Verwenden Sie Trident

Trident

NetApp
January 14, 2026

Inhalt

Verwenden Sie 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-, NVMe/TCP- oder FC-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 Red hat Enterprise Linux CoreOS (RHCOS) sind die Tools standardmäßig installiert.

NFS Tools

"Installieren Sie die NFS Tools" Wenn Sie: `ontap-nas`, , , `ontap-nas-economy` `ontap-nas-flexgroup` , `azure-netapp-files` `gcp-cvs` .

ISCSI-Tools

"Installieren Sie die iSCSI-Tools" Wenn Sie: `ontap-san`, , `ontap-san-economy` `solidfire-san` .

NVMe-Tools

"Installation der NVMe Tools" Falls Sie das Protokoll Nonvolatile Memory Express (NVMe) over TCP (NVMe/TCP) verwenden `ontap-san`.

> NetApp empfiehlt für NVMe/TCP ONTAP 9.12 oder höher.

SCSI-über-FC-Tools

finden Weitere Informationen zur Konfiguration von FC- und FC-NVMe-SAN-Hosts unter"Möglichkeiten zur Konfiguration von FC- FC-NVMe SAN-Hosts"Sie.

"Installieren Sie die FC Tools" Wenn Sie mit sanType `fcp` (SCSI über FC) verwenden `ontap-san`.

Zu berücksichtigende Punkte: * SCSI über FC wird in OpenShift- und KubeVirt-Umgebungen unterstützt. * SCSI über FC wird auf Docker nicht unterstützt. * ISCSI Selbstheilung gilt nicht für SCSI über FC.

Ermittlung des Node-Service

Trident versucht automatisch zu erkennen, ob auf dem Node iSCSI- oder NFS-Services ausgeführt werden können.

> 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

Trident erstellt Ereignisse für den Node, um die erkannten Services zu identifizieren. 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

Trident erkennt Dienste, die für jeden Knoten auf dem Trident-Knoten CR aktiviert sind. 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

Trident kann automatisch eine iSCSI-Sitzung einrichten, LUNs scannen, Multipath-Geräte erkennen, formatieren und in einen Pod einbinden.

ISCSI-Funktionen zur Selbstreparatur

Bei ONTAP Systemen führt Trident die iSCSI-Selbstreparatur alle fünf Minuten aus, um folgende Vorteile zu nutzen:

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. Trident bestimmt die Reparaturprioritäten und den Zeitpunkt, an dem Reparaturen vorbeugen 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` Container auf dem jeweiligen Demonset-Pod. Um Protokolle anzuzeigen, müssen Sie während der Trident-Installation auf „true“ gesetzt haben `debug`.

Trident iSCSI-Funktionen zur Selbstheilung verhindern Folgendes:

- Veraltete oder ungesunde iSCSI-Sitzungen, die nach einem Problem mit der Netzwerkverbindung auftreten können Im Falle einer veralteten Sitzung wartet Trident sieben Minuten, bevor er sich abmeldet, um die Verbindung zu 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

Punkte, die Sie vor dem Upgrade von Trident beachten sollten

- Wenn nur Initiatorgruppen pro Node (eingeführt in 23.04+) verwendet werden, initiiert iSCSI Self-Healing SCSI-Rescans für alle Geräte im SCSI-Bus.
- Wenn nur Back-End-scoped-Initiatorgruppen (veraltet ab 23.04) verwendet werden, initiiert iSCSI-Selbstreparatur SCSI-Rescans für exakte LUN-IDs im SCSI-Bus.
- Wenn eine Kombination von Initiatorgruppen pro Node und mit Back-End-Scoped-Initiatorgruppen verwendet wird, initiiert iSCSI Self-Healing SCSI-Rescans für exakte LUN-IDs im SCSI-Bus.

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**.
- Wenn Sie RHCOS Version 4.5 oder höher oder eine andere RHEL-kompatible Linux-Distribution mit dem Treiber und Element OS 12.5 oder früher verwenden `solidfire-san`, 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 verfügbar.

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

- Geben Sie bei der Verwendung von Worker-Nodes, die RHEL/Red hat Enterprise Linux CoreOS (RHCOS) mit iSCSI-PVs ausführen, die Option mountOption in der StorageClass an `discard`, um Inline-Speicherplatz zurückzunehmen. Siehe "Red hat Dokumentation".

RHEL 8 ODER HÖHER

1. Installieren Sie die folgenden Systempakete:

```
sudo yum install -y lsscsi iscsi-initiator-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. Multipathing aktivieren:

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

> Stellen Sie sicher, dass `/etc/multipath.conf` enthält `find_multipaths no` unter `defaults`.

4. Stellen Sie sicher, dass `iscsid` und `multipathd` ausgeführt werden:

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

5. 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
```

> Stellen Sie sicher, dass `/etc/multipath.conf` enthält `find_multipaths no` unter `defaults`.

5. Stellen Sie sicher, dass `open-iscsi` und `multipath-tools` aktiviert sind und ausgeführt werden:

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

> Für Ubuntu 18.04 müssen Sie Zielports mit ermitteln `iscsiadm`, bevor der iSCSI-Daemon gestartet `open-iscsi` wird. Sie können den Dienst auch so ändern `iscsi`, dass er automatisch gestartet `iscsid` wird.

Konfigurieren oder deaktivieren Sie die iSCSI-Selbstheilung

Sie können die folgenden Trident iSCSI-Selbstreparatureinstellungen 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 die `iscsiSelfHealingInterval` Parameter und `iscsiSelfHealingWaitTime` während der Helm-Installation oder der Helm-Aktualisierung.

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-100.2502.0.tgz --set
iscsiSelfHealingInterval=3m0s --set iscsiSelfHealingWaitTime=6m0s -n
trident
```

Tridentctl

Um iSCSI-Selbstreparatureinstellungen zu konfigurieren oder zu ändern, übergeben Sie die `iscsi-self-healing-interval` Parameter und `iscsi-self-healing-wait-time` 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
```

Trident ändert den `ctrl_device_tmo` Wert, um zu gewährleisten, dass NVMe bei einem Ausfall nicht auf dem Pfad aufgibt. Ändern Sie diese Einstellung nicht.

SCSI über FC Volumes

Jetzt kann das Fibre Channel-Protokoll (FC) mit Trident verwendet werden, um Storage-Ressourcen auf ONTAP Systemen bereitzustellen und zu managen.

Voraussetzungen

Konfigurieren Sie die erforderlichen Netzwerk- und Node-Einstellungen für FC.

Netzwerkeinstellungen

1. Erhalten Sie den WWPN der Zielschnittstellen. Weitere Informationen finden Sie unter "Netzwerkschnittstelle wird angezeigt" .
2. Abrufen der WWPN für die Schnittstellen auf Initiator (Host).

   Weitere Informationen finden Sie in den entsprechenden Dienstprogrammen des Host-Betriebssystems.
3. Konfigurieren Sie das Zoning auf dem FC-Switch mithilfe von WWPNs des Hosts und Ziels.

   Weitere Informationen finden Sie in der Dokumentation des jeweiligen Switch-Anbieters.

   Details finden Sie in der folgenden ONTAP Dokumentation:

   - "Übersicht über Fibre Channel und FCoE Zoning"

- "Möglichkeiten zur Konfiguration von FC- FC-NVMe SAN-Hosts"

Installieren Sie die FC Tools

Installieren Sie die FC-Tools unter Verwendung der Befehle für Ihr Betriebssystem.

- Geben Sie bei der Verwendung von Worker-Nodes, die RHEL/Red hat Enterprise Linux CoreOS (RHCOS) mit FC PVs ausführen, die Option mountOption in der StorageClass an `discard`, um Inline-Speicherplatz zurückzunehmen. Siehe "Red hat Dokumentation".

RHEL 8 ODER HÖHER

1. Installieren Sie die folgenden Systempakete:

```
sudo yum install -y lsscsi device-mapper-multipath
```

2. Multipathing aktivieren:

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

> Stellen Sie sicher, dass `/etc/multipath.conf` enthält `find_multipaths no` unter `defaults`.

