgenden Optionen in einem speziellen Abschnitt der Konfigurationsdatei angeben. Siehe [Beispielkonfigurationen](#) Entsprechende Details.

Parameter	Beschreibung	Standard
<code>exportRule</code>	Exportregeln für neue Volumes <code>exportRule</code> Muss eine kommagetrennte Liste beliebiger Kombinationen von IPv4-Adressen oder IPv4-Subnetzen in CIDR-Notation sein. Für SMB Volumes ignoriert.	„0.0.0.0/0“
<code>snapshotDir</code>	Steuert die Sichtbarkeit des .Snapshot-Verzeichnisses	„Falsch“
<code>size</code>	Die Standardgröße der neuen Volumes	„100 GB“
<code>unixPermissions</code>	die unix-Berechtigungen neuer Volumes (4 Oktal-Ziffern). Für SMB Volumes ignoriert.	„“ (Vorschau-Funktion, erfordert Whitelisting im Abonnement)

Beispielkonfigurationen

Die folgenden Beispiele zeigen grundlegende Konfigurationen, bei denen die meisten Parameter standardmäßig belassen werden. Dies ist der einfachste Weg, ein Backend zu definieren.

Minimalkonfiguration

Dies ist die absolute minimale Backend-Konfiguration. Mit dieser Konfiguration erkennt Astra Trident alle NetApp-Konten, Kapazitätspools und Subnetze, die an Azure NetApp Files am konfigurierten Standort delegiert wurden. Zudem werden neue Volumes zufällig in einem dieser Pools und Subnetze platziert. Weil `nasType` weggelassen, das `nfs` Standard gilt und das Backend wird für NFS-Volumes bereitgestellt.

Diese Konfiguration ist ideal, wenn Sie gerade erst mit Azure NetApp Files beginnen und Dinge ausprobieren möchten, aber in der Praxis möchten Sie einen zusätzlichen Umfang für die bereitgestellten Volumes angeben.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
```

Verwaltete Identitäten für AKS

Diese Backend-Konfiguration unterlässt `subscriptionID`, `tenantID`, `clientID`, und `clientSecret`, Die bei der Verwendung von verwalteten Identitäten optional sind.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  capacityPools: ["ultra-pool"]
  resourceGroups: ["aks-ami-eastus-rg"]
  netappAccounts: ["smb-na"]
  virtualNetwork: eastus-prod-vnet
  subnet: eastus-anf-subnet
```

Cloud-Identität für AKS

Diese Backend-Konfiguration unterlässt `tenantID`, `clientID`, und `clientSecret`, Die bei Verwendung einer Cloud-Identität optional sind.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  capacityPools: ["ultra-pool"]
  resourceGroups: ["aks-ami-eastus-rg"]
  netappAccounts: ["smb-na"]
  virtualNetwork: eastus-prod-vnet
  subnet: eastus-anf-subnet
  location: eastus
  subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
```

Spezifische Service-Level-Konfiguration mit Filtern nach Kapazitäts-Pools

Bei dieser Back-End-Konfiguration werden Volumes in Azure platziert `eastus` Standort in einem `Ultra` Kapazitäts-Pool: Astra Trident erkennt automatisch alle an Azure NetApp Files delegierten Subnetze an diesem Standort und platziert ein neues Volume zufällig in einem davon.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
serviceLevel: Ultra
capacityPools:
- application-group-1/account-1/ultra-1
- application-group-1/account-1/ultra-2
```

Erweiterte Konfiguration

Diese Back-End-Konfiguration reduziert den Umfang der Volume-Platzierung auf ein einzelnes Subnetz und ändert auch einige Standardwerte für die Volume-Bereitstellung.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
serviceLevel: Ultra
capacityPools:
- application-group-1/account-1/ultra-1
- application-group-1/account-1/ultra-2
virtualNetwork: my-virtual-network
subnet: my-subnet
networkFeatures: Standard
nfsMountOptions: vers=3,proto=tcp,timeo=600
limitVolumeSize: 500Gi
defaults:
  exportRule: 10.0.0.0/24,10.0.1.0/24,10.0.2.100
  snapshotDir: 'true'
  size: 200Gi
  unixPermissions: '0777'
```

Konfiguration des virtuellen Pools

Diese Back-End-Konfiguration definiert mehrere Storage-Pools in einer einzelnen Datei. Dies ist nützlich, wenn Sie über mehrere Kapazitäts-Pools verfügen, die unterschiedliche Service-Level unterstützen, und Sie Storage-Klassen in Kubernetes erstellen möchten, die diese unterstützen. Virtuelle Pool-Labels wurden verwendet, um die Pools basierend auf zu differenzieren `performance`.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
resourceGroups:
- application-group-1
networkFeatures: Basic
nfsMountOptions: vers=3,proto=tcp,timeo=600
labels:
  cloud: azure
storage:
- labels:
    performance: gold
    serviceLevel: Ultra
    capacityPools:
    - ultra-1
    - ultra-2
    networkFeatures: Standard
- labels:
    performance: silver
    serviceLevel: Premium
    capacityPools:
    - premium-1
- labels:
    performance: bronze
    serviceLevel: Standard
    capacityPools:
    - standard-1
    - standard-2
```

Definitionen der Storage-Klassen

Im Folgenden `StorageClass` Definitionen beziehen sich auf die oben genannten Speicherpools.

Beispieldefinitionen mit `parameter.selector` Feld

Wird Verwendet `parameter.selector` Sie können für jedes angeben `StorageClass` Der virtuelle Pool, der zum Hosten eines Volumes genutzt wird. Im Volume werden die Aspekte definiert, die im ausgewählten Pool definiert sind.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=gold"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: silver
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: bronze
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=bronze"
allowVolumeExpansion: true
```

Beispieldefinitionen für SMB Volumes

Wird Verwendet `nasType`, `node-stage-secret-name`, und `node-stage-secret-namespace`, Sie können ein SMB-Volume angeben und die erforderlichen Active Directory-Anmeldeinformationen angeben.

Grundkonfiguration im Standard-Namespace

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
```

Verschiedene Schlüssel pro Namespace verwenden

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

Verschiedene Geheimnisse pro Band verwenden

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: ${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```



`nasType: smb` Filter für Pools, die SMB-Volumes unterstützen `nasType: nfs` Oder `nasType: null` Filter für NFS Pools.

Erstellen Sie das Backend

Führen Sie nach dem Erstellen der Back-End-Konfigurationsdatei den folgenden Befehl aus:

```
tridentctl create backend -f <backend-file>
```

Wenn die Backend-Erstellung fehlschlägt, ist mit der Back-End-Konfiguration ein Fehler aufgetreten. Sie können die Protokolle zur Bestimmung der Ursache anzeigen, indem Sie den folgenden Befehl ausführen:

```
tridentctl logs
```

Nachdem Sie das Problem mit der Konfigurationsdatei identifiziert und korrigiert haben, können Sie den Befehl „Erstellen“ erneut ausführen.

Cloud Volumes Service für Google Cloud-Back-End konfigurieren

Erfahren Sie, wie Sie NetApp Cloud Volumes Service für Google Cloud mit den vorgegebenen Beispielkonfigurationen als Backend für Ihre Astra Trident Installation konfigurieren.

Treiberdetails zu Google Cloud

Astra Trident bietet die `gcp-cvs` Treiber für die Kommunikation mit dem Cluster. Unterstützte Zugriffsmodi sind: *ReadWriteOnce* (RWO), *ReadOnly Many* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Treiber	Protokoll	VolumeModu s	Unterstützte Zugriffsmodi	Unterstützte Filesysteme
<code>gcp-cvs</code>	NFS	Dateisystem	RWO, ROX, RWX, RWOP	<code>nfs</code>

Erfahren Sie mehr über den Astra Trident Support für Cloud Volumes Service für Google Cloud

Astra Trident kann Cloud Volumes Service Volumes in einem von zwei erstellen "[Servicetypen](#)":

- **CVS-Performance:** Der Standard Astra Trident Service-Typ. Dieser Performance-optimierte Service-Typ ist ideal für Produktions-Workloads, die Performance schätzen. Der CVS-Performance-Servicetyp ist eine Hardwareoption, die Volumes mit einer Größe von mindestens 100 gib unterstützt. Sie können eine von auswählen "[Drei Service-Level](#)":
 - `standard`
 - `premium`
 - `extreme`
- **CVS:** Der CVS-Servicetyp bietet eine hohe zonale Verfügbarkeit bei begrenzten bis moderaten Leistungsstufen. Der CVS-Servicetyp ist eine Software-Option, die Storage Pools zur Unterstützung von

Volumes mit einer Größe von 1 gib verwendet. Der Speicherpool kann bis zu 50 Volumes enthalten, in denen sich alle Volumes die Kapazität und Performance des Pools teilen. Sie können eine von auswählen ["Zwei Service-Level"](#):

- standardsw
- zoneredundantstandardsw

Was Sie benötigen

Um den zu konfigurieren und zu verwenden ["Cloud Volumes Service für Google Cloud"](#) Back-End, Sie benötigen Folgendes:

- Ein Google Cloud Konto, das mit NetApp Cloud Volumes Service konfiguriert ist
- Projektnummer Ihres Google Cloud-Kontos
- Google Cloud-Servicekonto bei `netappcloudvolumes.admin` Rolle
- API-Schlüsseldatei für Ihr Cloud Volumes Service-Konto

Back-End-Konfigurationsoptionen

Jedes Back-End stellt Volumes in einer einzigen Google Cloud-Region bereit. Um Volumes in anderen Regionen zu erstellen, können Sie zusätzliche Back-Ends definieren.

Parameter	Beschreibung	Standard
version		Immer 1
storageDriverName	Name des Speichertreibers	„gcp-cvs“
backendName	Benutzerdefinierter Name oder das Storage-Backend	Treibernamen + „_“ + Teil des API-Schlüssels
storageClass	Optionaler Parameter zur Angabe des CVS-Servicetyps. Nutzung <code>software</code> Wählen Sie den CVS-Diensttyp aus. Anderenfalls übernimmt Astra Trident den Servicetyp CVS-Performance (<code>hardware</code>).	
storagePools	CVS-Diensttyp nur. Optionaler Parameter zur Angabe von Speicherpools für die Volume-Erstellung.	
projectNumber	Google Cloud Account Projektnummer. Der Wert ist auf der Startseite des Google Cloud Portals zu finden.	
hostProjectNumber	Erforderlich bei Verwendung eines gemeinsamen VPC-Netzwerks. In diesem Szenario <code>projectNumber</code> ist das Service-Projekt, und <code>hostProjectNumber</code> ist das Hostprojekt.	

Parameter	Beschreibung	Standard
apiRegion	<p>In der Google Cloud-Region, in der Astra Trident Cloud Volumes Service Volumes erstellt. Wenn regionenübergreifende Kubernetes-Cluster erstellt werden, werden Volumes in einem erstellt apiRegion Können in Workloads verwendet werden, die auf Nodes über mehrere Google Cloud Regionen hinweg geplant sind.</p> <p>Der regionale Verkehr verursacht zusätzliche Kosten.</p>	
apiKey	<p>API-Schlüssel für das Google Cloud-Dienstkonto bei netappcloudvolumes.admin Rolle:</p> <p>Er enthält den JSON-formatierten Inhalt der privaten Schlüsseldatei eines Google Cloud-Dienstkontos (wortgetreu in die Back-End-Konfigurationsdatei kopiert).</p>	
proxyURL	<p>Proxy-URL, wenn Proxyserver für die Verbindung mit dem CVS-Konto benötigt wird. Der Proxy-Server kann entweder ein HTTP-Proxy oder ein HTTPS-Proxy sein.</p> <p>Bei einem HTTPS-Proxy wird die Zertifikatvalidierung übersprungen, um die Verwendung von selbstsignierten Zertifikaten im Proxyserver zu ermöglichen.</p> <p>Proxy-Server mit aktivierter Authentifizierung werden nicht unterstützt.</p>	
nfsMountOptions	Engmaschige Kontrolle der NFS-Mount-Optionen	„Nfsvers=3“
limitVolumeSize	Bereitstellung fehlgeschlagen, wenn die angeforderte Volume-Größe über diesem Wert liegt.	„“ (nicht standardmäßig durchgesetzt)
serviceLevel	<p>Das CVS-Performance oder CVS Service-Level für neue Volumes.</p> <p>CVS-Performance Werte sind standard, premium, Oder extreme.</p> <p>CVS-Werte sind standardsw Oder zoneredundantstandardsw.</p>	<p>CVS-Performance ist der Standard.</p> <p>Der CVS-Standardwert ist „standardsw“.</p>
network	Für Cloud Volumes Service Volumes verwendetes Google Cloud Netzwerk	„Standard“
debugTraceFlags	<p>Fehler-Flags bei der Fehlerbehebung beheben. Beispiel: \{"api":false, "method":true}.</p> <p>Verwenden Sie dies nur, wenn Sie Fehler beheben und einen detaillierten Log Dump benötigen.</p>	Null

Parameter	Beschreibung	Standard
allowedTopologies	<p>Damit Sie regionsübergreifenden Zugriff ermöglichen, wird Ihre StorageClass-Definition für verwendet <code>allowedTopologies</code> Muss alle Regionen umfassen.</p> <p>Beispiel:</p> <ul style="list-style-type: none"> - <code>key: topology.kubernetes.io/region</code> <code>values:</code> - <code>us-east1</code> - <code>europa-west1</code> 	

Optionen zur Volume-Bereitstellung

Sie können die Standard-Volume-Bereitstellung im steuern `defaults` Abschnitt der Konfigurationsdatei.

Parameter	Beschreibung	Standard
exportRule	Die Exportregeln für neue Volumes. Muss eine kommasetrennte Liste beliebiger Kombinationen von IPv4-Adressen oder IPv4-Subnetzen in CIDR-Notation sein.	„0.0.0.0/0“
snapshotDir	Zugriff auf die <code>.snapshot</code> Verzeichnis	„Falsch“
snapshotReserve	Prozentsatz des für Snapshots reservierten Volumes	"" (CVS Standard 0 akzeptieren)
size	<p>Die Größe neuer Volumes.</p> <p>Die Mindestmenge von CVS-Performance beträgt 100 gib.</p> <p>CVS mindestens 1 gib.</p>	<p>Der Servicetyp CVS-Performance ist standardmäßig auf „100 gib“ eingestellt.</p> <p>CVS-Diensttyp setzt keine Standardeinstellung, erfordert jedoch mindestens 1 gib.</p>

Beispiele für CVS-Performance-Diensttypen

Die folgenden Beispiele enthalten Beispielkonfigurationen für den CVS-Performance-Servicetyp.

Dies ist die minimale Backend-Konfiguration, die den standardmäßigen CVS-Performance-Servicetyp mit dem Standard-Service Level verwendet.

28

```
auth_uri: https://accounts.google.com/o/oauth2/auth
token_uri: https://oauth2.googleapis.com/token
auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
```

[illegible]


```
auth_uri: https://accounts.google.com/o/oauth2/auth
token_uri: https://oauth2.googleapis.com/token
auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
proxyURL: http://proxy-server-hostname/
nfsMountOptions: vers=3,proto=tcp,timeo=600
limitVolumeSize: 10Ti
serviceLevel: premium
defaults:
  snapshotDir: 'true'
  snapshotReserve: '5'
  exportRule: 10.0.0.0/24,10.0.1.0/24,10.0.2.100
  size: 5Ti
```

Beispiel 3: Virtuelle Pool-Konfiguration

Dieses Beispiel verwendet `storage` Um virtuelle Pools und die zu konfigurieren `StorageClasses` Die sich auf sie beziehen. Siehe [Definitionen der Storage-Klassen](#) Um zu sehen, wie die Speicherklassen definiert wurden.

Hier werden für alle virtuellen Pools, die das festlegen, spezifische Standardeinstellungen festgelegt `snapshotReserve` Bei 5% und der `exportRule` Zu 0.0.0.0/0. Die virtuellen Pools werden im definiert `storage` Abschnitt. Jeder individuelle virtuelle Pool definiert seine eigenen `serviceLevel`, Und einige Pools überschreiben die Standardwerte. Virtuelle Pool-Labels wurden verwendet, um die Pools basierend auf zu differenzieren `performance` Und `protection`.

[illegible]

```

znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m
XsYg6gyxy4zq7OlwWgLwGa==
-----END PRIVATE KEY-----
client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
client_id: '123456789012345678901'
auth_uri: https://accounts.google.com/o/oauth2/auth
token_uri: https://oauth2.googleapis.com/token
auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
nfsMountOptions: vers=3,proto=tcp,timeo=600
defaults:
  snapshotReserve: '5'
  exportRule: 0.0.0.0/0
labels:
  cloud: gcp
region: us-west2
storage:
- labels:
  performance: extreme
  protection: extra
  serviceLevel: extreme
  defaults:
    snapshotDir: 'true'
    snapshotReserve: '10'
    exportRule: 10.0.0.0/24
- labels:
  performance: extreme
  protection: standard
  serviceLevel: extreme
- labels:
  performance: premium
  protection: extra
  serviceLevel: premium
  defaults:
    snapshotDir: 'true'
    snapshotReserve: '10'
- labels:
  performance: premium
  protection: standard
  serviceLevel: premium
- labels:
  performance: standard

```

```
serviceLevel: standard
```

Definitionen der Storage-Klassen

Die folgenden StorageClass-Definitionen gelten für das Beispiel der virtuellen Pool-Konfiguration. Wird verwendet `parameters.selector`, Sie können für jede StorageClass den virtuellen Pool angeben, der zum Host eines Volumes verwendet wird. Im Volume werden die Aspekte definiert, die im ausgewählten Pool definiert sind.

Beispiel für Storage-Klasse

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-extra-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=extreme; protection=extra"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-standard-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=premium; protection=standard"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-premium-extra-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=premium; protection=extra"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-premium
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=premium; protection=standard"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-standard
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=standard"
allowVolumeExpansion: true
```

```
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extra-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=extra"
allowVolumeExpansion: true
```

- Die erste StorageClass (`cvs-extreme-extra-protection`) Karten zum ersten virtuellen Pool. Dies ist der einzige Pool, der eine extreme Performance mit einer Snapshot-Reserve von 10 % bietet.
- Die letzte StorageClass (`cvs-extra-protection`) Ruft alle Speicher-Pool, die eine Snapshot-Reserve von 10% bietet. Astra Trident entscheidet, welcher Virtual Pool ausgewählt wird und stellt sicher, dass die Anforderungen an die Snapshot-Reserve erfüllt werden.

Beispiele für CVS-Diensttypen

Die folgenden Beispiele enthalten Beispielkonfigurationen für den CVS-Servicetyp.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
storageClass: software
apiRegion: us-east4
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qP8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
```

```
client_id: '123456789012345678901'
auth_uri: https://accounts.google.com/o/oauth2/auth
token_uri: https://oauth2.googleapis.com/token
auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
serviceLevel: standardsw
```


Beispiel 2: Konfiguration des Storage Pools

Diese Beispiel-Back-End-Konfiguration verwendet `storagePools`. So konfigurieren Sie einen Speicherpool:

```
---
version: 1
storageDriverName: gcp-cvs
backendName: gcp-std-so-with-pool
projectNumber: '531265380079'
apiRegion: europe-west1
apiKey:
  type: service_account
  project_id: cloud-native-data
  private_key_id: "<id_value>"
  private_key: |-
    -----BEGIN PRIVATE KEY-----
    MIEvAIBADANBgkqhkiG9w0BAQEFAASCBywggSiAgEAAoIBAQDaT+Oui9FBAw19
    L1AGEkrYU5xd9K5NlO5jMkIFND5wCD+Nv+jd1GvtFRLaLK5RvXyF5wzvztmODNS+
    qtScpQ+5cFpQkuGtv9U9+N6qtuVYYO3b504Kp5CtqVPJCgMJaK2j8pZTIqUiMum/
    5/Y9oTbZrjAHSMsgJm2nHzFq2X0rqVMAHghI6ATm4DOuWx8XGWKTGIPlc0qPqJlqS
    LLaWOH4VIZQZCAyW5IU9PCAmwqHgdG0uhFNfCgMmED6PBUvVLsLvcq86X+QSWR9k
    ETqElj/sGCenPF7ti1DhGBFafd9hPnxg9PZY29ArEZwY9G/ZjZQX7WPgs0VvxiNR
    DxZRC3GXAgMBAAECggEACn5c59bG/qnVEVI1CwMAalM5M2z09JFhlL1ljKwntNPj
    Vilw2eTW2+UE7HbJru/S7KQgA5Dnn9kvCraEahPRuddUMrD0vG4kTl/IODV6uFuk
    Y0sZfbqd4jMUQ21smvGsqFzwloYWS5qzO1W83ivXH/HW/iqkmY2eW+EPRS/hwSSu
    SscR+SojI7PB0BWSJhlV4yqYf3vcD/D95el2CVHfRCkL85DKumeZ+yHENpiXGZAE
    t8xSs4a50OPm6NHhevCw2a/UQ95/foXNUR450HtbjieJo5o+FF6EYZQGfU2ZHZO8
    37FBKuaJkdGW5xqaI9TL7aqkGkFMF4F2qvOZM+vy8QKBgQD4oVuOkJDlhkTHP86W
    esFlwlkpWyJR9ZA7LI0g/rVpslnX+XdDq0WQf4umDLNau5hYEH9LU6ZSGs1Xk3/B
    NHwR6OXFuqEKNiu83d0zSlHhTy7PZpOZdj5a/vVvQfPDMz7OvsqLRd7YCAbdzuQ0
    +Ahq0Ztwvg0HQ64hdW0ukpYRRwKBgQDgyHj98oqswoYuIa+pPlYs0pPwLmjwKyNm
    /HayzCp+Qjiyy7Tzg8AUqlH1Ou83Xbv428jvg7kDh07PCCKFq+mMmfqHmTpb0Maq
    KpKnZg4ipsqPlyHNNEOrmcailXbwIhCLewMqMrggUiLOmCw4PscL5nK+4GKu2XE1
    jLqjWAZFMQKBgFHkQ9XXRAJlkr3XpGHOgn890pZOkCVSrqu6aUef/5KYlFCt8ew
    F/+aIxM2iQSVmWQYOvVCnhuY/F2GfAQ7d0om3decuwIOCX/xy7PjHMkLXa2uaZs4
    WR17sLduj62RqXRLX0c0QkwBiNFyHbRcpdkZJQujbYMHba+7j7SxT4BtAoGAWMWT
    UucocRXZm/pdvz9wteNH3YDwnJLMxm1KC06qMXbBoYrliY4sm3ywJWMC+iCd/H8A
    Gecxd/xVu5mA2L2N3KMq18Zhz8Th0G5DwKyDRJgOQ0Q46yuNXOoYEjlo4Wjyk8Me
    +tlQ8iK98E0UmZnhTgfSpSNElbz2AqnzQ3MN9uECgYAqdvdVPnKGfvdTz2DjyMoJ
    E89UIC41WjjJGmHsd8W65+3X0RwMzKMT6aZc5tK9J5dHvmWIETnbM+1TImdbBFga
    NWOC6f3r2xbGXHhaWSl+nobpTuvlo56ZRJVvVk7lFMsidzMuHH8pxfgNJemwA4P
    ThDHcejv035NNV6Kyo00tA==
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@cloud-native-
  data.iam.gserviceaccount.com
```

```
client_id: '107071413297115343396'
auth_uri: https://accounts.google.com/o/oauth2/auth
token_uri: https://oauth2.googleapis.com/token
auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40cloud-native-data.iam.gserviceaccount.com
storageClass: software
zone: europe-west1-b
network: default
storagePools:
- 1bc7f380-3314-6005-45e9-c7dc8c2d7509
serviceLevel: Standardsw
```

Was kommt als Nächstes?

Führen Sie nach dem Erstellen der Back-End-Konfigurationsdatei den folgenden Befehl aus:

```
tridentctl create backend -f <backend-file>
```

Wenn die Backend-Erstellung fehlschlägt, ist mit der Back-End-Konfiguration ein Fehler aufgetreten. Sie können die Protokolle zur Bestimmung der Ursache anzeigen, indem Sie den folgenden Befehl ausführen:

```
tridentctl logs
```

Nachdem Sie das Problem mit der Konfigurationsdatei identifiziert und korrigiert haben, können Sie den Befehl „Erstellen“ erneut ausführen.

Konfigurieren Sie ein NetApp HCI- oder SolidFire-Backend

Erstellen und Verwenden eines Element Backend mit der Astra Trident Installation

Details zum Elementtreiber

Astra Trident bietet die `solidfire-san` Speichertreiber für die Kommunikation mit dem Cluster. Unterstützte Zugriffsmodi sind: *ReadWriteOnce* (RWO), *ReadOnly Many* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Der `solidfire-san` Der Speichertreiber unterstützt die Volume-Modi *File* und *Block*. Für das Filesystem VolumeMode erstellt Astra Trident ein Volume und erstellt ein Dateisystem. Der Dateisystem-Typ wird von StorageClass angegeben.

Treiber	Protokoll	VolumeMode	Unterstützte Zugriffsmodi	Unterstützte Filesysteme
solidfire-san	ISCSI	Block-Storage	RWO, ROX, RWX, RWOP	Kein Dateisystem. Rohes Blockgerät.
solidfire-san	ISCSI	Dateisystem	RWO, RWOP	xfs, ext3, ext4

Bevor Sie beginnen

Sie benötigen Folgendes, bevor Sie ein Element-Backend erstellen.

- Ein unterstütztes Storage-System, auf dem die Element Software ausgeführt wird.
- Anmeldedaten für einen NetApp HCI/SolidFire Cluster-Administrator oder einen Mandantenbenutzer, der Volumes managen kann
- Alle Kubernetes-Worker-Nodes sollten die entsprechenden iSCSI-Tools installiert haben. Siehe ["Informationen zur Vorbereitung auf den Worker-Node"](#).

Back-End-Konfigurationsoptionen

Die Back-End-Konfigurationsoptionen finden Sie in der folgenden Tabelle:

Parameter	Beschreibung	Standard
version		Immer 1
storageDriverName	Name des Speichertreibers	Immer „solidfire-san“
backendName	Benutzerdefinierter Name oder das Storage-Backend	IP-Adresse „SolidFire_“ + Storage (iSCSI)
Endpoint	MVIP für den SolidFire-Cluster mit Mandanten-Anmeldedaten	
SVIP	Speicher-IP-Adresse und -Port	
labels	Satz willkürlicher JSON-formatierter Etiketten für Volumes.	“
TenantName	Zu verwendende Mandantenbezeichnung (wird erstellt, wenn sie nicht gefunden wurde)	
InitiatorIFace	Beschränken Sie den iSCSI-Datenverkehr auf eine bestimmte Host-Schnittstelle	„Standard“
UseCHAP	Verwenden Sie CHAP zur Authentifizierung von iSCSI. Astra Trident verwendet CHAP.	Richtig
AccessGroups	Liste der zu verwendenden Zugriffsgruppen-IDs	Findet die ID einer Zugriffsgruppe namens „Dreizack“
Types	QoS-Spezifikationen	

Parameter	Beschreibung	Standard
limitVolumeSize	Bereitstellung fehlgeschlagen, wenn die angeforderte Volume-Größe über diesem Wert liegt	„ (nicht standardmäßig durchgesetzt)
debugTraceFlags	Fehler-Flags bei der Fehlerbehebung beheben. Beispiel: { „API“:false, „Methode“:true}	Null



Verwenden Sie es nicht debugTraceFlags Es sei denn, Sie beheben Fehler und benötigen einen detaillierten Log Dump.

Beispiel 1: Back-End-Konfiguration für solidfire-san Treiber mit drei Lautstärketypen

Dieses Beispiel zeigt eine Backend-Datei mit CHAP-Authentifizierung und Modellierung von drei Volume-Typen mit spezifischen QoS-Garantien. Sehr wahrscheinlich würden Sie dann Storage-Klassen definieren, um jeden davon mit dem zu nutzen IOPS Parameter für Storage-Klasse.

```
---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0
SVIP: "<svip>:3260"
TenantName: "<tenant>"
labels:
  k8scluster: dev1
  backend: dev1-element-cluster
UseCHAP: true
Types:
- Type: Bronze
  Qos:
    minIOPS: 1000
    maxIOPS: 2000
    burstIOPS: 4000
- Type: Silver
  Qos:
    minIOPS: 4000
    maxIOPS: 6000
    burstIOPS: 8000
- Type: Gold
  Qos:
    minIOPS: 6000
    maxIOPS: 8000
    burstIOPS: 10000
```

Beispiel 2: Back-End- und Storage-Class-Konfiguration für solidfire-san Treiber mit virtuellen Pools

Dieses Beispiel zeigt die mit virtuellen Pools zusammen mit StorageClasses konfigurierte Back-End-Definitionsdatei.

Astra Trident kopiert beim Provisioning die auf einem Storage-Pool vorhandenen Labels auf die Back-End-Storage-LUN. Storage-Administratoren können Labels je virtuellen Pool definieren und Volumes nach Label gruppieren.

In der unten gezeigten Beispiel-Backend-Definitionsdatei werden für alle Speicherpools spezifische Standardwerte festgelegt, die die definieren `type` Bei Silver. Die virtuellen Pools werden im definiert `storage` Abschnitt. In diesem Beispiel legen einige Speicherpools ihren eigenen Typ fest, und einige Pools überschreiben die oben festgelegten Standardwerte.

```
---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0
SVIP: "<svip>:3260"
TenantName: "<tenant>"
UseCHAP: true
Types:
- Type: Bronze
  Qos:
    minIOPS: 1000
    maxIOPS: 2000
    burstIOPS: 4000
- Type: Silver
  Qos:
    minIOPS: 4000
    maxIOPS: 6000
    burstIOPS: 8000
- Type: Gold
  Qos:
    minIOPS: 6000
    maxIOPS: 8000
    burstIOPS: 10000
type: Silver
labels:
  store: solidfire
  k8scluster: dev-1-cluster
region: us-east-1
storage:
- labels:
    performance: gold
    cost: '4'
  zone: us-east-1a
  type: Gold
```

```
- labels:
  performance: silver
  cost: '3'
  zone: us-east-1b
  type: Silver
- labels:
  performance: bronze
  cost: '2'
  zone: us-east-1c
  type: Bronze
- labels:
  performance: silver
  cost: '1'
  zone: us-east-1d
```

Die folgenden StorageClass-Definitionen beziehen sich auf die oben genannten virtuellen Pools. Verwenden der `parameters.selector` Feld gibt in jeder StorageClass an, welche virtuellen Pools zum Hosten eines Volumes verwendet werden können. Auf dem Volume werden die Aspekte im ausgewählten virtuellen Pool definiert.

Die erste StorageClass (`solidfire-gold-four`) Wird dem ersten virtuellen Pool zugeordnet. Dies ist der einzige Pool, der Gold Performance mit einem bietet `Volume Type QoS` Von Gold. Die letzte StorageClass (`solidfire-silver`) Bezeichnet jeden Speicherpool, der eine silberne Leistung bietet. Astra Trident entscheidet, welcher virtuelle Pool ausgewählt wird und stellt sicher, dass die Storage-Anforderungen erfüllt werden.

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-gold-four
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=gold; cost=4"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-three
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver; cost=3"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-bronze-two
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=bronze; cost=2"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-one
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver; cost=1"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver"
  fsType: "ext4"

```

Weitere Informationen

- ["Volume-Zugriffsgruppen"](#)

ONTAP SAN-Treiber

Übersicht über ONTAP SAN-Treiber

Erfahren Sie mehr über die Konfiguration eines ONTAP Backend mit ONTAP- und Cloud Volumes ONTAP-SAN-Treibern.

Details zum ONTAP-SAN-Treiber

Astra Trident bietet die folgenden SAN-Storage-Treiber für die Kommunikation mit dem ONTAP Cluster. Unterstützte Zugriffsmodi sind: *ReadWriteOnce* (RWO), *ReadOnly Many* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).



Wenn Sie Astra Control für Schutz, Recovery und Mobilität verwenden, lesen Sie bitte [Treiberkompatibilität bei Astra Control](#).

Treiber	Protokoll	VolumeModus	Unterstützte Zugriffsmodi	Unterstützte Filesysteme
ontap-san	ISCSI	Block-Storage	RWO, ROX, RWX, RWOP	Kein Filesystem, rohes Block-Gerät
ontap-san	ISCSI	Dateisystem	RWO, RWOP ROX und RWX sind im Filesystem-Volume-Modus nicht verfügbar.	xfs, ext3, ext4
ontap-san	NVMe/TCP Siehe Weitere Überlegungen zu NVMe/TCP .	Block-Storage	RWO, ROX, RWX, RWOP	Kein Filesystem, rohes Block-Gerät
ontap-san	NVMe/TCP Siehe Weitere Überlegungen zu NVMe/TCP .	Dateisystem	RWO, RWOP ROX und RWX sind im Filesystem-Volume-Modus nicht verfügbar.	xfs, ext3, ext4
ontap-san-economy	ISCSI	Block-Storage	RWO, ROX, RWX, RWOP	Kein Filesystem, rohes Block-Gerät

Treiber	Protokoll	VolumeModus	Unterstützte Zugriffsmodi	Unterstützte Filesysteme
ontap-san-economy	ISCSI	Dateisystem	RWO, RWOP ROX und RWX sind im Filesystem-Volume-Modus nicht verfügbar.	xfs, ext3, ext4

Treiberkompatibilität bei Astra Control

Astra Control bietet nahtlosen Schutz, Disaster Recovery und Mobilität (Verschieben von Volumes zwischen Kubernetes Clustern) für Volumes, die mit der erstellt wurden `ontap-nas`, `ontap-nas-flexgroup`, und `ontap-san` Treiber. Siehe ["Voraussetzungen für die Astra Control Replikation"](#) Entsprechende Details.



- Nutzung `ontap-san-economy` Nur wenn die Nutzungszahl für persistente Volumes voraussichtlich höher ist als ["Unterstützte ONTAP-Volume-Größen"](#).
- Nutzung `ontap-nas-economy` Nur wenn die Nutzungszahl für persistente Volumes voraussichtlich höher ist als ["Unterstützte ONTAP-Volume-Größen"](#) Und das `ontap-san-economy` Treiber kann nicht verwendet werden.
- Verwenden Sie ihn nicht `ontap-nas-economy` Wenn Sie die Notwendigkeit von Datensicherung, Disaster Recovery oder Mobilität erwarten.

Benutzerberechtigungen

Astra Trident erwartet, dass er entweder als ONTAP- oder SVM-Administrator ausgeführt wird, in der Regel mit dem `admin` Cluster-Benutzer oder ein `vsadmin` SVM-Benutzer oder ein Benutzer mit einem anderen Namen und derselben Rolle. Astra Trident erwartet, dass bei Amazon FSX für Implementierungen von NetApp ONTAP, über das Cluster entweder als ONTAP- oder SVM-Administrator ausgeführt wird `fsxadmin` Benutzer oder A `vsadmin` SVM-Benutzer oder ein Benutzer mit einem anderen Namen und derselben Rolle. Der `fsxadmin` Der Benutzer ist ein eingeschränkter Ersatz für den Cluster-Admin-Benutzer.



Wenn Sie den verwenden `limitAggregateUsage` Parameter, Berechtigungen für Cluster-Admin sind erforderlich. Bei der Verwendung von Amazon FSX für NetApp ONTAP mit Astra Trident, das `limitAggregateUsage` Der Parameter funktioniert nicht mit dem `vsadmin` Und `fsxadmin` Benutzerkonten. Der Konfigurationsvorgang schlägt fehl, wenn Sie diesen Parameter angeben.

Es ist zwar möglich, eine restriktivere Rolle in ONTAP zu erstellen, die ein Trident-Treiber verwenden kann, wir empfehlen sie jedoch nicht. Bei den meisten neuen Versionen von Trident sind zusätzliche APIs erforderlich, die berücksichtigt werden müssten, was Upgrades schwierig und fehleranfällig macht.

Weitere Überlegungen zu NVMe/TCP

Astra Trident unterstützt das Non-Volatile Memory Express-Protokoll (NVMe) über das `ontap-san` Treiber einschließlich:

- IPv6
- Snapshots und Klone von NVMe Volumes
- Größe eines NVMe Volumes ändern

- Importieren eines NVMe Volumes, das außerhalb von Astra Trident erstellt wurde, damit sein Lebenszyklus durch Astra Trident gemanagt werden kann
- NVMe-natives Multipathing
- Ordnungsgemäßes oder unzumutbar Herunterfahren der K8s-Nodes (24.02)

Astra Trident unterstützt nicht:

- Dh-HMAC-CHAP, das von nativ von NVMe unterstützt wird
- Multipathing für Device Mapper (DM)
- LUKS-Verschlüsselung

Vorbereiten der Konfiguration des Back-End mit ONTAP-SAN-Treibern

Verstehen Sie die Anforderungen und Authentifizierungsoptionen für die Konfiguration eines ONTAP-Backends mit ONTAP-SAN-Treibern.

Anforderungen

Für alle ONTAP Back-Ends benötigt Astra Trident mindestens ein Aggregat, das der SVM zugewiesen ist.

Denken Sie daran, dass Sie auch mehr als einen Treiber ausführen können und Speicherklassen erstellen können, die auf den einen oder anderen verweisen. Beispielsweise könnten Sie A konfigurieren `san-dev` Klasse, die den verwendet `ontap-san` Fahrer und A `san-default` Klasse, die den verwendet `ontap-san-economy` Eins.

Alle Kubernetes-Worker-Nodes müssen über die entsprechenden iSCSI-Tools verfügen. Siehe ["Bereiten Sie den Knoten „Worker“ vor"](#) Entsprechende Details.

Authentifizieren Sie das ONTAP-Backend

Astra Trident bietet zwei Arten der Authentifizierung eines ONTAP-Backend.

- Anmeldeinformationsbasiert: Benutzername und Passwort für einen ONTAP-Benutzer mit den erforderlichen Berechtigungen. Es wird empfohlen, eine vordefinierte Sicherheits-Login-Rolle zu verwenden, wie z. B. `admin` Oder `vsadmin` Für maximale Kompatibilität mit ONTAP Versionen.
- Zertifikatsbasiert: Astra Trident kann auch mit einem ONTAP Cluster kommunizieren. Verwenden Sie dazu ein Zertifikat, das auf dem Backend installiert ist. Hier muss die Backend-Definition Base64-kodierte Werte des Client-Zertifikats, des Schlüssels und des vertrauenswürdigen CA-Zertifikats enthalten, sofern verwendet (empfohlen).

Sie können vorhandene Back-Ends aktualisieren, um zwischen auf Anmeldeinformationen basierenden und zertifikatbasierten Methoden zu verschieben. Es wird jedoch immer nur eine Authentifizierungsmethode unterstützt. Um zu einer anderen Authentifizierungsmethode zu wechseln, müssen Sie die vorhandene Methode von der Backend-Konfiguration entfernen.



Wenn Sie versuchen, **sowohl Anmeldeinformationen als auch Zertifikate** bereitzustellen, schlägt die Backend-Erstellung mit einem Fehler fehl, dass mehr als eine Authentifizierungsmethode in der Konfigurationsdatei angegeben wurde.

Aktivieren Sie die Anmeldeinformationsbasierte Authentifizierung

Astra Trident erfordert die Zugangsdaten für einen Administrator mit SVM-Umfang/Cluster-Umfang, um mit dem Backend von ONTAP zu kommunizieren. Es wird empfohlen, die Standard-vordefinierten Rollen wie zu verwenden `admin` Oder `vsadmin`. So ist gewährleistet, dass die Kompatibilität mit künftigen ONTAP Versionen gewährleistet ist, die FunktionsAPIs der künftigen Astra Trident Versionen bereitstellen können. Eine benutzerdefinierte Sicherheits-Login-Rolle kann mit Astra Trident erstellt und verwendet werden, wird aber nicht empfohlen.

Eine Beispiel-Back-End-Definition sieht folgendermaßen aus:

YAML

```
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: password
```

JSON

```
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password"
}
```

Beachten Sie, dass die Backend-Definition der einzige Ort ist, an dem die Anmeldeinformationen im reinen Text gespeichert werden. Nach der Erstellung des Backend werden Benutzernamen/Passwörter mit Base64 codiert und als Kubernetes Secrets gespeichert. Die Erstellung oder Aktualisierung eines Backend ist der einzige Schritt, der Kenntnisse über die Anmeldeinformationen erfordert. Daher ist dieser Vorgang nur für Administratoren und wird vom Kubernetes-/Storage-Administrator ausgeführt.

Aktivieren Sie die zertifikatbasierte Authentifizierung

Neue und vorhandene Back-Ends können ein Zertifikat verwenden und mit dem ONTAP-Back-End kommunizieren. In der Backend-Definition sind drei Parameter erforderlich.

- `ClientCertificate`: Base64-codierter Wert des Clientzertifikats.
- `ClientPrivateKey`: Base64-kodierte Wert des zugeordneten privaten Schlüssels.

- **Trusted CACertificate:** Base64-codierter Wert des vertrauenswürdigen CA-Zertifikats. Bei Verwendung einer vertrauenswürdigen CA muss dieser Parameter angegeben werden. Dies kann ignoriert werden, wenn keine vertrauenswürdige CA verwendet wird.

Ein typischer Workflow umfasst die folgenden Schritte.

Schritte

1. Erzeugen eines Clientzertifikats und eines Schlüssels. Legen Sie beim Generieren den allgemeinen Namen (CN) für den ONTAP-Benutzer fest, der sich authentifizieren soll als.

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=admin"
```

2. Fügen Sie dem ONTAP-Cluster ein vertrauenswürdiges CA-Zertifikat hinzu. Dies kann möglicherweise bereits vom Storage-Administrator übernommen werden. Ignorieren, wenn keine vertrauenswürdige CA verwendet wird.

```
security certificate install -type server -cert-name <trusted-ca-cert-
name> -vserver <vserver-name>
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled
true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca
<cert-authority>
```

3. Installieren Sie das Client-Zertifikat und den Schlüssel (von Schritt 1) auf dem ONTAP-Cluster.

```
security certificate install -type client-ca -cert-name <certificate-
name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

4. Bestätigen Sie, dass die ONTAP-Sicherheitsanmeldungsrolle unterstützt wird `cert` Authentifizierungsmethode.

```
security login create -user-or-group-name admin -application ontapi
-authentication-method cert
security login create -user-or-group-name admin -application http
-authentication-method cert
```

5. Testen Sie die Authentifizierung mithilfe des generierten Zertifikats. <ONTAP Management LIF> und <vServer Name> durch Management-LIF-IP und SVM-Namen ersetzen.

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns="http://www.netapp.com/filer/admin" version="1.21"
vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. Encodieren von Zertifikat, Schlüssel und vertrauenswürdigen CA-Zertifikat mit Base64.

```
base64 -w 0 k8senv.pem >> cert_base64
base64 -w 0 k8senv.key >> key_base64
base64 -w 0 trustedca.pem >> trustedca_base64
```

7. Erstellen Sie das Backend mit den Werten, die aus dem vorherigen Schritt ermittelt wurden.

```
cat cert-backend.json
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "SanBackend",
  "managementLIF": "1.2.3.4",
  "svm": "vserver_test",
  "clientCertificate": "Faaaakkkkeeee...Vaaalllluuuuueeee",
  "clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
  "trustedCACertificate": "QNFinfO...SiqOyN",
  "storagePrefix": "myPrefix_"
}

tridentctl create backend -f cert-backend.json -n trident
+-----+-----+-----+-----+
+-----+-----+
|   NAME   | STORAGE DRIVER |                               UUID                               |
STATE | VOLUMES |
+-----+-----+-----+-----+
+-----+-----+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
online |         0 |
+-----+-----+-----+-----+
+-----+-----+
```

Aktualisieren Sie Authentifizierungsmethoden, oder drehen Sie die Anmeldedaten

Sie können ein vorhandenes Backend aktualisieren, um eine andere Authentifizierungsmethode zu verwenden oder ihre Anmeldedaten zu drehen. Das funktioniert auf beide Arten: Back-Ends, die einen Benutzernamen/ein

Passwort verwenden, können aktualisiert werden, um Zertifikate zu verwenden; Back-Ends, die Zertifikate verwenden, können auf Benutzername/Passwort-basiert aktualisiert werden. Dazu müssen Sie die vorhandene Authentifizierungsmethode entfernen und die neue Authentifizierungsmethode hinzufügen. Verwenden Sie dann die aktualisierte Backend.json-Datei, die die erforderlichen Parameter enthält `tridentctl backend update`.

```
cat cert-backend-updated.json
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "SanBackend",
  "managementLIF": "1.2.3.4",
  "svm": "vserver_test",
  "username": "vsadmin",
  "password": "password",
  "storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend SanBackend -f cert-backend-updated.json -n
trident
```

NAME	STORAGE DRIVER	UUID
SanBackend	ontap-san	586b1cd5-8cf8-428d-a76c-2872713612c1



Bei der Änderung von Passwörtern muss der Speicheradministrator das Kennwort für den Benutzer auf ONTAP aktualisieren. Auf diese Weise folgt ein Backend-Update. Beim Drehen von Zertifikaten können dem Benutzer mehrere Zertifikate hinzugefügt werden. Das Backend wird dann aktualisiert und verwendet das neue Zertifikat. Danach kann das alte Zertifikat aus dem ONTAP Cluster gelöscht werden.

Durch die Aktualisierung eines Backend wird der Zugriff auf Volumes, die bereits erstellt wurden, nicht unterbrochen, und auch die danach erstellten Volume-Verbindungen werden beeinträchtigt. Ein erfolgreiches Backend-Update zeigt, dass Astra Trident mit dem ONTAP-Backend kommunizieren und zukünftige Volume-Operationen verarbeiten kann.

Verbindungen mit bidirektionalem CHAP authentifizieren

Astra Trident kann iSCSI-Sitzungen mit bidirektionalem CHAP für die authentifizieren `ontap-san` Und `ontap-san-economy` Treiber. Hierfür muss die Aktivierung von erforderlich sein `useCHAP` Option in der Back-End-Definition. Wenn eingestellt auf `true`, Astra Trident konfiguriert die Standard-Initiator-Sicherheit der

SVM auf bidirektionales CHAP und setzt den Benutzernamen und die Geheimnisse aus der Backend-Datei. NetApp empfiehlt die Verwendung von bidirektionalem CHAP zur Authentifizierung von Verbindungen. Die folgende Beispielkonfiguration ist verfügbar:

```
---
version: 1
storageDriverName: ontap-san
backendName: ontap_san_chap
managementLIF: 192.168.0.135
svm: ontap_iscsi_svm
useCHAP: true
username: vsadmin
password: password
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSD6cNwxyz
```



Der `useCHAP` Parameter ist eine Boolesche Option, die nur einmal konfiguriert werden kann. Die Standardeinstellung ist „false“. Nachdem Sie die Einstellung auf „true“ gesetzt haben, können Sie sie nicht auf „false“ setzen.

Zusätzlich zu `useCHAP=true`, Das `chapInitiatorSecret`, `chapTargetInitiatorSecret`, `chapTargetUsername`, und `chapUsername` Felder müssen in die Backend-Definition aufgenommen werden. Die Geheimnisse können geändert werden, nachdem ein Backend durch Ausführen erstellt wird `tridentctl update`.

So funktioniert es

Nach Einstellung `useCHAP` Der Storage-Administrator weist Astra Trident an, CHAP im Storage-Back-End zu konfigurieren. Dazu gehört Folgendes:

- Einrichten von CHAP auf der SVM:
 - Wenn der Standard-Initiator-Sicherheitstyp der SVM `none` ist (standardmäßig festgelegt) **und** keine bereits vorhandenen LUNs im Volume vorhanden sind, setzt Astra Trident den Standard-Sicherheitstyp auf `CHAP` Und fahren Sie mit der Konfiguration des CHAP-Initiators und des Zielbenutzernamens und der Schlüssel fort.
 - Wenn die SVM LUNs enthält, aktiviert Astra Trident nicht CHAP auf der SVM. Dadurch wird sichergestellt, dass der Zugriff auf die LUNs, die bereits auf der SVM vorhanden sind, nicht eingeschränkt wird.
- Konfigurieren des CHAP-Initiators und des Ziel-Usernamens und der Schlüssel; diese Optionen müssen in der Back-End-Konfiguration angegeben werden (siehe oben).

Nach der Erstellung des Backend erstellt Astra Trident eine entsprechende `tridentbackend` CRD: Speichert die CHAP-Geheimnisse und Benutzernamen als Kubernetes-Geheimnisse. Alle PVS, die von Astra Trident auf diesem Backend erstellt werden, werden über CHAP gemountet und angeschlossen.

Anmeldedaten rotieren und Back-Ends aktualisieren

Sie können die CHAP-Anmeldeinformationen aktualisieren, indem Sie die CHAP-Parameter im aktualisieren backend.json Datei: Dazu müssen die CHAP-Schlüssel aktualisiert und der verwendet werden tridentctl update Befehl zum Übergeben dieser Änderungen.



Wenn Sie die CHAP-Schlüssel für ein Backend aktualisieren, müssen Sie verwenden tridentctl Um das Backend zu aktualisieren. Aktualisieren Sie die Anmeldeinformationen im Storage-Cluster nicht über die Benutzeroberfläche von CLI/ONTAP, da Astra Trident diese Änderungen nicht übernehmen kann.

```
cat backend-san.json
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "ontap_san_chap",
  "managementLIF": "192.168.0.135",
  "svm": "ontap_iscsi_svm",
  "useCHAP": true,
  "username": "vsadmin",
  "password": "password",
  "chapInitiatorSecret": "cl9qxUpDaTeD",
  "chapTargetInitiatorSecret": "rqxigXgkeUpDaTeD",
  "chapTargetUsername": "iJF4heBRT0TCwxyz",
  "chapUsername": "uh2aNCLSD6cNwxyz",
}
```

```
./tridentctl update backend ontap_san_chap -f backend-san.json -n trident
+-----+-----+-----+-----+
+-----+-----+
|  NAME          | STORAGE DRIVER |                               UUID                               |
STATE | VOLUMES |
+-----+-----+-----+-----+
+-----+-----+
| ontap_san_chap | ontap-san      | aa458f3b-ad2d-4378-8a33-1a472ffbeb5c |
online |      7 |
+-----+-----+-----+-----+
+-----+-----+
```

Bestehende Verbindungen bleiben unbeeinträchtigt, sie bleiben auch weiterhin aktiv, wenn die Anmeldedaten vom Astra Trident auf der SVM aktualisiert werden. Neue Verbindungen verwenden die aktualisierten Anmeldedaten und vorhandene Verbindungen bleiben weiterhin aktiv. Wenn Sie alte PVS trennen und neu verbinden, werden sie die aktualisierten Anmeldedaten verwenden.

ONTAP SAN-Konfigurationsoptionen und -Beispiele

Erfahren Sie, wie Sie ONTAP SAN Treiber für Ihre Astra Trident Installation erstellen und

verwenden. Dieser Abschnitt enthält Beispiele und Details zur Back-End-Konfiguration für die Zuordnung von Back-Ends zu StorageClasses.

Back-End-Konfigurationsoptionen

Die Back-End-Konfigurationsoptionen finden Sie in der folgenden Tabelle:

Parameter	Beschreibung	Standard
version		Immer 1
storageDriverName	Name des Speichertreibers	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economy
backendName	Benutzerdefinierter Name oder das Storage-Backend	Treibername + „_“ + DatenLIF
managementLIF	<p>Die IP-Adresse einer Cluster- oder SVM-Management-LIF.</p> <p>Es kann ein vollständig qualifizierter Domänenname (FQDN) angegeben werden.</p> <p>Kann so eingestellt werden, dass IPv6-Adressen verwendet werden, wenn Astra Trident mit dem IPv6-Flag installiert wurde. IPv6-Adressen müssen in eckigen Klammern definiert werden, z. B. [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].</p> <p>Informationen zur nahtlosen MetroCluster-Umschaltung finden Sie im Beispiel: MetroCluster.</p>	„10.0.0.1“, „[2001:1234:abcd::fefe]“
dataLIF	<p>IP-Adresse des LIF-Protokolls.</p> <p>Nicht für iSCSI angeben. Astra Trident verwendet "ONTAP selektive LUN-Zuordnung" Um die iSCSI LIFs zu ermitteln, die für die Einrichtung einer Multi-Path-Sitzung erforderlich sind. Wenn eine Warnung erzeugt wird dataLIF Ist explizit definiert.</p> <p>Für MetroCluster weglassen. Siehe Beispiel: MetroCluster.</p>	Abgeleitet von SVM
svm	<p>Zu verwendende Storage Virtual Machine</p> <p>Für MetroCluster weglassen. Siehe Beispiel: MetroCluster.</p>	Abgeleitet wenn eine SVM managementLIF Angegeben ist

Parameter	Beschreibung	Standard
useCHAP	Verwenden Sie CHAP, um iSCSI für ONTAP-SAN-Treiber zu authentifizieren [Boolesch]. Auf einstellen <code>true</code> Damit Astra Trident bidirektionales CHAP als Standardauthentifizierung für die im Backend angegebene SVM konfiguriert und verwendet. Siehe " Vorbereiten der Konfiguration des Back-End mit ONTAP-SAN-Treibern " Entsprechende Details.	<code>false</code>
chapInitiatorSecret	CHAP-Initiatorschlüssel. Erforderlich, wenn <code>useCHAP=true</code>	" "
labels	Satz willkürlicher JSON-formatierter Etiketten für Volumes	" "
chapTargetInitiatorSecret	Schlüssel für CHAP-Zielinitiator. Erforderlich, wenn <code>useCHAP=true</code>	" "
chapUsername	Eingehender Benutzername. Erforderlich, wenn <code>useCHAP=true</code>	" "
chapTargetUsername	Zielbenutzername. Erforderlich, wenn <code>useCHAP=true</code>	" "
clientCertificate	Base64-codierter Wert des Clientzertifikats. Wird für zertifikatbasierte Authentifizierung verwendet	" "
clientPrivateKey	Base64-kodierte Wert des privaten Client-Schlüssels. Wird für zertifikatbasierte Authentifizierung verwendet	" "
trustedCACertificate	Base64-kodierte Wert des vertrauenswürdigen CA-Zertifikats. Optional Wird für die zertifikatbasierte Authentifizierung verwendet.	" "
username	Benutzername für die Kommunikation mit dem ONTAP Cluster erforderlich. Wird für die Anmeldeinformationsbasierte Authentifizierung verwendet.	" "
password	Passwort, das für die Kommunikation mit dem ONTAP Cluster erforderlich ist. Wird für die Anmeldeinformationsbasierte Authentifizierung verwendet.	" "
svm	Zu verwendende Storage Virtual Machine	Abgeleitet wenn eine SVM <code>managementLIF</code> Angegeben ist
storagePrefix	Das Präfix wird beim Bereitstellen neuer Volumes in der SVM verwendet. Kann später nicht mehr geändert werden. Um diesen Parameter zu aktualisieren, müssen Sie ein neues Backend erstellen.	<code>trident</code>

Parameter	Beschreibung	Standard
limitAggregateUsage	<p>Bereitstellung fehlgeschlagen, wenn die Nutzung über diesem Prozentsatz liegt.</p> <p>Wenn Sie ein Amazon FSX für das NetApp ONTAP-Back-End verwenden, geben Sie diese bitte nicht an limitAggregateUsage. Die vorhanden fsxadmin Und vsadmin Enthalten Sie nicht die erforderlichen Berechtigungen, um die Aggregatnutzung abzurufen und sie mit Astra Trident zu begrenzen.</p>	„ (nicht standardmäßig durchgesetzt)
limitVolumeSize	<p>Bereitstellung fehlgeschlagen, wenn die angeforderte Volume-Größe über diesem Wert liegt.</p> <p>Schränkt auch die maximale Größe der Volumes ein, die es für qtrees und LUNs managt.</p>	„ (standardmäßig nicht erzwungen)
lunsPerFlexvol	Die maximale Anzahl an LUNs pro FlexVol muss im Bereich [50, 200] liegen.	100
debugTraceFlags	<p>Fehler-Flags bei der Fehlerbehebung beheben. Beispiel, {„API“:false, „method“:true}</p> <p>Verwenden Sie diese Funktion nur, wenn Sie eine Fehlerbehebung durchführen und einen detaillierten Protokollauszug benötigen.</p>	null
useREST	<p>Boolescher Parameter zur Verwendung von ONTAP REST-APIs. Technische Vorschau</p> <p>useREST Wird als Tech-Vorschau bereitgestellt, das für Testumgebungen und nicht für Produktions-Workloads empfohlen wird. Wenn eingestellt auf true, Astra Trident wird ONTAP REST APIs zur Kommunikation mit dem Backend verwenden. Diese Funktion erfordert ONTAP 9.11.1 und höher. Darüber hinaus muss die verwendete ONTAP-Login-Rolle Zugriff auf den haben ontap Applikation. Dies wird durch die vordefinierte zufrieden vsadmin Und cluster-admin Rollen:</p> <p>useREST Wird mit MetroCluster nicht unterstützt.</p> <p>useREST Ist vollständig für NVMe/TCP qualifiziert.</p>	false
sanType	Verwenden Sie, um auszuwählen iscsi Für iSCSI oder nvme Für NVMe/TCP	iscsi Falls leer

Back-End-Konfigurationsoptionen für die Bereitstellung von Volumes

Sie können die Standardbereitstellung mit diesen Optionen im steuern defaults Abschnitt der Konfiguration. Ein Beispiel finden Sie unten in den Konfigurationsbeispielen.

Parameter	Beschreibung	Standard
spaceAllocation	Speicherplatzzuweisung für LUNs	„Wahr“
spaceReserve	Modus für Speicherplatzreservierung; „none“ (Thin) oder „Volume“ (Thick)	„Keine“
snapshotPolicy	Die Snapshot-Richtlinie zu verwenden	„Keine“
qosPolicy	<p>QoS-Richtliniengruppe zur Zuweisung für erstellte Volumes Wählen Sie eine der qosPolicy oder adaptiveQosPolicy pro Storage Pool/Backend.</p> <p>Die Verwendung von QoS Policy Groups mit Astra Trident erfordert ONTAP 9.8 oder höher. Wir empfehlen die Verwendung einer nicht gemeinsam genutzten QoS-Richtliniengruppe und stellen sicher, dass die Richtliniengruppe auf jede Komponente einzeln angewendet wird. Eine Richtliniengruppe für Shared QoS führt zur Durchsetzung der Obergrenze für den Gesamtdurchsatz aller Workloads.</p>	“
adaptiveQosPolicy	Adaptive QoS-Richtliniengruppe mit Zuordnung für erstellte Volumes Wählen Sie eine der qosPolicy oder adaptiveQosPolicy pro Storage Pool/Backend	“
snapshotReserve	Prozentsatz des für Snapshots reservierten Volumes	„0“ wenn snapshotPolicy Ist „keine“, andernfalls „
splitOnClone	Teilen Sie einen Klon bei der Erstellung von seinem übergeordneten Objekt auf	„Falsch“
encryption	<p>Aktivieren Sie NetApp Volume Encryption (NVE) auf dem neuen Volume, standardmäßig aktiviert <code>false</code>. NVE muss im Cluster lizenziert und aktiviert sein, damit diese Option verwendet werden kann.</p> <p>Wenn NAE auf dem Backend aktiviert ist, wird jedes im Astra Trident bereitgestellte Volume NAE aktiviert.</p> <p>Weitere Informationen finden Sie unter: "Astra Trident arbeitet mit NVE und NAE zusammen".</p>	„Falsch“
luksEncryption	<p>Aktivieren Sie die LUKS-Verschlüsselung. Siehe "Linux Unified Key Setup (LUKS) verwenden".</p> <p>LUKS-Verschlüsselung wird für NVMe/TCP nicht unterstützt.</p>	“
securityStyle	Sicherheitstyp für neue Volumes	unix
tieringPolicy	Tiering-Richtlinie, die zu „keinen“ verwendet wird	„Nur snapshot“ für eine SVM-DR-Konfiguration vor ONTAP 9.5

Beispiele für die Volume-Bereitstellung

Hier ein Beispiel mit definierten Standardwerten:

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: trident_svm
username: admin
password: <password>
labels:
  k8scluster: dev2
  backend: dev2-sanbackend
storagePrefix: alternate-trident
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: standard
  spaceAllocation: 'false'
  snapshotPolicy: default
  snapshotReserve: '10'
```



Für alle mit dem erstellten Volumes `ontap-san` Treiber: Astra Trident fügt der FlexVol zusätzliche Kapazität von 10 % hinzu, um die LUN-Metadaten zu bewältigen. Die LUN wird genau mit der Größe bereitgestellt, die der Benutzer in der PVC anfordert. Astra Trident fügt 10 Prozent zum FlexVol hinzu (wird in ONTAP als verfügbare Größe dargestellt). Benutzer erhalten jetzt die Menge an nutzbarer Kapazität, die sie angefordert haben. Diese Änderung verhindert auch, dass LUNs schreibgeschützt werden, sofern der verfügbare Speicherplatz nicht vollständig genutzt wird. Dies gilt nicht für die Wirtschaft von `ontap-san`.

Für Back-Ends, die definieren `snapshotReserve`, Astra Trident berechnet die Größe der Volumes wie folgt:

```
Total volume size = [(PVC requested size) / (1 - (snapshotReserve
percentage) / 100)] * 1.1
```

Das 1.1 ist der zusätzliche 10-Prozent-Astra Trident fügt dem FlexVol hinzu, um die LUN-Metadaten zu bewältigen. Für `snapshotReserve = 5 %`, und die PVC-Anforderung = 5 gib, die Gesamtgröße des Volumes beträgt 5,79 gib und die verfügbare Größe 5,5 gib. Der `volume show` Der Befehl sollte Ergebnisse anzeigen, die diesem Beispiel ähnlich sind:

Vserver	Volume	Aggregate	State	Type	Size	Available	Used%
		_pvc_89f1c156_3801_4de4_9f9d_034d54c395f4	online	RW	10GB	5.00GB	0%
		_pvc_e42ec6fe_3baa_4af6_996d_134adbbb8e6d	online	RW	5.79GB	5.50GB	0%
		_pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba	online	RW	1GB	511.8MB	0%

3 entries were displayed.

Die Größenanpassung ist derzeit die einzige Möglichkeit, die neue Berechnung für ein vorhandenes Volume zu verwenden.

Minimale Konfigurationsbeispiele

Die folgenden Beispiele zeigen grundlegende Konfigurationen, bei denen die meisten Parameter standardmäßig belassen werden. Dies ist der einfachste Weg, ein Backend zu definieren.



Wenn Sie Amazon FSX auf NetApp ONTAP mit Astra Trident verwenden, empfehlen wir, DNS-Namen für LIFs anstelle von IP-Adressen anzugeben.

Beispiel: ONTAP SAN

Dies ist eine grundlegende Konfiguration mit dem `ontap-san` Treiber.

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
labels:
  k8scluster: test-cluster-1
  backend: testcluster1-sanbackend
username: vsadmin
password: <password>
```

Beispiel für die SAN-Ökonomie von ONTAP

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
username: vsadmin
password: <password>
```

Beispiel: MetroCluster

Sie können das Backend so konfigurieren, dass die Backend-Definition nach Umschaltung und einem Wechsel während nicht manuell aktualisiert werden muss "[SVM-Replizierung und Recovery](#)".

Für nahtloses Switchover und Switchback geben Sie die SVM über an `managementLIF` Und lassen Sie die aus `dataLIF` Und `svm` Parameter. Beispiel:

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 192.168.1.66
username: vsadmin
password: password
```

Beispiel für die zertifikatbasierte Authentifizierung

In diesem Beispiel der Grundkonfiguration `clientCertificate`, `clientPrivateKey`, und `trustedCACertificate` (Optional, wenn Sie eine vertrauenswürdige CA verwenden) werden ausgefüllt `backend.json` Und nehmen Sie die base64-kodierten Werte des Clientzertifikats, des privaten Schlüssels und des vertrauenswürdigen CA-Zertifikats.

```
---
version: 1
storageDriverName: ontap-san
backendName: DefaultSANBackend
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSD6cNwxyz
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
```

Beispiele für bidirektionales CHAP

Diese Beispiele erstellen ein Backend mit `useCHAP` Auf einstellen `true`.

Beispiel für ONTAP-SAN-CHAP

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
labels:
  k8scluster: test-cluster-1
  backend: testcluster1-sanbackend
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSD6cNwxyz
username: vsadmin
password: <password>
```

Beispiel für ONTAP SAN Economy CHAP

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSD6cNwxyz
username: vsadmin
password: <password>
```


Beispiel für NVMe/TCP

Sie müssen eine SVM auf Ihrem ONTAP Back-End mit NVMe konfiguriert haben. Dies ist eine grundlegende Backend-Konfiguration für NVMe/TCP.

```
---
version: 1
backendName: NVMeBackend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_nvme
username: vsadmin
password: password
sanType: nvme
useREST: true
```

Beispiele für Back-Ends mit virtuellen Pools

In diesen Beispiel-Back-End-Definitionsdateien werden spezifische Standardwerte für alle Speicherpools festgelegt, z. B. `spaceReserve` Bei `keiner`, `spaceAllocation` Bei `false`, und `encryption` Bei `false`. Die virtuellen Pools werden im Abschnitt `Speicher` definiert.

Astra Trident bestimmt die Bereitstellungsetiketten im Feld „Kommentare“. Kommentare werden auf dem FlexVol gesetzt. Astra Trident kopiert alle Labels auf einem virtuellen Pool auf das Storage-Volume während der Bereitstellung. Storage-Administratoren können Labels je virtuellen Pool definieren und Volumes nach Label gruppieren.

In diesen Beispielen legen einige Speicherpools eigene fest `spaceReserve`, `spaceAllocation`, und `encryption` Werte und einige Pools überschreiben die Standardwerte.



```

---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSD6cNwxyz
username: vsadmin
password: <password>
defaults:
  spaceAllocation: 'false'
  encryption: 'false'
  qosPolicy: standard
labels:
  store: san_store
  kubernetes-cluster: prod-cluster-1
region: us_east_1
storage:
- labels:
  protection: gold
  creditpoints: '40000'
  zone: us_east_1a
  defaults:
    spaceAllocation: 'true'
    encryption: 'true'
    adaptiveQosPolicy: adaptive-extreme
- labels:
  protection: silver
  creditpoints: '20000'
  zone: us_east_1b
  defaults:
    spaceAllocation: 'false'
    encryption: 'true'
    qosPolicy: premium
- labels:
  protection: bronze
  creditpoints: '5000'
  zone: us_east_1c
  defaults:
    spaceAllocation: 'true'
    encryption: 'false'

```

Beispiel für die SAN-Ökonomie von ONTAP

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSD6cNwxyz
username: vsadmin
password: <password>
defaults:
  spaceAllocation: 'false'
  encryption: 'false'
labels:
  store: san_economy_store
region: us_east_1
storage:
- labels:
  app: oracledb
  cost: '30'
  zone: us_east_1a
  defaults:
    spaceAllocation: 'true'
    encryption: 'true'
- labels:
  app: postgresdb
  cost: '20'
  zone: us_east_1b
  defaults:
    spaceAllocation: 'false'
    encryption: 'true'
- labels:
  app: mysqldb
  cost: '10'
  zone: us_east_1c
  defaults:
    spaceAllocation: 'true'
    encryption: 'false'
- labels:
  department: legal
  creditpoints: '5000'
  zone: us_east_1c
```

```
defaults:
  spaceAllocation: 'true'
  encryption: 'false'
```

Beispiel für NVMe/TCP

```
---
version: 1
storageDriverName: ontap-san
sanType: nvme
managementLIF: 10.0.0.1
svm: nvme_svm
username: vsadmin
password: <password>
useREST: true
defaults:
  spaceAllocation: 'false'
  encryption: 'true'
storage:
- labels:
  app: testApp
  cost: '20'
  defaults:
    spaceAllocation: 'false'
    encryption: 'false'
```

Back-Ends StorageClasses zuordnen

Die folgenden StorageClass-Definitionen finden Sie im [Beispiele für Back-Ends mit virtuellen Pools](#).

Verwenden der `parameters.selector` Jede StorageClass ruft auf, welche virtuellen Pools zum Hosten eines Volumes verwendet werden können. Auf dem Volume werden die Aspekte im ausgewählten virtuellen Pool definiert.

- Der `protection-gold` StorageClass wird dem ersten virtuellen Pool in zugeordnet `ontap-san` Back-End: Dies ist der einzige Pool mit Gold-Level-Schutz.

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=gold"
  fsType: "ext4"

```

- Der **protection-not-gold StorageClass** wird dem zweiten und dritten virtuellen Pool in zugeordnet **ontap-san Back-End**: Dies sind die einzigen Pools, die ein anderes Schutzniveau als Gold bieten.

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection!=gold"
  fsType: "ext4"

```

- Der **app-mysqldb StorageClass** wird dem dritten virtuellen Pool in zugeordnet **ontap-san-economy Back-End**: Dies ist der einzige Pool, der Storage-Pool-Konfiguration für die **mysqldb-App** bietet.

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"

```

- Der **protection-silver-creditpoints-20k StorageClass** wird dem zweiten virtuellen Pool in zugeordnet **ontap-san Back-End**: Dies ist der einzige Pool mit Silber-Level-Schutz und 20000 Kreditpunkte.

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"

```

- Der `creditpoints-5k` StorageClass wird dem dritten virtuellen Pool in zugeordnet `ontap-san` Back-End und der vierte virtuelle Pool im `ontap-san-economy` Back-End: Dies sind die einzigen Poolangebote mit 5000 Kreditpunkten.

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: csi.trident.netapp.io
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"

```

- Der `my-test-app-sc` StorageClass wird dem zugeordnet `testAPP` Virtueller Pool im `ontap-san` Treiber mit `sanType: nvme`. Dies ist das einzige Poolangebot `testApp`.