3. Stellen Sie sicher, dass `multipathd` Folgendes ausgeführt wird:

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

Ubuntu

1. Installieren Sie die folgenden Systempakete:

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

2. 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
```

> Stellen Sie sicher, dass `/etc/multipath.conf` enthält `find_multipaths no` unter `defaults`.

3. Stellen Sie sicher, dass `multipath-tools` aktiviert und ausgeführt wird:

```
sudo systemctl status multipath-tools
```

Konfiguration und Management von Back-Ends

Back-Ends konfigurieren

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

Trident bietet automatisch Back-Ends-Storage-Pools an, 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"
- "Google Cloud NetApp Volumes-Back-End konfigurieren"
- "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"
- "Verwenden Sie Trident mit Amazon FSX für NetApp ONTAP"

Azure NetApp Dateien

Konfigurieren Sie ein Azure NetApp Files-Backend

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

Azure NetApp Files-Treiberdetails

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

Treiber	Protokoll	VolumeMo dus	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, die kleiner als 50 gib sind. Trident erstellt automatisch 50-gib-Volumes, wenn ein kleineres Volume angefordert wird.
- Trident unterstützt nur SMB Volumes, die in Pods gemountet sind, die nur auf Windows Nodes ausgeführt werden.

Verwaltete Identitäten für AKS

Trident unterstützt "Verwaltete Identitäten"Cluster mit Azure Kubernetes Services. Um die Vorteile einer optimierten Verwaltung von Anmeldeinformationen zu nutzen, die von verwalteten Identitäten angeboten wird, müssen Sie über Folgendes verfügen:

- Implementierung eines Kubernetes Clusters mit AKS
- Verwaltete Identitäten, die auf dem AKS kubernetes-Cluster konfiguriert sind
- Trident installiert, die die zu spezifizieren `"Azure"` enthält `cloudProvider` .

Betreiber von Trident

Um Trident mit dem Trident-Operator zu installieren, bearbeiten Sie, `tridentorchestrator_cr.yaml` um auf `"Azure"` einzustellen `cloudProvider`. 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 Trident-Sets mit der Umgebungsvariable auf Azure `$CP` installiert `cloudProvider`:

```
helm install trident trident-operator-100.2502.0.tgz --create
-namespace --namespace <trident-namespace> --set cloudProvider=$CP
```

<code>-Datei findet </code>

Das folgende Beispiel installiert Trident und setzt das `cloudProvider` Flag auf `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:

- Implementierung eines Kubernetes Clusters mit AKS
- Workload-Identität und oidc-Issuer, die auf dem AKS Kubernetes-Cluster konfiguriert sind
- Trident wurde installiert, das die zum Angeben `"Azure"` und `cloudIdentity` Angeben der Workload-Identität enthält `cloudProvider`

Betreiber von Trident

Um Trident mithilfe des Trident-Operators zu installieren, bearbeiten Sie die `tridentorchestrator_cr.yaml` Einstellung `cloudProvider` auf und setzen Sie `cloudIdentity` `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx` sie auf `"Azure"` .

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-xxxxxxxxxxx' # Edit
```

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-xxxxxxxxxxx'"
```

Das folgende Beispiel installiert Trident und setzt `cloudProvider` auf Azure unter Verwendung der Umgebungsvariable `$CP` und setzt die `cloudIdentity` unter Verwendung der Umgebungsvariable `$CI`:

```
helm install trident trident-operator-100.2502.0.tgz --set cloudProvider=$CP --set cloudIdentity="$CI"
```

<code>-Datei findet </code>

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-xxxxxxxxxxx"
```

Das folgende Beispiel installiert Trident und setzt das `cloud-provider` Flag auf `$CP`, und `cloud-identity` auf `$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 Backend zu konfigurieren und zu verwenden "Azure NetApp Dateien", benötigen Sie 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` Von einem 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-Dienst. Die App-Registrierung sollte Folgendes verwenden:
  - Der Eigentümer oder die Rolle des Beitragenden "Vordefiniert von Azure".
  - A "Benutzerdefinierte Beitragsrolle" auf Abonnementebene (`assignableScopes`) mit den folgenden Berechtigungen, die auf das beschränkt sind, was Trident benötigt. Nach dem Erstellen der benutzerdefinierten Rolle, "Weisen Sie die Rolle über das Azure-Portal zu".

Rolle für benutzerdefinierte Mitwirkende

```
{
  "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/MountTarge
ts/read",
          "Microsoft.Network/virtualNetworks/read",
          "Microsoft.Network/virtualNetworks/subnets/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/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": []
        }
      ]
    }
}
```

- Der Azure `location`, der mindestens einen enthält "Delegiertes Subnetz". Ab Trident 22.01 ist der `location` Parameter ein Pflichtfeld auf der obersten Ebene der Backend-Konfigurationsdatei. In virtuellen Pools angegebene Standortwerte werden ignoriert.
- Um zu verwenden `Cloud Identity`, erhalten Sie 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-xxxxxxxxxxx`.

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 2022 ausgeführt wird. Trident unterstützt nur SMB Volumes, die in Pods gemountet sind, die nur auf Windows Nodes ausgeführt werden.
- Mindestens ein Trident-Schlüssel, der Ihre Active Directory-Anmeldeinformationen enthält, damit Azure NetApp Files sich bei Active Directory authentifizieren kann. So generieren Sie ein Geheimnis `smbcreds`:

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

- Ein CSI-Proxy, der als Windows-Dienst konfiguriert ist. Informationen zum Konfigurieren `csi-proxy` von finden Sie unter "GitHub: CSI-Proxy" oder "GitHub: CSI Proxy für Windows" für Kubernetes-Nodes, die unter 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

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

Trident unterstützt keine manuellen QoS-Kapazitätspools.

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 einer App-Registrierung Optional, wenn verwaltete Identitäten oder Cloud-Identität auf einem AKS-Cluster verwendet wird.	
`clientID`	Die Client-ID 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.	
`serviceLevel`	Einer von `Standard`, , `Premium` oder `Ultra`	„“ (zufällig)
`location`	Name des Azure-Standorts, an dem die neuen Volumes erstellt werden Optional, wenn verwaltete Identitäten auf einem AKS-Cluster aktiviert sind	

Parameter	Beschreibung	Standard
`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 Subnetzes, an das delegiert wurde `Microsoft.Netapp/volumes`	„“
`networkFeatures`	Satz von vnet-Features für ein Volume, kann oder `Standard` sein `Basic`. Netzwerkfunktionen sind nicht in allen Regionen verfügbar und müssen möglicherweise in einem Abonnement aktiviert werden. Wenn die `networkFeatures`Funktion nicht aktiviert ist, schlägt die Volume-Bereitstellung fehl.	„“
`nfsMountOptions`	Engmaschige Kontrolle der NFS-Mount-Optionen Für SMB Volumes ignoriert. Um Volumes mit NFS-Version 4.1 zu mounten, fügen Sie in die Liste mit kommagetrennten Mount-Optionen ein `nfsvers=4`, um NFS v4.1 auszuwählen. Mount-Optionen, die in einer Storage-Klassen-Definition festgelegt sind, überschreiben Mount-Optionen, die in der Backend-Konfiguration festgelegt sind.	„Nfsvers=3“
`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, "method": true, "discovery": true}`. Verwenden Sie dies nur, wenn Sie Fehler beheben und einen detaillierten Log Dump benötigen.	Null

Parameter	Beschreibung	Standard
`nasType`	Konfiguration der Erstellung von NFS- oder SMB-Volumes Optionen sind `nfs`, `smb` oder Null. Einstellung auf null setzt standardmäßig auf NFS-Volumes.	`nfs`
`supportedTopologies`	Stellt eine Liste von Regionen und Zonen dar, die von diesem Backend unterstützt werden. Weitere Informationen finden Sie unter "Verwenden Sie die CSI-Topologie".	

Weitere Informationen zu 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, ist es wahrscheinlich, dass Ihre App-Registrierung nicht über die erforderlichen Berechtigungen und Ressourcen (Subnetz, virtuelles Netzwerk, Kapazitätspool) verfügt. Wenn Debug aktiviert ist, protokolliert Trident die beim Erstellen des Backends erkannten Azure-Ressourcen. Überprüfen Sie, ob eine geeignete Rolle verwendet wird.