```

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: my-test-app-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=testApp"
  fsType: "ext4"

```

Astra Trident entscheidet, welcher virtuelle Pool ausgewählt wird und stellt sicher, dass die Storage-Anforderungen erfüllt werden.

ONTAP-NAS-Treiber

Übersicht über den ONTAP NAS-Treiber

Erfahren Sie mehr über die Konfiguration eines ONTAP-Backend mit ONTAP- und Cloud Volumes ONTAP-NAS-Treibern.

Details zum ONTAP NAS-Treiber

Astra Trident bietet die folgenden NAS-Storage-Treiber für die Kommunikation mit dem ONTAP Cluster. Unterstützte Zugriffsmodi sind: *ReadWriteOnce* (RWO), *ReadOnly Many* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).



Wenn Sie Astra Control für Schutz, Recovery und Mobilität verwenden, lesen Sie bitte [Treiberkompatibilität bei Astra Control](#).

Treiber	Protokoll	VolumeModus	Unterstützte Zugriffsmodi	Unterstützte Filesysteme
ontap-nas	NFS SMB	Dateisystem	RWO, ROX, RWX, RWOP	„“, nfs, smb
ontap-nas-economy	NFS SMB	Dateisystem	RWO, ROX, RWX, RWOP	„“, nfs, smb
ontap-nas-flexgroup	NFS SMB	Dateisystem	RWO, ROX, RWX, RWOP	„“, nfs, smb

Treiberkompatibilität bei Astra Control

Astra Control bietet nahtlosen Schutz, Disaster Recovery und Mobilität (Verschieben von Volumes zwischen Kubernetes Clustern) für Volumes, die mit der erstellt wurden `ontap-nas`, `ontap-nas-flexgroup`, und `ontap-san` Treiber. Siehe "[Voraussetzungen für die Astra Control Replikation](#)" Entsprechende Details.



- Nutzung `ontap-san-economy` Nur wenn die Nutzungszahl für persistente Volumes voraussichtlich höher ist als "[Unterstützte ONTAP-Volume-Größen](#)".
- Nutzung `ontap-nas-economy` Nur wenn die Nutzungszahl für persistente Volumes voraussichtlich höher ist als "[Unterstützte ONTAP-Volume-Größen](#)" Und das `ontap-san-economy` Treiber kann nicht verwendet werden.
- Verwenden Sie ihn nicht `ontap-nas-economy` Wenn Sie die Notwendigkeit von Datensicherung, Disaster Recovery oder Mobilität erwarten.

Benutzerberechtigungen

Astra Trident erwartet, dass er entweder als ONTAP- oder SVM-Administrator ausgeführt wird, in der Regel mit dem `admin` Cluster-Benutzer oder ein `vsadmin` SVM-Benutzer oder ein Benutzer mit einem anderen Namen und derselben Rolle.

Astra Trident erwartet, dass bei Amazon FSX für Implementierungen von NetApp ONTAP, über das Cluster entweder als ONTAP- oder SVM-Administrator ausgeführt wird `fsxadmin` Benutzer oder A `vsadmin` SVM-Benutzer oder ein Benutzer mit einem anderen Namen und derselben Rolle. Der `fsxadmin` Der Benutzer ist ein eingeschränkter Ersatz für den Cluster-Admin-Benutzer.



Wenn Sie den verwenden `limitAggregateUsage` Parameter, Berechtigungen für Cluster-Admin sind erforderlich. Bei der Verwendung von Amazon FSX für NetApp ONTAP mit Astra Trident, das `limitAggregateUsage` Der Parameter funktioniert nicht mit dem `vsadmin` Und `fsxadmin` Benutzerkonten. Der Konfigurationsvorgang schlägt fehl, wenn Sie diesen Parameter angeben.

Es ist zwar möglich, eine restriktivere Rolle in ONTAP zu erstellen, die ein Trident-Treiber verwenden kann, wir empfehlen sie jedoch nicht. Bei den meisten neuen Versionen von Trident sind zusätzliche APIs erforderlich, die berücksichtigt werden müssten, was Upgrades schwierig und fehleranfällig macht.

Bereiten Sie sich auf die Konfiguration eines Backend mit ONTAP-NAS-Treibern vor

Verstehen Sie die Anforderungen, Authentifizierungsoptionen und Exportrichtlinien für die Konfiguration eines ONTAP-Backends mit ONTAP-NAS-Treibern.

Anforderungen

- Für alle ONTAP Back-Ends benötigt Astra Trident mindestens ein Aggregat, das der SVM zugewiesen ist.
- Sie können mehrere Treiber ausführen und Speicherklassen erstellen, die auf den einen oder den anderen zeigen. Beispielsweise könnten Sie eine Gold-Klasse konfigurieren, die den verwendet `ontap-nas` Fahrer und eine Bronze-Klasse, die den verwendet `ontap-nas-economy` Eins.
- Alle Kubernetes-Worker-Nodes müssen über die entsprechenden NFS-Tools verfügen. Siehe ["Hier"](#) Entnehmen.
- Astra Trident unterstützt SMB Volumes, die nur auf Windows Nodes laufenden Pods gemountet werden. Siehe [Vorbereitung zur Bereitstellung von SMB Volumes](#) Entsprechende Details.

Authentifizieren Sie das ONTAP-Backend

Astra Trident bietet zwei Arten der Authentifizierung eines ONTAP-Backend.

- Anmeldeinformationsbasiert: Dieser Modus erfordert ausreichende Berechtigungen für das ONTAP-Backend. Es wird empfohlen, ein Konto zu verwenden, das mit einer vordefinierten Sicherheits-Login-Rolle verknüpft ist, z. B. `admin` Oder `vsadmin` Für maximale Kompatibilität mit ONTAP Versionen.
- Zertifikatsbasiert: Für die Kommunikation mit einem ONTAP-Cluster ist in diesem Modus ein auf dem Backend installiertes Zertifikat erforderlich. Hier muss die Backend-Definition Base64-kodierte Werte des Client-Zertifikats, des Schlüssels und des vertrauenswürdigen CA-Zertifikats enthalten, sofern verwendet (empfohlen).

Sie können vorhandene Back-Ends aktualisieren, um zwischen auf Anmeldeinformationen basierenden und zertifikatbasierten Methoden zu verschieben. Es wird jedoch immer nur eine Authentifizierungsmethode unterstützt. Um zu einer anderen Authentifizierungsmethode zu wechseln, müssen Sie die vorhandene Methode von der Backend-Konfiguration entfernen.



Wenn Sie versuchen, **sowohl Anmeldeinformationen als auch Zertifikate** bereitzustellen, schlägt die Backend-Erstellung mit einem Fehler fehl, dass mehr als eine Authentifizierungsmethode in der Konfigurationsdatei angegeben wurde.

Aktivieren Sie die Anmeldeinformationsbasierte Authentifizierung

Astra Trident erfordert die Zugangsdaten für einen Administrator mit SVM-Umfang/Cluster-Umfang, um mit dem Backend von ONTAP zu kommunizieren. Es wird empfohlen, die Standard-vordefinierten Rollen wie zu

verwenden `admin` Oder `vsadmin`. So ist gewährleistet, dass die Kompatibilität mit künftigen ONTAP Versionen gewährleistet ist, die FunktionsAPIs der künftigen Astra Trident Versionen bereitstellen können. Eine benutzerdefinierte Sicherheits-Login-Rolle kann mit Astra Trident erstellt und verwendet werden, wird aber nicht empfohlen.

Eine Beispiel-Back-End-Definition sieht folgendermaßen aus:

YAML

```
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

JSON

```
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password"
}
```

Beachten Sie, dass die Backend-Definition der einzige Ort ist, an dem die Anmeldeinformationen im reinen Text gespeichert werden. Nach der Erstellung des Backend werden Benutzernamen/Passwörter mit Base64 codiert und als Kubernetes Secrets gespeichert. Die Erstellung/Aktualisierung eines Backend ist der einzige Schritt, der Kenntnisse der Anmeldeinformationen erfordert. Daher ist dieser Vorgang nur für Administratoren und wird vom Kubernetes-/Storage-Administrator ausgeführt.

Aktivieren Sie die zertifikatbasierte Authentifizierung

Neue und vorhandene Back-Ends können ein Zertifikat verwenden und mit dem ONTAP-Back-End kommunizieren. In der Backend-Definition sind drei Parameter erforderlich.

- `ClientCertificate`: Base64-codierter Wert des Clientzertifikats.
- `ClientPrivateKey`: Base64-kodierte Wert des zugeordneten privaten Schlüssels.
- `Trusted CACertificate`: Base64-codierter Wert des vertrauenswürdigen CA-Zertifikats. Bei Verwendung einer vertrauenswürdigen CA muss dieser Parameter angegeben werden. Dies kann ignoriert werden,

wenn keine vertrauenswürdige CA verwendet wird.

Ein typischer Workflow umfasst die folgenden Schritte.

Schritte

1. Erzeugen eines Clientzertifikats und eines Schlüssels. Legen Sie beim Generieren den allgemeinen Namen (CN) für den ONTAP-Benutzer fest, der sich authentifizieren soll als.

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key  
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=vsadmin"
```

2. Fügen Sie dem ONTAP-Cluster ein vertrauenswürdiges CA-Zertifikat hinzu. Dies kann möglicherweise bereits vom Storage-Administrator übernommen werden. Ignorieren, wenn keine vertrauenswürdige CA verwendet wird.

```
security certificate install -type server -cert-name <trusted-ca-cert-name> -vserver <vserver-name>  
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled  
true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca  
<cert-authority>
```

3. Installieren Sie das Client-Zertifikat und den Schlüssel (von Schritt 1) auf dem ONTAP-Cluster.

```
security certificate install -type client-ca -cert-name <certificate-name> -vserver <vserver-name>  
security ssl modify -vserver <vserver-name> -client-enabled true
```

4. Bestätigen Sie, dass die ONTAP-Sicherheitsanmeldungsrolle unterstützt wird `cert` Authentifizierungsmethode.

```
security login create -user-or-group-name vsadmin -application ontapi  
-authentication-method cert -vserver <vserver-name>  
security login create -user-or-group-name vsadmin -application http  
-authentication-method cert -vserver <vserver-name>
```

5. Testen Sie die Authentifizierung mithilfe des generierten Zertifikats. <ONTAP Management LIF> und <vServer Name> durch Management-LIF-IP und SVM-Namen ersetzen. Sie müssen sicherstellen, dass die Service-Richtlinie für das LIF auf festgelegt ist `default-data-management`.

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns="http://www.netapp.com/filer/admin" version="1.21"
vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. Encodieren von Zertifikat, Schlüssel und vertrauenswürdigen CA-Zertifikat mit Base64.

```
base64 -w 0 k8senv.pem >> cert_base64
base64 -w 0 k8senv.key >> key_base64
base64 -w 0 trustedca.pem >> trustedca_base64
```

7. Erstellen Sie das Backend mit den Werten, die aus dem vorherigen Schritt ermittelt wurden.

```
cat cert-backend-updated.json
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "NasBackend",
  "managementLIF": "1.2.3.4",
  "dataLIF": "1.2.3.8",
  "svm": "vserver_test",
  "clientCertificate": "Faaaakkkkeeee...Vaaalllluuueeeee",
  "clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
  "storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident

+-----+-----+-----+-----+
+-----+-----+
|      NAME      | STORAGE DRIVER |                      UUID                      |
STATE | VOLUMES |
+-----+-----+-----+-----+
+-----+-----+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |          9 |
+-----+-----+-----+-----+
+-----+-----+

```

Aktualisieren Sie Authentifizierungsmethoden, oder drehen Sie die Anmeldedaten

Sie können ein vorhandenes Backend aktualisieren, um eine andere Authentifizierungsmethode zu verwenden oder ihre Anmeldedaten zu drehen. Das funktioniert auf beide Arten: Back-Ends, die einen Benutzernamen/ein Passwort verwenden, können aktualisiert werden, um Zertifikate zu verwenden; Back-Ends, die Zertifikate verwenden, können auf Benutzername/Passwort-basiert aktualisiert werden. Dazu müssen Sie die vorhandene Authentifizierungsmethode entfernen und die neue Authentifizierungsmethode hinzufügen. Verwenden Sie dann die aktualisierte Backend.json-Datei, die die erforderlichen Parameter enthält `tridentctl update backend`.

```
cat cert-backend-updated.json
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "NasBackend",
  "managementLIF": "1.2.3.4",
  "dataLIF": "1.2.3.8",
  "svm": "vserver_test",
  "username": "vsadmin",
  "password": "password",
  "storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident

+-----+-----+-----+-----+
+-----+-----+
|  NAME      | STORAGE DRIVER |                               UUID                               |
STATE | VOLUMES |
+-----+-----+-----+-----+
+-----+-----+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |         9 |
+-----+-----+-----+-----+
+-----+-----+
```



Bei der Änderung von Passwörtern muss der Speicheradministrator das Kennwort für den Benutzer auf ONTAP aktualisieren. Auf diese Weise folgt ein Backend-Update. Beim Drehen von Zertifikaten können dem Benutzer mehrere Zertifikate hinzugefügt werden. Das Backend wird dann aktualisiert und verwendet das neue Zertifikat. Danach kann das alte Zertifikat aus dem ONTAP Cluster gelöscht werden.

Durch die Aktualisierung eines Backend wird der Zugriff auf Volumes, die bereits erstellt wurden, nicht unterbrochen, und auch die danach erstellten Volume-Verbindungen werden beeinträchtigt. Ein erfolgreiches Backend-Update zeigt, dass Astra Trident mit dem ONTAP-Backend kommunizieren und zukünftige Volume-Operationen verarbeiten kann.

Management der NFS-Exportrichtlinien

Astra Trident verwendet NFS-Exportrichtlinien, um den Zugriff auf die Volumes zu kontrollieren, die er bereitstellt.

Astra Trident bietet zwei Optionen für die Arbeit mit Exportrichtlinien:

- Astra Trident kann die Exportrichtlinie selbst dynamisch managen. In diesem Betriebsmodus spezifiziert der Storage-Administrator eine Liste mit CIDR-Blöcken, die zulässige IP-Adressen darstellen. Astra Trident fügt automatisch Node-IPs hinzu, die in diese Bereiche fallen, zur Exportrichtlinie hinzu. Wenn keine CIDRs angegeben werden, wird alternativ jede auf den Knoten gefundene globale Unicast-IP mit globalem Umfang zur Exportrichtlinie hinzugefügt.
- Storage-Administratoren können eine Exportrichtlinie erstellen und Regeln manuell hinzufügen. Astra Trident verwendet die Standard-Exportrichtlinie, es sei denn, in der Konfiguration ist ein anderer Name der Exportrichtlinie angegeben.

Dynamisches Managen von Exportrichtlinien

Astra Trident bietet die Möglichkeit, Richtlinien für den Export von ONTAP Back-Ends dynamisch zu managen. So kann der Storage-Administrator einen zulässigen Adressraum für Worker-Node-IPs festlegen, anstatt explizite Regeln manuell zu definieren. Dies vereinfacht das Management von Exportrichtlinien erheblich. Änderungen der Exportrichtlinie erfordern keine manuellen Eingriffe des Storage-Clusters mehr. Darüber hinaus hilft dies, den Zugriff auf den Storage-Cluster nur auf Worker-Nodes mit IPs im angegebenen Bereich zu beschränken, was ein fein abgestimmtes und automatisiertes Management unterstützt.



Verwenden Sie keine Network Address Translation (NAT), wenn Sie dynamische Exportrichtlinien verwenden. Bei NAT erkennt der Speicher-Controller die Frontend-NAT-Adresse und nicht die tatsächliche IP-Host-Adresse, so dass der Zugriff verweigert wird, wenn in den Exportregeln keine Übereinstimmung gefunden wird.

Beispiel

Es müssen zwei Konfigurationsoptionen verwendet werden. Hier ist eine Beispiel-Backend-Definition:

```
---
version: 1
storageDriverName: ontap-nas
backendName: ontap_nas_auto_export
managementLIF: 192.168.0.135
svm: svm1
username: vsadmin
password: password
autoExportCIDRs:
- 192.168.0.0/24
autoExportPolicy: true
```



Wenn Sie diese Funktion verwenden, müssen Sie sicherstellen, dass für die Root-Verbindung in Ihrer SVM eine zuvor erstellte Exportrichtlinie mit einer Exportregel vorhanden ist, die den CIDR-Block des Nodes zulässt (z. B. die standardmäßige Exportrichtlinie). Folgen Sie stets den von NetApp empfohlenen Best Practices, um eine SVM für Astra Trident zu zuweisen.

Hier ist eine Erklärung, wie diese Funktion funktioniert, anhand des obigen Beispiels:

- `autoExportPolicy` Ist auf festgelegt `true`. Dies zeigt an, dass Astra Trident eine Exportrichtlinie für den erstellen wird `svm1` SVM und das Hinzufügen und Löschen von Regeln mit behandeln `autoExportCIDRs` Adressblöcke. Beispiel: Ein Backend mit UUID `403b5326-8482-40db-96d0-d83fb3f4daec` und `autoExportPolicy` Auf einstellen `true` Erstellt eine Exportrichtlinie mit dem Namen `trident-403b5326-8482-40db-96d0-d83fb3f4daec` Auf der SVM.
- `autoExportCIDRs` Enthält eine Liste von Adressblöcken. Dieses Feld ist optional und standardmäßig `[„0.0.0.0/0“, „:/0“]`. Falls nicht definiert, fügt Astra Trident alle Unicast-Adressen mit globellem Umfang hinzu, die auf den Worker-Nodes gefunden wurden.

In diesem Beispiel ist der `192.168.0.0/24` Adressbereich wird bereitgestellt. Das zeigt an, dass die Kubernetes-Node-IPs, die in diesen Adressbereich fallen, der vom Astra Trident erstellten Exportrichtlinie hinzugefügt werden. Wenn Astra Trident einen Knoten registriert, auf dem er ausgeführt wird, ruft er die IP-Adressen des Knotens ab und überprüft sie auf die in angegebenen Adressblöcke `autoExportCIDRs`. Nach dem Filtern der IPs erstellt Astra Trident Regeln für die Exportrichtlinie für die erkannte Client-IPs. Dabei gilt für jeden Node eine Regel, die er identifiziert.

Sie können aktualisieren `autoExportPolicy` Und `autoExportCIDRs` Für Back-Ends, nachdem Sie sie erstellt haben. Sie können neue CIDRs für ein Backend anhängen, das automatisch verwaltet wird oder vorhandene CIDRs löschen. Beim Löschen von CIDRs Vorsicht walten lassen, um sicherzustellen, dass vorhandene Verbindungen nicht unterbrochen werden. Sie können auch wählen, zu deaktivieren `autoExportPolicy` Für ein Backend und kehren Sie zu einer manuell erstellten Exportrichtlinie zurück. Dazu muss die Einstellung festgelegt werden `exportPolicy` Parameter in Ihrer Backend-Konfiguration.

Nachdem Astra Trident ein Backend erstellt oder aktualisiert hat, können Sie das Backend mit überprüfen `tridentctl` Oder das entsprechende `tridentbackend` CRD:

```
./tridentctl get backends ontap_nas_auto_export -n trident -o yaml
items:
- backendUUID: 403b5326-8482-40db-96d0-d83fb3f4daec
  config:
    aggregate: ""
    autoExportCIDRs:
    - 192.168.0.0/24
    autoExportPolicy: true
    backendName: ontap_nas_auto_export
    chapInitiatorSecret: ""
    chapTargetInitiatorSecret: ""
    chapTargetUsername: ""
    chapUsername: ""
    dataLIF: 192.168.0.135
    debug: false
    debugTraceFlags: null
    defaults:
      encryption: "false"
      exportPolicy: <automatic>
      fileType: ext4
```

Wenn Nodes zu einem Kubernetes-Cluster hinzugefügt und beim Astra Trident Controller registriert werden, werden die Exportrichtlinien vorhandener Back-Ends aktualisiert (vorausgesetzt, sie sind in den in angegebenen Adressbereich enthalten `autoExportCIDRs` Für das Backend).

Wenn ein Node entfernt wird, überprüft Astra Trident alle Back-Ends, die online sind, um die Zugriffsregel für den Node zu entfernen. Indem Astra Trident diese Node-IP aus den Exportrichtlinien für gemanagte Back-Ends entfernt, verhindert er abnormale Mounts, sofern diese IP nicht von einem neuen Node im Cluster verwendet wird.

Aktualisieren Sie bei zuvor vorhandenen Back-Ends das Backend mit `tridentctl update backend`. Stellt sicher, dass Astra Trident die Exportrichtlinien automatisch verwaltet. Dadurch wird eine neue Exportrichtlinie erstellt, die nach der UUID des Backends benannt ist und Volumes, die auf dem Backend vorhanden sind, verwenden die neu erstellte Exportrichtlinie, wenn sie wieder gemountet werden.



Wenn Sie ein Backend mit automatisch gemanagten Exportrichtlinien löschen, wird die dynamisch erstellte Exportrichtlinie gelöscht. Wenn das Backend neu erstellt wird, wird es als neues Backend behandelt und erzeugt eine neue Exportrichtlinie.

Wenn die IP-Adresse eines aktiven Node aktualisiert wird, müssen Sie den Astra Trident Pod auf dem Node neu starten. Astra Trident aktualisiert dann die Exportrichtlinie für Back-Ends, die es verwaltet, um diese IP-Änderung zu berücksichtigen.

Vorbereitung zur Bereitstellung von SMB Volumes

Mit ein wenig Vorbereitung können Sie SMB Volumes mit bereitstellen `ontap-nas` Treiber.



Zur Erstellung eines müssen Sie auf der SVM sowohl NFS- als auch SMB/CIFS-Protokolle konfigurieren `ontap-nas-economy` SMB Volume für ONTAP vor Ort: Ist eines dieser Protokolle nicht konfiguriert, schlägt die Erstellung von SMB Volumes fehl.

Bevor Sie beginnen

Bevor Sie SMB-Volumes bereitstellen können, müssen Sie über Folgendes verfügen:

- Kubernetes-Cluster mit einem Linux-Controller-Knoten und mindestens einem Windows-Worker-Node, auf dem Windows Server 2019 ausgeführt wird. Astra Trident unterstützt SMB Volumes, die nur auf Windows Nodes laufenden Pods gemountet werden.
- Mindestens ein Astra Trident-Geheimnis, der Ihre Active Directory-Anmeldedaten enthält. Um Geheimnis zu erzeugen `smbcreds`:

```
kubectl create secret generic smbcreds --from-literal username=user  
--from-literal password='password'
```

- Ein CSI-Proxy, der als Windows-Dienst konfiguriert ist. Zum Konfigurieren von A `csi-proxy` Weitere Informationen finden Sie unter "[GitHub: CSI-Proxy](#)" Oder "[GitHub: CSI Proxy für Windows](#)" Für Kubernetes-Knoten, die auf Windows ausgeführt werden.

Schritte

1. Bei On-Premises-ONTAP können Sie optional eine SMB-Freigabe erstellen oder Astra Trident eine für Sie erstellen.



SMB-Freigaben sind für Amazon FSX for ONTAP erforderlich.

Sie können SMB-Admin-Freigaben auf zwei Arten erstellen: Mit ["Microsoft Management Console"](#) Snap-in für freigegebene Ordner oder mit der ONTAP-CLI. So erstellen Sie SMB-Freigaben mithilfe der ONTAP-CLI:

- a. Erstellen Sie bei Bedarf die Verzeichnispfadstruktur für die Freigabe.

Der `vserver cifs share create` Der Befehl überprüft während der Freigabenerstellung den in der Option `-path` angegebenen Pfad. Wenn der angegebene Pfad nicht vorhanden ist, schlägt der Befehl fehl.

- b. Erstellen einer mit der angegebenen SVM verknüpften SMB-Freigabe:

```
vserver cifs share create -vserver vserver_name -share-name
share_name -path path [-share-properties share_properties,...]
[other_attributes] [-comment text]
```

- c. Vergewissern Sie sich, dass die Freigabe erstellt wurde:

```
vserver cifs share show -share-name share_name
```



Siehe ["Erstellen Sie eine SMB-Freigabe"](#) Vollständige Informationen.

2. Beim Erstellen des Backend müssen Sie Folgendes konfigurieren, um SMB-Volumes festzulegen. Alle FSX-Konfigurationsoptionen für ONTAP-Backend finden Sie unter ["FSX für ONTAP Konfigurationsoptionen und Beispiele"](#).

Parameter	Beschreibung	Beispiel
smbShare	<p>Sie können eine der folgenden Optionen angeben: Den Namen einer SMB-Freigabe, die mit der Microsoft Management Console oder der ONTAP-CLI erstellt wurde, einen Namen, über den Astra Trident die SMB-Freigabe erstellen kann, oder Sie können den Parameter leer lassen, um den Zugriff auf gemeinsame Freigaben auf Volumes zu verhindern.</p> <p>Dieser Parameter ist für On-Premises-ONTAP optional.</p> <p>Dieser Parameter ist für Amazon FSX for ONTAP-Back-Ends erforderlich und darf nicht leer sein.</p>	smb-share
nasType	Muss auf eingestellt sein smb. Wenn Null, wird standardmäßig auf gesetzt <code>nfs</code> .	smb

Parameter	Beschreibung	Beispiel
securityStyle	Sicherheitstyp für neue Volumes. Muss auf eingestellt sein ntfs Oder mixed Für SMB Volumes.	ntfs Oder mixed Für SMB Volumes
unixPermissions	Modus für neue Volumes. Muss für SMB Volumes leer gelassen werden.	“

ONTAP-NAS-Konfigurationsoptionen und Beispiele

Lernen Sie, wie Sie ONTAP NAS-Treiber mit Ihrer Astra Trident Installation erstellen und verwenden. Dieser Abschnitt enthält Beispiele und Details zur Back-End-Konfiguration für die Zuordnung von Back-Ends zu StorageClasses.

Back-End-Konfigurationsoptionen

Die Back-End-Konfigurationsoptionen finden Sie in der folgenden Tabelle:

Parameter	Beschreibung	Standard
version		Immer 1
storageDriverName	Name des Speichertreibers	„ontap-nas“, „ontap-nas-Economy“, „ontap-nas-flexgroup“, „ontap-san“, „ontap-san-Economy“
backendName	Benutzerdefinierter Name oder das Storage-Backend	Treibername + „_“ + DatenLIF
managementLIF	<p>IP-Adresse eines Clusters oder einer SVM-Management-LIF</p> <p>Es kann ein vollständig qualifizierter Domänenname (FQDN) angegeben werden.</p> <p>Kann so eingestellt werden, dass IPv6-Adressen verwendet werden, wenn Astra Trident mit dem IPv6-Flag installiert wurde. IPv6-Adressen müssen in eckigen Klammern definiert werden, z. B. [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].</p> <p>Informationen zur nahtlosen MetroCluster-Umschaltung finden Sie im Beispiel: MetroCluster.</p>	„10.0.0.1“, „[2001:1234:abcd::fefe]“

Parameter	Beschreibung	Standard
dataLIF	<p>IP-Adresse des LIF-Protokolls.</p> <p>Wir empfehlen Ihnen, anzugeben <code>dataLIF</code>. Falls nicht vorgesehen, ruft Astra Trident Daten-LIFs von der SVM ab. Sie können einen vollständig qualifizierten Domännennamen (FQDN) angeben, der für die NFS-Mount-Vorgänge verwendet werden soll. Damit können Sie ein Round-Robin-DNS zum Load-Balancing über mehrere Daten-LIFs erstellen.</p> <p>Kann nach der Anfangseinstellung geändert werden. Siehe .</p> <p>Kann so eingestellt werden, dass IPv6-Adressen verwendet werden, wenn Astra Trident mit dem IPv6-Flag installiert wurde. IPv6-Adressen müssen in eckigen Klammern definiert werden, z. B. <code>[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]</code>.</p> <p>Für MetroCluster weglassen. Siehe Beispiel: MetroCluster.</p>	Angegebene Adresse oder abgeleitet von SVM, falls nicht angegeben (nicht empfohlen)
svm	<p>Zu verwendende Storage Virtual Machine</p> <p>Für MetroCluster weglassen. Siehe Beispiel: MetroCluster.</p>	Abgeleitet wenn eine SVM <code>managementLIF</code> Angegeben ist
autoExportPolicy	<p>Aktivieren Sie die automatische Erstellung von Exportrichtlinien und aktualisieren Sie [Boolean].</p> <p>Verwenden der <code>autoExportPolicy</code> Und <code>autoExportCIDRs</code> Optionen: Astra Trident kann Exportrichtlinien automatisch verwalten.</p>	Falsch
autoExportCIDRs	<p>Liste der CIDRs, nach denen die Node-IPs von Kubernetes gefiltert werden sollen <code>autoExportPolicy</code> Ist aktiviert.</p> <p>Verwenden der <code>autoExportPolicy</code> Und <code>autoExportCIDRs</code> Optionen: Astra Trident kann Exportrichtlinien automatisch verwalten.</p>	[„0.0.0.0/0“, „:/0“]
labels	Satz willkürlicher JSON-formatierter Etiketten für Volumes	“ ”
clientCertificate	Base64-codierter Wert des Clientzertifikats. Wird für zertifikatbasierte Authentifizierung verwendet	“ ”
clientPrivateKey	Base64-kodierte Wert des privaten Client-Schlüssels. Wird für zertifikatbasierte Authentifizierung verwendet	“ ”

Parameter	Beschreibung	Standard
trustedCACertificate	Base64-kodierte Wert des vertrauenswürdigen CA-Zertifikats. Optional Wird für zertifikatbasierte Authentifizierung verwendet	„“
username	Benutzername für die Verbindung mit dem Cluster/SVM. Wird für Anmeldeinformationsbasierte verwendet	
password	Passwort für die Verbindung mit dem Cluster/SVM Wird für Anmeldeinformationsbasierte verwendet	
storagePrefix	Das Präfix wird beim Bereitstellen neuer Volumes in der SVM verwendet. Kann nicht aktualisiert werden, nachdem Sie sie festgelegt haben	trident
limitAggregateUsage	Bereitstellung fehlgeschlagen, wenn die Nutzung über diesem Prozentsatz liegt. Gilt nicht für Amazon FSX für ONTAP	„“ (nicht standardmäßig durchgesetzt)
limitVolumeSize	Bereitstellung fehlgeschlagen, wenn die angeforderte Volume-Größe über diesem Wert liegt. Schränkt auch die maximale Größe der Volumes ein, die es für qtrees und LUNs verwaltet, und auf ein qtreesPerFlexvol Mit Option kann die maximale Anzahl von qtrees pro FlexVol angepasst werden.	„“ (standardmäßig nicht erzwungen)
lunsPerFlexvol	Die maximale Anzahl an LUNs pro FlexVol muss im Bereich [50, 200] liegen.	„100“
debugTraceFlags	Fehler-Flags bei der Fehlerbehebung beheben. Beispiel, {„API“:false, „method“:true} Verwenden Sie es nicht debugTraceFlags Es sei denn, Sie beheben Fehler und benötigen einen detaillierten Log Dump.	Null
nasType	Konfiguration der Erstellung von NFS- oder SMB-Volumes Die Optionen lauten nfs, smb Oder null. Einstellung auf null setzt standardmäßig auf NFS-Volumes.	nfs

Parameter	Beschreibung	Standard
nfsMountOptions	<p>Kommagetrennte Liste von NFS-Mount-Optionen.</p> <p>Die Mount-Optionen für Kubernetes-persistente Volumes werden normalerweise in Storage-Klassen angegeben. Wenn jedoch keine Mount-Optionen in einer Storage-Klasse angegeben sind, stellt Astra Trident die Mount-Optionen bereit, die in der Konfigurationsdatei des Storage-Back-End angegeben sind.</p> <p>Wenn in der Storage-Klasse oder der Konfigurationsdatei keine Mount-Optionen angegeben sind, stellt Astra Trident keine Mount-Optionen für ein damit verbundener persistentes Volume fest.</p>	“ ”
qtreesPerFlexvol	Maximale Ques pro FlexVol, muss im Bereich [50, 300] liegen	„200“
smbShare	<p>Sie können eine der folgenden Optionen angeben: Den Namen einer SMB-Freigabe, die mit der Microsoft Management Console oder der ONTAP-CLI erstellt wurde, einen Namen, über den Astra Trident die SMB-Freigabe erstellen kann, oder Sie können den Parameter leer lassen, um den Zugriff auf gemeinsame Freigaben auf Volumes zu verhindern.</p> <p>Dieser Parameter ist für On-Premises-ONTAP optional.</p> <p>Dieser Parameter ist für Amazon FSX for ONTAP-Back-Ends erforderlich und darf nicht leer sein.</p>	smb-share
useREST	<p>Boolescher Parameter zur Verwendung von ONTAP REST-APIs. Technische Vorschau</p> <p>useREST Wird als Tech-Vorschau bereitgestellt, das für Testumgebungen und nicht für Produktions-Workloads empfohlen wird. Wenn eingestellt auf true, Astra Trident wird ONTAP REST APIs zur Kommunikation mit dem Backend verwenden. Diese Funktion erfordert ONTAP 9.11.1 und höher. Darüber hinaus muss die verwendete ONTAP-Login-Rolle Zugriff auf den haben <code>ontap</code> Applikation. Dies wird durch die vordefinierte zufrieden <code>vsadmin</code> Und <code>cluster-admin</code> Rollen:</p> <p>useREST Wird mit MetroCluster nicht unterstützt.</p>	Falsch

Back-End-Konfigurationsoptionen für die Bereitstellung von Volumes

Sie können die Standardbereitstellung mit diesen Optionen im `steuern defaults` Abschnitt der Konfiguration. Ein Beispiel finden Sie unten in den Konfigurationsbeispielen.

Parameter	Beschreibung	Standard
spaceAllocation	Speicherplatzzuweisung für LUNs	„Wahr“
spaceReserve	Modus für Speicherplatzreservierung; „none“ (Thin) oder „Volume“ (Thick)	„Keine“
snapshotPolicy	Die Snapshot-Richtlinie zu verwenden	„Keine“
qosPolicy	QoS-Richtliniengruppe zur Zuweisung für erstellte Volumes Wählen Sie eine der qosPolicy oder adaptiveQosPolicy pro Storage Pool/Backend	“
adaptiveQosPolicy	Adaptive QoS-Richtliniengruppe mit Zuordnung für erstellte Volumes Wählen Sie eine der qosPolicy oder adaptiveQosPolicy pro Storage Pool/Backend. Nicht unterstützt durch ontap-nas-Ökonomie	“
snapshotReserve	Prozentsatz des für Snapshots reservierten Volumes	„0“ wenn snapshotPolicy Ist „keine“, andernfalls „“
splitOnClone	Teilen Sie einen Klon bei der Erstellung von seinem übergeordneten Objekt auf	„Falsch“
encryption	Aktivieren Sie NetApp Volume Encryption (NVE) auf dem neuen Volume, standardmäßig aktiviert false. NVE muss im Cluster lizenziert und aktiviert sein, damit diese Option verwendet werden kann. Wenn NAE auf dem Backend aktiviert ist, wird jedes im Astra Trident bereitgestellte Volume NAE aktiviert. Weitere Informationen finden Sie unter: " Astra Trident arbeitet mit NVE und NAE zusammen ".	„Falsch“
tieringPolicy	Tiering-Richtlinie, die zu „keinen“ verwendet wird	„Nur snapshot“ für eine SVM-DR-Konfiguration vor ONTAP 9.5
unixPermissions	Modus für neue Volumes	„777“ für NFS Volumes; leer (nicht zutreffend) für SMB Volumes
snapshotDir	Steuert den Zugriff auf das .snapshot Verzeichnis	„Falsch“
exportPolicy	Zu verwendende Exportrichtlinie	„Standard“
securityStyle	Sicherheitstyp für neue Volumes. NFS unterstützt mixed Und unix Sicherheitsstile. SMB-Support mixed Und ntfs Sicherheitsstile.	NFS-Standard ist unix. Der SMB-Standardwert ist ntfs.



Die Verwendung von QoS Policy Groups mit Astra Trident erfordert ONTAP 9.8 oder höher. Es wird empfohlen, eine nicht gemeinsam genutzte QoS-Richtliniengruppe zu verwenden und sicherzustellen, dass die Richtliniengruppe auf jede Komponente einzeln angewendet wird. Eine Richtliniengruppe für Shared QoS führt zur Durchsetzung der Obergrenze für den Gesamtdurchsatz aller Workloads.

Beispiele für die Volume-Bereitstellung

Hier ein Beispiel mit definierten Standardwerten:

```
---
version: 1
storageDriverName: ontap-nas
backendName: customBackendName
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
labels:
  k8scluster: dev1
  backend: dev1-nasbackend
svm: trident_svm
username: cluster-admin
password: <password>
limitAggregateUsage: 80%
limitVolumeSize: 50Gi
nfsMountOptions: nfsvers=4
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: premium
  exportPolicy: myk8scluster
  snapshotPolicy: default
  snapshotReserve: '10'
```

Für `ontap-nas` Und `ontap-nas-flexgroups` Astra Trident verwendet jetzt eine neue Berechnung, um sicherzustellen, dass die FlexVol korrekt mit dem Prozentwert der Snapshot Reserve und PVC dimensioniert ist. Wenn der Benutzer eine PVC anfordert, erstellt Astra Trident unter Verwendung der neuen Berechnung die ursprüngliche FlexVol mit mehr Speicherplatz. Diese Berechnung stellt sicher, dass der Benutzer den beschreibbaren Speicherplatz erhält, für den er in der PVC benötigt wird, und nicht weniger Speicherplatz als der angeforderte. Vor Version 2.07, wenn der Benutzer eine PVC anfordert (z. B. 5 gib), bei der SnapshotReserve auf 50 Prozent, erhalten sie nur 2,5 gib schreibbaren Speicherplatz. Der Grund dafür ist, dass der Benutzer das gesamte Volume und angefordert hat `snapshotReserve` ist ein Prozentsatz davon. Mit Trident 21.07 sind die Benutzeranforderungen der beschreibbare Speicherplatz, und Astra Trident definiert den `snapshotReserve` Zahl als Prozentsatz des gesamten Volumens. Dies gilt

nicht für `ontap-nas-economy`. Im folgenden Beispiel sehen Sie, wie das funktioniert:

Die Berechnung ist wie folgt:

```
Total volume size = (PVC requested size) / (1 - (snapshotReserve
percentage) / 100)
```

Für die `snapshotReserve` = 50 %, und die PVC-Anfrage = 5 gib, beträgt die Gesamtgröße des Volumes $2/5 = 10$ gib, und die verfügbare Größe beträgt 5 gib. Dies entspricht dem, was der Benutzer in der PVC-Anfrage angefordert hat. Der `volume show` Der Befehl sollte Ergebnisse anzeigen, die diesem Beispiel ähnlich sind:

Vserver	Volume	Aggregate	State	Type	Size	Available	Used%
		_pvc_89f1c156_3801_4de4_9f9d_034d54c395f4	online	RW	10GB	5.00GB	0%
		_pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba	online	RW	1GB	511.8MB	0%

2 entries were displayed.

Vorhandene Back-Ends aus vorherigen Installationen stellen Volumes wie oben beschrieben beim Upgrade von Astra Trident bereit. Bei Volumes, die Sie vor dem Upgrade erstellt haben, sollten Sie die Größe ihrer Volumes entsprechend der zu beobachtenden Änderung anpassen. Beispiel: Ein 2 gib PVC mit `snapshotReserve=50` Früher hat ein Volume ergeben, das 1 gib beschreibbaren Speicherplatz bereitstellt. Wenn Sie die Größe des Volumes auf 3 gib ändern, z. B. stellt die Applikation auf einem 6 gib an beschreibbarem Speicherplatz bereit.

Minimale Konfigurationsbeispiele

Die folgenden Beispiele zeigen grundlegende Konfigurationen, bei denen die meisten Parameter standardmäßig belassen werden. Dies ist der einfachste Weg, ein Backend zu definieren.



Wenn Sie Amazon FSX auf NetApp ONTAP mit Trident verwenden, empfiehlt es sich, DNS-Namen für LIFs anstelle von IP-Adressen anzugeben.

Beispiel für die NAS-Ökonomie von ONTAP

```
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```


Beispiel für ONTAP NAS FlexGroup

```
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

Beispiel: MetroCluster

Sie können das Backend so konfigurieren, dass die Backend-Definition nach Umschaltung und einem Wechsel während nicht manuell aktualisiert werden muss ["SVM-Replizierung und Recovery"](#).

Für nahtloses Switchover und Switchback geben Sie die SVM über an `managementLIF` Und lassen Sie die aus `dataLIF` Und `svm` Parameter. Beispiel:

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 192.168.1.66
username: vsadmin
password: password
```

Beispiel: SMB Volumes

```
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
nasType: smb
securityStyle: ntfs
unixPermissions: ""
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

Beispiel für die zertifikatbasierte Authentifizierung

Dies ist ein minimales Beispiel für die Back-End-Konfiguration. `clientCertificate`, `clientPrivateKey`, und `trustedCACertificate` (Optional, wenn Sie eine vertrauenswürdige CA verwenden) werden ausgefüllt `backend.json` Und nehmen Sie die base64-kodierten Werte des Clientzertifikats, des privaten Schlüssels und des vertrauenswürdigen CA-Zertifikats.

```
---
version: 1
backendName: DefaultNASBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.15
svm: nfs_svm
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

Beispiel für eine Richtlinie für den automatischen Export

In diesem Beispiel erfahren Sie, wie Sie Astra Trident anweisen können, dynamische Exportrichtlinien zu verwenden, um die Exportrichtlinie automatisch zu erstellen und zu verwalten. Das funktioniert auch für das `ontap-nas-economy` Und `ontap-nas-flexgroup` Treiber.

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
labels:
  k8scluster: test-cluster-east-1a
  backend: test1-nasbackend
autoExportPolicy: true
autoExportCIDRs:
- 10.0.0.0/24
username: admin
password: password
nfsMountOptions: nfsvers=4
```

Beispiel für IPv6-Adressen

Dieses Beispiel zeigt managementLIF Verwenden einer IPv6-Adresse.

```
---
version: 1
storageDriverName: ontap-nas
backendName: nas_ipv6_backend
managementLIF: "[5c5d:5edf:8f:7657:bef8:109b:1b41:d491]"
labels:
  k8scluster: test-cluster-east-1a
  backend: test1-ontap-ipv6
svm: nas_ipv6_svm
username: vsadmin
password: password
```

Amazon FSX für ONTAP mit SMB-Volumes – Beispiel

Der smbShare Parameter ist für FSX for ONTAP mit SMB Volumes erforderlich.

```
---
version: 1
backendName: SMBBackend
storageDriverName: ontap-nas
managementLIF: example.mgmt.fqdn.aws.com
nasType: smb
dataLIF: 10.0.0.15
svm: nfs_svm
smbShare: smb-share
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

Beispiele für Back-Ends mit virtuellen Pools

In den unten gezeigten Beispieldateien für die Backend-Definition werden spezifische Standardwerte für alle Speicherpools festgelegt, z. B. spaceReserve Bei keiner, spaceAllocation Bei false, und encryption Bei false. Die virtuellen Pools werden im Abschnitt Speicher definiert.

Astra Trident bestimmt die Bereitstellungsetiketten im Feld „Kommentare“. Kommentare werden auf FlexVol für gesetzt ontap-nas Oder FlexGroup für ontap-nas-flexgroup. Astra Trident kopiert alle Labels auf einem virtuellen Pool auf das Storage-Volume während der Bereitstellung. Storage-Administratoren können Labels je virtuellen Pool definieren und Volumes nach Label gruppieren.

In diesen Beispielen legen einige Speicherpools eigene fest `spaceReserve`, `spaceAllocation`, und `encryption` Werte und einige Pools überschreiben die Standardwerte.

Beispiel: ONTAP NAS

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
svm: svm_nfs
username: admin
password: <password>
nfsMountOptions: nfsvers=4
defaults:
  spaceReserve: none
  encryption: 'false'
  qosPolicy: standard
labels:
  store: nas_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
- labels:
  app: msoffice
  cost: '100'
  zone: us_east_1a
  defaults:
    spaceReserve: volume
    encryption: 'true'
    unixPermissions: '0755'
    adaptiveQosPolicy: adaptive-premium
- labels:
  app: slack
  cost: '75'
  zone: us_east_1b
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0755'
- labels:
  department: legal
  creditpoints: '5000'
  zone: us_east_1b
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0755'
- labels:
  app: wordpress
```

```
    cost: '50'
  zone: us_east_1c
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0775'
- labels:
  app: mysqldb
  cost: '25'
  zone: us_east_1d
  defaults:
    spaceReserve: volume
    encryption: 'false'
    unixPermissions: '0775'
```

Beispiel für ONTAP NAS FlexGroup

```
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: <password>
defaults:
  spaceReserve: none
  encryption: 'false'
labels:
  store: flexgroup_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
- labels:
  protection: gold
  creditpoints: '50000'
  zone: us_east_1a
  defaults:
    spaceReserve: volume
    encryption: 'true'
    unixPermissions: '0755'
- labels:
  protection: gold
  creditpoints: '30000'
  zone: us_east_1b
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0755'
- labels:
  protection: silver
  creditpoints: '20000'
  zone: us_east_1c
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0775'
- labels:
  protection: bronze
  creditpoints: '10000'
  zone: us_east_1d
  defaults:
```

```
spaceReserve: volume  
encryption: 'false'  
unixPermissions: '0775'
```


Beispiel für die NAS-Ökonomie von ONTAP

```
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: <password>
defaults:
  spaceReserve: none
  encryption: 'false'
labels:
  store: nas_economy_store
region: us_east_1
storage:
- labels:
  department: finance
  creditpoints: '6000'
  zone: us_east_1a
  defaults:
    spaceReserve: volume
    encryption: 'true'
    unixPermissions: '0755'
- labels:
  protection: bronze
  creditpoints: '5000'
  zone: us_east_1b
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0755'
- labels:
  department: engineering
  creditpoints: '3000'
  zone: us_east_1c
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0775'
- labels:
  department: humanresource
  creditpoints: '2000'
  zone: us_east_1d
  defaults:
    spaceReserve: volume
```

```
encryption: 'false'
unixPermissions: '0775'
```

Back-Ends StorageClasses zuordnen

Die folgenden StorageClass-Definitionen finden Sie unter [Beispiele für Back-Ends mit virtuellen Pools](#).

Verwenden der `parameters.selector` Jede StorageClass ruft auf, welche virtuellen Pools zum Hosten eines Volumes verwendet werden können. Auf dem Volume werden die Aspekte im ausgewählten virtuellen Pool definiert.

- Der `protection-gold` StorageClass wird dem ersten und zweiten virtuellen Pool in zugeordnet `ontap-nas-flexgroup` Back-End: Dies sind die einzigen Pools, die Gold-Level-Schutz bieten.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=gold"
  fsType: "ext4"
```

- Der `protection-not-gold` StorageClass wird dem dritten und vierten virtuellen Pool in zugeordnet `ontap-nas-flexgroup` Back-End: Dies sind die einzigen Pools, die Schutz Level nicht Gold bieten.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
```

- Der `app-mysqldb` StorageClass wird dem vierten virtuellen Pool in zugeordnet `ontap-nas` Back-End: Dies ist der einzige Pool, der Storage-Pool-Konfiguration für `mysqldb`-Typ-App bietet.

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"

```

- The protection-silver-creditpoints-20k StorageClass wird dem dritten virtuellen Pool in zugeordnet ontap-nas-flexgroup Back-End: Dies ist der einzige Pool mit Silber-Level-Schutz und 20000 Kreditpunkte.

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"

```

- Der creditpoints-5k StorageClass wird dem dritten virtuellen Pool in zugeordnet ontap-nas Back-End und der zweite virtuelle Pool im ontap-nas-economy Back-End: Dies sind die einzigen Poolangebote mit 5000 Kreditpunkten.

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: csi.trident.netapp.io
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"

```

Astra Trident entscheidet, welcher virtuelle Pool ausgewählt wird und stellt sicher, dass die Storage-Anforderungen erfüllt werden.

Aktualisierung dataLIF Nach der Erstkonfiguration

Sie können die Daten-LIF nach der Erstkonfiguration ändern, indem Sie den folgenden Befehl ausführen, um die neue Backend-JSON-Datei mit aktualisierten Daten-LIF bereitzustellen.

```
tridentctl update backend <backend-name> -f <path-to-backend-json-file-with-updated-data>
```



Wenn PVCs an einen oder mehrere Pods angeschlossen sind, müssen Sie alle entsprechenden Pods herunterfahren und sie dann wieder zurückbringen, damit die neue logische Daten wirksam werden.

Amazon FSX für NetApp ONTAP

Setzen Sie Astra Trident mit Amazon FSX für NetApp ONTAP ein

"Amazon FSX für NetApp ONTAP" ist ein vollständig gemanagter AWS Service, mit dem Kunden Filesysteme auf Basis des NetApp ONTAP Storage-Betriebssystems starten und ausführen können. Mit FSX für ONTAP können Sie bekannte NetApp Funktionen sowie die Performance und Administration nutzen und gleichzeitig die Einfachheit, Agilität, Sicherheit und Skalierbarkeit beim Speichern von Daten in AWS nutzen. FSX für ONTAP unterstützt ONTAP Dateisystemfunktionen und Administrations-APIs.

Überblick

Ein Dateisystem ist die primäre Ressource in Amazon FSX, analog zu einem ONTAP-Cluster vor Ort. Innerhalb jeder SVM können Sie ein oder mehrere Volumes erstellen, bei denen es sich um Daten-Container handelt, die die Dateien und Ordner im Filesystem speichern. Amazon FSX für NetApp ONTAP wird Data ONTAP als gemanagtes Dateisystem in der Cloud zur Verfügung stellen. Der neue Dateisystemtyp heißt **NetApp ONTAP**.

Mit Astra Trident mit Amazon FSX für NetApp ONTAP können Sie sicherstellen, dass Kubernetes Cluster, die in Amazon Elastic Kubernetes Service (EKS) ausgeführt werden, persistente Block- und Datei-Volumes bereitstellen, die durch ONTAP gesichert sind.

Überlegungen

- SMB Volumes:
 - SMB Volumes werden mit unterstützt `ontap-nas` Nur Treiber.
 - SMB-Volumes werden mit dem Astra Trident EKS Add-on nicht unterstützt.
 - Astra Trident unterstützt SMB Volumes, die nur auf Windows Nodes laufenden Pods gemountet werden.
- Vor Astra Trident 24.02 konnten auf Amazon FSX-Dateisystemen erstellte Volumes mit aktivierten automatischen Backups nicht von Trident gelöscht werden. Um dieses Problem in Astra Trident 24.02 oder höher zu vermeiden, geben Sie den an `fsxFilesystemID`, `AWS apiRegion`, `AWS apikey`` Und `AWS `secretKey` In der Back-End-Konfigurationsdatei für AWS FSX für ONTAP.



Wenn Sie eine IAM-Rolle für Astra Trident angeben, können Sie die Angabe des auslassen `apiRegion`, `apiKey`, und `secretKey` Felder explizit in Astra Trident eintragen. Weitere Informationen finden Sie unter "[FSX für ONTAP Konfigurationsoptionen und Beispiele](#)".

FSX für ONTAP-Treiber Details

Sie können Astra Trident mithilfe der folgenden Treiber in Amazon FSX für NetApp ONTAP integrieren:

- `ontap-san`: Jedes bereitgestellte PV ist eine LUN innerhalb seines eigenen Amazon FSX für NetApp ONTAP Volume.
- `ontap-san-economy`: Jedes bereitgestellte PV ist eine LUN mit einer konfigurierbaren Anzahl an LUNs pro Amazon FSX für das NetApp ONTAP Volume.
- `ontap-nas`: Jedes bereitgestellte PV ist ein vollständiger Amazon FSX für NetApp ONTAP Volume.
- `ontap-nas-economy`: Jedes bereitgestellte PV ist ein qtree mit einer konfigurierbaren Anzahl von qtrees pro Amazon FSX für NetApp ONTAP Volume.
- `ontap-nas-flexgroup`: Jedes bereitgestellte PV ist ein vollständiger Amazon FSX für NetApp ONTAP FlexGroup Volume.

Informationen zum Treiber finden Sie unter ["NAS-Treiber"](#) Und ["SAN-Treiber"](#).

Authentifizierung

Astra Trident bietet zwei Authentifizierungsmodi.

- **Zertifikatsbasiert**: Astra Trident kommuniziert mit der SVM auf Ihrem FSX Dateisystem mit einem Zertifikat, das auf Ihrer SVM installiert ist.
- **Anmeldeinformationsbasiert**: Sie können den verwenden `fsxadmin` Benutzer für Ihr Dateisystem oder die `vsadmin` Benutzer für Ihre SVM konfiguriert.



Astra Trident erwartet einen weiteren Betrieb `vsadmin` SVM-Benutzer oder als Benutzer mit einem anderen Namen, der dieselbe Rolle hat. Amazon FSX für NetApp ONTAP hat eine `fsxadmin` Benutzer, die nur einen eingeschränkten Ersatz für die ONTAP bieten `admin` Cluster-Benutzer. Wir empfehlen Ihnen sehr, es zu verwenden `vsadmin` Mit Astra Trident:

Sie können Back-Ends aktualisieren, um zwischen auf Anmeldeinformationen basierenden und zertifikatbasierten Methoden zu verschieben. Wenn Sie jedoch versuchen, **Anmeldeinformationen und Zertifikate** bereitzustellen, schlägt die Backend-Erstellung fehl. Um zu einer anderen Authentifizierungsmethode zu wechseln, müssen Sie die vorhandene Methode von der Backend-Konfiguration entfernen.

Weitere Informationen zur Aktivierung der Authentifizierung finden Sie in der Authentifizierung für Ihren Treibertyp:

- ["ONTAP NAS-Authentifizierung"](#)
- ["ONTAP SAN-Authentifizierung"](#)

Cloud-Identität für EKS

Die Cloud-Identität ermöglicht Kubernetes-Pods den Zugriff auf AWS Ressourcen durch Authentifizierung als AWS IAM-Rolle anstatt durch Angabe explizite AWS-Anmeldedaten.

Um die Vorteile der Cloud-Identität in AWS zu nutzen, müssen Sie über folgende Voraussetzungen verfügen:

- Implementierung eines Kubernetes Clusters mit EKS

- Astra Trident installiert, einschließlich `cloudProvider` Angeben "AWS" Und `cloudIdentity` Festlegen der AWS IAM-Rolle

Betreiber von Trident

Um Astra Trident mit dem Trident-Operator zu installieren, bearbeiten Sie `tridentorchestrator_cr.yaml` Einstellen `cloudProvider` Bis "AWS" Und gesetzt `cloudIdentity` Zur AWS IAM-Rolle.

Beispiel:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "AWS"
  cloudIdentity: "'eks.amazonaws.com/role-arn:
arn:aws:iam::123456:role/astratrident-role'"
```

Helm

Legen Sie die Werte für **Cloud Provider** und **Cloud Identity** unter Verwendung der folgenden Umgebungsvariablen fest:

```
export CP="AWS"
export CI="'eks.amazonaws.com/role-arn:
arn:aws:iam::123456:role/astratrident-role'"
```

Im folgenden Beispiel werden Astra Trident und Sätze installiert `cloudProvider` Bis AWS Verwenden der Umgebungsvariable `$CP` Und legt die 'CloudIdentity' über die Umgebungsvariable fest `$CI`:

```
helm install trident trident-operator-100.2402.0.tgz --set
cloudProvider=$CP --set cloudIdentity=$CI
```

<code>tridentctl</code>

Legen Sie die Werte für **Cloud Provider** und **Cloud Identity** unter Verwendung der folgenden Umgebungsvariablen fest:

```
export CP="AWS"
export CI="'eks.amazonaws.com/role-arn:
arn:aws:iam::123456:role/astratrident-role'"
```

Im folgenden Beispiel wird Astra Trident installiert und legt den fest `cloud-provider` Flag an `$CP`, und `cloud-identity` Bis `$CI`:

```
tridentctl install --cloud-provider=$CP --cloud-identity="$CI" -n
trident
```

Weitere Informationen

- ["Dokumentation zu Amazon FSX für NetApp ONTAP"](#)
- ["Blogbeitrag zu Amazon FSX für NetApp ONTAP"](#)

Integration von Amazon FSX für NetApp ONTAP

Sie können Ihr Filesystem Amazon FSX für NetApp ONTAP mit Astra Trident integrieren, um sicherzustellen, dass Kubernetes Cluster, die in Amazon Elastic Kubernetes Service (EKS) ausgeführt werden, persistente Block- und File-Volumes mit ONTAP bereitstellen können.

Anforderungen

Zusätzlich zu ["Anforderungen von Astra Trident"](#) Zur Integration von FSX für ONTAP mit Astra Trident benötigen Sie Folgendes:

- Ein vorhandener Amazon EKS-Cluster oder selbst verwalteter Kubernetes-Cluster mit `kubectl` Installiert.
- Ein vorhandenes Amazon FSX for NetApp ONTAP-Filesystem und eine Storage Virtual Machine (SVM), die über die Worker-Nodes Ihres Clusters erreichbar ist.
- Worker-Nodes, die vorbereitet sind ["NFS oder iSCSI"](#).



Achten Sie darauf, dass Sie die für Amazon Linux und Ubuntu erforderlichen Schritte zur Knotenvorbereitung befolgen ["Amazon Machine Images"](#) (Amis) je nach EKS AMI-Typ.

- Astra Trident unterstützt SMB Volumes, die nur auf Windows Nodes laufenden Pods gemountet werden. Siehe [Vorbereitung zur Bereitstellung von SMB Volumes](#) Entsprechende Details.

Integration von ONTAP-SAN- und NAS-Treibern



Wenn Sie für SMB Volumes konfigurieren, müssen Sie lesen [Vorbereitung zur Bereitstellung von SMB Volumes](#) Bevor Sie das Backend erstellen.

Schritte

1. Implementieren Sie Astra Trident mit einer der Lösungen ["Implementierungsoptionen"](#).
2. Sammeln Sie den SVM-Management-LIF-DNS-Namen. Suchen Sie zum Beispiel mit der AWS CLI nach `DNSName` Eintrag unter `Endpoints` → `Management` Nach Ausführung des folgenden Befehls:

```
aws fsx describe-storage-virtual-machines --region <file system region>
```

3. Erstellen und Installieren von Zertifikaten für ["NAS-Back-End-Authentifizierung"](#) Oder ["SAN-Back-End-Authentifizierung"](#).



Sie können sich bei Ihrem Dateisystem anmelden (zum Beispiel Zertifikate installieren) mit SSH von überall, wo Sie Ihr Dateisystem erreichen können. Verwenden Sie die `fsxadmin` Benutzer, das Kennwort, das Sie beim Erstellen Ihres Dateisystems konfiguriert haben, und der Management-DNS-Name von `aws fsx describe-file-systems`.

4. Erstellen Sie eine Backend-Datei mithilfe Ihrer Zertifikate und des DNS-Namens Ihrer Management LIF, wie im folgenden Beispiel dargestellt:

YAML

```
version: 1
storageDriverName: ontap-san
backendName: customBackendName
managementLIF: svm-XXXXXXXXXXXXXXXXXX.fs-XXXXXXXXXXXXXXXXXX.fsx.us-
east-2.aws.internal
svm: svm01
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
```

JSON

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "customBackendName",
  "managementLIF": "svm-XXXXXXXXXXXXXXXXXX.fs-
XXXXXXXXXXXXXXXXXXXXX.fsx.us-east-2.aws.internal",
  "svm": "svm01",
  "clientCertificate": "ZXR0ZXJwYXB...ICMgJ3BhcGVyc2",
  "clientPrivateKey": "vciwKIyAgZG...0cnksIGRlc2NyaX",
  "trustedCACertificate": "zcyBbaG...b3Igb3duIGNsYXNz"
}
```

Alternativ können Sie eine Back-End-Datei mit den im AWS Secret Manager gespeicherten SVM-Zugangsdaten (Benutzername und Passwort) erstellen, wie im folgenden Beispiel dargestellt:

YAML

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
spec:
  version: 1
  storageDriverName: ontap-nas
  backendName: tbc-ontap-nas
  svm: svm-name
  aws:
    fsxFileSystemID: fs-xxxxxxxxxx
  managementLIF:
  credentials:
    name: "arn:aws:secretsmanager:us-west-2:xxxxxxx:secret:secret-
name"
    type: awsarn
```

JSON

```
{
  "apiVersion": "trident.netapp.io/v1",
  "kind": "TridentBackendConfig",
  "metadata": {
    "name": "backend-tbc-ontap-nas"
  },
  "spec": {
    "version": 1,
    "storageDriverName": "ontap-nas",
    "backendName": "tbc-ontap-nas",
    "svm": "svm-name",
    "aws": {
      "fsxFileSystemID": "fs-xxxxxxxxxx"
    },
    "managementLIF": null,
    "credentials": {
      "name": "arn:aws:secretsmanager:us-west-
2:xxxxxxx:secret:secret-name",
      "type": "awsarn"
    }
  }
}
```

Informationen zum Erstellen von Back-Ends finden Sie unter folgenden Links:

- ["Konfigurieren Sie ein Backend mit ONTAP NAS-Treibern"](#)
- ["Konfigurieren Sie ein Backend mit ONTAP-SAN-Treibern"](#)

Vorbereitung zur Bereitstellung von SMB Volumes

Sie können SMB-Volumes mit bereitstellen `ontap-nas` Treiber. Bevor Sie fertig sind [Integration von ONTAP-SAN- und NAS-Treibern](#) Führen Sie die folgenden Schritte aus.

Bevor Sie beginnen

Bevor Sie SMB-Volumes mit bereitstellen können `ontap-nas` Treiber, müssen Sie Folgendes haben.

- Kubernetes-Cluster mit einem Linux-Controller-Knoten und mindestens einem Windows-Worker-Node, auf dem Windows Server 2019 ausgeführt wird. Astra Trident unterstützt SMB Volumes, die nur auf Windows Nodes laufenden Pods gemountet werden.
- Mindestens ein Astra Trident-Geheimnis, der Ihre Active Directory-Anmeldedaten enthält. Um Geheimnis zu erzeugen `smbcreds`:

```
kubectl create secret generic smbcreds --from-literal username=user
--from-literal password='password'
```

- Ein CSI-Proxy, der als Windows-Dienst konfiguriert ist. Zum Konfigurieren von A `csi-proxy` Weitere Informationen finden Sie unter ["GitHub: CSI-Proxy"](#) Oder ["GitHub: CSI Proxy für Windows"](#) Für Kubernetes-Knoten, die auf Windows ausgeführt werden.

Schritte

1. Erstellen von SMB-Freigaben Sie können SMB-Admin-Freigaben auf zwei Arten erstellen: Mit ["Microsoft Management Console"](#) Snap-in für freigegebene Ordner oder mit der ONTAP-CLI. So erstellen Sie SMB-Freigaben mithilfe der ONTAP-CLI:

- a. Erstellen Sie bei Bedarf die Verzeichnispfadstruktur für die Freigabe.

Der `vserver cifs share create` Der Befehl überprüft während der Freigabenerstellung den in der Option `-path` angegebenen Pfad. Wenn der angegebene Pfad nicht vorhanden ist, schlägt der Befehl fehl.

- b. Erstellen einer mit der angegebenen SVM verknüpften SMB-Freigabe:

```
vserver cifs share create -vserver vserver_name -share-name
share_name -path path [-share-properties share_properties,...]
[other_attributes] [-comment text]
```

- c. Vergewissern Sie sich, dass die Freigabe erstellt wurde:

```
vserver cifs share show -share-name share_name
```



Siehe ["Erstellen Sie eine SMB-Freigabe"](#) Vollständige Informationen.

2. Beim Erstellen des Backend müssen Sie Folgendes konfigurieren, um SMB-Volumes festzulegen. Alle FSX-Konfigurationsoptionen für ONTAP-Backend finden Sie unter ["FSX für ONTAP Konfigurationsoptionen und Beispiele"](#).

Parameter	Beschreibung	Beispiel
smbShare	Sie können eine der folgenden Optionen angeben: Den Namen einer SMB-Freigabe, die mit der Microsoft Management Console oder der ONTAP-CLI erstellt wurde, oder einen Namen, mit dem Astra Trident die SMB-Freigabe erstellen kann. Dieser Parameter ist für Amazon FSX for ONTAP Back-Ends erforderlich.	smb-share
nasType	Muss auf eingestellt sein smb. Wenn Null, wird standardmäßig auf gesetzt nfs.	smb
securityStyle	Sicherheitstyp für neue Volumes. Muss auf eingestellt sein ntfs Oder mixed Für SMB Volumes.	ntfs Oder mixed Für SMB Volumes
unixPermissions	Modus für neue Volumes. Muss für SMB Volumes leer gelassen werden.	“

FSX für ONTAP Konfigurationsoptionen und Beispiele

Erfahren Sie mehr über Back-End-Konfigurationsoptionen für Amazon FSX für ONTAP. Dieser Abschnitt enthält Beispiele für die Back-End-Konfiguration.

Back-End-Konfigurationsoptionen

Die Back-End-Konfigurationsoptionen finden Sie in der folgenden Tabelle:

Parameter	Beschreibung	Beispiel
version		Immer 1
storageDriverName	Name des Speichertreibers	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economy
backendName	Benutzerdefinierter Name oder das Storage-Backend	Treibername + „_“ + DatenLIF

Parameter	Beschreibung	Beispiel
managementLIF	<p>IP-Adresse eines Clusters oder einer SVM-Management-LIF</p> <p>Es kann ein vollständig qualifizierter Domänenname (FQDN) angegeben werden.</p> <p>Kann so eingestellt werden, dass IPv6-Adressen verwendet werden, wenn Astra Trident mit dem IPv6-Flag installiert wurde. IPv6-Adressen müssen in eckigen Klammern definiert werden, z. B. [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].</p>	„10.0.0.1“, „[2001:1234:abcd::fefe]“
dataLIF	<p>IP-Adresse des LIF-Protokolls.</p> <p>ONTAP NAS drivers: Wir empfehlen die Angabe von dataLIF. Falls nicht vorgesehen, ruft Astra Trident Daten-LIFs von der SVM ab. Sie können einen vollständig qualifizierten Domännennamen (FQDN) angeben, der für die NFS-Mount-Vorgänge verwendet werden soll. Damit können Sie ein Round-Robin-DNS zum Load-Balancing über mehrere Daten-LIFs erstellen. Kann nach der Anfangseinstellung geändert werden. Siehe .</p> <p>ONTAP-SAN-Treiber: Geben Sie nicht für iSCSI an. Astra Trident verwendet die ONTAP Selective LUN Map, um die iSCSI LIFs zu ermitteln, die für die Einrichtung einer Multi-Path-Sitzung erforderlich sind. Eine Warnung wird erzeugt, wenn dataLIF explizit definiert ist.</p> <p>Kann so eingestellt werden, dass IPv6-Adressen verwendet werden, wenn Astra Trident mit dem IPv6-Flag installiert wurde. IPv6-Adressen müssen in eckigen Klammern definiert werden, z. B. [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].</p>	

Parameter	Beschreibung	Beispiel
autoExportPolicy	Aktivieren Sie die automatische Erstellung von Exportrichtlinien und aktualisieren Sie [Boolean]. Verwenden der autoExportPolicy Und autoExportCIDRs Optionen: Astra Trident kann Exportrichtlinien automatisch verwalten.	false
autoExportCIDRs	Liste der CIDRs, nach denen die Node-IPs von Kubernetes gefiltert werden sollen autoExportPolicy Ist aktiviert. Verwenden der autoExportPolicy Und autoExportCIDRs Optionen: Astra Trident kann Exportrichtlinien automatisch verwalten.	„[„0.0.0.0/0“, „:/0“]“
labels	Satz willkürlicher JSON-formatierter Etiketten für Volumes	“ ”
clientCertificate	Base64-codierter Wert des Clientzertifikats. Wird für zertifikatbasierte Authentifizierung verwendet	“ ”
clientPrivateKey	Base64-kodierte Wert des privaten Client-Schlüssels. Wird für zertifikatbasierte Authentifizierung verwendet	“ ”
trustedCACertificate	Base64-kodierte Wert des vertrauenswürdigen CA-Zertifikats. Optional Wird für die zertifikatbasierte Authentifizierung verwendet.	“ ”
username	Benutzername zum Herstellen einer Verbindung zum Cluster oder zur SVM. Wird für die Anmeldeinformationsbasierte Authentifizierung verwendet. Beispiel: Vsadmin.	
password	Passwort für die Verbindung mit dem Cluster oder der SVM Wird für die Anmeldeinformationsbasierte Authentifizierung verwendet.	
svm	Zu verwendende Storage Virtual Machine	Abgeleitet, wenn eine SVM Management LIF angegeben ist.

Parameter	Beschreibung	Beispiel
storagePrefix	<p>Das Präfix wird beim Bereitstellen neuer Volumes in der SVM verwendet.</p> <p>Kann nach der Erstellung nicht geändert werden. Um diesen Parameter zu aktualisieren, müssen Sie ein neues Backend erstellen.</p>	trident
limitAggregateUsage	<p>Nicht für Amazon FSX für NetApp ONTAP angeben.</p> <p>Die vorhanden <code>fsxadmin</code> Und <code>vsadmin</code> Enthalten Sie nicht die erforderlichen Berechtigungen, um die Aggregatnutzung abzurufen und sie mit Astra Trident zu begrenzen.</p>	Verwenden Sie ihn nicht.
limitVolumeSize	<p>Bereitstellung fehlgeschlagen, wenn die angeforderte Volume-Größe über diesem Wert liegt.</p> <p>Schränkt auch die maximale Größe der Volumes ein, die es für qtrees und LUNs verwaltet, und auf ein <code>qtreesPerFlexvol</code> Mit Option kann die maximale Anzahl von qtrees pro FlexVol angepasst werden.</p>	„“ (nicht standardmäßig durchgesetzt)
lunsPerFlexvol	<p>Die maximale Anzahl an LUNs pro FlexVol muss im Bereich [50, 200] liegen.</p> <p>Nur SAN</p>	100
debugTraceFlags	<p>Fehler-Flags bei der Fehlerbehebung beheben. Beispiel: { „API“:false, „Methode“:true}</p> <p>Verwenden Sie es nicht <code>debugTraceFlags</code> Es sei denn, Sie beheben Fehler und benötigen einen detaillierten Log Dump.</p>	Null

Parameter	Beschreibung	Beispiel
nfsMountOptions	<p>Kommagetrennte Liste von NFS-Mount-Optionen.</p> <p>Die Mount-Optionen für Kubernetes-persistente Volumes werden normalerweise in Storage-Klassen angegeben. Wenn jedoch keine Mount-Optionen in einer Storage-Klasse angegeben sind, stellt Astra Trident die Mount-Optionen bereit, die in der Konfigurationsdatei des Storage-Back-End angegeben sind.</p> <p>Wenn in der Storage-Klasse oder der Konfigurationsdatei keine Mount-Optionen angegeben sind, stellt Astra Trident keine Mount-Optionen für ein damit verbundener persistentes Volume fest.</p>	“ ”
nasType	<p>Konfiguration der Erstellung von NFS- oder SMB-Volumes</p> <p>Die Optionen lauten <code>nfs</code>, <code>smb</code>, Oder Null.</p> <p>Muss auf eingestellt sein <code>smb</code> Für SMB-Volumes. Einstellung auf null setzt standardmäßig auf NFS-Volumes.</p>	<code>nfs</code>
qtreesPerFlexvol	Maximale Ques pro FlexVol, muss im Bereich [50, 300] liegen	200
smbShare	<p>Sie können eine der folgenden Optionen angeben: Den Namen einer SMB-Freigabe, die mit der Microsoft Management Console oder der ONTAP-CLI erstellt wurde, oder einen Namen, mit dem Astra Trident die SMB-Freigabe erstellen kann.</p> <p>Dieser Parameter ist für Amazon FSX for ONTAP Back-Ends erforderlich.</p>	<code>smb-share</code>

Parameter	Beschreibung	Beispiel
useREST	<p>Boolescher Parameter zur Verwendung von ONTAP REST-APIs. Technische Vorschau</p> <p>useREST Wird als Tech-Vorschau bereitgestellt, das für Testumgebungen und nicht für Produktions-Workloads empfohlen wird. Wenn eingestellt auf <code>true</code>, Astra Trident wird ONTAP REST APIs zur Kommunikation mit dem Backend verwenden.</p> <p>Diese Funktion erfordert ONTAP 9.11.1 und höher. Darüber hinaus muss die verwendete ONTAP-Login-Rolle Zugriff auf den haben <code>ontap</code> Applikation. Dies wird durch die vordefinierte zufrieden <code>vsadmin</code> Und <code>cluster-admin</code> Rollen:</p>	false
aws	<p>In der Konfigurationsdatei für AWS FSX für ONTAP können Sie Folgendes angeben:</p> <ul style="list-style-type: none"> - <code>fsxFilesystemID</code>: Geben Sie die ID des AWS FSX Dateisystems an. - <code>apiRegion</code>: Name der AWS API-Region. - <code>apikey</code>: AWS API-Schlüssel. - <code>secretKey</code>: AWS geheimer Schlüssel. 	<pre>"" "" ""</pre>
credentials	<p>Geben Sie die FSX SVM-Anmeldeinformationen an, die in AWS Secret Manager zu speichern sind.</p> <ul style="list-style-type: none"> - <code>name</code>: Amazon Resource Name (ARN) des Geheimnisses, das die Zugangsdaten von SVM enthält. - <code>type</code>: Auf eingestellt <code>awsarn</code>. <p>Siehe "Erstellen Sie einen AWS Secrets Manager-Schlüssel" Finden Sie weitere Informationen.</p>	

Aktualisierung dataLIF Nach der Erstkonfiguration

Sie können die Daten-LIF nach der Erstkonfiguration ändern, indem Sie den folgenden Befehl ausführen, um die neue Backend-JSON-Datei mit aktualisierten Daten-LIF bereitzustellen.