Die Werte für `resourceGroups`, , , `netappAccounts` `capacityPools` , `virtualNetwork` und `subnet` können 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.

Die `resourceGroups` Werte , `netappAccounts` und `capacityPools` sind Filter, die die ermittelten Ressourcen auf die Ressourcen beschränken, die für dieses Speicher-Backend verfügbar sind und in jeder 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. Weitere Informationen finden Sie unter Beispielkonfigurationen .

Parameter	Beschreibung	Standard
`exportRule`	Exportregeln für neue Volumes `exportRule` Muss eine kommagetrennte Liste einer beliebigen Kombination von IPv4-Adressen oder IPv4-Subnetzen in CIDR-Notation sein. Für SMB Volumes ignoriert.	„0.0.0.0/0“
`snapshotDir`	Steuert die Sichtbarkeit des .Snapshot-Verzeichnisses	„Wahr“ für NFSv4 „falsch“ für NFSv3
`size`	Die Standardgröße der neuen Volumes	„100G“
`unixPermissions`	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 Trident alle NetApp-Konten, Kapazitätspools und an Azure NetApp Files delegierte Subnetze am konfigurierten Standort und platziert neue Volumes zufällig in einem dieser Pools und Subnetze. Da `nasType` nicht angegeben ist, gilt der `nfs` Standard 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.

```
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  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 optional sind, wenn Sie eine Cloud-Identität verwenden.

```
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

Diese Backend-Konfiguration platziert Volumes an Azure `eastus` in einem `Ultra` Kapazitäts-Pool. 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-Etiketten wurden verwendet, um die Pools anhandzu 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
```

Konfiguration unterstützter Topologien

Trident erleichtert die Bereitstellung von Volumes für Workloads, basierend auf Regionen und Verfügbarkeitszonen. Der `supportedTopologies` Block in dieser Backend-Konfiguration dient zur Bereitstellung einer Liste von Regionen und Zonen pro Backend. Die hier angegebenen Region- und Zonenwerte müssen mit den Region- und Zonenwerten der Beschriftungen auf jedem Kubernetes-Cluster-Node übereinstimmen. Diese Regionen und Zonen stellen die Liste der zulässigen Werte dar, die in einer Lagerklasse bereitgestellt werden können. Für Storage-Klassen, die eine Teilmenge der Regionen und Zonen enthalten, die in einem Back-End bereitgestellt werden, erstellt Trident Volumes in der genannten Region und Zone. Weitere Informationen finden Sie unter "Verwenden Sie die CSI-Topologie".

```
---
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
supportedTopologies:
  - topology.kubernetes.io/region: eastus
    topology.kubernetes.io/zone: eastus-1
  - topology.kubernetes.io/region: eastus
    topology.kubernetes.io/zone: eastus-2
```

Definitionen der Storage-Klassen

Die folgenden `StorageClass` Definitionen beziehen sich auf die Speicherpools oben.

Beispieldefinitionen mit `parameter.selector` Feld

Mit `parameter.selector` können Sie für jeden virtuellen Pool angeben `StorageClass`, der zum Hosten eines Volumes verwendet 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

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

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.

Google Cloud NetApp Volumes

Google Cloud NetApp Volumes-Back-End konfigurieren

Sie können jetzt Google Cloud NetApp Volumes als Backend für Trident konfigurieren. NFS- und SMB-Volumes können über ein Google Cloud NetApp-Back-End angebunden werden.

Treiberdetails zu Google Cloud NetApp Volumes

Trident stellt den `google-cloud-netapp-volumes` Treiber für die Kommunikation mit dem Cluster bereit. Unterstützte Zugriffsmodi sind: *ReadWriteOnce* (RWO), *ReadOnly Many* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Treiber	Protokoll	VolumeModus	Unterstützte Zugriffsmodi	Unterstützte Filesysteme
`google-cloud-netapp-volumes`	NFS SMB	Dateisystem	RWO, ROX, RWX, RWOP	`nfs`, `smb`

Cloud-Identität für GKE

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

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

- Ein mit GKE implementierter Kubernetes-Cluster.
- Auf dem GKE-Cluster konfigurierte Workload-Identität und auf den Node-Pools konfigurierten GKE-Metadatenserver.

- Ein GCP-Service-Konto mit der Google Cloud NetApp Volumes Admin-Rolle (Rollen/NetApp.admin) oder einer benutzerdefinierten Rolle.
- Trident installiert, das den CloudProvider enthält, um „GCP“ und CloudIdentity anzugeben, die das neue GCP-Dienstkonto angeben. Ein Beispiel ist unten angegeben.

Betreiber von Trident

Um Trident mithilfe des Trident-Operators zu installieren, bearbeiten Sie die `tridentorchestrator_cr.yaml` Einstellung `cloudProvider` auf und setzen Sie `cloudIdentity` `iam.gke.io/gcp-service-account: cloudvolumes-admin-sa@mygcpproject.iam.gserviceaccount.com` sie auf `"GCP"` .

Beispiel:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "GCP"
  cloudIdentity: 'iam.gke.io/gcp-service-account: cloudvolumes-
admin-sa@mygcpproject.iam.gserviceaccount.com'
```

Helm

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

```
export CP="GCP"
export ANNOTATION="'iam.gke.io/gcp-service-account: cloudvolumes-admin-
sa@mygcpproject.iam.gserviceaccount.com'"
```

Das folgende Beispiel installiert Trident und setzt `cloudProvider` auf GCP unter Verwendung der Umgebungsvariable `$CP` und setzt die `cloudIdentity` unter Verwendung der Umgebungsvariable `$ANNOTATION`:

```
helm install trident trident-operator-100.2502.0.tgz --set
cloudProvider=$CP --set cloudIdentity="$ANNOTATION"
```

<code>-Datei findet </code>

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

```
export CP="GCP"
export ANNOTATION="'iam.gke.io/gcp-service-account: cloudvolumes-admin-
sa@mygcpproject.iam.gserviceaccount.com'"
```

Das folgende Beispiel installiert Trident und setzt das `cloud-provider` Flag auf `$CP`, und `cloud-identity` auf `$ANNOTATION`:

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

Bereiten Sie sich auf die Konfiguration eines Google Cloud NetApp Volumes-Back-End vor

Bevor Sie Ihr Google Cloud NetApp Volumes-Backend konfigurieren können, müssen Sie sicherstellen, dass die folgenden Anforderungen erfüllt sind.

Voraussetzungen für NFS Volumes

Wenn Sie Google Cloud NetApp Volumes zum ersten Mal oder an einem neuen Speicherort verwenden, ist eine Erstkonfiguration erforderlich, um Google Cloud NetApp Volumes einzurichten und ein NFS-Volume zu erstellen. Siehe "Bevor Sie beginnen".

Stellen Sie vor der Konfiguration des Google Cloud NetApp Volumes-Back-End sicher, dass folgende Voraussetzungen bestehen:

- Ein Google Cloud Konto, das mit dem Google Cloud NetApp Volumes Service konfiguriert ist. Siehe "Google Cloud NetApp Volumes".
- Projektnummer Ihres Google Cloud-Kontos. Siehe "Projekte identifizieren".
- Ein Google Cloud-Service-Konto mit der Rolle NetApp Volumes Admin (`roles/netapp.admin`). Siehe "Rollen und Berechtigungen für Identitäts- und Zugriffsmanagement".
- API-Schlüsseldatei für Ihr GCNV-Konto. Siehe "Erstellen eines Service-Kontokonschlüssels"
- Ein Speicherpool. Siehe "Überblick über Speicherpools".

Weitere Informationen zum Einrichten des Zugriffs auf Google Cloud NetApp Volumes finden Sie unter "Zugriff auf Google Cloud NetApp Volumes einrichten".

Konfigurationsoptionen und Beispiele für die Backend-Konfiguration von Google Cloud NetApp Volumes

Informieren Sie sich über die Back-End-Konfigurationsoptionen für Google Cloud NetApp Volumes und sehen Sie sich Konfigurationsbeispiele an.

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	Der Wert von `storageDriverName` muss als „google-Cloud-netapp-Volumes“ angegeben werden.

Parameter	Beschreibung	Standard
`backendName`	(Optional) Benutzerdefinierter Name des Speicher-Backends	Treibername + „_“ + Teil des API-Schlüssels
`storagePools`	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.	
`location`	Die Google Cloud-Umgebung, an der Trident GCNV Volumes erstellt. Bei der Erstellung regionsübergreifender Kubernetes-Cluster können in A erstellte Volumes `location` für Workloads verwendet werden, die auf Nodes in mehreren Google Cloud-Regionen geplant sind. Der regionale Verkehr verursacht zusätzliche Kosten.	
`apiKey`	API-Schlüssel für das Google Cloud-Servicekonto mit der `netapp.admin` Rolle. Er enthält den JSON-formatierten Inhalt der privaten Schlüsseldatei eines Google Cloud-Dienstkontos (wortgetreu in die Back-End-Konfigurationsdatei kopiert). Das `apiKey` muss Schlüssel-Wert-Paare für die folgenden Schlüssel enthalten: `type`, `project_id`, `client_email` `client_id` `auth_uri`, , , `token_uri` `auth_provider_x509_cert_url`, und `client_x509_cert_url`.	
`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`	Service-Level eines Storage-Pools und seiner Volumes. Die Werte sind `flex`, , `standard` `premium`oder `extreme`.	
`network`	Für GCNV-Volumes verwendetes Google Cloud-Netzwerk	
`debugTraceFlags`	Fehler-Flags bei der Fehlerbehebung beheben. Beispiel, `{"api":false, "method":true}`. Verwenden Sie dies nur, wenn Sie Fehler beheben und einen detaillierten Log Dump benötigen.	Null
`nasType`	Konfiguration der Erstellung von NFS- oder SMB-Volumes Optionen sind `nfs`, `smb` oder Null. Einstellung auf null setzt standardmäßig auf NFS-Volumes.	`nfs`

Parameter	Beschreibung	Standard
`supportedTopologies`	Stellt eine Liste von Regionen und Zonen dar, die von diesem Backend unterstützt werden. Weitere Informationen finden Sie unter "Verwenden Sie die CSI-Topologie". Beispiel: `supportedTopologies: - topology.kubernetes.io/region: asia-east1 topology.kubernetes.io/zone: asia-east1-a`	

Optionen zur Volume-Bereitstellung

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

Parameter	Beschreibung	Standard
`exportRule`	Die Exportregeln für neue Volumes. Muss eine kommagetrennte Liste einer beliebigen Kombination von IPv4-Adressen sein.	„0.0.0.0/0“
`snapshotDir`	Zugriff auf das `.snapshot` Verzeichnis	„Wahr“ für NFSv4 „falsch“ für NFSv3
`snapshotReserve`	Prozentsatz des für Snapshots reservierten Volumes	„“ (Standardeinstellung 0 akzeptieren)
`unixPermissions`	die unix-Berechtigungen neuer Volumes (4 Oktal-Ziffern).	„“

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 Trident alle an Google Cloud NetApp Volumes delegierten Storage-Pools am konfigurierten Standort und platziert neue Volumes zufällig in einem dieser Pools. Da `nasType` nicht angegeben ist, gilt der `nfs` Standard und das Backend wird für NFS Volumes bereitgestellt.

Diese Konfiguration ist ideal, wenn Sie gerade erst mit Google Cloud NetApp Volumes beginnen und alles ausprobieren möchten, aber in der Praxis müssen Sie höchstwahrscheinlich einen zusätzlichen Umfang für die bereitgestellten Volumes angeben.

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----\n
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\n
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\n
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\n
    XsYg6gyxy4zq7OlwWgLwGa==\n
    -----END PRIVATE KEY-----\n

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  serviceLevel: premium
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    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/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Konfiguration für SMB Volumes

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv1
  namespace: trident
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123456789"
  location: asia-east1
  serviceLevel: flex
  nasType: smb
  apiKey:
    type: service_account
    project_id: cloud-native-data
    client_email: trident-sample@cloud-native-
data.iam.gserviceaccount.com
    client_id: "123456789737813416734"
    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/trident-
sample%40cloud-native-data.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Konfiguration mit StoragePools-Filter

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  serviceLevel: premium
  storagePools:
    - premium-pool1-europe-west6
    - premium-pool2-europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    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/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Konfiguration des virtuellen Pools

Diese Backend-Konfiguration definiert mehrere virtuelle Pools in einer einzelnen Datei. Virtuelle Pools werden im Abschnitt definiert `storage` . Sie sind nützlich, wenn Sie mehrere Storage-Pools haben, die unterschiedliche Service-Level unterstützen, und Sie Storage-Klassen in Kubernetes erstellen möchten, die diese repräsentieren. Zur Unterscheidung der Pools werden Bezeichnungen für virtuelle Pools verwendet. Im Beispiel unten werden beispielsweise `performance` Label und `serviceLevel` type zur Unterscheidung virtueller Pools verwendet.

Sie können auch einige Standardwerte für alle virtuellen Pools festlegen und die Standardwerte für einzelne virtuelle Pools überschreiben. Im folgenden Beispiel `snapshotReserve` und `exportRule` dienen als Standard für alle virtuellen Pools.

Weitere Informationen finden Sie unter "Virtuelle Pools".

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
```

```
    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/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
  defaults:
    snapshotReserve: "10"
    exportRule: 10.0.0.0/24
  storage:
    - labels:
        performance: extreme
      serviceLevel: extreme
      defaults:
        snapshotReserve: "5"
        exportRule: 0.0.0.0/0
    - labels:
        performance: premium
      serviceLevel: premium
    - labels:
        performance: standard
      serviceLevel: standard
```

Cloud-Identität für GKE

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcp-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: '012345678901'
  network: gcnv-network
  location: us-west2
  serviceLevel: Premium
  storagePool: pool-premium1
```

Konfiguration unterstützter Topologien

Trident erleichtert die Bereitstellung von Volumes für Workloads, basierend auf Regionen und Verfügbarkeitszonen. Der `supportedTopologies` Block in dieser Backend-Konfiguration dient zur Bereitstellung einer Liste von Regionen und Zonen pro Backend. Die hier angegebenen Region- und Zonenwerte müssen mit den Region- und Zonenwerten der Beschriftungen auf jedem Kubernetes-Cluster-Node übereinstimmen. Diese Regionen und Zonen stellen die Liste der zulässigen Werte dar, die in einer Lagerklasse bereitgestellt werden können. Für Storage-Klassen, die eine Teilmenge der Regionen und Zonen enthalten, die in einem Back-End bereitgestellt werden, erstellt Trident Volumes in der genannten Region und Zone. Weitere Informationen finden Sie unter "Verwenden Sie die CSI-Topologie".

```
---
version: 1
storageDriverName: google-cloud-netapp-volumes
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: asia-east1
serviceLevel: flex
supportedTopologies:
  - topology.kubernetes.io/region: asia-east1
    topology.kubernetes.io/zone: asia-east1-a
  - topology.kubernetes.io/region: asia-east1
    topology.kubernetes.io/zone: asia-east1-b
```

Was kommt als Nächstes?

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

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

Führen Sie den folgenden Befehl aus, um zu überprüfen, ob das Backend erfolgreich erstellt wurde:

```
kubectl get tridentbackendconfig

NAME                BACKEND NAME        BACKEND UUID
PHASE   STATUS
backend-tbc-gcnv    backend-tbc-gcnv    b2fd1ff9-b234-477e-88fd-713913294f65
Bound   Success
```

Wenn die Backend-Erstellung fehlschlägt, ist mit der Back-End-Konfiguration ein Fehler aufgetreten. Sie können das Backend mit dem Befehl beschreiben `kubectl get tridentbackendconfig <backend-name>` oder die Protokolle anzeigen, um die Ursache zu ermitteln, indem Sie den folgenden Befehl ausführen:

```
tridentctl logs
```

Nachdem Sie das Problem mit der Konfigurationsdatei identifiziert und behoben haben, können Sie das Backend löschen und den Befehl create erneut ausführen.