```
tridentctl update backend <backend-name> -f <path-to-backend-json-file-with-updated-data>
```



Wenn PVCs an einen oder mehrere Pods angeschlossen sind, müssen Sie alle entsprechenden Pods herunterfahren und sie dann wieder zurückbringen, damit die neue logische Daten wirksam werden.

Back-End-Konfigurationsoptionen für die Bereitstellung von Volumes

Sie können die Standardbereitstellung mit diesen Optionen im steuern `defaults` Abschnitt der Konfiguration. Ein Beispiel finden Sie unten in den Konfigurationsbeispielen.

Parameter	Beschreibung	Standard
<code>spaceAllocation</code>	Speicherplatzzuweisung für LUNs	<code>true</code>
<code>spaceReserve</code>	Space Reservation Mode; „none“ (Thin) oder „Volume“ (Thick)	<code>none</code>
<code>snapshotPolicy</code>	Die Snapshot-Richtlinie zu verwenden	<code>none</code>
<code>qosPolicy</code>	<p>QoS-Richtliniengruppe zur Zuweisung für erstellte Volumes Wählen Sie eine der <code>qosPolicy</code> oder <code>adaptiveQosPolicy</code> pro Storage-Pool oder Backend.</p> <p>Die Verwendung von QoS Policy Groups mit Astra Trident erfordert ONTAP 9.8 oder höher.</p> <p>Wir empfehlen die Verwendung einer nicht gemeinsam genutzten QoS-Richtliniengruppe und stellen sicher, dass die Richtliniengruppe auf jede Komponente einzeln angewendet wird. Eine Richtliniengruppe für Shared QoS führt zur Durchsetzung der Obergrenze für den Gesamtdurchsatz aller Workloads.</p>	“ ”
<code>adaptiveQosPolicy</code>	<p>Adaptive QoS-Richtliniengruppe mit Zuordnung für erstellte Volumes Wählen Sie eine der <code>qosPolicy</code> oder <code>adaptiveQosPolicy</code> pro Storage-Pool oder Backend.</p> <p>Nicht unterstützt durch <code>ontap-nas</code>-Ökonomie</p>	“ ”
<code>snapshotReserve</code>	Prozentsatz des für Snapshots reservierten Volumes „0“	Wenn <code>snapshotPolicy</code> <code>ist none</code> , else „

Parameter	Beschreibung	Standard
splitOnClone	Teilen Sie einen Klon bei der Erstellung von seinem übergeordneten Objekt auf	false
encryption	<p>Aktivieren Sie NetApp Volume Encryption (NVE) auf dem neuen Volume, standardmäßig aktiviert false. NVE muss im Cluster lizenziert und aktiviert sein, damit diese Option verwendet werden kann.</p> <p>Wenn NAE auf dem Backend aktiviert ist, wird jedes im Astra Trident bereitgestellte Volume NAE aktiviert.</p> <p>Weitere Informationen finden Sie unter: "Astra Trident arbeitet mit NVE und NAE zusammen".</p>	false
luksEncryption	<p>Aktivieren Sie die LUKS-Verschlüsselung. Siehe "Linux Unified Key Setup (LUKS) verwenden".</p> <p>Nur SAN</p>	"
tieringPolicy	<p>Tiering-Richtlinie für die Nutzung none</p>	snapshot-only Für Konfiguration vor ONTAP 9.5 SVM-DR
unixPermissions	<p>Modus für neue Volumes.</p> <p>Leere leer für SMB Volumen.</p>	"
securityStyle	<p>Sicherheitstyp für neue Volumes.</p> <p>NFS unterstützt mixed Und unix Sicherheitsstile.</p> <p>SMB-Support mixed Und ntfs Sicherheitsstile.</p>	<p>NFS-Standard ist unix.</p> <p>Der SMB-Standardwert ist ntfs.</p>

Beispielkonfigurationen

Konfiguration der Storage-Klasse für SMB Volumes

Wird verwendet `nasType`, `node-stage-secret-name`, und `node-stage-secret-namespace`, Sie können ein SMB-Volume angeben und die erforderlichen Active Directory-Anmeldeinformationen angeben. SMB Volumes werden mit unterstützt `ontap-nas` Nur Treiber.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: nas-smb-sc
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
```

Konfiguration für AWS FSX für ONTAP mit Secret Manager

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
spec:
  version: 1
  storageDriverName: ontap-nas
  backendName: tbc-ontap-nas
  svm: svm-name
  aws:
    fsxFileSystemID: fs-xxxxxxxxxx
  managementLIF:
  credentials:
    name: "arn:aws:secretsmanager:us-west-2:xxxxxxx:secret:secret-
name"
    type: awsarn
```

Konfiguration des Astra Trident EKS Add-On Version 23.10 im EKS Cluster

Astra Trident optimiert das Amazon FSX für NetApp ONTAP Storage-Management in Kubernetes, damit sich Ihre Entwickler und Administratoren voll und ganz auf den Applikationseinsatz konzentrieren können. Das Add-on für Astra Trident EKS enthält die neuesten Sicherheits-Patches und Bug Fixes. Es wurde von AWS für die Zusammenarbeit mit Amazon EKS validiert. Mit dem EKS-Add-on können Sie

sicherstellen, dass Ihre Amazon EKS-Cluster sicher und stabil sind und den Arbeitsaufwand für die Installation, Konfiguration und Aktualisierung von Add-Ons verringern.

Voraussetzungen

Stellen Sie vor dem Konfigurieren des Astra Trident Add-ons für AWS EKS sicher, dass folgende Voraussetzungen erfüllt sind:

- Ein Amazon EKS Cluster-Konto mit Add-on-Abonnement
- AWS Berechtigungen für den AWS Marketplace:
"aws-marketplace:ViewSubscriptions",
"aws-marketplace:Subscribe",
"aws-marketplace:Unsubscribe"
- AMI-Typ: Amazon Linux 2 (AL2_x86_64) oder Amazon Linux 2 Arm(AL2_ARM_64)
- Knotentyp: AMD oder ARM
- Ein bestehendes Amazon FSX für NetApp ONTAP-Filesystem

Schritte

1. Navigieren Sie auf Ihrem EKS Kubernetes-Cluster zur Registerkarte **Add-ons**.
2. Gehen Sie zu **AWS Marketplace Add-ons** und wählen Sie die Kategorie *Storage*.
3. Suchen Sie **AstraTrident by NetApp** und aktivieren Sie das Kontrollkästchen für das Astra Trident Add-on.
4. Wählen Sie die gewünschte Version des Add-ons aus.
5. Wählen Sie die Option IAM-Rolle aus, die vom Knoten übernommen werden soll.
6. Konfigurieren Sie die gewünschten optionalen Einstellungen, und wählen Sie **Weiter**.
7. Wählen Sie **Erstellen**.
8. Überprüfen Sie, ob der Status des Add-ons *Active* lautet.

Installieren/deinstallieren Sie das Astra Trident EKS Add-on über CLI

Installation des Astra Trident EKS Add-On über CLI:

Im folgenden Beispiel wird das Add-on für Astra Trident EKS installiert:

```
eksctl create addon --cluster K8s-arm --name netapp_trident-operator --version v23.10.0-eksbuild.  
eksctl create addon --cluster K8s-arm --name netapp_trident-operator --version v23.10.0-eksbuild.1 (Mit einer dedizierten Version)
```

Deinstallieren Sie das Astra Trident EKS-Add-On über CLI:

Mit dem folgenden Befehl wird das Astra Trident EKS Add-on deinstalliert:

```
eksctl delete addon --cluster K8s-arm --name netapp_trident-operator
```

Back-Ends mit kubectl erstellen

Ein Backend definiert die Beziehung zwischen Astra Trident und einem Storage-System. Er erzählt Astra Trident, wie man mit diesem Storage-System kommuniziert und wie Astra Trident Volumes darauf bereitstellen sollte. Nach der Installation von Astra Trident ist der nächste Schritt die Erstellung eines Backend. Der `TridentBackendConfig` Mit Custom Resource Definition (CRD) können Sie Trident Back-Ends direkt über die Kubernetes Schnittstelle erstellen und managen. Dies können Sie mit `tun kubectl` Oder das vergleichbare CLI Tool für Ihre Kubernetes Distribution.

`TridentBackendConfig`

`TridentBackendConfig` (`tbc`, `tbconfig`, `tbackendconfig`) Ist ein Front-End, Namensvetter CRD, mit dem Sie Astra Trident Back-Ends mit verwalten können `kubectl`. Kubernetes- und Storage-Administratoren können Back-Ends jetzt direkt über die Kubernetes-CLI erstellen und managen, ohne dass ein dediziertes Dienstprogramm für die Befehlszeilenschnittstelle erforderlich ist (`tridentctl`).

Bei der Erstellung eines `TridentBackendConfig` Objekt, geschieht Folgendes:

- Ein Back-End wird automatisch von Astra Trident auf Basis der von Ihnen zu erstellenden Konfiguration erstellt. Dies wird intern als `A` dargestellt `TridentBackend` (`tbe`, `tridentbackend`) CR.
- Der `TridentBackendConfig` Ist eindeutig an `A` gebunden `TridentBackend` Das wurde von Astra Trident entwickelt.

Beide `TridentBackendConfig` Pflegt eine 1:1-Zuordnung mit einem `TridentBackend`. Die erstere Schnittstelle, die dem Benutzer zum Design und zur Konfiguration von Back-Ends zur Verfügung gestellt wird. Letztere ist, wie Trident das tatsächliche Backend-Objekt darstellt.



`TridentBackend` CRS werden automatisch von Astra Trident erstellt. Sie sollten diese nicht ändern. Wenn Sie an Back-Ends Aktualisierungen vornehmen möchten, ändern Sie das `TridentBackendConfig` Objekt:

Im folgenden Beispiel finden Sie Informationen zum Format des `TridentBackendConfig` CR:

```

apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret

```

Sie können sich auch die Beispiele im ansehen ["trident-Installationsprogramm"](#) Verzeichnis für Beispielkonfigurationen für die gewünschte Speicherplattform/den gewünschten Service.

Der spec Nimmt Back-End-spezifische Konfigurationsparameter ein. In diesem Beispiel verwendet das Backend `ontap-san` Speichertreiber und verwendet die hier tabellarischen Konfigurationsparameter. Eine Liste der Konfigurationsoptionen für den gewünschten Speichertreiber finden Sie im ["Back-End-Konfigurationsinformationen für Ihren Speichertreiber"](#).

Der spec Abschnitt enthält auch `credentials` Und `deletionPolicy` Felder, die neu in den eingeführt werden `TridentBackendConfig` CR:

- `credentials`: Dieser Parameter ist ein Pflichtfeld und enthält die Anmeldeinformationen, die zur Authentifizierung mit dem Speichersystem/Service verwendet werden. Dies ist auf ein vom Benutzer erstelltes Kubernetes Secret festgelegt. Die Anmeldeinformationen können nicht im Klartext weitergegeben werden und führen zu einem Fehler.
- `deletionPolicy`: Dieses Feld definiert, was passieren soll, wenn der `TridentBackendConfig` Wird gelöscht. Es kann einen von zwei möglichen Werten annehmen:
 - `delete`: Dies führt zur Löschung beider `TridentBackendConfig` CR und das zugehörige Backend. Dies ist der Standardwert.
 - `retain`: Wenn a `TridentBackendConfig` CR wird gelöscht, die Backend-Definition ist weiterhin vorhanden und kann mit verwaltet werden `tridentctl`. Einstellen der Löschrictlinie auf `retain` Benutzer können ein Downgrade auf eine frühere Version (vor 21.04) durchführen und die erstellten Back-Ends behalten. Der Wert für dieses Feld kann nach einem aktualisiert werden `TridentBackendConfig` Wird erstellt.



Der Name eines Backend wird mit festgelegt `spec.backendName`. Wenn nicht angegeben, wird der Name des Backend auf den Namen des gesetzt `TridentBackendConfig` Objekt (`metadata.name`). Es wird empfohlen, mit explizit Back-End-Namen festzulegen `spec.backendName`.



Back-Ends, die mit `tridentctl` erstellt wurden, sind `tridentctl` nicht zugeordnet. `TridentBackendConfig` Objekt: Sie können solche Back-Ends mit `kubectl` verwalten. Durch Erstellen von `TridentBackendConfig` CR. Es muss sorgfältig darauf geachtet werden, identische Konfigurationsparameter festzulegen (z. B. `spec.backendName`, `spec.storagePrefix`, `spec.storageDriverName`, und so weiter). Astra Trident bindet automatisch die neu erstellte `TridentBackendConfig` mit dem bereits vorhandenen Backend.

Schritte im Überblick

Um ein neues Backend mit `kubectl` zu erstellen, sollten Sie Folgendes tun:

1. Erstellen Sie ein **"Kubernetes Secret"**. Das Geheimnis enthält die Zugangsdaten, die Astra Trident zur Kommunikation mit dem Storage-Cluster/Service benötigt.
2. Erstellen Sie ein `TridentBackendConfig` Objekt: Dies enthält Angaben zum Storage-Cluster/Service und verweist auf das im vorherigen Schritt erstellte Geheimnis.

Nachdem Sie ein Backend erstellt haben, können Sie den Status mit `kubectl get tbc <tbc-name> -n <trident-namespace>` beobachten und sammeln Sie weitere Details.

Schritt: Ein Kubernetes Secret erstellen

Erstellen Sie einen geheimen Schlüssel, der die Anmeldedaten für den Zugriff auf das Backend enthält. Dies ist nur bei jedem Storage Service/jeder Plattform möglich. Hier ein Beispiel:

```
kubectl -n trident create -f backend-tbc-ontap-san-secret.yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-san-secret
type: Opaque
stringData:
  username: cluster-admin
  password: password
```

In dieser Tabelle sind die Felder zusammengefasst, die für jede Speicherplattform im Secret enthalten sein müssen:

Beschreibung der geheimen Felder der Speicherplattform	Geheim	Feldbeschreibung
Azure NetApp Dateien	Client-ID	Die Client-ID aus einer App-Registrierung
Cloud Volumes Service für GCP	Private_Schlüssel_id	ID des privaten Schlüssels. Teil des API-Schlüssels für GCP-Servicekonto mit CVS-Administratorrolle

Beschreibung der geheimen Felder der Speicherplattform	Geheim	Feldbeschreibung
Cloud Volumes Service für GCP	Privater_Schlüssel	Privater Schlüssel. Teil des API-Schlüssels für GCP-Servicekonto mit CVS-Administratorrolle
Element (NetApp HCI/SolidFire)	Endpunkt	MVIP für den SolidFire-Cluster mit Mandanten-Anmeldedaten
ONTAP	Benutzername	Benutzername für die Verbindung mit dem Cluster/SVM. Wird für die Anmeldeinformationsbasierte Authentifizierung verwendet
ONTAP	Passwort	Passwort für die Verbindung mit dem Cluster/SVM Wird für die Anmeldeinformationsbasierte Authentifizierung verwendet
ONTAP	KundenPrivateKey	Base64-kodierte Wert des privaten Client-Schlüssels. Wird für die zertifikatbasierte Authentifizierung verwendet
ONTAP	ChapUsername	Eingehender Benutzername. Erforderlich, wenn usCHAP=true verwendet wird. Für <code>ontap-san</code> Und <code>ontap-san-economy</code>
ONTAP	ChapInitiatorSecret	CHAP-Initiatorschlüssel. Erforderlich, wenn usCHAP=true verwendet wird. Für <code>ontap-san</code> Und <code>ontap-san-economy</code>
ONTAP	ChapTargetBenutzername	Zielbenutzername. Erforderlich, wenn usCHAP=true verwendet wird. Für <code>ontap-san</code> Und <code>ontap-san-economy</code>
ONTAP	ChapTargetInitiatorSecret	Schlüssel für CHAP-Zielinitiator. Erforderlich, wenn usCHAP=true verwendet wird. Für <code>ontap-san</code> Und <code>ontap-san-economy</code>

Auf das in diesem Schritt erstellte Geheimnis wird im verwiesen `spec.credentials` Feld von `TridentBackendConfig` Objekt, das im nächsten Schritt erstellt wird.

Schritt 2: Erstellen Sie die `TridentBackendConfig` CR

Sie sind jetzt bereit, Ihre zu erstellen `TridentBackendConfig` CR. In diesem Beispiel wird ein Backend verwendet, das den verwendet `ontap-san` Treiber wird mithilfe des erstellt `TridentBackendConfig` Unten gezeigte Objekte:

```
kubectl -n trident create -f backend-tbc-ontap-san.yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret
```

Schritt 3: Überprüfen Sie den Status des `TridentBackendConfig` CR

Nun, da Sie die erstellt haben `TridentBackendConfig` CR, Sie können den Status überprüfen. Das folgende Beispiel zeigt:

```
kubectl -n trident get tbc backend-tbc-ontap-san
```

NAME	BACKEND	NAME	UUID
backend-tbc-ontap-san	ontap-san-backend		8d24fce7-6f60-4d4a-8ef6-bab2699e6ab8
Bound	Success		

Ein Back-End wurde erfolgreich erstellt und an das gebunden `TridentBackendConfig` CR.

Die Phase kann einen der folgenden Werte annehmen:

- **Bound:** Das `TridentBackendConfig` CR ist mit einem Backend verknüpft, und dieses Backend enthält `configRef` Auf einstellen `TridentBackendConfig` CR-UID.
- **Unbound:** Dargestellt mit `""`. Der `TridentBackendConfig` Objekt ist nicht an ein Backend gebunden. Neu erstellt `TridentBackendConfig` CRS befinden sich standardmäßig in dieser Phase. Wenn die Phase sich ändert, kann sie nicht wieder auf Unbound zurückgesetzt werden.
- **Deleting:** Das `TridentBackendConfig` CR `deletionPolicy` Wurde auf Löschen festgelegt. Wenn der `TridentBackendConfig` CR wird gelöscht und wechselt in den Löschzustand.

- Wenn im Backend keine PVCs (Persistent Volume Claims) vorhanden sind, löschen Sie den `TridentBackendConfig`. Wird dazu führen, dass Astra Trident das Backend sowie das löscht `TridentBackendConfig` CR.
- Wenn ein oder mehrere VES im Backend vorhanden sind, wechselt es in den Löschzustand. Der `TridentBackendConfig` Anschließend wechselt CR in die Löschphase. Das Backend und `TridentBackendConfig` Werden erst gelöscht, nachdem alle PVCs gelöscht wurden.
- **Lost:** Das Backend, das mit dem verbunden ist `TridentBackendConfig` CR wurde versehentlich oder absichtlich gelöscht und das `TridentBackendConfig` CR hat noch einen Verweis auf das gelöschte Backend. Der `TridentBackendConfig` CR kann weiterhin unabhängig vom gelöscht werden `deletionPolicy` Wert:
- **Unknown:** Astra Trident kann den Zustand oder die Existenz des mit dem verbundenen Backend nicht bestimmen `TridentBackendConfig` CR. Beispiel: Wenn der API-Server nicht antwortet oder wenn der `tridentbackends.trident.netapp.io` CRD fehlt. Dies kann Eingriffe erfordern.

In dieser Phase wird erfolgreich ein Backend erstellt! Es gibt mehrere Operationen, die zusätzlich gehandhabt werden können, wie z. B. ["Back-End-Updates und Löschungen am Back-End"](#).

(Optional) Schritt 4: Weitere Informationen

Sie können den folgenden Befehl ausführen, um weitere Informationen über Ihr Backend zu erhalten:

```
kubectl -n trident get tbc backend-tbc-ontap-san -o wide
```

NAME	BACKEND NAME	BACKEND UUID	
PHASE	STATUS	STORAGE DRIVER	DELETION POLICY
backend-tbc-ontap-san	ontap-san-backend	8d24fce7-6f60-4d4a-8ef6-	
bab2699e6ab8	Bound	Success	ontap-san delete

Zusätzlich können Sie auch einen YAML/JSON Dump von erhalten `TridentBackendConfig`.

```
kubectl -n trident get tbc backend-tbc-ontap-san -o yaml
```

```

apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  creationTimestamp: "2021-04-21T20:45:11Z"
  finalizers:
    - trident.netapp.io
  generation: 1
  name: backend-tbc-ontap-san
  namespace: trident
  resourceVersion: "947143"
  uid: 35b9d777-109f-43d5-8077-c74a4559d09c
spec:
  backendName: ontap-san-backend
  credentials:
    name: backend-tbc-ontap-san-secret
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  storageDriverName: ontap-san
  svm: trident_svm
  version: 1
status:
  backendInfo:
    backendName: ontap-san-backend
    backendUUID: 8d24fce7-6f60-4d4a-8ef6-bab2699e6ab8
  deletionPolicy: delete
  lastOperationStatus: Success
  message: Backend 'ontap-san-backend' created
  phase: Bound

```

backendInfo Enthält backendName Und das backendUUID Des Back-End, das als Antwort auf das erstellt wurde TridentBackendConfig CR. Der lastOperationStatus Feld gibt den Status des letzten Vorgangs des an TridentBackendConfig CR, der vom Benutzer ausgelöst werden kann (z. B. hat der Benutzer etwas in geändert spec) Oder ausgelöst durch Astra Trident (z. B. während Astra Trident Neustart). Er kann entweder erfolgreich oder fehlgeschlagen sein. phase Stellt den Status der Beziehung zwischen dem dar TridentBackendConfig CR und das Backend. Im obigen Beispiel phase Hat den Wert gebunden, was bedeutet, dass der TridentBackendConfig CR ist mit dem Backend verknüpft.

Sie können die ausführen `kubectl -n trident describe tbc <tbc-cr-name>` Befehl, um Details zu den Ereignisprotokollen zu erhalten.



Sie können ein Back-End, das einen zugeordneten enthält, nicht aktualisieren oder löschen TridentBackendConfig Objekt wird verwendet `tridentctl`. Um die Schritte zu verstehen, die mit dem Wechsel zwischen verbunden sind `tridentctl` Und TridentBackendConfig, "[Sehen Sie hier](#)".

Back-Ends managen

Führen Sie das Back-End-Management mit kubectl durch

Erfahren Sie, wie Sie mit Backend-Management-Operationen durchführen `kubectl`.

Löschen Sie ein Back-End

Durch Löschen von A `TridentBackendConfig`, Sie weisen Astra Trident an, Back-Ends zu löschen/zu behalten (basierend auf `deletionPolicy`). Um ein Backend zu löschen, stellen Sie sicher, dass `deletionPolicy` ist auf Löschen festgelegt. Um nur die zu löschen `TridentBackendConfig`, Stellen Sie das sicher `deletionPolicy` Auf beibehalten eingestellt. Dadurch wird sichergestellt, dass das Backend weiterhin vorhanden ist und mit verwaltet werden kann `tridentctl`.

Führen Sie den folgenden Befehl aus:

```
kubectl delete tbc <tbc-name> -n trident
```

Astra Trident löscht nicht die Kubernetes Secrets, die von verwendet wurden `TridentBackendConfig`. Der Kubernetes-Benutzer ist für die Bereinigung von Geheimnissen verantwortlich. Beim Löschen von Geheimnissen ist Vorsicht zu nehmen. Sie sollten Geheimnisse nur löschen, wenn sie nicht von den Back-Ends verwendet werden.

Zeigen Sie die vorhandenen Back-Ends an

Führen Sie den folgenden Befehl aus:

```
kubectl get tbc -n trident
```

Sie können auch ausführen `tridentctl get backend -n trident` Oder `tridentctl get backend -o yaml -n trident` Um eine Liste aller vorhandenen Back-Ends zu erhalten. Diese Liste umfasst auch Back-Ends, die mit erstellt wurden `tridentctl`.

Aktualisieren Sie ein Backend

Es gibt mehrere Gründe für die Aktualisierung eines Backend:

- Die Anmeldeinformationen für das Speichersystem wurden geändert. Um Anmeldedaten zu aktualisieren, wird das in verwendete Kubernetes Secret verwendet `TridentBackendConfig` Objekt muss aktualisiert werden. Astra Trident aktualisiert automatisch das Backend mit den neuesten Zugangsdaten. Führen Sie den folgenden Befehl aus, um den Kubernetes Secret zu aktualisieren:

```
kubectl apply -f <updated-secret-file.yaml> -n trident
```

- Parameter (wie der Name der verwendeten ONTAP-SVM) müssen aktualisiert werden.
 - Sie können aktualisieren `TridentBackendConfig` Objekte können direkt über Kubernetes mit dem folgenden Befehl abgerufen werden:

```
kubectl apply -f <updated-backend-file.yaml>
```

- Alternativ können Sie Änderungen an der vorhandenen vornehmen `TridentBackendConfig` CR mit folgendem Befehl:

```
kubectl edit tbc <tbc-name> -n trident
```



- Wenn ein Backend-Update fehlschlägt, bleibt das Backend in seiner letzten bekannten Konfiguration erhalten. Sie können die Protokolle anzeigen, um die Ursache durch Ausführen zu bestimmen `kubectl get tbc <tbc-name> -o yaml -n trident` Oder `kubectl describe tbc <tbc-name> -n trident`.
- Nachdem Sie das Problem mit der Konfigurationsdatei erkannt und behoben haben, können Sie den Befehl `Update` erneut ausführen.

Back-End-Management mit `tridentctl`

Erfahren Sie, wie Sie mit Backend-Management-Operationen durchführen `tridentctl`.

Erstellen Sie ein Backend

Nachdem Sie ein erstellt haben "[Back-End-Konfigurationsdatei](#)", Ausführen des folgenden Befehls:

```
tridentctl create backend -f <backend-file> -n trident
```

Wenn die Back-End-Erstellung fehlschlägt, ist mit der Back-End-Konfiguration ein Fehler aufgetreten. Sie können die Protokolle zur Bestimmung der Ursache anzeigen, indem Sie den folgenden Befehl ausführen:

```
tridentctl logs -n trident
```

Nachdem Sie das Problem mit der Konfigurationsdatei identifiziert und behoben haben, können Sie einfach die ausführen `create` Befehl erneut.

Löschen Sie ein Back-End

Gehen Sie wie folgt vor, um ein Backend von Astra Trident zu löschen:

1. Abrufen des Back-End-Namens:

```
tridentctl get backend -n trident
```

2. Back-End löschen:

```
tridentctl delete backend <backend-name> -n trident
```



Wenn Astra Trident Volumes und Snapshots aus diesem Backend bereitgestellt hat, die immer noch vorhanden sind, verhindert das Löschen des Backend, dass neue Volumes bereitgestellt werden. Das Backend wird weiterhin in einem „Deleting“ Zustand vorhanden sein und Trident wird weiterhin diese Volumes und Snapshots verwalten, bis sie gelöscht werden.

Zeigen Sie die vorhandenen Back-Ends an

Gehen Sie zum Anzeigen der von Trident verwendeten Back-Ends wie folgt vor:

- Führen Sie den folgenden Befehl aus, um eine Zusammenfassung anzuzeigen:

```
tridentctl get backend -n trident
```

- Um alle Details anzuzeigen, führen Sie den folgenden Befehl aus:

```
tridentctl get backend -o json -n trident
```

Aktualisieren Sie ein Backend

Führen Sie nach dem Erstellen einer neuen Backend-Konfigurationsdatei den folgenden Befehl aus:

```
tridentctl update backend <backend-name> -f <backend-file> -n trident
```

Wenn das Backend-Update fehlschlägt, ist bei der Backend-Konfiguration ein Fehler aufgetreten oder Sie haben ein ungültiges Update versucht. Sie können die Protokolle zur Bestimmung der Ursache anzeigen, indem Sie den folgenden Befehl ausführen:

```
tridentctl logs -n trident
```

Nachdem Sie das Problem mit der Konfigurationsdatei identifiziert und behoben haben, können Sie einfach die ausführen `update` Befehl erneut.

Identifizieren Sie die Storage-Klassen, die ein Backend nutzen

Dies ist ein Beispiel für die Art von Fragen, die Sie mit der JSON beantworten können `tridentctl` Ausgänge für Backend-Objekte. Dazu wird der verwendet `jq` Dienstprogramm, das Sie installieren müssen.

```
tridentctl get backend -o json | jq '[.items[] | {backend: .name, storageClasses: [.storage[].storageClasses]|unique}]'
```

Dies gilt auch für Back-Ends, die mit erstellt wurden `TridentBackendConfig`.

Wechseln Sie zwischen den Back-End-Managementoptionen

Erfahren Sie in Astra Trident, wie Back-Ends auf verschiedene Art und Weise gemanagt werden.

Optionen für das Management von Back-Ends

Mit der Einführung von `TridentBackendConfig`, Administratoren haben jetzt zwei unterschiedliche Arten von Back-Ends zu verwalten. Dies stellt die folgenden Fragen:

- Mit können Back-Ends erstellt werden `tridentctl` Gemanagt werden mit `TridentBackendConfig`?
- Mit können Back-Ends erstellt werden `TridentBackendConfig` Gemanagt werden mit `tridentctl`?

Managen `tridentctl` Back-Ends mit `TridentBackendConfig`

In diesem Abschnitt werden die Schritte aufgeführt, die für das Management von Back-Ends erforderlich sind, die mit erstellt wurden `tridentctl` Erstellen Sie direkt über die Kubernetes Schnittstelle `TridentBackendConfig` Objekte:

Dies gilt für die folgenden Szenarien:

- Bereits vorhandene Back-Ends, die keine haben `TridentBackendConfig` Weil sie mit erstellt wurden `tridentctl`.
- Neue Back-Ends, mit denen erstellt wurden `tridentctl`, Während andere `TridentBackendConfig` Objekte sind vorhanden.

In beiden Szenarien werden Back-Ends weiterhin vorhanden sein, wobei Astra Trident Volumes terminieren und darauf arbeiten wird. Administratoren können hier eine von zwei Möglichkeiten wählen:

- Fahren Sie mit der Verwendung fort `tridentctl` Um Back-Ends zu managen, die mit ihr erstellt wurden.
- Back-Ends werden mit erstellt `tridentctl` Zu einer neuen `TridentBackendConfig` Objekt: Dies würde bedeuten, dass die Back-Ends mit gemanagt werden `kubectl` Und nicht `tridentctl`.

Um ein bereits vorhandenes Backend mit zu verwalten `kubectl`, Sie müssen ein erstellen `TridentBackendConfig` Das bindet an das vorhandene Backend. Hier eine Übersicht über die Funktionsweise:

1. Kubernetes Secret erstellen: Das Geheimnis enthält die Zugangsdaten, die Astra Trident zur Kommunikation mit dem Storage-Cluster/Service benötigt.
2. Erstellen Sie ein `TridentBackendConfig` Objekt: Dies enthält Angaben zum Storage-Cluster/Service und verweist auf das im vorherigen Schritt erstellte Geheimnis. Es muss sorgfältig darauf achten, identische Konfigurationsparameter festzulegen (z. B. `spec.backendName`, `spec.storagePrefix`, `spec.storageDriverName`, Und so weiter). `spec.backendName` Muss auf den Namen des vorhandenen Backend eingestellt werden.

Schritt 0: Identifizieren Sie das Backend

Um ein zu erstellen `TridentBackendConfig` Die an ein vorhandenes Backend bindet, müssen Sie die Backend-Konfiguration abrufen. In diesem Beispiel nehmen wir an, dass ein Backend mithilfe der folgenden

JSON-Definition erstellt wurde:

```
tridentctl get backend ontap-nas-backend -n trident
```

```
+-----+-----+
+-----+-----+-----+-----+
|          NAME          | STORAGE DRIVER |          UUID          |
| STATE | VOLUMES |                |
+-----+-----+
+-----+-----+-----+-----+
| ontap-nas-backend      | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |          25 |
+-----+-----+
+-----+-----+-----+-----+
```

```
cat ontap-nas-backend.json
```

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.10.10.1",
  "dataLIF": "10.10.10.2",
  "backendName": "ontap-nas-backend",
  "svm": "trident_svm",
  "username": "cluster-admin",
  "password": "admin-password",

  "defaults": {
    "spaceReserve": "none",
    "encryption": "false"
  },
  "labels": {"store": "nas_store"},
  "region": "us_east_1",
  "storage": [
    {
      "labels": {"app": "msoffice", "cost": "100"},
      "zone": "us_east_1a",
      "defaults": {
        "spaceReserve": "volume",
        "encryption": "true",
        "unixPermissions": "0755"
      }
    },
    {
      "labels": {"app": "mysqldb", "cost": "25"},
      "zone": "us_east_1d",
      "defaults": {
```

```

        "spaceReserve": "volume",
        "encryption": "false",
        "unixPermissions": "0775"
    }
}
]
}

```

Schritt: Ein Kubernetes Secret erstellen

Erstellen Sie einen geheimen Schlüssel, der die Anmeldeinformationen für das Backend enthält, wie in diesem Beispiel gezeigt:

```

cat tbc-ontap-nas-backend-secret.yaml

apiVersion: v1
kind: Secret
metadata:
  name: ontap-nas-backend-secret
type: Opaque
stringData:
  username: cluster-admin
  password: admin-password

kubectl create -f tbc-ontap-nas-backend-secret.yaml -n trident
secret/backend-tbc-ontap-san-secret created

```

Schritt 2: Erstellen Sie ein TridentBackendConfig CR

Im nächsten Schritt wird ein erstellt TridentBackendConfig CR, das automatisch an die bereits vorhandene bindet ontap-nas-backend (Wie in diesem Beispiel). Stellen Sie sicher, dass folgende Anforderungen erfüllt sind:

- Der gleiche Backend-Name wird in definiert spec.backendName.
- Die Konfigurationsparameter sind mit dem ursprünglichen Back-End identisch.
- Virtuelle Pools (falls vorhanden) müssen dieselbe Reihenfolge wie im ursprünglichen Backend beibehalten.
- Anmeldedaten werden bei einem Kubernetes Secret und nicht im Klartext bereitgestellt.