Definitionen der Storage-Klassen

Im Folgenden finden Sie eine grundlegende `StorageClass` Definition, die sich auf das Backend oben bezieht.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-nfs-sc
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
```

Beispieldefinitionen mit dem `parameter.selector` Feld:

Mit `parameter.selector` können Sie für jeden angeben `StorageClass` "Virtueller Pool" , der zum Hosten eines Volumes verwendet wird. Im Volume werden die Aspekte definiert, die im ausgewählten Pool definiert sind.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: extreme-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=extreme
  backendType: google-cloud-netapp-volumes

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: premium-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=premium
  backendType: google-cloud-netapp-volumes

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: standard-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=standard
  backendType: google-cloud-netapp-volumes
```

Weitere Informationen zu Speicherklassen finden Sie unter "Erstellen Sie eine Speicherklasse".

Beispieldefinitionen für SMB Volumes

Mit `nasType`, `node-stage-secret-name` und `node-stage-secret-namespace`können Sie ein SMB-Volume angeben und die erforderlichen Active Directory-Anmeldeinformationen eingeben. Jeder Active Directory-Benutzer/jedes Active Directory-Kennwort mit beliebigen oder keinen Berechtigungen kann für den Schlüssel der Knotenstufe verwendet werden.

Grundkonfiguration im Standard-Namespace

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  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: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  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: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  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.

Beispiel für eine PVC-Definition

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: gcnv-nfs-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 100Gi
  storageClassName: gcnv-nfs-sc
```

Um zu überprüfen, ob die PVC gebunden ist, führen Sie den folgenden Befehl aus:

```
kubectl get pvc gcnv-nfs-pvc

NAME          STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS AGE
gcnv-nfs-pvc  Bound    pvc-b00f2414-e229-40e6-9b16-ee03eb79a213  100Gi
RWX            gcnv-nfs-sc  1m
```

Cloud Volumes Service für Google Cloud-Back-End konfigurieren

Erfahren Sie, wie Sie NetApp Cloud Volumes Service für Google Cloud mithilfe der bereitgestellten Beispielkonfigurationen als Back-End für Ihre Trident-Installation konfigurieren.

Treiberdetails zu Google Cloud

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

Treiber	Protokoll	VolumeModus	Unterstützte Zugriffsmodi	Unterstützte Filesysteme
`gcp-cvs`	NFS	Dateisystem	RWO, ROX, RWX, RWOP	`nfs`

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

Trident kann Cloud Volumes Service Volumes in einer von zwei erstellen"Servicetypen":

- **CVS-Performance**: Der Standard-Trident-Diensttyp. 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 der "Drei Service-Level"folgenden Optionen wählen:
  - `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 der "Zwei Service-Level"folgenden Optionen wählen:
  - `standardsw`
  - `zoneredundantstandardsw`

Was Sie benötigen

Um das Backend zu konfigurieren und zu verwenden "Cloud Volumes Service für Google Cloud", benötigen Sie Folgendes:

- Ein Google Cloud Konto, das mit NetApp Cloud Volumes Service konfiguriert ist
- Projektnummer Ihres Google Cloud-Kontos
- Google Cloud Service-Konto mit der `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	Treibername + „_“ + Teil des API-Schlüssels
`storageClass`	Optionaler Parameter zur Angabe des CVS-Servicetyps. Verwenden Sie `software`, um den CVS-Diensttyp auszuwählen. Andernfalls übernimmt Trident den CVS-Performance Servicetyp (`hardware`).	
`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.	

Parameter	Beschreibung	Standard
`hostProjectNumber`	Erforderlich bei Verwendung eines gemeinsamen VPC-Netzwerks. In diesem Szenario `projectNumber` handelt es sich um das Service-Projekt und `hostProjectNumber` das Host-Projekt.	
`apiRegion`	Die Google Cloud-Region, in der Trident Cloud Volumes Service Volumes erstellt Bei der Erstellung regionsübergreifender Kubernetes-Cluster können in einem erstellte Volumes `apiRegion` für Workloads verwendet werden, die auf Nodes in mehreren Google Cloud-Regionen geplant sind. Der regionale Verkehr verursacht zusätzliche Kosten.	
`apiKey`	API-Schlüssel für das Google Cloud-Servicekonto mit der `netappcloudvolumes.admin` Rolle. Er enthält den JSON-formatierten Inhalt der privaten Schlüsseldatei eines Google Cloud-Dienstkontos (wortgetreu in die Back-End-Konfigurationsdatei kopiert).	
`proxyURL`	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. Bei einem HTTPS-Proxy wird die Zertifikatvalidierung übersprungen, um die Verwendung von selbstsignierten Zertifikaten im Proxyserver zu ermöglichen. Proxy-Server mit aktivierter Authentifizierung werden nicht unterstützt.	
`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`	Das CVS-Performance oder CVS Service-Level für neue Volumes. CVS-Leistungswerte sind `standard`, `premium` oder `extreme`. CVS-Werte sind `standardsw` oder `zoneredundantstandardsw`.	CVS-Performance ist der Standard. Der CVS-Standardwert ist „standardsw“.
`network`	Für Cloud Volumes Service Volumes verwendetes Google Cloud Netzwerk	„Standard“
`debugTraceFlags`	Fehler-Flags bei der Fehlerbehebung beheben. Beispiel, `\{"api":false, "method":true}`. Verwenden Sie dies nur, wenn Sie Fehler beheben und einen detaillierten Log Dump benötigen.	Null

`allowedTopologies`	Um den regionsübergreifenden Zugriff zu ermöglichen, muss die StorageClass-Definition für `allowedTopologies` alle Regionen umfassen. Beispiel:
	`- key: topology.kubernetes.io/region`
	`values:`
	`- us-east1`
	`- europe-west1`	

Optionen zur Volume-Bereitstellung

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

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

Beispiele für CVS-Performance-Diensttypen

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

Beispiel 1: Minimale Konfiguration

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

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: "012345678901"
apiRegion: us-west2
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: <id_value>
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----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
```

Beispiel 2: Service Level-Konfiguration

Dieses Beispiel stellt die Back-End-Konfigurationsoptionen dar, einschließlich Service Level und Volume-Standardeinstellungen.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
apiRegion: us-west2
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----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
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: Konfiguration des virtuellen Pools

Dieses Beispiel verwendet `storage`, um virtuelle Pools und die zu konfigurieren `StorageClasses`, die auf sie verweisen. SieheDefinitionen der Storage-Klassen, um zu sehen, wie die Speicherklassen definiert wurden.

Hier werden spezifische Standardwerte für alle virtuellen Pools festgelegt, die den auf 5 % und den auf `exportRule` 0.0.0.0/0 setzen `snapshotReserve`. Die virtuellen Pools werden im Abschnitt definiert `storage`. Jeder einzelne virtuelle Pool definiert seinen eigenen `serviceLevel`, und einige Pools überschreiben die Standardwerte. Virtuelle Pool-Etiketten wurden verwendet, um die Pools basierend auf und `protection`zu unterscheiden `performance`.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
apiRegion: us-west2
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----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. Mit `parameters.selector` können Sie für jede StorageClass den virtuellen Pool angeben, der zum Hosten 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`) wird dem ersten virtuellen Pool zugeordnet. Dies ist der einzige Pool, der eine extreme Performance mit einer Snapshot-Reserve von 10 % bietet.
- Die letzte StorageClass (`cvs-extra-protection`) ruft jeden Speicherpool auf, der eine Snapshot-Reserve von 10% bietet. Trident entscheidet, welcher virtuelle Pool ausgewählt wird, und stellt sicher, dass die Anforderung der Snapshot-Reserve erfüllt wird.

Beispiele für CVS-Diensttypen

Die folgenden Beispiele enthalten Beispielkonfigurationen für den CVS-Servicetyp.

Beispiel 1: Minimalkonfiguration

Dies ist die minimale Backend-Konfiguration `storageClass` zur Angabe des CVS-Diensttyps und des Standard- `standardsw`Service-Levels.

```
---
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-----
    <key_value>
    -----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-Backend-Konfiguration verwendet `storagePools`, um einen Speicherpool zu konfigurieren.

```
---
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-----
    <key_value>
    -----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

Erfahren Sie, wie Sie mit Ihrer Trident Installation ein Element Backend erstellen und verwenden.

Details zum Elementtreiber

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

Der `solidfire-san` Speichertreiber unterstützt die Volume-Modi *File* und *Block*. Für den `Filesystem` Volumemodus erstellt Trident ein Volume und 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	„SolidFire_“ + Storage (iSCSI) IP-Adresse

Parameter	Beschreibung	Standard
`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. Trident verwendet CHAP.	Richtig
`AccessGroups`	Liste der zu verwendenden Zugriffsgruppen-IDs	Sucht die ID einer Zugriffsgruppe namens „Trident“
`Types`	QoS-Spezifikationen	
`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, „method“:true}	Null

Verwenden Sie diese Funktion `debugTraceFlags` nur, wenn Sie eine Fehlerbehebung durchführen und einen detaillierten Protokollauszug benötigen.

Beispiel 1: Backend-Konfiguration für `solidfire-san` Treiber mit drei Volume-Typen

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 diese mit dem Storage-Klassen-Parameter zu nutzen `IOPS`.

```
---
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-Klassenkonfiguration für `solidfire-san` Treiber mit virtuellen Pools

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

Trident kopiert bei der Bereitstellung Labels, die sich in einem Storage-Pool befinden, auf die Back-End-Storage-LUN. Storage-Administratoren können Labels je virtuellen Pool definieren und Volumes nach Label gruppieren.

In der unten abgebildeten Beispieldefinitionsdatei für das Backend werden spezifische Standardwerte für alle Speicherpools festgelegt, die die auf „Silber“ setzen `type`. Die virtuellen Pools werden im Abschnitt definiert `storage`. 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. Mit dem

`parameters.selector` Feld ruft jede StorageClass ab, 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 eine Goldleistung mit einem Gold bietet `Volume Type QoS`. Die letzte StorageClass (`solidfire-silver`) ruft jeden Speicherpool auf, der eine silberne Performance bietet. Trident entscheidet, welcher virtuelle Pool ausgewählt wird, und stellt sicher, dass die Speicheranforderungen 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

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

Treiber	Protokoll	VolumeModus	Unterstützte Zugriffsmodi	Unterstützte Filesysteme
`ontap-san`	ISCSI SCSI über FC	Block-Storage	RWO, ROX, RWX, RWOP	Kein Filesystem, rohes Block-Gerät
`ontap-san`	ISCSI SCSI über FC	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

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

- Verwenden Sie `ontap-san-economy` diese Option nur, wenn die Anzahl der persistenten Volumes voraussichtlich höher ist als "Unterstützte ONTAP-Volume-Größen".
- Verwenden Sie `ontap-nas-economy` diese Option nur, wenn die Anzahl der persistenten Volumes voraussichtlich höher ist als "Unterstützte ONTAP-Volume-Größen" und der `ontap-san-economy` Treiber nicht verwendet werden kann.
- Verwenden Sie diese Option nicht `ontap-nas-economy`, wenn Sie voraussehen, dass Datensicherung, Disaster Recovery oder Mobilität erforderlich sind.
- NetApp empfiehlt nicht die Verwendung von FlexVol Autogrow in allen ONTAP-Treibern außer ONTAP-san. Als Workaround unterstützt Trident die Verwendung von Snapshot-Reserve und skaliert FlexVol-Volumen entsprechend.

Benutzerberechtigungen

Trident geht davon aus, dass es entweder als ONTAP- oder SVM-Administrator ausgeführt wird, wobei der Cluster-Benutzer oder ein `vsadmin` SVM-Benutzer oder ein Benutzer mit einem anderen Namen und derselben Rolle verwendet `admin` wird. Bei Implementierungen von Amazon FSX for NetApp ONTAP rechnet Trident damit, als ONTAP- oder SVM-Administrator ausgeführt zu werden. Dabei verwendet er den Cluster-`fsxadmin`Benutzer, einen `vsadmin` SVM-Benutzer oder einen Benutzer mit einem anderen Namen mit derselben Rolle. Der `fsxadmin` Benutzer ist ein eingeschränkter Ersatz für den Cluster-Admin-Benutzer.

Wenn Sie den Parameter verwenden `limitAggregateUsage`, sind Administratorberechtigungen für den Cluster erforderlich. Wenn Amazon FSX for NetApp ONTAP mit Trident verwendet wird, funktioniert der `limitAggregateUsage` Parameter nicht mit den `vsadmin` Benutzerkonten und `fsxadmin`. 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

Trident unterstützt das NVMe-Protokoll (Non-Volatile Memory Express) unter Verwendung des `ontap-san` Treibers, einschließlich:

- IPv6
- Snapshots und Klone von NVMe Volumes
- Größe eines NVMe Volumes ändern
- Importieren eines NVMe Volumes, das außerhalb von Trident erstellt wurde, damit sein Lebenszyklus durch Trident gemanagt werden kann
- NVMe-natives Multipathing
- Ordnungsgemäßes oder unzumutbar Herunterfahren der K8s-Nodes (24.06)

Trident unterstützt Folgendes nicht:

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

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

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

Anforderungen

Für alle ONTAP-Backends erfordert Trident, dass dem SVM mindestens ein Aggregat zugewiesen wird.

Informationen zum Zuweisen von Aggregaten zu SVM in ASA R2-Systemen finden Sie in diesem Knowledge Base-Artikel: "Das Erstellen einer Speichereinheit durch den SVM-Administrator mithilfe der CLI schlägt mit der Fehlermeldung „Für Speicherdienste sind keine Kandidatenaggregate verfügbar“ fehl." .

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

Alle Kubernetes-Worker-Nodes müssen über die entsprechenden iSCSI-Tools verfügen. Weitere Informationen finden Sie unter "Bereiten Sie den Knoten „Worker“ vor" .

Authentifizieren Sie das ONTAP-Backend

Trident bietet zwei Arten der Authentifizierung eines ONTAP-Backends.

- Anmeldeinformationsbasiert: Benutzername und Passwort für einen ONTAP-Benutzer mit den erforderlichen Berechtigungen. Es wird empfohlen, eine vordefinierte Sicherheits-Login-Rolle zu verwenden, wie `admin` oder `vsadmin`, um maximale Kompatibilität mit ONTAP-Versionen zu gewährleisten.
- Zertifikat-basiert: Trident kann auch über ein auf dem Backend installiertes Zertifikat mit einem ONTAP-Cluster kommunizieren. 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

Für die Kommunikation mit dem ONTAP-Back-End ist die Zugangsdaten an einen Administrator mit SVM-Umfang/Cluster-Umfang erforderlich Trident. Es wird empfohlen, standardmäßige, vordefinierte Rollen wie oder `vsadmin`zu verwenden `admin`. So wird die Kompatibilität mit zukünftigen ONTAP Versionen sichergestellt, die möglicherweise die FunktionAPIs für zukünftige Trident Versionen offenlegen. Eine benutzerdefinierte Sicherheits-Login-Rolle kann erstellt und mit Trident verwendet werden, wird aber nicht empfohlen.

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

YAML

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

JSON

```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. Vergewissern Sie sich, dass die ONTAP-Sicherheits-Anmeldungsrolle die Authentifizierungsmethode unterstützt `cert`.

```
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 vertrauenswürdigem 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...Vaaalllluuuueeee",
"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 Datei Backend.json, 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                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
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 an, dass Trident mit dem ONTAP Back-End kommunizieren und zukünftige Volume-Operationen verarbeiten kann.

Benutzerdefinierte ONTAP-Rolle für Trident erstellen

Sie können eine ONTAP-Cluster-Rolle mit minimaler Privileges erstellen, sodass Sie nicht die ONTAP-Administratorrolle verwenden müssen, um Vorgänge in Trident auszuführen. Wenn Sie den Benutzernamen in eine Trident-Back-End-Konfiguration aufnehmen, verwendet Trident die ONTAP-Cluster-Rolle, die Sie für die Durchführung der Vorgänge erstellt haben.

Weitere Informationen zum Erstellen benutzerdefinierter Trident-Rollen finden Sie unter"Trident Custom-Role Generator".

Verwenden der ONTAP CLI

1. Erstellen Sie eine neue Rolle mit dem folgenden Befehl:

```
security login role create <role_name\> -cmddirname "command" -access all
–vserver <svm_name\>
```

2. Erstellen Sie einen Benutzernamen für den Trident-Benutzer:

```
security login create -username <user_name\> -application ontapi
-authmethod <password\> -role <name_of_role_in_step_1\> –vserver
<svm_name\> -comment "user_description"
```

3. Ordnen Sie die Rolle dem Benutzer zu:

```
security login modify username <user_name\> –vserver <svm_name\> -role
<role_name\> -application ontapi -application console -authmethod
<password\>
```

Verwenden Von System Manager

Führen Sie die folgenden Schritte im ONTAP System Manager durch:

1. **Erstellen Sie eine benutzerdefinierte Rolle**:
   a. Um eine benutzerdefinierte Rolle auf Cluster-Ebene zu erstellen, wählen Sie **Cluster > Einstellungen** aus.

      (Oder) um eine benutzerdefinierte Rolle auf SVM-Ebene zu erstellen, wählen Sie **Storage > Storage VMs > > `required SVM` Einstellungen > Benutzer und Rollen** aus.

   b. Wählen Sie das Pfeilsymbol (→) neben **Users and Roles**.
   c. Wählen Sie unter **Rollen +Hinzufügen** aus.
   d. Definieren Sie die Regeln für die Rolle und klicken Sie auf **Speichern**.
2. **Rolle dem Trident-Benutzer zuordnen**: + Führen Sie auf der Seite **Benutzer und Rollen** folgende Schritte aus:
   a. Wählen Sie unter **Benutzer** das Symbol Hinzufügen **+**.
   b. Wählen Sie den gewünschten Benutzernamen aus, und wählen Sie im Dropdown-Menü für **Rolle** eine Rolle aus.
   c. Klicken Sie Auf **Speichern**.

Weitere Informationen finden Sie auf den folgenden Seiten:

- "Benutzerdefinierte Rollen für die Administration von ONTAP" Oder "Definieren benutzerdefinierter Rollen"
- "Arbeiten Sie mit Rollen und Benutzern"

Verbindungen mit bidirektionalem CHAP authentifizieren

Trident kann iSCSI-Sitzungen mit bidirektionalem CHAP für den und `ontap-san-economy`-Treiber authentifizieren `ontap-san`. Dazu muss die Option in Ihrer Backend-Definition aktiviert `useCHAP` werden. Wenn auf festgelegt `true`, konfiguriert Trident die standardmäßige Initiatorsicherheit der SVM auf bidirektionales CHAP und legt den Benutzernamen und die Schlüssel aus der Backend-Datei fest. 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` `chapTargetUsername` müssen die `chapInitiatorSecret` Felder , , `chapTargetInitiatorSecret` und `chapUsername` in die Backend-Definition einbezogen werden. Die Secrets können geändert werden, nachdem ein Backend durch Ausführen erstellt `tridentctl update` wurde.

So funktioniert es

Durch die Einstellung `useCHAP` auf true weist der Speicheradministrator Trident an, CHAP auf dem Speicher-Back-End zu konfigurieren. Dazu gehört Folgendes:

- Einrichten von CHAP auf der SVM:
  - Wenn der standardmäßige Sicherheitstyp des Initiators der SVM none (standardmäßig festgelegt) ist **und**, wenn keine bereits vorhandenen LUNs im Volume vorhanden sind, setzt Trident den Standardsicherheitstyp auf `CHAP` und fährt mit der Konfiguration des CHAP-Initiators und des Zielbenutzernamens und der -Schlüssel fort.
  - Wenn die SVM LUNs enthält, aktiviert Trident CHAP auf der SVM nicht. 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 Backends erstellt Trident eine entsprechende `tridentbackend` CRD und speichert die CHAP-Geheimnisse und Benutzernamen als Kubernetes-Geheimnisse. Alle PVS, die von Trident auf diesem Backend erstellt werden, werden über CHAP gemountet und angehängt.

Anmeldedaten rotieren und Back-Ends aktualisieren

Sie können die CHAP-Anmeldeinformationen aktualisieren, indem Sie die CHAP-Parameter in der Datei aktualisieren `backend.json`. Dies erfordert die Aktualisierung der CHAP-Schlüssel und die Verwendung des `tridentctl update` Befehls, um diese Änderungen widerzuspiegeln.

> Wenn Sie die CHAP-Schlüssel für ein Backend aktualisieren, müssen Sie `tridentctl` das Backend aktualisieren. Aktualisieren Sie die Zugangsdaten auf dem Storage-Cluster nicht über die ONTAP-CLI oder den ONTAP-System-Manager, da Trident diese Änderungen nicht aufnehmen 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 nicht betroffen und bleiben weiterhin aktiv, wenn die Zugangsdaten von

Trident auf der SVM aktualisiert werden. Für neue Verbindungen werden die aktualisierten Anmeldeinformationen verwendet, und bestehende 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 mit Ihrer 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-san` Oder `ontap-san-economy`
`backendName`	Benutzerdefinierter Name oder das Storage-Backend	Treibername + „_“ + DatenLIF
`managementLIF`	Die IP-Adresse einer Cluster- oder SVM-Management-LIF. Es kann ein vollständig qualifizierter Domänenname (FQDN) angegeben werden. Kann so eingestellt werden, dass IPv6-Adressen verwendet werden, wenn 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]`. Informationen über die nahtlose MetroCluster-Umschaltung finden Sie im Beispiel: MetroCluster. Wenn Sie „vsadmin“-Anmeldedaten verwenden, `managementLIF` muss dies die der SVM sein. Bei Verwendung der „admin“-Anmeldedaten muss es sich um die des Clusters handeln. `managementLIF`	„10.0.0.1“, „[2001:1234:abcd::fefe]“

Parameter	Beschreibung	Standard
`dataLIF`	IP-Adresse des LIF-Protokolls. Kann so eingestellt werden, dass IPv6-Adressen verwendet werden, wenn 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]`. **Nicht für iSCSI angeben.** Trident verwendet"ONTAP selektive LUN-Zuordnung", um die iSCSI LIFs zu ermitteln, die für eine Multi-Path-Sitzung erforderlich sind. Eine Warnung wird erzeugt, wenn `dataLIF` explizit definiert ist. **Für MetroCluster weglassen.** Siehe Beispiel: MetroCluster.	Abgeleitet von SVM
`svm`	Zu verwendende virtuelle Speichermaschine **omit für MetroCluster.** Siehe Beispiel: MetroCluster.	Abgeleitet, wenn eine SVM `managementLIF` angegeben wird
`useCHAP`	Verwenden Sie CHAP, um iSCSI für ONTAP-SAN-Treiber zu authentifizieren [Boolesch]. Legen Sie für Trident fest `true`, um bidirektionales CHAP als Standardauthentifizierung für die im Backend angegebene SVM zu konfigurieren und zu verwenden. Weitere Informationen finden Sie unter "Vorbereiten der Back-End-Konfiguration mit ONTAP-SAN-Treibern" .	`false`
`chapInitiatorSecret`	CHAP-Initiatorschlüssel. Erforderlich, wenn `useCHAP=true`	„“
`labels`	Satz willkürlicher JSON-formatierter Etiketten für Volumes	„“
`chapTargetInitiatorSecret`	Schlüssel für CHAP-Zielinitiator. Erforderlich, wenn `useCHAP=true`	„“
`chapUsername`	Eingehender Benutzername. Erforderlich, wenn `useCHAP=true`	„“
`chapTargetUsername`	Zielbenutzername. Erforderlich, wenn `useCHAP=true`	„“
`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.	„“

Parameter	Beschreibung	Standard
`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 `managementLIF` angegeben wird