In diesem Fall die TridentBackendConfig Wird so aussehen:

```

cat backend-tbc-ontap-nas.yaml
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-ontap-nas-backend
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.10.10.1
  dataLIF: 10.10.10.2
  backendName: ontap-nas-backend
  svm: trident_svm
  credentials:
    name: mysecret
  defaults:
    spaceReserve: none
    encryption: 'false'
  labels:
    store: nas_store
  region: us_east_1
  storage:
  - labels:
      app: msoffice
      cost: '100'
      zone: us_east_1a
      defaults:
        spaceReserve: volume
        encryption: 'true'
        unixPermissions: '0755'
  - labels:
      app: mysqldb
      cost: '25'
      zone: us_east_1d
      defaults:
        spaceReserve: volume
        encryption: 'false'
        unixPermissions: '0775'

kubectl create -f backend-tbc-ontap-nas.yaml -n trident
tridentbackendconfig.trident.netapp.io/tbc-ontap-nas-backend created

```

Schritt 3: Überprüfen Sie den Status des TridentBackendConfig **CR**

Nach dem TridentBackendConfig Wurde erstellt, seine Phase muss sein Bound. Sie sollte außerdem den gleichen Backend-Namen und die gleiche UUID wie das vorhandene Backend widerspiegeln.

```
kubectl get tbc tbc-ontap-nas-backend -n trident
```

NAME	BACKEND NAME	BACKEND UUID
tbc-ontap-nas-backend	ontap-nas-backend	52f2eb10-e4c6-4160-99fc-96b3be5ab5d7

```
tridentctl get backend -n trident
```

NAME	STATE	VOLUMES	STORAGE DRIVER	UUID
ontap-nas-backend	online	25	ontap-nas	52f2eb10-e4c6-4160-99fc-96b3be5ab5d7

Das Backend wird nun vollständig mit dem verwaltet tbc-ontap-nas-backend TridentBackendConfig Objekt:

Managen TridentBackendConfig **Back-Ends** mit tridentctl

`tridentctl` Kann zur Auflistung von Back-Ends verwendet werden, die mit erstellt wurden `TridentBackendConfig`. Darüber hinaus können Administratoren solche Back-Ends mithilfe von auch vollständig managen `tridentctl` Durch Löschen `TridentBackendConfig` Mit Sicherheit `spec.deletionPolicy` Ist auf festgelegt `retain`.

Schritt 0: Identifizieren Sie das Backend

Nehmen wir beispielsweise an, dass das folgende Backend mit erstellt wurde TridentBackendConfig:

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
```

NAME	BACKEND NAME	BACKEND UUID
backend-tbc-ontap-san	ontap-san-backend	81abcb27-ea63-49bb-b606-0a5315ac5f82

```
tridentctl get backend ontap-san-backend -n trident
```

NAME	STORAGE DRIVER	UUID
ontap-san-backend	ontap-san	81abcb27-ea63-49bb-b606-0a5315ac5f82

Von der Ausgabe, ist es gesehen, dass TridentBackendConfig Wurde erfolgreich erstellt und ist an ein Backend gebunden [UUID des Backends beobachten].

Schritt 1: Bestätigen deletionPolicy Ist auf festgelegt retain

Lassen Sie uns den Wert von betrachten deletionPolicy. Dies muss eingestellt werden retain. Dadurch wird sichergestellt, dass, wenn ein TridentBackendConfig CR wird gelöscht, die Backend-Definition ist weiterhin vorhanden und kann mit verwaltet werden tridentctl.

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
```

NAME	BACKEND NAME	BACKEND UUID
backend-tbc-ontap-san	ontap-san-backend	81abcb27-ea63-49bb-b606-0a5315ac5f82

```
# Patch value of deletionPolicy to retain
kubectl patch tbc backend-tbc-ontap-san --type=merge -p
'{"spec":{"deletionPolicy":"retain"}}' -n trident
tridentbackendconfig.trident.netapp.io/backend-tbc-ontap-san patched

#Confirm the value of deletionPolicy
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
```

NAME	BACKEND NAME	BACKEND UUID
backend-tbc-ontap-san	ontap-san-backend	81abcb27-ea63-49bb-b606-0a5315ac5f82



Fahren Sie nur mit dem nächsten Schritt fort `deletionPolicy` ist auf festgelegt `retain`.

Schritt 2: Löschen Sie den `TridentBackendConfig` CR

Der letzte Schritt besteht darin, den zu löschen `TridentBackendConfig` CR. Nach Bestätigung des `deletionPolicy` ist auf festgelegt `retain`, Sie können mit der Löschung fortfahren:

```
kubectl delete tbc backend-tbc-ontap-san -n trident
tridentbackendconfig.trident.netapp.io "backend-tbc-ontap-san" deleted

tridentctl get backend ontap-san-backend -n trident
+-----+-----+
+-----+-----+-----+-----+
|          NAME          | STORAGE DRIVER |                               UUID                               |
| STATE | VOLUMES |                               |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
| ontap-san-backend | ontap-san      | 81abcb27-ea63-49bb-b606-0a5315ac5f82 |
| online |      33 |                               |
+-----+-----+-----+-----+
```

Bei der Löschung der `TridentBackendConfig` Object, Astra Trident entfernt es einfach, ohne das Backend zu löschen.

Erstellen und Managen von Storage-Klassen

Erstellen Sie eine Speicherklasse

Konfigurieren Sie ein Kubernetes `StorageClass`-Objekt und erstellen Sie die Storage-Klasse, um Astra Trident über die Bereitstellung von Volumes zu informieren.

Konfigurieren Sie ein Kubernetes `StorageClass`-Objekt

Der "[Kubernetes StorageClass-Objekt](#)" Astra Trident wird als bereitstellung für diese Klasse identifiziert. Astra Trident weist Anweisungen zur Provisionierung eines Volumes aus. Beispiel:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: <Name>
provisioner: csi.trident.netapp.io
mountOptions: <Mount Options>
parameters:
  <Trident Parameters>
allowVolumeExpansion: true
volumeBindingMode: Immediate
```

Siehe "[Kubernetes und Trident Objekte](#)" Erfahren Sie, wie Storage-Klassen mit dem interagieren PersistentVolumeClaim Und Parameter für die Steuerung, wie Astra Trident Volumes provisioniert.

Erstellen Sie eine Speicherklasse

Nachdem Sie das StorageClass-Objekt erstellt haben, können Sie die Storage-Klasse erstellen. [Proben der Lagerklasse](#) Enthält einige grundlegende Proben, die Sie verwenden oder ändern können.

Schritte

1. Verwenden Sie dieses Objekt von Kubernetes `kubectl` Um sie in Kubernetes zu erstellen.

```
kubectl create -f sample-input/storage-class-basic-csi.yaml
```

2. Sie sollten jetzt in Kubernetes und Astra Trident eine **Basis-csi** Storage-Klasse sehen, und Astra Trident hätte die Pools auf dem Backend entdeckt haben sollen.

```

kubectl get sc basic-csi
NAME          PROVISIONER          AGE
basic-csi     csi.trident.netapp.io 15h

./tridentctl -n trident get storageclass basic-csi -o json
{
  "items": [
    {
      "Config": {
        "version": "1",
        "name": "basic-csi",
        "attributes": {
          "backendType": "ontap-nas"
        },
        "storagePools": null,
        "additionalStoragePools": null
      },
      "storage": {
        "ontapnas_10.0.0.1": [
          "aggr1",
          "aggr2",
          "aggr3",
          "aggr4"
        ]
      }
    }
  ]
}

```

Proben der Lagerklasse

Astra Trident bietet das ["Einfache Definitionen von Storage-Klassen für spezifische Back-Ends"](#).

Alternativ können Sie bearbeiten `sample-input/storage-class-csi.yaml.templ` Datei, die im Lieferumfang des Installationsprogramms enthalten ist und ersetzt wird `BACKEND_TYPE` Mit dem Namen des Speichertreibers.


```
./tridentctl -n trident get backend
+-----+-----+-----+-----+
+-----+-----+
|      NAME      | STORAGE DRIVER |          UUID          |
STATE | VOLUMES |
+-----+-----+-----+-----+
+-----+-----+
| nas-backend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |         0 |
+-----+-----+-----+-----+
+-----+-----+

cp sample-input/storage-class-csi.yaml.templ sample-input/storage-class-
basic-csi.yaml

# Modify __BACKEND_TYPE__ with the storage driver field above (e.g.,
ontap-nas)
vi sample-input/storage-class-basic-csi.yaml
```

Management von Storage-Klassen

Sie können vorhandene Storage-Klassen anzeigen, eine Standard-Storage-Klasse festlegen, das Back-End der Speicherklasse identifizieren und Speicherklassen löschen.

Sehen Sie sich die vorhandenen Speicherklassen an

- Um vorhandene Kubernetes-Storage-Klassen anzuzeigen, führen Sie den folgenden Befehl aus:

```
kubectl get storageclass
```

- Um die Details der Kubernetes-Storage-Klasse anzuzeigen, führen Sie den folgenden Befehl aus:

```
kubectl get storageclass <storage-class> -o json
```

- Führen Sie den folgenden Befehl aus, um die synchronisierten Storage-Klassen von Astra Trident anzuzeigen:

```
tridentctl get storageclass
```

- Um die synchronisierten Storage-Klassendetails von Astra Trident anzuzeigen, führen Sie den folgenden Befehl aus:

```
tridentctl get storageclass <storage-class> -o json
```

Legen Sie eine Standardspeicherklasse fest

Mit Kubernetes 1.6 können Sie eine Standard-Storage-Klasse festlegen. Dies ist die Storage-Klasse, die zur Bereitstellung eines Persistent Volume verwendet wird, wenn ein Benutzer in einer Persistent Volume Claim (PVC) nicht eine Angabe vorgibt.

- Definieren Sie eine Standard-Storage-Klasse, indem Sie die Anmerkung festlegen `storageclass.kubernetes.io/is-default-class` In der Definition der Storage-Klassen wie den „true“. Gemäß der Spezifikation wird jeder andere Wert oder jede Abwesenheit der Anmerkung als falsch interpretiert.
- Sie können eine vorhandene Storage-Klasse als Standard-Storage-Klasse konfigurieren, indem Sie den folgenden Befehl verwenden:

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

- In ähnlicher Weise können Sie die standardmäßige Storage-Klassenbeschriftung mithilfe des folgenden Befehls entfernen:

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

Es gibt auch Beispiele im Trident Installationspaket, die diese Annotation enthält.



Ihr Cluster sollte immer nur eine Standard-Storage-Klasse aufweisen. Kubernetes verhindert technisch nicht, dass Sie mehr als eine haben, aber es verhält sich so, als ob es überhaupt keine Standard-Storage-Klasse gibt.

Das Backend für eine Storage-Klasse ermitteln

Dies ist ein Beispiel für die Art von Fragen, die Sie mit der JSON beantworten können `tridentctl` Ausgänge für Astra Trident Backend-Objekte. Dazu wird der verwendet `jq` Dienstprogramm, das Sie möglicherweise zuerst installieren müssen.

```
tridentctl get storageclass -o json | jq '[.items[] | {storageClass:  
.Config.name, backends: [.storage]|unique}]'
```

Löschen Sie eine Speicherklasse

Führen Sie den folgenden Befehl aus, um eine Storage-Klasse aus Kubernetes zu löschen:

```
kubectl delete storageclass <storage-class>
```

<storage-class> Sollten durch Ihre Storage-Klasse ersetzt werden.

Alle persistenten Volumes, die durch diese Storage-Klasse erstellt wurden, werden unverändert beibehalten und Astra Trident wird sie weiterhin managen.



Astra Trident setzt ein Leereinschub um `fsType` Für die von ihm erstellten Volumes. Bei iSCSI-Back-Ends wird die Durchsetzung empfohlen `parameters.fsType` In der `StorageClass`. Sie sollten vorhandene `StorageClasses` löschen und mit neu erstellen `parameters.fsType` Angegeben.

Provisionierung und Management von Volumes

Bereitstellen eines Volumes

Erstellen Sie ein `PersistentVolume` (PV) und ein `PersistentVolumeClaim` (PVC), das die konfigurierte Kubernetes `StorageClass` verwendet, um Zugriff auf das PV anzufordern. Anschließend können Sie das PV an einem Pod montieren.

Überblick

A "[*PersistentVolume*](#)" (PV) ist eine physische Speicherressource, die vom Clusteradministrator auf einem Kubernetes-Cluster bereitgestellt wird. Der "[*PersistentVolumeClaim*](#)" (PVC) ist eine Anforderung für den Zugriff auf das `PersistentVolume` auf dem Cluster.

Die PVC kann so konfiguriert werden, dass eine Speicherung einer bestimmten Größe oder eines bestimmten Zugriffsmodus angefordert wird. Mithilfe der zugehörigen `StorageClass` kann der Clusteradministrator mehr als die Größe des `PersistentVolume` und den Zugriffsmodus steuern, z. B. die Performance oder das Service-Level.

Nachdem Sie das PV und die PVC erstellt haben, können Sie das Volume in einem Pod einbinden.

Beispielmanifeste

PersistentVolume-Beispielmanifest

Dieses Beispielmanifest zeigt ein Basis-PV von 10Gi, das mit StorageClass verknüpft ist `basic-csi`.

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-storage
  labels:
    type: local
spec:
  storageClassName: basic-csi
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/my/host/path"
```

PersistentVolumeClaim-Beispielmanifeste

Diese Beispiele zeigen grundlegende PVC-Konfigurationsoptionen.

PVC mit RWO-Zugang

Dieses Beispiel zeigt eine grundlegende PVC mit RWO-Zugriff, die einer StorageClass mit dem Namen zugeordnet ist `basic-csi`.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-storage
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: basic-csi
```

PVC mit NVMe/TCP

Dieses Beispiel zeigt eine grundlegende PVC für NVMe/TCP mit RWO-Zugriff, die einer StorageClass mit dem Namen zugeordnet ist `protection-gold`.

```
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-san-nvme
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 300Mi
  storageClassName: protection-gold
```

Pod-Manifest-Proben

Diese Beispiele zeigen grundlegende Konfigurationen zum Anschließen der PVC an einen Pod.

Basiskonfiguration

```
kind: Pod
apiVersion: v1
metadata:
  name: pv-pod
spec:
  volumes:
    - name: pv-storage
      persistentVolumeClaim:
        claimName: basic
  containers:
    - name: pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: pv-storage
```

Grundlegende NVMe/TCP-Konfiguration

```
---
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx
  name: nginx
spec:
  containers:
    - image: nginx
      name: nginx
      resources: {}
      volumeMounts:
        - mountPath: "/usr/share/nginx/html"
          name: task-pv-storage
  dnsPolicy: ClusterFirst
  restartPolicy: Always
  volumes:
    - name: task-pv-storage
      persistentVolumeClaim:
        claimName: pvc-san-nvme
```

Erstellen Sie das PV und die PVC

Schritte

1. Erstellen Sie das PV.

```
kubectl create -f pv.yaml
```

2. Überprüfen Sie den PV-Status.

```
kubectl get pv
NAME          CAPACITY  ACCESS MODES  RECLAIM POLICY  STATUS  CLAIM
STORAGECLASS  REASON    AGE
pv-storage    4Gi       RWO           Retain          Available
7s
```

3. Erstellen Sie die PVC.

```
kubectl create -f pvc.yaml
```

4. Überprüfen Sie den PVC-Status.

```
kubectl get pvc
```

NAME	STATUS	VOLUME	CAPACITY	ACCESS MODES	STORAGECLASS	AGE
pvc-storage	Bound	pv-name	2Gi	RWO		5m

5. Mounten Sie das Volume in einem Pod.

```
kubectl create -f pv-pod.yaml
```



Sie können den Fortschritt mit überwachen `kubectl get pod --watch`.

6. Vergewissern Sie sich, dass das Volume auf gemountet ist `/my/mount/path`.

```
kubectl exec -it task-pv-pod -- df -h /my/mount/path
```

7. Sie können den Pod jetzt löschen. Die Pod Applikation wird nicht mehr existieren, aber das Volume bleibt erhalten.

```
kubectl delete pod task-pv-pod
```

Siehe "[Kubernetes und Trident Objekte](#)" Erfahren Sie, wie Storage-Klassen mit dem interagieren `PersistentVolumeClaim` Und Parameter für die Steuerung, wie Astra Trident Volumes provisioniert.

Erweitern Sie Volumes

Astra Trident bietet Kubernetes-Benutzern die Möglichkeit, ihre Volumes nach Erstellung zu erweitern. Hier finden Sie Informationen zu den erforderlichen Konfigurationen zum erweitern von iSCSI- und NFS-Volumes.

Erweitern Sie ein iSCSI-Volume

Sie können ein iSCSI Persistent Volume (PV) mithilfe der CSI-provisionierung erweitern.



Die Erweiterung des iSCSI-Volumes wird von unterstützt `ontap-san`, `ontap-san-economy`, `solidfire-san` Treiber und erfordert Kubernetes 1.16 und höher.

Schritt: Storage Class für Volume-Erweiterung konfigurieren

Bearbeiten Sie die StorageClass-Definition, um die festzulegen `allowVolumeExpansion` Feld an `true`.


```
cat storageclass-ontapsan.yaml
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-san
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
allowVolumeExpansion: True
```

Bearbeiten Sie für eine bereits vorhandene StorageClass, um die einzuschließen `allowVolumeExpansion` Parameter.

Schritt 2: Erstellen Sie ein PVC mit der von Ihnen erstellten StorageClass

Bearbeiten Sie die PVC-Definition, und aktualisieren Sie die `spec.resources.requests.storage` Um die neu gewünschte Größe zu reflektieren, die größer als die ursprüngliche Größe sein muss.

```
cat pvc-ontapsan.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: san-pvc
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-san
```

Astra Trident erstellt ein persistentes Volume (PV) und verknüpft es mit dieser Persistent Volume Claim (PVC).

```
kubectl get pvc
```

NAME	STATUS	VOLUME	CAPACITY
san-pvc	Bound	pvc-8a814d62-bd58-4253-b0d1-82f2885db671	1Gi

```
kubectl get pv
```

NAME	RECLAIM POLICY	STATUS	CLAIM	CAPACITY	STORAGECLASS	ACCESS MODES	REASON	AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671	Delete	Bound	default/san-pvc	1Gi	ontap-san	RWO		10s

Schritt 3: Definieren Sie einen Behälter, der das PVC befestigt

Schließen Sie das PV an einen Pod an, um die Größe zu ändern. Beim Ändern der Größe eines iSCSI-PV gibt es zwei Szenarien:

- Wenn das PV an einen POD angeschlossen ist, erweitert Astra Trident das Volume auf dem Storage-Back-End, setzt das Gerät neu ein und vergrößert das Dateisystem neu.
- Bei dem Versuch, die Größe eines nicht angeschlossenen PV zu ändern, erweitert Astra Trident das Volume auf dem Storage-Backend. Nachdem die PVC an einen Pod gebunden ist, lässt Trident das Gerät neu in die Größe des Dateisystems einarbeiten. Kubernetes aktualisiert dann die PVC-Größe, nachdem der Expand-Vorgang erfolgreich abgeschlossen ist.

In diesem Beispiel wird ein POD erstellt, der die verwendet `san-pvc`.

```

kubect1 get pod
NAME          READY   STATUS    RESTARTS   AGE
ubuntu-pod    1/1     Running   0           65s

kubect1 describe pvc san-pvc
Name:          san-pvc
Namespace:     default
StorageClass:  ontap-san
Status:        Bound
Volume:        pvc-8a814d62-bd58-4253-b0d1-82f2885db671
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
               volume.beta.kubernetes.io/storage-provisioner:
               csi.trident.netapp.io
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      1Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Mounted By:    ubuntu-pod

```

Schritt 4: Erweitern Sie das PV

Um die Größe des PV zu ändern, das von 1Gi auf 2Gi erstellt wurde, bearbeiten Sie die PVC-Definition und aktualisieren Sie die `spec.resources.requests.storage` Bis 2Gi.

```

kubectl edit pvc san-pvc
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: "2019-10-10T17:32:29Z"
  finalizers:
    - kubernetes.io/pvc-protection
  name: san-pvc
  namespace: default
  resourceVersion: "16609"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/san-pvc
  uid: 8a814d62-bd58-4253-b0d1-82f2885db671
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
  ...

```

Schritt 5: Validieren Sie die Erweiterung

Sie können die korrekte Ausführung der Erweiterung überprüfen, indem Sie die Größe der PVC, PV und des Astra Trident Volume überprüfen:

```
kubectl get pvc san-pvc
NAME          STATUS    VOLUME                                     CAPACITY
ACCESS MODES  STORAGECLASS  AGE
san-pvc      Bound       pvc-8a814d62-bd58-4253-b0d1-82f2885db671  2Gi
RWO           ontap-san    11m

kubectl get pv
NAME          CAPACITY  ACCESS MODES
RECLAIM POLICY STATUS    CLAIM          STORAGECLASS  REASON    AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671  2Gi        RWO
Delete              Bound       default/san-pvc  ontap-san    12m

tridentctl get volumes -n trident
+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
|          NAME          | SIZE | STORAGE CLASS |
+-----+-----+-----+-----+-----+-----+
|          BACKEND UUID  | STATE | MANAGED |
+-----+-----+-----+-----+-----+-----+
| pvc-8a814d62-bd58-4253-b0d1-82f2885db671 | 2.0 GiB | ontap-san |
| block      | a9b7bfff-0505-4e31-b6c5-59f492e02d33 | online | true      |
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
```

Erweitern Sie ein NFS-Volume

Astra Trident unterstützt die Volume-Erweiterung für auf bereitgestellte NFS PVS `ontap-nas`, `ontap-nas-economy`, `ontap-nas-flexgroup`, `gcp-cvs`, und `azure-netapp-files` Back-Ends:

Schritt: Storage Class für Volume-Erweiterung konfigurieren

Um die Größe eines NFS PV zu ändern, muss der Administrator zunächst die Storage-Klasse konfigurieren, um die Volume-Erweiterung durch Einstellen der zu ermöglichen `allowVolumeExpansion` Feld an `true`:

```
cat storageclass-ontapnas.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontapnas
provisioner: csi.trident.netapp.io
parameters:
  backendType: ontap-nas
allowVolumeExpansion: true
```

Wenn Sie bereits eine Storage-Klasse ohne diese Option erstellt haben, können Sie die vorhandene Storage-Klasse einfach mit `kubectl edit storageclass` bearbeiten, um eine Volume-Erweiterung zu ermöglichen.

Schritt 2: Erstellen Sie ein PVC mit der von Ihnen erstellten StorageClass

```
cat pvc-ontapnas.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: ontapnas20mb
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 20Mi
  storageClassName: ontapnas
```

Astra Trident sollte ein 20MiB NFS PV für diese PVC erstellen:

```
kubectl get pvc
NAME                STATUS    VOLUME
CAPACITY            ACCESS MODES  STORAGECLASS  AGE
ontapnas20mb        Bound      pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7  20Mi
RWO                  ontapnas      9s

kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                CAPACITY  ACCESS MODES
RECLAIM POLICY      STATUS    CLAIM                STORAGECLASS  REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7  20Mi      RWO
Delete              Bound     default/ontapnas20mb  ontapnas
2m42s
```

Schritt 3: Erweitern Sie das PV

Um die Größe des neu erstellten 20MiB PV auf 1 gib zu ändern, bearbeiten Sie die PVC und den Satz `spec.resources.requests.storage` Bis 1 gib:

```

kubectl edit pvc ontapnas20mb
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: 2018-08-21T18:26:44Z
  finalizers:
  - kubernetes.io/pvc-protection
  name: ontapnas20mb
  namespace: default
  resourceVersion: "1958015"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/ontapnas20mb
  uid: c1bd7fa5-a56f-11e8-b8d7-fa163e59eaab
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  ...

```

Schritt 4: Validierung der Erweiterung

Sie können die korrekte Größenänderung validieren, indem Sie die Größe des PVC, des PV und des Astra Trident Volume überprüfen:

```
kubectl get pvc ontapnas20mb
```

NAME	STATUS	VOLUME
ontapnas20mb	Bound	pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
RWO	ontapnas	4m44s


```
kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
```

NAME	CAPACITY	ACCESS MODES
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7	1Gi	RWO
Delete	Bound	default/ontapnas20mb
5m35s		ontapnas


```
tridentctl get volume pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 -n trident
```

NAME	SIZE	STORAGE CLASS
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7	1.0 GiB	ontapnas
file	c5a6f6a4-b052-423b-80d4-8fb491a14a22	online
		true

Volumes importieren

Sie können vorhandene Storage Volumes mit als Kubernetes PV importieren
tridentctl import.

Überblick und Überlegungen

Ein Volume kann in Astra Trident importiert werden, um:

- Containerisierung einer Applikation und Wiederverwendung des vorhandenen Datensatzes
- Verwenden Sie einen Klon eines Datensatzes für eine kurzlebige Applikation
- Wiederherstellung eines fehlerhaften Kubernetes-Clusters
- Migration von Applikationsdaten bei der Disaster Recovery

Überlegungen

Lesen Sie vor dem Importieren eines Volumes die folgenden Überlegungen durch.

- Astra Trident kann nur ONTAP Volumes vom Typ RW (Lese-/Schreibzugriff) importieren. Volumes im DP-Typ (Datensicherung) sind SnapMirror Ziel-Volumes. Sie sollten die Spiegelungsbeziehung unterbrechen, bevor Sie das Volume in Astra Trident importieren.

- Wir empfehlen, Volumes ohne aktive Verbindungen zu importieren. Um ein aktiv verwendetes Volume zu importieren, klonen Sie das Volume, und führen Sie dann den Import durch.



Dies ist besonders für Block-Volumes wichtig, da Kubernetes die vorherige Verbindung nicht mitbekommt und problemlos ein aktives Volume an einen Pod anbinden kann. Dies kann zu Datenbeschädigungen führen.

- Aber `StorageClass` Muss auf einer PVC angegeben werden, Astra Trident verwendet diesen Parameter während des Imports nicht. Während der Volume-Erstellung werden Storage-Klassen eingesetzt, um basierend auf den Storage-Merkmalen aus verfügbaren Pools auszuwählen. Da das Volume bereits vorhanden ist, ist beim Import keine Poolauswahl erforderlich. Daher schlägt der Import auch dann nicht fehl, wenn das Volume auf einem Back-End oder Pool vorhanden ist, das nicht mit der in der PVC angegebenen Speicherklasse übereinstimmt.
- Die vorhandene Volumengröße wird in der PVC ermittelt und festgelegt. Nachdem das Volumen vom Speichertreiber importiert wurde, wird das PV mit einem `ClaimRef` an die PVC erzeugt.
 - Die Rückgewinnungsrichtlinie ist zunächst auf festgelegt `retain` Im PV. Nachdem Kubernetes die PVC und das PV erfolgreich bindet, wird die Zurückgewinnungsrichtlinie aktualisiert und an die Zurückgewinnungsrichtlinie der Storage-Klasse angepasst.
 - Wenn die Richtlinie zur Zurückgewinnung der Storage-Klasse lautet `delete`, Das Speichervolumen wird gelöscht, wenn das PV gelöscht wird.
- Astra Trident verwaltet standardmäßig die PVC und benennt die FlexVol und die LUN auf dem Backend um. Sie können die passieren `--no-manage` Flag zum Importieren eines nicht verwalteten Volumes. Wenn Sie verwenden `--no-manage`, Astra Trident führt keine zusätzlichen Operationen auf der PVC oder PV für den Lebenszyklus der Objekte. Das Speicher-Volume wird nicht gelöscht, wenn das PV gelöscht wird und andere Vorgänge wie Volume-Klon und Volume-Größe ebenfalls ignoriert werden.



Diese Option ist nützlich, wenn Sie Kubernetes für Workloads in Containern verwenden möchten, aber ansonsten den Lebenszyklus des Storage Volumes außerhalb von Kubernetes managen möchten.

- Der PVC und dem PV wird eine Anmerkung hinzugefügt, die einem doppelten Zweck dient, anzugeben, dass das Volumen importiert wurde und ob PVC und PV verwaltet werden. Diese Anmerkung darf nicht geändert oder entfernt werden.

Importieren Sie ein Volume

Verwenden Sie können `tridentctl import` Um ein Volume zu importieren.

Schritte

1. Erstellen der PVC-Datei (Persistent Volume Claim) (beispielsweise `pvc.yaml`), die verwendet werden, um die PVC zu erstellen. Die PVC-Datei sollte enthalten `name`, `namespace`, `accessModes`, und `storageClassName`. Optional können Sie angeben `unixPermissions` In Ihrer PVC-Definition.

Im Folgenden finden Sie ein Beispiel für eine Mindestspezifikation:

```

kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: my_claim
  namespace: my_namespace
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: my_storage_class

```



Verwenden Sie keine zusätzlichen Parameter wie den PV-Namen oder die Volume-Größe. Dies kann dazu führen, dass der Importbefehl fehlschlägt.

2. Verwenden Sie die `tridentctl import volume` Befehl zur Angabe des Namens des Astra Trident Back-End, das das Volume enthält, sowie des Namens, der das Volume auf dem Storage eindeutig identifiziert (z. B. ONTAP FlexVol, Element Volume, Cloud Volumes Service-Pfad). Der `-f` Argument ist erforderlich, um den Pfad zur PVC-Datei anzugeben.

```

tridentctl import volume <backendName> <volumeName> -f <path-to-pvc-
file>

```

Beispiele

Lesen Sie die folgenden Beispiele für den Import von Volumes für unterstützte Treiber.

ONTAP NAS und ONTAP NAS FlexGroup

Astra Trident unterstützt den Volume-Import mithilfe von `ontap-nas` Und `ontap-nas-flexgroup` Treiber.



- Der `ontap-nas-economy` Der Treiber kann qtrees nicht importieren und verwalten.
- Der `ontap-nas` Und `ontap-nas-flexgroup` Treiber erlauben keine doppelten Volume-Namen.

Jedes Volume wurde mit erstellt `ontap-nas` Treiber ist ein FlexVol auf dem ONTAP Cluster. Importieren von FlexVols mit dem `ontap-nas` Der Treiber funktioniert genauso. Eine FlexVol, die bereits auf einem ONTAP Cluster vorhanden ist, kann als importiert werden `ontap-nas` PVC: Ebenso können FlexGroup Volumes importiert werden als `ontap-nas-flexgroup` VES.

Beispiele für ONTAP NAS

Die folgende Darstellung zeigt ein Beispiel für ein verwaltetes Volume und einen nicht verwalteten Volume-Import.

Gemanagtes Volume

Im folgenden Beispiel wird ein Volume mit dem Namen `importiert managed_volume` Auf einem Backend mit dem Namen `ontap_nas`:

```
tridentctl import volume ontap_nas managed_volume -f <path-to-pvc-file>
```

		NAME		SIZE		STORAGE CLASS			
PROTOCOL		BACKEND UUID			STATE		MANAGED		
	pvc-bf5ad463-afbb-11e9-8d9f-5254004dfdb7		1.0 GiB		standard				
file		c5a6f6a4-b052-423b-80d4-8fb491a14a22		online		true			

Nicht verwaltetes Volume

Bei Verwendung des `--no-manage` Argument, Astra Trident benennt das Volume nicht um.

Das folgende Beispiel importiert `unmanaged_volume` Auf dem `ontap_nas` Back-End:

```
tridentctl import volume nas_blog unmanaged_volume -f <path-to-pvc-file> --no-manage
```

		NAME		SIZE		STORAGE CLASS			
PROTOCOL		BACKEND UUID			STATE		MANAGED		
	pvc-df07d542-afbc-11e9-8d9f-5254004dfdb7		1.0 GiB		standard				
file		c5a6f6a4-b052-423b-80d4-8fb491a14a22		online		false			

ONTAP SAN

Astra Trident unterstützt den Volume-Import mithilfe von `ontap-san` Treiber. Der Import von Volumes wird nicht unterstützt `ontap-san-economy` Treiber.

Astra Trident kann ONTAP SAN FlexVols importieren, die eine einzige LUN enthalten. Dies entspricht dem `ontap-san` Treiber, der für jede PVC und eine LUN innerhalb der FlexVol eine FlexVol erstellt. Astra Trident importiert die FlexVol und ordnet sie der PVC-Definition zu.

Die folgende Darstellung zeigt ein Beispiel für ein verwaltetes Volume und einen nicht verwalteten Volume-Import.

Für gemanagte Volumes benennt Astra Trident die FlexVol in den um `pvc-<uuid>` Formatieren Sie und die LUN innerhalb der FlexVol bis `lun0`.

```
tridentctl import volume ontapsan_san_default ontap-san-managed -f pvc-  
basic-import.yaml -n trident -d
```

```
+-----+-----+-----+-----+
+-----+-----+-----+-----+
|          NAME          |  SIZE  | STORAGE CLASS |
+-----+-----+-----+-----+
| PROTOCOL |  BACKEND UUID  |  STATE  | MANAGED |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
| pvc-d6ee4f54-4e40-4454-92fd-d00fc228d74a | 20 MiB | basic          |
+-----+-----+-----+-----+
| block    | cd394786-ddd5-4470-adc3-10c5ce4ca757 | online | true        |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```

Das folgende Beispiel importiert `unmanaged example volume` Auf dem `ontap san` Back-End:

```
tridentctl import volume -n trident san_blog unmanaged_example_volume
-f pvc-import.yaml --no-manage
```

```
+-----+-----+-----+-----+
+-----+-----+-----+-----+
|          NAME          |  SIZE  | STORAGE CLASS |
+-----+-----+-----+-----+
| PROTOCOL | BACKEND UUID | STATE | MANAGED |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
| pvc-1fc999c9-ce8c-459c-82e4-ed4380a4b228 | 1.0 GiB | san-blog      |
+-----+-----+-----+-----+
| block    | e3275890-7d80-4af6-90cc-c7a0759f555a | online | false      |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```


154

zu importieren.

Vserver	Igroup	Protocol	OS Type	Initiators
svm0	k8s-nodename.example.com-fe5d36f2-cded-4f38-9eb0-c7719fc2f9f3	iscsi	linux	iqn.1994-05.com.redhat:4c2e1cf35e0
svm0	unmanaged-example-igroup	mixed	linux	iqn.1994-05.com.redhat:4c2e1cf35e0

Element

Astra Trident unterstützt die NetApp Element Software und den NetApp HCI Volume-Import über die `solidfire-san` Treiber.



Der Elementtreiber unterstützt doppelte Volume-Namen. Astra Trident gibt jedoch einen Fehler zurück, wenn es doppelte Volume-Namen gibt. Um dies zu umgehen, klonen Sie das Volume, geben Sie einen eindeutigen Volume-Namen ein und importieren Sie das geklonte Volume.

Beispiel für ein Element

Im folgenden Beispiel wird ein importiert `element-managed` Volume am Backend `element_default`.

```
tridentctl import volume element_default element-managed -f pvc-basic-import.yaml -n trident -d
```

NAME	SIZE	STORAGE CLASS
PROTOCOL BACKEND UUID	STATE	MANAGED
pvc-970ce1ca-2096-4ecd-8545-ac7edc24a8fe	10 GiB	basic-element
block d3ba047a-ea0b-43f9-9c42-e38e58301c49	online	true

Google Cloud Platform

Astra Trident unterstützt den Volume-Import mithilfe von `gcp-cvs` Treiber.



Um ein Volume zu importieren, das von NetApp Cloud Volumes Service in die Google Cloud Platform unterstützt wird, identifizieren Sie das Volume anhand seines Volume-Pfads. Der Volume-Pfad ist der Teil des Exportpfades des Volumes nach dem :/. Beispiel: Wenn der Exportpfad lautet 10.0.0.1:/adroit-jolly-swift, Der Volume-Pfad ist adroit-jolly-swift.

Beispiel für die Google Cloud Platform

Im folgenden Beispiel wird ein importiert gcp-cvs Volume am Backend gcpcvs_YEppr Mit dem Volume-Pfad von adroit-jolly-swift.

```
tridentctl import volume gcpcvs_YEppr adroit-jolly-swift -f <path-to-pvc-file> -n trident
```

PROTOCOL	NAME	BACKEND UUID	SIZE	STORAGE CLASS	STATE	MANAGED
	pvc-a46ccab7-44aa-4433-94b1-e47fc8c0fa55	e1a6e65b-299e-4568-ad05-4f0a105c888f	93 GiB	gcp-storage	online	true

Azure NetApp Dateien

Astra Trident unterstützt den Volume-Import mithilfe von azure-netapp-files Treiber.



Um ein Azure NetApp Files-Volume zu importieren, identifizieren Sie das Volume anhand seines Volume-Pfads. Der Volume-Pfad ist der Teil des Exportpfades des Volumes nach dem :/. Beispiel: Wenn der Mount-Pfad lautet 10.0.0.2:/importvoll1, Der Volume-Pfad ist importvoll1.

Beispiel: Azure NetApp Files

Im folgenden Beispiel wird ein importiert azure-netapp-files Volume am Backend azurenetafiles_40517 Mit dem Volume-Pfad importvoll1.

```
tridentctl import volume azurenetappfiles_40517 importvoll1 -f <path-to-pvc-file> -n trident
```

PROTOCOL	NAME	SIZE	STORAGE CLASS
file	1c01274f-d94b-44a3-98a3-04c953c9a51e	100 GiB	anf-storage

Ein NFS-Volume kann über Namespaces hinweg genutzt werden

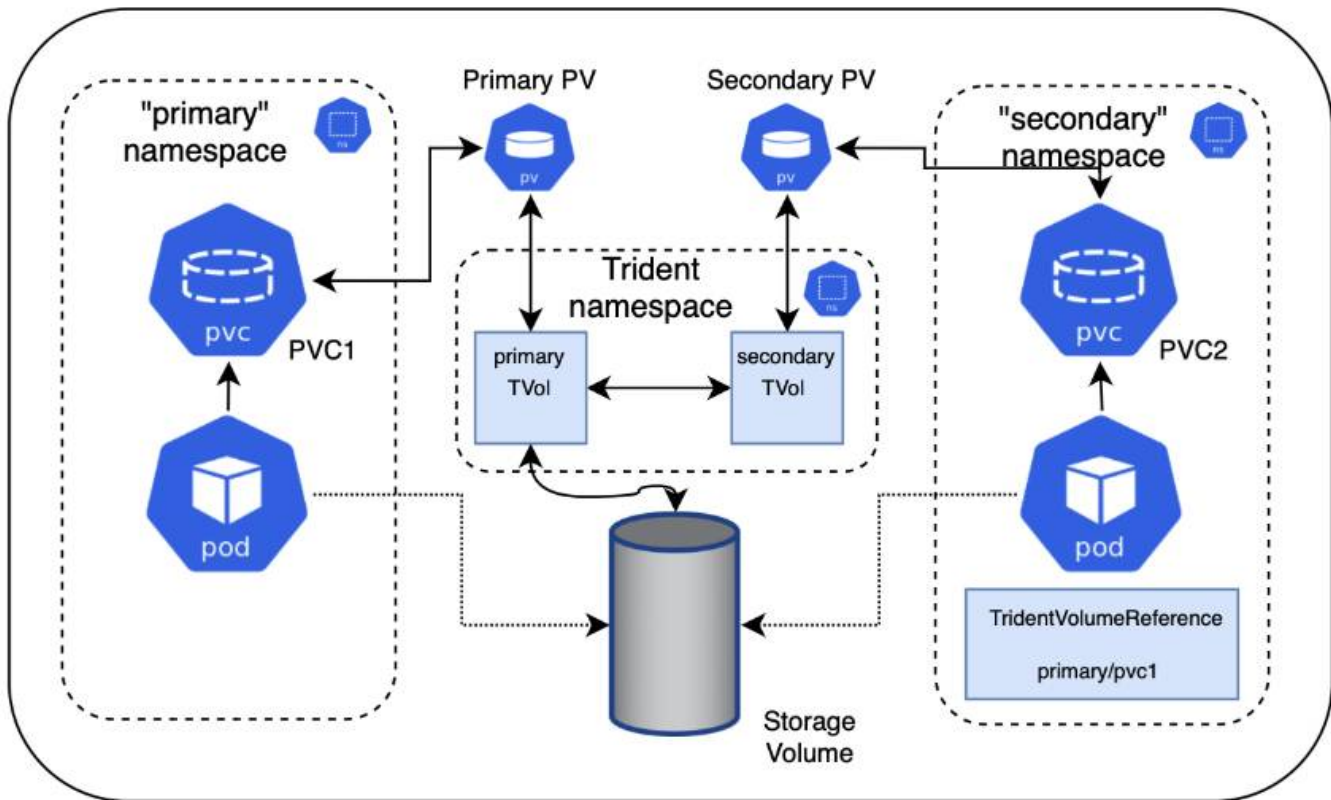
Mit Astra Trident können Sie ein Volume in einem primären Namespace erstellen und es in einem oder mehreren sekundären Namespaces teilen.

Funktionen

Mit dem Astra TridentVolumeReference CR können Sie ReadWriteManche (RWX) NFS-Volumes sicher über einen oder mehrere Kubernetes-Namespaces teilen. Diese native Kubernetes-Lösung bietet folgende Vorteile:

- Mehrere Stufen der Zugriffssteuerung zur Sicherstellung der Sicherheit
- Funktioniert mit allen Trident NFS-Volume-Treibern
- Tridentctl oder andere nicht-native Kubernetes-Funktionen sind nicht von Bedeutung

Dieses Diagramm zeigt die NFS-Volume-Freigabe über zwei Kubernetes-Namespaces.



Schnellstart

Sie können in nur wenigen Schritten NFS-Volume Sharing einrichten.

1

Konfigurieren Sie die PVC-Quelle für die gemeinsame Nutzung des Volumes

Der Eigentümer des Quell-Namespace erteilt die Berechtigung, auf die Daten im Quell-PVC zuzugreifen.

2

Berechtigung zum Erstellen eines CR im Ziel-Namespace gewähren

Der Clusteradministrator erteilt dem Eigentümer des Ziel-Namespace die Berechtigung, das TridentVolumeReference CR zu erstellen.

3

Erstellen Sie im Ziel-Namespace tridentVolumeReference

Der Eigentümer des Ziel-Namespace erstellt das TridentVolumeReference CR, um sich auf das Quell-PVC zu beziehen.

4

Erstellen Sie das untergeordnete PVC im Ziel-Namespace

Der Eigentümer des Ziel-Namespace erstellt das untergeordnete PVC, um die Datenquelle aus dem Quell-PVC zu verwenden.

Konfigurieren Sie die Namensräume für Quelle und Ziel

Um die Sicherheit zu gewährleisten, erfordert die Namespace-übergreifende Freigabe Zusammenarbeit und Aktion durch den Eigentümer des Quell-Namespace, den Cluster-Administrator und den Ziel-Namespace-Eigentümer. In jedem Schritt wird die Benutzerrolle festgelegt.

Schritte

1. **Source Namespace Owner:** Erstellen Sie das PVC (`pvc1`) Im Quell-Namespace, der die Erlaubnis gibt, mit dem Ziel-Namespace zu teilen (`namespace2`) Mit dem `shareToNamespace` Anmerkung:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1
  namespace: namespace1
  annotations:
    trident.netapp.io/shareToNamespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

Astra Trident erstellt das PV und das Back-End NFS Storage Volume.



- Sie können das PVC über eine durch Kommas getrennte Liste mehreren Namespaces freigeben. Beispiel: `trident.netapp.io/shareToNamespace: namespace2, namespace3, namespace4`.
- Sie können mit allen Namespaces freigeben `*`. Beispiel: `trident.netapp.io/shareToNamespace: *`
- Sie können das PVC so aktualisieren, dass es die enthält `shareToNamespace` Kommentare können jederzeit hinzugefügt werden.

2. **Cluster Admin:** Erstellen Sie die benutzerdefinierte Rolle und `kubeconfig`, um dem Ziel-Namespace-Eigentümer die Berechtigung zu erteilen, das `TridentVolumeReference` CR im Ziel-Namespace zu erstellen.
3. **Zielgebietes-Namespace-Eigentümer:** Erstellen Sie ein `TridentVolumeReference` CR im Ziel-Namespace, der sich auf den Quell-Namespace bezieht `pvc1`.

```

apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1

```

4. **Eigentümer des Ziel-Namespace:** Erstellen Sie ein PVC (pvc2) Im Ziel-Namespace (namespace2) Mit dem shareFromPVC Anmerkung zur Angabe der Quelle PVC.

```

kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  annotations:
    trident.netapp.io/shareFromPVC: namespace1/pvc1
  name: pvc2
  namespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi

```



Die Größe der Ziel-PVC muss kleiner oder gleich der Quelle PVC sein.

Ergebnisse

Astra Trident liest den shareFromPVC Anmerkung auf dem Ziel-PVC und erstellt das Ziel-PV als untergeordnetes Volumen ohne eigene Speicherressource, die auf das Quell-PV verweist und die PV-Quellressource teilt. Die Ziel-PVC und das PV erscheinen wie normal gebunden.

Löschen eines freigegebenen Volumes

Sie können ein Volume löschen, das über mehrere Namespaces hinweg gemeinsam genutzt wird. Astra Trident entfernt den Zugriff auf das Volume im Quell-Namespace und behält auch andere Namespaces, die das Volume gemeinsam nutzen. Wenn alle Namespaces entfernt werden, die auf dem Volume verweisen, löscht Astra Trident das Volume.

Nutzung `tridentctl get` Zum Abfragen von untergeordneten Volumes

Verwenden der `tridentctl` Das Dienstprogramm kann ausgeführt werden `get` Befehl zum Abrufen untergeordneter Volumes. Weitere Informationen finden Sie unter [Link:../Trident-](#)

Referenz/tridentctl.html[tridentctl Befehle und Optionen].

Usage:

```
tridentctl get [option]
```

Markierungen:

- `-h, --help`: Hilfe für Volumen.
- `--parentOfSubordinate string`: Abfrage auf untergeordnetes Quellvolumen begrenzen.
- `--subordinateOf string`: Abfrage auf Untergeebene beschränken.

Einschränkungen

- Astra Trident kann nicht verhindern, dass Ziel-Namespace auf dem Shared Volume schreiben. Sie sollten Dateisperren oder andere Prozesse verwenden, um das Überschreiben von gemeinsam genutzten Volume-Daten zu verhindern.
- Sie können den Zugriff auf die Quelle PVC nicht widerrufen, indem Sie die entfernen `shareToNamespace` Oder `shareFromNamespace` Anmerkungen oder Löschen des `TridentVolumeReference` CR. Um den Zugriff zu widerrufen, müssen Sie das untergeordnete PVC löschen.
- Snapshots, Klone und Spiegelungen sind auf untergeordneten Volumes nicht möglich.

Finden Sie weitere Informationen

Weitere Informationen zum Namespace-übergreifenden Volume-Zugriff:

- Besuchen Sie ["Teilen von Volumes zwischen Namespaces: Sagen Sie hallo für Namespace-übergreifenden Volume-Zugriff"](#).

Replizieren Sie Volumes mit SnapMirror

Mit Astra Control Provisioner können Sie Spiegelungsbeziehungen zwischen einem Quell-Volume auf einem Cluster und dem Ziel-Volume auf dem Peering-Cluster erstellen, um Daten für die Disaster Recovery zu replizieren. Sie können eine benutzerdefinierte Ressourcendefinition (CRD, Named Custom Resource Definition) verwenden, um die folgenden Vorgänge auszuführen:

- Erstellen von Spiegelbeziehungen zwischen Volumes (VES)
- Entfernen Sie Spiegelungsbeziehungen zwischen Volumes
- Brechen Sie die Spiegelbeziehungen auf
- Bewerben des sekundären Volumes bei Ausfällen (Failover)
- Verlustfreie Transition von Applikationen von Cluster zu Cluster (während geplanter Failover oder Migrationen)

Replikationsvoraussetzungen

Stellen Sie sicher, dass die folgenden Voraussetzungen erfüllt sind, bevor Sie beginnen:

ONTAP Cluster

- **Astra Control Provisioner:** Astra Control Provisioner Version 23.10 oder höher muss sowohl auf den Quell- als auch auf den Ziel-Kubernetes-Clustern vorhanden sein, die ONTAP als Backend verwenden.
- **Lizenzen:** Asynchrone Lizenzen von ONTAP SnapMirror, die das Datensicherungspaket verwenden, müssen sowohl auf den Quell- als auch auf den Ziel-ONTAP-Clustern aktiviert sein. Weitere Informationen finden Sie unter "[Übersicht über die SnapMirror Lizenzierung in ONTAP](#)".

Peering

- **Cluster und SVM:** Die ONTAP Speicher-Back-Ends müssen aktiviert werden. Weitere Informationen finden Sie unter "[Übersicht über Cluster- und SVM-Peering](#)".



Vergewissern Sie sich, dass die in der Replizierungsbeziehung zwischen zwei ONTAP-Clustern verwendeten SVM-Namen eindeutig sind.

- **Astra Control Provisioner und SVM:** Die Peering von Remote-SVMs müssen für die Astra Control Bereitstellung im Ziel-Cluster verfügbar sein.

Unterstützte Treiber

- Die Volume-Replizierung wird von ontap-nas und ontap-san Treibern unterstützt.

Erstellen Sie eine gespiegelte PVC

Führen Sie die folgenden Schritte aus, und verwenden Sie die CRD-Beispiele, um eine Spiegelungsbeziehung zwischen primären und sekundären Volumes zu erstellen.

Schritte

1. Führen Sie auf dem primären Kubernetes-Cluster die folgenden Schritte aus:
 - a. Erstellen Sie ein StorageClass-Objekt mit dem `trident.netapp.io/replication: true` Parameter.

Beispiel

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-nas
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  fsType: "nfs"
  trident.netapp.io/replication: "true"
```

- b. PVC mit zuvor erstellter StorageClass erstellen.

Beispiel

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: csi-nas
```

- c. Erstellen Sie eine MirrorRelation CR mit lokalen Informationen.

Beispiel

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: promoted
  volumeMappings:
    - localPVCName: csi-nas
```

Astra Control Provisioner ruft die internen Informationen für das Volume und den aktuellen DP-Status des Volumes ab und füllt dann das Statusfeld der MirrorRelationship aus.

- d. Holen Sie sich den TridentMirrorRelationship CR, um den internen Namen und die SVM der PVC zu erhalten.

```
kubectl get tmr csi-nas
```

```

kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
  generation: 1
spec:
  state: promoted
  volumeMappings:
    - localPVCName: csi-nas
status:
  conditions:
    - state: promoted
    localVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
    localPVCName: csi-nas
    observedGeneration: 1

```

2. Führen Sie auf dem sekundären Kubernetes-Cluster die folgenden Schritte aus:

a. Erstellen Sie eine StorageClass mit dem Parameter `trident.netapp.io/replication: true`.

Beispiel

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-nas
provisioner: csi.trident.netapp.io
parameters:
  trident.netapp.io/replication: true

```

b. Erstellen Sie eine MirrorRelationship-CR mit Ziel- und Quellinformationen.

Beispiel

```

kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: established
  volumeMappings:
    - localPVCName: csi-nas
      remoteVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"

```

Astra Control Provisioner erstellt eine SnapMirror Beziehung zum Namen der konfigurierten Beziehungsrichtlinie (oder dem Standard für ONTAP) und initialisiert sie.

- c. PVC mit zuvor erstellter StorageClass erstellen, um als sekundäres Ziel zu fungieren (SnapMirror Ziel).

Beispiel

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
  annotations:
    trident.netapp.io/mirrorRelationship: csi-nas
spec:
  accessModes:
    - ReadWriteMany
resources:
  requests:
    storage: 1Gi
storageClassName: csi-nas
```

Astra Control Provisioner überprüft die CRD für die TridentMirrorRelationship und erstellt das Volume nicht, wenn die Beziehung nicht vorhanden ist. Falls die Beziehung besteht, stellt Astra Control Provisioner sicher, dass das neue FlexVol Volume auf eine SVM platziert wird, die mit der in MirrorRelation definierten Remote SVM verbunden ist.

Volume-Replikationsstatus

Eine Trident Mirror-Beziehung (TMR) ist eine CRD, die ein Ende einer Replizierungsbeziehung zwischen PVCs darstellt. Das Ziel-TMR verfügt über einen Status, der Astra Control Provisioner über den gewünschten Status informiert. Das Ziel-TMR hat die folgenden Zustände:

- **Etabliert:** Die lokale PVC ist das Zielvolumen einer Spiegelbeziehung, und das ist eine neue Beziehung.
- **Befördert:** Die lokale PVC ist ReadWrite und montierbar, ohne dass aktuell eine Spiegelbeziehung besteht.
- **Wiederhergestellt:** Die lokale PVC ist das Zielvolumen einer Spiegelbeziehung und war zuvor auch in dieser Spiegelbeziehung.
 - Der neu eingerichtete Status muss verwendet werden, wenn das Ziel-Volume jemals in einer Beziehung zum Quell-Volume stand, da es den Inhalt des Ziel-Volume überschreibt.
 - Der neu eingerichtete Status schlägt fehl, wenn das Volume zuvor nicht in einer Beziehung zur Quelle stand.

Fördern Sie die sekundäre PVC während eines ungeplanten Failover

Führen Sie den folgenden Schritt auf dem sekundären Kubernetes-Cluster aus:

- Aktualisieren Sie das Feld *spec.State* von TridentMirrorRelationship auf *promoted*.

Fördern Sie die sekundäre PVC während eines geplanten Failover

Führen Sie während eines geplanten Failover (Migration) die folgenden Schritte durch, um die sekundäre PVC hochzustufen:

Schritte

1. Erstellen Sie auf dem primären Kubernetes-Cluster einen Snapshot der PVC und warten Sie, bis der Snapshot erstellt wurde.
2. Erstellen Sie auf dem primären Kubernetes-Cluster SnapshotInfo CR, um interne Details zu erhalten.

Beispiel

```
kind: SnapshotInfo
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  snapshot-name: csi-nas-snapshot
```

3. Aktualisieren Sie im sekundären Kubernetes-Cluster das Feld *spec.State* des *tridentMirrorRelationship* CR auf *promoted* und *spec.promotedSnapshotHandle* als InternalName des Snapshots.
4. Bestätigen Sie auf sekundärem Kubernetes-Cluster den Status (Feld *Status.State*) von *TridentMirrorRelationship* auf hochgestuft.

Stellen Sie nach einem Failover eine gespiegelte Beziehung wieder her

Wählen Sie vor dem Wiederherstellen einer Spiegelbeziehung die Seite aus, die Sie als neuen primären festlegen möchten.

Schritte

1. Stellen Sie auf dem sekundären Kubernetes-Cluster sicher, dass die Werte für das Feld *spec.remoteVolumeHandle* auf dem *TridentMirrorRelationship* aktualisiert werden.
2. Aktualisieren Sie im sekundären Kubernetes-Cluster das Feld *spec.mirror* von *TridentMirrorRelationship* auf *reestablished*.

Zusätzliche Vorgänge

Astra Control Provisioner unterstützt die folgenden Vorgänge für primäre und sekundäre Volumes:

Replizieren der primären PVC auf eine neue sekundäre PVC

Stellen Sie sicher, dass Sie bereits über eine primäre PVC und eine sekundäre PVC verfügen.

Schritte

1. Löschen Sie die CRDs *PersistentVolumeClaim* und *TridentMirrorRelationship* aus dem eingerichteten sekundären Cluster (Ziel).
2. Löschen Sie die CRD für *TridentMirrorRelationship* aus dem primären (Quell-) Cluster.
3. Erstellen Sie eine neue *TridentMirrorRelationship* CRD auf dem primären (Quell-) Cluster für die neue sekundäre (Ziel-) PVC, die Sie einrichten möchten.

Ändern der Größe einer gespiegelten, primären oder sekundären PVC

Die PVC-Größe kann wie gewohnt geändert werden. ONTAP erweitert automatisch alle Zielflvxole, wenn die Datenmenge die aktuelle Größe überschreitet.

Entfernen Sie die Replikation aus einer PVC

Um die Replikation zu entfernen, führen Sie einen der folgenden Vorgänge auf dem aktuellen sekundären Volume aus:

- Löschen Sie `MirrorRelation` auf der sekundären PVC. Dadurch wird die Replikationsbeziehung unterbrochen.
- Oder aktualisieren Sie das Feld `spec.State` auf *promoted*.

Löschen einer PVC (die zuvor gespiegelt wurde)

Astra Control Provisioner überprüft nach replizierten PVCs und gibt die Replizierungsbeziehung frei, bevor versucht wird, das Volume zu löschen.

Löschen eines TMR

Das Löschen eines TMR auf einer Seite einer gespiegelten Beziehung führt dazu, dass der verbleibende TMR in den Status *promoted* übergeht, bevor Astra Control Provisioner den Löschvorgang abgeschlossen hat. Wenn der für den Löschvorgang ausgewählte TMR bereits den Status *promoted* hat, gibt es keine bestehende Spiegelbeziehung und der TMR wird entfernt und Astra Control Provisioner wird die lokale PVC auf *ReadWrite* hochstufen. Durch dieses Löschen werden `SnapMirror` Metadaten für das lokale Volume in ONTAP freigegeben. Wenn dieses Volume in Zukunft in einer Spiegelbeziehung verwendet wird, muss es beim Erstellen der neuen Spiegelbeziehung ein neues TMR mit einem *established* Volume-Replikationsstatus verwenden.

Aktualisieren Sie Spiegelbeziehungen, wenn ONTAP online ist

Spiegelbeziehungen können jederzeit nach ihrer Einrichtung aktualisiert werden. Sie können die Felder oder verwenden `state: promoted` `state: reestablished`, um die Beziehungen zu aktualisieren. Wenn Sie ein Zielvolume auf ein reguläres `ReadWrite`-Volume heraufstufen, können Sie *promotedSnapshotHandle* verwenden, um einen bestimmten Snapshot anzugeben, auf dem das aktuelle Volume wiederhergestellt werden soll.

Aktualisieren Sie Spiegelbeziehungen, wenn ONTAP offline ist

Sie können ein CRD verwenden, um ein `SnapMirror Update` durchzuführen, ohne dass Astra Control direkt mit dem ONTAP Cluster verbunden ist. Im folgenden Beispielformat finden Sie das `TridentActionMirrorUpdate`:

Beispiel

```
apiVersion: trident.netapp.io/v1
kind: TridentActionMirrorUpdate
metadata:
  name: update-mirror-b
spec:
  snapshotHandle: "pvc-1234/snapshot-1234"
  tridentMirrorRelationshipName: mirror-b
```

`status.state` Gibt den Status von TridentActionMirrorUpdate CRD wieder. Es kann einen Wert von *suileded*, *in progress* oder *failed* annehmen.

Astra Control Provisioner Aktivieren

Trident Version 23.10 und höher bieten die Option zur Verwendung von Astra Control Provisioner, damit Benutzer von Astra Control auf erweiterte Funktionen zur Storage-Bereitstellung zugreifen können. Astra Control Provisioner bietet diese erweiterte Funktionalität zusätzlich zu den auf Astra Trident basierenden Standardfunktionen. Mit diesem Verfahren können Sie Astra Control Provisioner aktivieren und installieren.

Im Abonnement für Astra Control Service ist automatisch die Lizenz für die Nutzung von Astra Control Provisioner enthalten.

Bei den neuesten Updates für Astra Control wird Astra Control Provisioner Astra Trident als Storage-bereitstellung und -Orchestrierung ersetzen und für die Verwendung von Astra Control obligatorisch sein. Aus diesem Grund wird dringend empfohlen, Astra Control für die Astra Control-Bereitstellung zu aktivieren. Astra Trident wird weiterhin Open Source bleiben und mit neuen CSI- und anderen Funktionen von NetApp veröffentlicht, gepflegt, unterstützt und auf dem neuesten Stand sein.

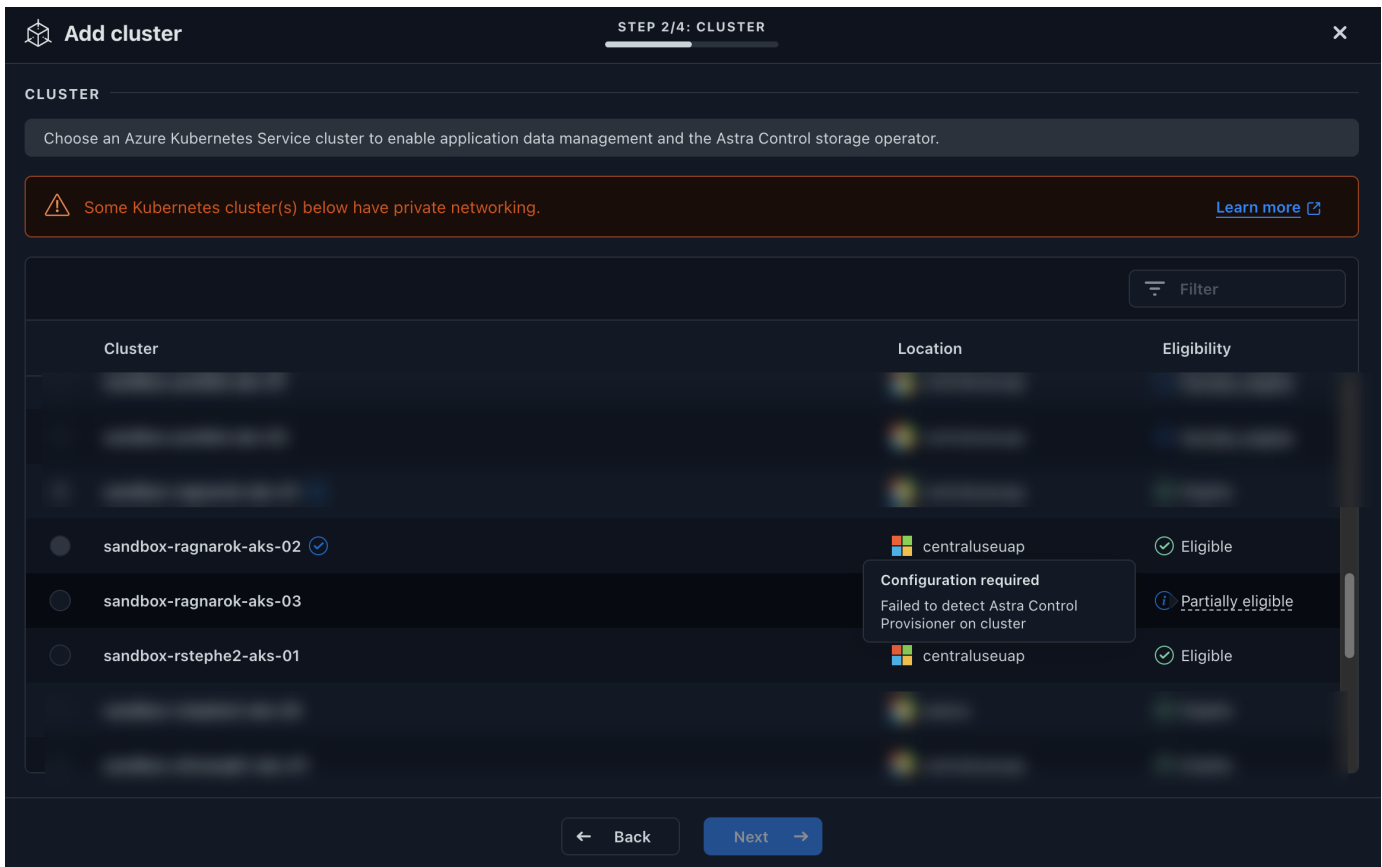
Wie kann ich feststellen, ob ich die Astra Control-Bereitstellung aktivieren muss?

Wenn Sie Astra Control Service einen Cluster hinzufügen, für den Astra Trident zuvor nicht installiert ist, wird der Cluster als markiert `Eligible`. Nach Ihnen ["Fügen Sie den Cluster zu Astra Control hinzu"](#) wird der Astra Control Provisioner automatisch aktiviert.

Wenn Ihr Cluster nicht markiert ist `Eligible`, wird er aufgrund einer der folgenden Zeichen markiert `Partially eligible`:

- Es verwendet eine ältere Version von Astra Trident
- In Astra Trident 23.10 wird noch nicht die bereitstellungsoption aktiviert
- Es handelt sich um einen Cluster-Typ, der keine automatische Aktivierung zulässt

In `Partially eligible` Fällen können Sie mit diesen Anweisungen die Astra Control Provisioner für Ihr Cluster manuell aktivieren.



Bevor Sie Astra Control Provisioner aktivieren

Wenn Sie bereits Astra Trident ohne Astra Control Provisioner verwenden und Astra Control Provisioner aktivieren möchten, gehen Sie zuerst wie folgt vor:

- **Wenn Sie Astra Trident installiert haben, bestätigen Sie, dass seine Version innerhalb eines Fensters mit vier Versionen ist:** Sie können ein direktes Upgrade auf Astra Trident 24.02 mit Astra Control Provisioner durchführen, wenn Ihr Astra Trident innerhalb eines Fensters mit vier Versionen von Version 24.02 ist. Sie können beispielsweise direkt von Astra Trident 23.04 auf 24.02 aktualisieren.
- **Bestätigen Sie, dass Ihr Cluster über eine AMD64-Systemarchitektur verfügt:** Das Astra Control Provisioner-Image wird sowohl in AMD64- als auch in ARM64-CPU-Architekturen bereitgestellt, aber nur AMD64 wird von Astra Control unterstützt.

Schritte

1. Rufen Sie die NetApp Astra Control Image-Registry auf:
 - a. Melden Sie sich an der Astra Control Service UI an und notieren Sie Ihre Astra Control Konto-ID.
 - i. Wählen Sie das Symbol oben rechts auf der Seite.
 - ii. Wählen Sie **API-Zugriff**.
 - iii. Notieren Sie sich Ihre Konto-ID.
 - b. Wählen Sie auf derselben Seite **API-Token generieren** aus und kopieren Sie die API-Token-Zeichenfolge in die Zwischenablage und speichern Sie sie in Ihrem Editor.
 - c. Melden Sie sich über Ihre bevorzugte Methode in der Astra Control Registry an:

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

```
crane auth login cr.astra.netapp.io -u <account-id> -p <api-token>
```

2. (Nur benutzerdefinierte Registrierungen) Befolgen Sie diese Schritte, um das Bild in Ihre benutzerdefinierte Registrierung zu verschieben. Wenn Sie keine Registrierung verwenden, befolgen Sie die Schritte des Trident-Operators im [Nächster Abschnitt](#).



Sie können Podman anstelle von Docker für die folgenden Befehle verwenden. Wenn Sie eine Windows-Umgebung verwenden, wird PowerShell empfohlen.

Docker

- a. Rufen Sie das Astra Control Provisioner-Image aus der Registrierung ab:



Das abgezogene Image unterstützt nicht mehrere Plattformen und unterstützt nur die gleiche Plattform wie der Host, der das Image gezogen hat, wie z. B. Linux AMD64.

```
docker pull cr.astra.netapp.io/astra/trident-acp:24.02.0
--platform <cluster platform>
```

Beispiel:

```
docker pull cr.astra.netapp.io/astra/trident-acp:24.02.0
--platform linux/amd64
```

- b. Markieren Sie das Bild:

```
docker tag cr.astra.netapp.io/astra/trident-acp:24.02.0
<my_custom_registry>/trident-acp:24.02.0
```

- c. Laden Sie das Bild in Ihre benutzerdefinierte Registrierung:

```
docker push <my_custom_registry>/trident-acp:24.02.0
```

Kran

- a. Kopieren Sie das Astra Control Provisioner-Manifest in Ihre benutzerdefinierte Registry:

```
crane copy cr.astra.netapp.io/astra/trident-acp:24.02.0
<my_custom_registry>/trident-acp:24.02.0
```

3. Stellen Sie fest, ob die ursprüngliche Astra Trident Installationsmethode einen verwendet hat.
4. Aktivieren Sie Astra Control Provisioner in Astra Trident mit der ursprünglich verwendeten Installationsmethode:

Astra Trident Betreiber

- a. "Laden Sie das Astra Trident Installationsprogramm herunter und extrahieren Sie es".
- b. Führen Sie diese Schritte aus, wenn Sie Astra Trident noch nicht installiert haben oder den Operator aus der ursprünglichen Astra Trident-Implementierung entfernt haben:
 - i. Erstellen des CRD:

```
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.1
6.yaml
```

- ii. Erstellen Sie den dreigesichtigen Namespace (`kubectl create namespace trident`) oder bestätigen Sie, dass der dreigesichtigen Namespace noch existiert (`kubectl get all -n trident`). Wenn der Namespace entfernt wurde, erstellen Sie ihn erneut.

- c. Update von Astra Trident auf 24.02.0:



Verwenden Sie für Cluster mit Kubernetes 1.24 oder früher `bundle_pre_1_25.yaml`: Verwenden Sie für Cluster mit Kubernetes 1.25 oder höher `bundle_post_1_25.yaml`.

```
kubectl -n trident apply -f trident-installer/deploy/<bundle-
name.yaml>
```

- d. Überprüfen Sie, ob Astra Trident ausgeführt wird:

```
kubectl get torc -n trident
```

Antwort:

NAME	AGE
trident	21m

- e. Wenn Sie eine Registry mit Geheimnissen haben, erstellen Sie ein Geheimnis, mit dem Sie das Astra Control Provisioner-Bild abrufen können:

```
kubectl create secret docker-registry <secret_name> -n trident
--docker-server=<my_custom_registry> --docker-username=<username>
--docker-password=<token>
```

- f. Bearbeiten Sie den TridentOrchestrator CR, und nehmen Sie die folgenden Änderungen vor:

```
kubectl edit torc trident -n trident
```

- i. Legen Sie einen benutzerdefinierten Registrierungsort für das Astra Trident-Image fest oder ziehen Sie es aus der Astra Control-Registry (`tridentImage: <my_custom_registry>/trident:24.02.0` oder `tridentImage: netapp/trident:24.02.0`).
- ii. Aktivieren Sie Astra Control Provisioner (`enableACP: true`).
- iii. Legen Sie den benutzerdefinierten Registrierungsport für das Astra Control Provisioner-Image fest oder ziehen Sie es aus der Astra Control Registry (`acpImage: <my_custom_registry>/trident-acp:24.02.0` oder `acpImage: cr.astra.netapp.io/astra/trident-acp:24.02.0`).
- iv. Wenn Sie in diesem Verfahren bereits einmal eingerichtet [Geheimnisse der Bildausziehung](#) haben, können Sie diese hier einstellen (`imagePullSecrets: - <secret_name>`). Verwenden Sie den gleichen geheimen Namen, den Sie in den vorherigen Schritten festgelegt haben.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  tridentImage: <registry>/trident:24.02.0
  enableACP: true
  acpImage: <registry>/trident-acp:24.02.0
  imagePullSecrets:
    - <secret_name>
```

- g. Speichern und beenden Sie die Datei. Der Bereitstellungsprozess wird automatisch gestartet.
- h. Überprüfen Sie, ob der Operator, die Bereitstellung und Replikasets erstellt wurden.

```
kubectl get all -n trident
```



Es sollte nur eine Instanz* des Operators in einem Kubernetes-Cluster geben. Erstellen Sie nicht mehrere Implementierungen des Astra Trident Operators.

- i. Überprüfen Sie, ob der `trident-acp` Container läuft und der `acpVersion` Status lautet 24.02.0 Installed:

```
kubectl get torc -o yaml
```

Antwort:

```

status:
  acpVersion: 24.02.0
  currentInstallationParams:
    ...
    acpImage: <registry>/trident-acp:24.02.0
    enableACP: "true"
    ...
  ...
status: Installed

```

Tridentctl

- "Laden Sie das Astra Trident Installationsprogramm herunter und extrahieren Sie es".
- "Wenn Sie bereits Astra Trident verwenden, deinstallieren Sie ihn aus dem Cluster, das ihn hostet".
- Astra Trident mit aktivierter Astra Control Provisioner installieren (`--enable-acp=true`):

```

./tridentctl -n trident install --enable-acp=true --acp
-image=mycustomregistry/trident-acp:24.02

```

- Aktivieren Sie die Astra Control Provisioner-Funktion:

```

./tridentctl -n trident version

```

Antwort:

```

+-----+-----+-----+ | SERVER
VERSION | CLIENT VERSION | ACP VERSION | +-----+
+-----+-----+-----+ | 24.02.0 | 24.02.0 | 24.02.0. |
+-----+-----+-----+

```

Helm

- Bei Installation von Astra Trident 23.07.1 oder einer früheren Version sind der Operator und andere Komponenten installiert "[Deinstallieren](#)".
- Wenn auf dem Kubernetes-Cluster 1.24 oder eine frühere Version ausgeführt wird, löschen Sie psp:

```

kubectl delete psp tridentoperatorpod

```

- Fügen Sie das Helm Repository von Astra Trident hinzu:


```
helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
```

d. Aktualisieren Sie das Helm-Diagramm:

```
helm repo update netapp-trident
```

Antwort:

```
Hang tight while we grab the latest from your chart
repositories...
...Successfully got an update from the "netapp-trident" chart
repository
Update Complete. ☐Happy Helming!☐
```

e. Auflisten der Bilder:

```
./tridentctl images -n trident
```

Antwort:

```
| v1.28.0          | netapp/trident:24.02.0|
|                  | docker.io/netapp/trident-
autosupport:24.02|
|                  | registry.k8s.io/sig-storage/csi-
provisioner:v4.0.0|
|                  | registry.k8s.io/sig-storage/csi-
attacher:v4.5.0|
|                  | registry.k8s.io/sig-storage/csi-
resizer:v1.9.3|
|                  | registry.k8s.io/sig-storage/csi-
snapshotter:v6.3.3|
|                  | registry.k8s.io/sig-storage/csi-node-
driver-registrar:v2.10.0 |
|                  | netapp/trident-operator:24.02.0 (optional)
```

f. Stellen Sie sicher, dass Dreizack-Bediener 24.02.0 verfügbar ist:

```
helm search repo netapp-trident/trident-operator --versions
```

Antwort:

NAME	CHART VERSION	APP VERSION	
DESCRIPTION			
netapp-trident/trident-operator	100.2402.0	24.02.0	A

g. Verwenden Sie `helm install` eine der folgenden Optionen, die diese Einstellungen enthalten, und führen Sie sie aus:

- Ein Name für Ihren Bereitstellungsart
- Die Version Astra Trident
- Der Name des Bildes für die Astra Control-Bereitstellung
- Das Flag, mit dem die provisionierung aktiviert wird
- (Optional) Ein lokaler Registrierungspfad. Wenn Sie eine lokale Registrierung verwenden, kann sich Ihr "[Trident Images](#)" in einer Registrierung oder in verschiedenen Registrierungen befinden, aber alle CSI-Images müssen sich in derselben Registrierung befinden.
- Der Trident Namespace

Optionen

- Bilder ohne Registrierung

```
helm install trident netapp-trident/trident-operator --version
100.2402.0 --set acpImage=cr.astra.netapp.io/astra/trident-
acp:24.02.0 --set enableACP=true --set operatorImage=netapp/trident-
operator:24.02.0 --set
tridentAutosupportImage=docker.io/netapp/trident-autosupport:24.02
--set tridentImage=netapp/trident:24.02.0 --namespace trident
```

- Bilder in einer oder mehreren Registern

```
helm install trident netapp-trident/trident-operator --version
100.2402.0 --set acpImage=<your-registry>:<acp image> --set
enableACP=true --set imageRegistry=<your-registry>/sig-storage --set
operatorImage=netapp/trident-operator:24.02.0 --set
tridentAutosupportImage=docker.io/netapp/trident-autosupport:24.02
--set tridentImage=netapp/trident:24.02.0 --namespace trident
```

Verwenden Sie können `helm list` So prüfen Sie Installationsdetails wie Name, Namespace, Diagramm, Status, App-Version, Und Revisionsnummer.

Falls Sie Probleme bei der Implementierung von Trident mit Helm haben, führen Sie diesen Befehl aus, um Astra Trident vollständig zu deinstallieren:

```
./tridentctl uninstall -n trident
```

Nicht "[Astra Trident CRDs vollständig entfernen](#)" als Teil Ihrer Deinstallation, bevor Sie versuchen, Astra Control Provisioner wieder zu aktivieren.

Ergebnis

Die Bereitstellungsfunktion von Astra Control ist aktiviert und Sie können alle Funktionen der verwendeten Version verwenden.

Nach der Installation von Astra Control provisioner wird im Cluster, das die provisionierung in der Astra Control UI hostet, ein Feld statt und die aktuelle installierte Versionsnummer angezeigt `ACP version Trident version`.

CLUSTER STATUS

Available

Version v1.24.9+rke2r2	Managed 2024/03/15 17:32 UTC	Kube-system namespace UID <div></div>	ACP Version <div></div>
Private route identifier <div>...</div>	Cloud instance private	Default bucket astra-bucket1 (inherited)	

Overview

Namespaces

Storage

Activity

Finden Sie weitere Informationen

- ["Dokumentation für Astra Trident Upgrades"](#)

Verwenden Sie die CSI-Topologie

Astra Trident kann Volumes selektiv erstellen und zu Nodes in einem Kubernetes Cluster verbinden, indem der verwendet wird ["Funktion CSI Topology"](#).

Überblick

Mithilfe der CSI Topology-Funktion kann der Zugriff auf Volumes auf einen Teil von Nodes basierend auf Regionen und Verfügbarkeitszonen begrenzt werden. Cloud-Provider ermöglichen Kubernetes-Administratoren inzwischen das Erstellen von Nodes, die zonenbasiert sind. Die Nodes können sich in verschiedenen Verfügbarkeitszonen innerhalb einer Region oder über verschiedene Regionen hinweg befinden. Astra Trident verwendet CSI Topology, um die Provisionierung von Volumes für Workloads in einer Multi-Zone-Architektur zu vereinfachen.



Erfahren Sie mehr über die Funktion CSI Topology ["Hier"](#).

Kubernetes bietet zwei unterschiedliche Modi für die Volume-Bindung:

- Mit `VolumeBindingMode` Auf einstellen `Immediate`, Astra Trident erstellt das Volume ohne Topologiebewusstsein. Die Volume-Bindung und die dynamische Bereitstellung werden bei der Erstellung des PVC behandelt. Dies ist die Standardeinstellung `VolumeBindingMode` Und ist für Cluster geeignet, die keine Topologiebeschränkungen mehr durchsetzen. Persistente Volumes werden erstellt, ohne von den Planungsanforderungen des anfragenden Pods abhängig zu sein.

- Mit `VolumeBindingMode` Auf einstellen `WaitForFirstConsumer`, Die Erstellung und Bindung eines Persistent Volume für ein PVC wird verzögert, bis ein Pod, der die PVC verwendet, geplant und erstellt wird. Auf diese Weise werden Volumes erstellt, um Planungseinschränkungen zu erfüllen, die durch Topologieanforderungen durchgesetzt werden.



Der `WaitForFirstConsumer` Für den Bindungsmodus sind keine Topologiebeschriftungen erforderlich. Diese kann unabhängig von der CSI Topology Funktion verwendet werden.

Was Sie benötigen

Für die Verwendung von CSI Topology benötigen Sie Folgendes:

- Einen Kubernetes-Cluster mit einem ["Unterstützte Kubernetes-Version"](#)

```
kubectl version
Client Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:50:19Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:41:49Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
```

- Nodes im Cluster sollten über Labels verfügen, die eine Topologiebewusstsein einführen (`topology.kubernetes.io/region` Und `topology.kubernetes.io/zone`). Diese Labels * sollten auf Knoten im Cluster vorhanden sein* bevor Astra Trident installiert ist, damit Astra Trident Topologieorientiert ist.

```
kubectl get nodes -o=jsonpath='{range .items[*]}[{.metadata.name},
{.metadata.labels}]{"\n"}{end}' | grep --color "topology.kubernetes.io"
[node1,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kubernetes.io/arch":"amd64","kubernetes.io/hostname":"node1","kubernetes.io/os":"linux","node-role.kubernetes.io/master":"","topology.kubernetes.io/region":"us-east1","topology.kubernetes.io/zone":"us-east1-a"}]
[node2,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kubernetes.io/arch":"amd64","kubernetes.io/hostname":"node2","kubernetes.io/os":"linux","node-role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-east1","topology.kubernetes.io/zone":"us-east1-b"}]
[node3,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kubernetes.io/arch":"amd64","kubernetes.io/hostname":"node3","kubernetes.io/os":"linux","node-role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-east1","topology.kubernetes.io/zone":"us-east1-c"}]
```

Schritt 1: Erstellen Sie ein Topologieorientiertes Backend

Astra Trident Storage-Back-Ends können für die selektive Bereitstellung von Volumes basierend auf Verfügbarkeitszonen ausgelegt werden. Jedes Backend kann optional mittragen `supportedTopologies` Block, der eine Liste der zu unterstützenden Zonen und Regionen darstellt. Bei `StorageClasses`, die ein solches Backend nutzen, wird ein Volume nur erstellt, wenn es von einer Applikation angefordert wird, die in einer unterstützten Region/Zone geplant ist.

Hier ist eine Beispiel-Backend-Definition:

YAML

```
---
version: 1
storageDriverName: ontap-san
backendName: san-backend-us-east1
managementLIF: 192.168.27.5
svm: iscsi_svm
username: admin
password: password
supportedTopologies:
- topology.kubernetes.io/region: us-east1
  topology.kubernetes.io/zone: us-east1-a
- topology.kubernetes.io/region: us-east1
  topology.kubernetes.io/zone: us-east1-b
```

JSON

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "san-backend-us-east1",
  "managementLIF": "192.168.27.5",
  "svm": "iscsi_svm",
  "username": "admin",
  "password": "password",
  "supportedTopologies": [
    {"topology.kubernetes.io/region": "us-east1",
     "topology.kubernetes.io/zone": "us-east1-a"},
    {"topology.kubernetes.io/region": "us-east1",
     "topology.kubernetes.io/zone": "us-east1-b"}
  ]
}
```



`supportedTopologies` Wird verwendet, um eine Liste von Regionen und Zonen pro Backend bereitzustellen. Diese Regionen und Zonen stellen die Liste der zulässigen Werte dar, die in einer StorageClass bereitgestellt werden können. Bei StorageClasses, die einen Teil der Regionen und Zonen enthalten, die in einem Backend bereitgestellt werden, erstellt Astra Trident ein Volume im Backend.

Sie können definieren `supportedTopologies` Auch pro Storagepool. Das folgende Beispiel zeigt:

```

---
version: 1
storageDriverName: ontap-nas
backendName: nas-backend-us-central1
managementLIF: 172.16.238.5
svm: nfs_svm
username: admin
password: password
supportedTopologies:
- topology.kubernetes.io/region: us-central1
  topology.kubernetes.io/zone: us-central1-a
- topology.kubernetes.io/region: us-central1
  topology.kubernetes.io/zone: us-central1-b
storage:
- labels:
    workload: production
    region: Iowa-DC
    zone: Iowa-DC-A
    supportedTopologies:
    - topology.kubernetes.io/region: us-central1
      topology.kubernetes.io/zone: us-central1-a
- labels:
    workload: dev
    region: Iowa-DC
    zone: Iowa-DC-B
    supportedTopologies:
    - topology.kubernetes.io/region: us-central1
      topology.kubernetes.io/zone: us-central1-b

```

In diesem Beispiel ist der `region` Und `zone` Etiketten stehen für die Position des Speicherpools. `topology.kubernetes.io/region` Und `topology.kubernetes.io/zone` Vorgeben, woher die Speicherpools verbraucht werden können.

Schritt: Definition von StorageClasses, die sich der Topologie bewusst sind

Auf der Grundlage der Topologiebeschriftungen, die den Nodes im Cluster zur Verfügung gestellt werden, können StorageClasses so definiert werden, dass sie Topologieinformationen enthalten. So werden die Storage-Pools festgelegt, die als Kandidaten für PVC-Anfragen dienen, und die Untergruppe der Nodes, die die von Trident bereitgestellten Volumes nutzen können.

Das folgende Beispiel zeigt:

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: netapp-san-us-east1
  provisioner: csi.trident.netapp.io
  volumeBindingMode: WaitForFirstConsumer
  allowedTopologies:
    - matchLabelExpressions:
        - key: topology.kubernetes.io/zone
          values:
            - us-east1-a
            - us-east1-b
        - key: topology.kubernetes.io/region
          values:
            - us-east1
  parameters:
    fsType: "ext4"

```

In der oben angegebenen StorageClass-Definition `volumeBindingMode` ist auf festgelegt `WaitForFirstConsumer`. VES, die mit dieser StorageClass angefordert werden, werden erst dann gehandelt, wenn sie in einem Pod referenziert werden. Und `allowedTopologies` stellt die Zonen und die Region bereit, die verwendet werden sollen. Der `netapp-san-us-east1` StorageClass erstellt VES auf dem `san-backend-us-east1` Back-End oben definiert.

Schritt 3: Erstellen und verwenden Sie ein PVC

Wenn die StorageClass erstellt und einem Backend zugeordnet wird, können Sie jetzt PVCs erstellen.

Siehe Beispiel `spec` unten:

```

---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-san
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 300Mi
  storageClassName: netapp-san-us-east1

```

Das Erstellen eines PVC mithilfe dieses Manifests würde Folgendes zur Folge haben:


```

kubect1 create -f pvc.yaml
persistentvolumeclaim/pvc-san created
kubect1 get pvc
NAME          STATUS      VOLUME      CAPACITY    ACCESS MODES    STORAGECLASS
AGE
pvc-san      Pending
2s
kubect1 describe pvc
Name:          pvc-san
Namespace:     default
StorageClass:  netapp-san-us-east1
Status:        Pending
Volume:
Labels:        <none>
Annotations:   <none>
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:
Access Modes:
VolumeMode:    Filesystem
Mounted By:    <none>
Events:
  Type      Reason              Age    From
  ----      -
Normal    WaitForFirstConsumer  6s     persistentvolume-controller
waiting
for first consumer to be created before binding

```

Verwenden Sie für Trident, ein Volume zu erstellen und es an die PVC zu binden, das in einem Pod verwendet wird. Das folgende Beispiel zeigt:

```

apiVersion: v1
kind: Pod
metadata:
  name: app-pod-1
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
          - matchExpressions:
              - key: topology.kubernetes.io/region
                operator: In
                values:
                  - us-east1
      preferredDuringSchedulingIgnoredDuringExecution:
        - weight: 1
          preference:
            matchExpressions:
              - key: topology.kubernetes.io/zone
                operator: In
                values:
                  - us-east1-a
                  - us-east1-b
  securityContext:
    runAsUser: 1000
    runAsGroup: 3000
    fsGroup: 2000
  volumes:
    - name: voll
      persistentVolumeClaim:
        claimName: pvc-san
  containers:
    - name: sec-ctx-demo
      image: busybox
      command: [ "sh", "-c", "sleep 1h" ]
      volumeMounts:
        - name: voll
          mountPath: /data/demo
      securityContext:
        allowPrivilegeEscalation: false

```

Diese PodSpec beauftragt Kubernetes, den Pod auf Nodes zu planen, die in vorhanden sind `us-east1`. Wählen Sie einen beliebigen Knoten aus, der im vorhanden ist `us-east1-a` Oder `us-east1-b` Zonen:

Siehe die folgende Ausgabe:

```
kubectl get pods -o wide
```

NAME	READY	STATUS	RESTARTS	AGE	IP	NODE
NOMINATED	NODE	READINESS	GATES			
app-pod-1	1/1	Running	0	19s	192.168.25.131	node2
<none>		<none>				

```
kubectl get pvc -o wide
```

NAME	STATUS	VOLUME	CAPACITY
ACCESS MODES	STORAGECLASS	AGE	VOLUMEMODE
pvc-san	Bound	pvc-ecb1e1a0-840c-463b-8b65-b3d033e2e62b	300Mi
RWO		netapp-san-us-east1	48s
		Filesystem	

Aktualisieren Sie Back-Ends, um einzuschließen supportedTopologies

Vorhandene Back-Ends können mit einer Liste von aktualisiert werden `supportedTopologies` Wird verwendet `tridentctl backend update`. Dies wirkt sich nicht auf Volumes aus, die bereits bereitgestellt wurden und nur für nachfolgende VES verwendet werden.

Weitere Informationen

- ["Management von Ressourcen für Container"](#)
- ["NodeSelector"](#)
- ["Affinität und Antiaffinität"](#)
- ["Tönungen und Tolerationen"](#)

Arbeiten Sie mit Snapshots

Kubernetes Volume Snapshots von Persistent Volumes (PVs) ermöglichen zeitpunktgenaue Kopien von Volumes. Sie können einen Snapshot eines mit Astra Trident erstellten Volumes erstellen, einen außerhalb von Astra Trident erstellten Snapshot importieren, ein neues Volume aus einem vorhandenen Snapshot erstellen und Volume-Daten aus Snapshots wiederherstellen.

Überblick

Volume Snapshot wird von unterstützt `ontap-nas`, `ontap-nas-flexgroup`, `ontap-san`, `ontap-san-economy`, `solidfire-san`, `gcp-cvs`, und `azure-netapp-files` Treiber.

Bevor Sie beginnen

Sie benötigen einen externen Snapshot-Controller und benutzerdefinierte Ressourcendefinitionen (CRDs), um mit Snapshots arbeiten zu können. Dies ist die Aufgabe des Kubernetes Orchestrator (z. B. Kubeadm, GKE, OpenShift).

Wenn die Kubernetes-Distribution den Snapshot-Controller und die CRDs nicht enthält, lesen Sie [Stellen Sie einen Volume-Snapshot-Controller bereit](#).



Erstellen Sie keinen Snapshot Controller, wenn Sie On-Demand Volume Snapshots in einer GKE-Umgebung erstellen. GKE verwendet einen integrierten, versteckten Snapshot-Controller.

Erstellen eines Volume-Snapshots

Schritte

1. Erstellen Sie ein VolumeSnapshotClass. Weitere Informationen finden Sie unter ["VolumeSnapshotKlasse"](#).
 - Der driver Verweist auf den Astra Trident CSI-Treiber.
 - deletionPolicy Kann sein Delete Oder Retain. Wenn eingestellt auf Retain, Der zugrunde liegende physische Snapshot auf dem Storage-Cluster wird auch dann beibehalten, wenn der VolumeSnapshot Objekt wurde gelöscht.

Beispiel

```
cat snap-sc.yaml
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: csi-snapclass
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

2. Erstellen Sie einen Snapshot einer vorhandenen PVC.

Beispiele

- In diesem Beispiel wird ein Snapshot eines vorhandenen PVC erstellt.

```
cat snap.yaml
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: pvc1-snap
spec:
  volumeSnapshotClassName: csi-snapclass
  source:
    persistentVolumeClaimName: pvc1
```

- In diesem Beispiel wird ein Volume-Snapshot-Objekt für eine PVC mit dem Namen erstellt pvc1 Der Name des Snapshots lautet pvc1-snap. Ein VolumeSnapshot ist analog zu einem PVC und einem zugeordnet VolumeSnapshotContent Objekt, das den tatsächlichen Snapshot darstellt.

```
kubectl create -f snap.yaml
volumesnapshot.snapshot.storage.k8s.io/pvc1-snap created

kubectl get volumesnapshots
NAME                AGE
pvc1-snap           50s
```

- Sie können den identifizieren `VolumeSnapshotContent` Objekt für das `pvc1-snap` `VolumeSnapshot` wird beschrieben. Der `Snapshot Content Name` Identifiziert das `VolumeSnapshotContent`-Objekt, das diesen Snapshot bereitstellt. Der `Ready To Use` Parameter gibt an, dass der Snapshot zum Erstellen einer neuen PVC verwendet werden kann.

```
kubectl describe volumesnapshots pvc1-snap
Name:          pvc1-snap
Namespace:     default
.
.
.
Spec:
  Snapshot Class Name:    pvc1-snap
  Snapshot Content Name:  snapcontent-e8d8a0ca-9826-11e9-9807-
525400f3f660
  Source:
    API Group:
    Kind:      PersistentVolumeClaim
    Name:      pvc1
Status:
  Creation Time:  2019-06-26T15:27:29Z
  Ready To Use:   true
  Restore Size:   3Gi
.
.
```

Erstellen Sie eine PVC aus einem Volume-Snapshot

Verwenden Sie können `dataSource` So erstellen Sie eine PVC mit einem `VolumeSnapshot` namens `<pvc-name>` Als Quelle der Daten. Nachdem die PVC erstellt wurde, kann sie an einem Pod befestigt und wie jedes andere PVC verwendet werden.



Die PVC wird im selben Backend wie das Quell-Volume erstellt. Siehe "[KB: Die Erstellung einer PVC aus einem Trident PVC-Snapshot kann nicht in einem alternativen Backend erstellt werden](#)".

Im folgenden Beispiel wird die PVC mit erstellt `pvc1-snap` Als Datenquelle speichern.

```
cat pvc-from-snap.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: golden
  resources:
    requests:
      storage: 3Gi
  dataSource:
    name: pvcl-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

Importieren Sie einen Volume-Snapshot

Astra Trident unterstützt das ["Vorab bereitgestellter Snapshot-Prozess von Kubernetes"](#). Damit der Clusteradministrator einen erstellen kann `VolumeSnapshotContent` Objekt- und Import von Snapshots, die außerhalb von Astra Trident erstellt wurden.

Bevor Sie beginnen

Astra Trident muss das übergeordnete Volume des Snapshots erstellt oder importiert haben.

Schritte

1. **Cluster admin:** Erstellen Sie eine `VolumeSnapshotContent` Objekt, das auf den Back-End-Snapshot verweist. Dadurch wird der Snapshot Workflow in Astra Trident gestartet.
 - Geben Sie den Namen des Back-End-Snapshots in an annotations Als `trident.netapp.io/internalSnapshotName: <"backend-snapshot-name">`.
 - Angeben `<name-of-parent-volume-in-trident>/<volume-snapshot-content-name>` In `snapshotHandle`. Dies ist die einzige Information, die Astra Trident vom externen Snapshot in zur Verfügung gestellt wird `ListSnapshots` Anruf.



Der `<volumeSnapshotContentName>` Aufgrund von Einschränkungen bei der CR-Benennung kann der Name des Back-End-Snapshots nicht immer übereinstimmen.

Beispiel

Im folgenden Beispiel wird ein erstellt `VolumeSnapshotContent` Objekt, das auf Back-End-Snapshot verweist `snap-01`.

```

apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotContent
metadata:
  name: import-snap-content
  annotations:
    trident.netapp.io/internalSnapshotName: "snap-01" # This is the
name of the snapshot on the backend
spec:
  deletionPolicy: Retain
  driver: csi.trident.netapp.io
  source:
    snapshotHandle: pvc-f71223b5-23b9-4235-bbfe-e269ac7b84b0/import-
snap-content # <import PV name or source PV name>/<volume-snapshot-
content-name>

```

2. **Cluster admin:** Erstellen Sie das VolumeSnapshot CR, der auf den verweist VolumeSnapshotContent Objekt: Dadurch wird der Zugriff auf die Verwendung des angefordert VolumeSnapshot In einem bestimmten Namespace.

Beispiel

Im folgenden Beispiel wird ein erstellt VolumeSnapshot CR benannt import-snap Die auf die verweisen VolumeSnapshotContent Genannt import-snap-content.

```

apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: import-snap
spec:
  # volumeSnapshotClassName: csi-snapclass (not required for pre-
provisioned or imported snapshots)
  source:
    volumeSnapshotContentName: import-snap-content

```

3. **Interne Verarbeitung (keine Aktion erforderlich):** der externe Snapshotter erkennt das neu erstellte VolumeSnapshotContent Und führt das aus ListSnapshots Anruf. Astra Trident erstellt die TridentSnapshot.
 - Der externe Schnapper legt den fest VolumeSnapshotContent Bis readyToUse Und das VolumeSnapshot Bis true.
 - Trident kehrt zurück readyToUse=true.
4. **Jeder Benutzer:** Erstellen Sie eine PersistentVolumeClaim Um auf das neue zu verweisen VolumeSnapshot, Wo der spec.dataSource (Oder spec.dataSourceRef) Name ist der VolumeSnapshot Name:

Beispiel

Im folgenden Beispiel wird eine PVC erstellt, die auf den verweist VolumeSnapshot Genannt import-snap.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: simple-sc
  resources:
    requests:
      storage: 1Gi
  dataSource:
    name: import-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

Stellen Sie Volume-Daten mithilfe von Snapshots wieder her

Das Snapshot-Verzeichnis ist standardmäßig ausgeblendet, um die maximale Kompatibilität von Volumes zu ermöglichen, die über bereitgestellt werden ontap-nas Und ontap-nas-economy Treiber. Aktivieren Sie die .snapshot Verzeichnis, um Daten von Snapshots direkt wiederherzustellen.

Verwenden Sie die ONTAP-CLI zur Wiederherstellung eines Volume-Snapshots, um einen in einem früheren Snapshot aufgezeichneten Zustand wiederherzustellen.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```



Wenn Sie eine Snapshot-Kopie wiederherstellen, wird die vorhandene Volume-Konfiguration überschrieben. Änderungen an den Volume-Daten nach der Erstellung der Snapshot Kopie gehen verloren.

Das Snapshot-Verzeichnis ist standardmäßig ausgeblendet, um die maximale Kompatibilität von Volumes zu ermöglichen, die über bereitgestellt werden ontap-nas Und ontap-nas-economy Treiber. Aktivieren Sie die .snapshot Verzeichnis, um Daten von Snapshots direkt wiederherzustellen.



Wenn Sie eine Snapshot-Kopie wiederherstellen, wird die vorhandene Volume-Konfiguration überschrieben. Änderungen an den Volume-Daten nach der Erstellung der Snapshot Kopie gehen verloren.

In-Place-Volume-Wiederherstellung aus einem Snapshot

Astra Control Provisioner ermöglicht mithilfe des (TASR) CR eine schnelle Wiederherstellung von in-Place-

Volumes aus einem Snapshot `TridentActionSnapshotRestore` . Dieser CR fungiert als eine zwingend notwendige Kubernetes-Aktion und bleibt nach Abschluss des Vorgangs nicht erhalten.

Astra Control Provisioner unterstützt Snapshot-Wiederherstellung auf dem `ontap-san`, `ontap-san-economy` `ontap-nas` `ontap-nas-flexgroup` `azure-netapp-files`, `gcp-cvs` `solidfire-san` und Fahrer.

Bevor Sie beginnen

Sie müssen über einen gebundenen PVC-Snapshot und einen verfügbaren Volume-Snapshot verfügen.

- Vergewissern Sie sich, dass der PVC-Status gebunden ist.

```
kubectl get pvc
```

- Überprüfen Sie, ob der Volume-Snapshot einsatzbereit ist.

```
kubectl get vs
```

Schritte

1. Erstellen Sie den TASR CR. In diesem Beispiel wird ein CR für PVC und Volume-Snapshot erstellt `pvc1` `pvc1-snapshot`.

```
cat tasr-pvc1-snapshot.yaml

apiVersion: v1
kind: TridentActionSnapshotRestore
metadata:
  name: this-doesnt-matter
  namespace: trident
spec:
  pvcName: pvc1
  volumeSnapshotName: pvc1-snapshot
```

2. Wenden Sie den CR an, um ihn aus dem Snapshot wiederherzustellen. Dieses Beispiel wird aus Snapshot wiederhergestellt `pvc1`.

```
kubectl create -f tasr-pvc1-snapshot.yaml

tridentactionsnapshotrestore.trident.netapp.io/this-doesnt-matter
created
```

Ergebnisse

Mit Astra Control Provisioner werden die Daten aus dem Snapshot wiederhergestellt. Sie können den Status

der Snapshot-Wiederherstellung überprüfen.

```
kubectl get tasr -o yaml

apiVersion: v1
items:
- apiVersion: trident.netapp.io/v1
  kind: TridentActionSnapshotRestore
  metadata:
    creationTimestamp: "2023-04-14T00:20:33Z"
    generation: 3
    name: this-doesnt-matter
    namespace: trident
    resourceVersion: "3453847"
    uid: <uid>
  spec:
    pvcName: pvc1
    volumeSnapshotName: pvc1-snapshot
  status:
    startTime: "2023-04-14T00:20:34Z"
    completionTime: "2023-04-14T00:20:37Z"
    state: Succeeded
kind: List
metadata:
  resourceVersion: ""
```



- In den meisten Fällen versucht die Astra Control Provisioner bei einem Ausfall nicht automatisch einen weiteren Vorgang auszuführen. Sie müssen den Vorgang erneut ausführen.
- Kubernetes-Benutzer ohne Administratorzugriff müssen möglicherweise vom Administrator zum Erstellen eines TASR CR in ihrem Applikations-Namespace erhalten.

Verwenden Sie die ONTAP-CLI zur Wiederherstellung eines Volume-Snapshots, um einen in einem früheren Snapshot aufgezeichneten Zustand wiederherzustellen.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

Löschen Sie ein PV mit den zugehörigen Snapshots

Wenn Sie ein persistentes Volume mit zugeordneten Snapshots löschen, wird das entsprechende Trident-Volume in einen „Löschzustand“ aktualisiert. Entfernen Sie die Volume Snapshots, um das Astra Trident Volume zu löschen.

Stellen Sie einen Volume-Snapshot-Controller bereit

Wenn Ihre Kubernetes-Distribution den Snapshot-Controller und CRDs nicht enthält, können Sie sie wie folgt bereitstellen.

Schritte

1. Erstellen von Volume Snapshot-CRDs.

```
cat snapshot-setup.sh
#!/bin/bash
# Create volume snapshot CRDs
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshotclasses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshotcontents.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshots.yaml
```

2. Erstellen Sie den Snapshot-Controller.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/deploy/kubernetes/snapshot-controller/setup-snapshot-controller.yaml
```



Öffnen Sie bei Bedarf `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` Und Aktualisierung namespace In Ihren Namespace.

Weiterführende Links

- ["Volume Snapshots"](#)
- ["VolumeSnapshotKlasse"](#)

Copyright-Informationen

Copyright © 2026 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, BEISPIELHAFT 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 IRGEND EINER 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.