`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.	`trident`
`aggregate`	Aggregat für die Bereitstellung (optional, wenn eingestellt, muss der SVM zugewiesen werden) Für den `ontap-nas-flexgroup` Treiber wird diese Option ignoriert. Falls nicht, können alle verfügbaren Aggregate verwendet werden, um ein FlexGroup Volume bereitzustellen. Wenn das Aggregat in einer SVM aktualisiert wird, wird es automatisch in Trident aktualisiert, indem es die SVM abfragt, ohne den Trident Controller neu starten zu müssen. Wenn Sie ein bestimmtes Aggregat in Trident für die Bereitstellung von Volumes konfiguriert haben, wird das Back-End Trident bei der Abfrage des SVM-Aggregats in den Status „Fehlgeschlagen“ verschoben. Sie müssen entweder das Aggregat zu einem auf der SVM vorhandenen Aggregat ändern oder es komplett entfernen, um das Back-End wieder online zu schalten. **Nicht für ASA r2** angeben.	„“
`limitAggregateUsage`	Bereitstellung fehlgeschlagen, wenn die Nutzung über diesem Prozentsatz liegt. Wenn Sie ein Amazon FSX für NetApp ONTAP-Backend verwenden, geben Sie nicht an `limitAggregateUsage`. Die angegebenen `fsxadmin` und `vsadmin` enthalten nicht die erforderlichen Berechtigungen, um die aggregierte Nutzung abzurufen und sie mit Trident zu begrenzen. **Nicht für ASA r2** angeben.	„“ (nicht standardmäßig durchgesetzt)
`limitVolumeSize`	Bereitstellung fehlgeschlagen, wenn die angeforderte Volume-Größe über diesem Wert liegt. Beschränkt außerdem die maximale Größe der Volumes, die es für LUNs managt.	„“ (nicht standardmäßig durchgesetzt)

Parameter	Beschreibung	Standard
`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} nicht verwenden, es sei denn, Sie beheben die Fehlerbehebung und benötigen einen detaillierten Log Dump.	`null`
`useREST`	Boolescher Parameter zur Verwendung von ONTAP REST-APIs. `useREST` Wenn auf festgelegt `true`, verwendet Trident ONTAP REST APIs, um mit dem Backend zu kommunizieren; wenn auf gesetzt `false`, verwendet Trident ONTAPI (ZAPI) Aufrufe, um mit dem Backend zu kommunizieren. Diese Funktion erfordert ONTAP 9.11.1 und höher. Darüber hinaus muss die verwendete ONTAP-Anmelderolle Zugriff auf die Anwendung haben `ontapi` . Dies wird durch die vordefinierten `vsadmin` Rollen und `cluster-admin` erreicht. Ab Trident 24.06-Version und ONTAP 9.15.1 oder höher `useREST` ist standardmäßig auf gesetzt `true`. Wechseln Sie `useREST` zu `false` ONTAPI (ZAPI)-Aufrufe verwenden. `useREST` Ist vollständig für NVMe/TCP qualifiziert. **Falls angegeben, immer für ASA r2** auf einstellen `true`.	`true` Für ONTAP 9.15.1 oder höher, andernfalls `false`.
`sanType`	Verwenden Sie diese Option, um für iSCSI, `nvme` für NVMe/TCP oder `fcp` für SCSI über Fibre Channel (FC) auszuwählen `iscsi`.	`iscsi` Falls leer
`formatOptions`	Verwenden Sie `formatOptions` zum Angeben von Befehlszeilenargumenten für den `mkfs` Befehl, die bei jedem Formatieren eines Volumes angewendet werden. Auf diese Weise können Sie die Lautstärke nach Ihren Wünschen formatieren. Stellen Sie sicher, dass Sie die Formatieroptionen ähnlich wie die der mkfs-Befehlsoptionen angeben, ohne den Gerätepfad. Beispiel: „-E nodiscard“ • `ontap-san` `ontap-san-economy`Nur für und Treiber unterstützt.*	
`limitVolumePoolSize`	Maximale anforderbare FlexVol-Größe bei Verwendung von LUNs im ONTAP-san-Economy-Backend.	„“ (nicht standardmäßig durchgesetzt)
`denyNewVolumePools`	Schränkt das Erstellen neuer FlexVol Volumes für LUNs ein `ontap-san-economy` Zur Bereitstellung neuer PVS werden nur vorbestehende FlexVols verwendet.	

Empfehlungen für die Verwendung von FormatOptions

Trident empfiehlt die folgende Option, um den Formatierungsprozess zu beschleunigen:

-E nodiscard:

- Beibehalten, versuchen Sie nicht, Blöcke zur mkfs-Zeit zu verwerfen (das Verwerfen von Blöcken ist zunächst auf Solid State-Geräten und selten/Thin Provisioning-Storage nützlich). Dies ersetzt die veraltete Option "-K" und ist auf alle Dateisysteme anwendbar (xfs, ext3 und ext4).

Back-End-Konfigurationsoptionen für die Bereitstellung von Volumes

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

Parameter	Beschreibung	Standard
`spaceAllocation`	Speicherplatzzuweisung für LUNs	„True“ **Falls angegeben, setzen Sie für ASA r2** auf `true`.
`spaceReserve`	Modus für Speicherplatzreservierung; „none“ (Thin) oder „Volume“ (Thick). **Für ASA r2** auf eingestellt `none`.	„Keine“
`snapshotPolicy`	Zu verwendende Snapshot-Richtlinie. **Für ASA r2** auf eingestellt `none`.	„Keine“
`qosPolicy`	QoS-Richtliniengruppe zur Zuweisung für erstellte Volumes Wählen Sie eine der qosPolicy oder adaptiveQosPolicy pro Storage Pool/Backend. Für die Verwendung von QoS-Richtliniengruppen mit Trident ist ONTAP 9 8 oder höher erforderlich. Sie sollten eine nicht gemeinsam genutzte QoS-Richtliniengruppe verwenden und sicherstellen, dass die Richtliniengruppe auf jede Komponente einzeln angewendet wird. Eine Shared-QoS-Richtliniengruppe erzwingt die Obergrenze für den Gesamtdurchsatz aller Workloads.	„“
`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. **Nicht für ASA r2** angeben.	„0“, wenn `snapshotPolicy` „keine“ ist, 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, Standardeinstellung ist `false`. NVE muss im Cluster lizenziert und aktiviert sein, damit diese Option verwendet werden kann. Wenn auf dem Backend NAE aktiviert ist, wird jedes in Trident bereitgestellte Volume NAE aktiviert. Weitere Informationen finden Sie unter "Funktionsweise von Trident mit NVE und NAE".	„False“ **Falls angegeben, setzen Sie für ASA r2** auf `true`.

Parameter	Beschreibung	Standard
`luksEncryption`	Aktivieren Sie die LUKS-Verschlüsselung. Siehe "Linux Unified Key Setup (LUKS) verwenden".	„“ für ASA r2 eingestellt `false`.
`tieringPolicy`	Tiering Policy zu verwenden "none" **nicht angeben für ASA r2**.	
`nameTemplate`	Vorlage zum Erstellen benutzerdefinierter Volume-Namen.	„“

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 Volumes, die mit dem Treiber erstellt `ontap-san` wurden, fügt Trident der FlexVol zusätzliche Kapazität von 10 % hinzu, um die LUN-Metadaten aufzunehmen. Die LUN wird genau mit der Größe bereitgestellt, die der Benutzer in der PVC anfordert. Trident addiert 10 Prozent zum FlexVol (wird als verfügbare Größe in ONTAP angezeigt). 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`, berechnet Trident die Größe der Volumes wie folgt:

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

Die 1.1 ist die zusätzliche 10 Prozent Trident fügt zu den FlexVol, um die LUN-Metadaten aufzunehmen. Für `snapshotReserve` = 5 % und die PVC-Anforderung = 5 gib beträgt die Gesamtgröße des Volumes 5,79 gib und die verfügbare Größe 5,5 gib. Der `volume show` Befehl sollte die Ergebnisse ähnlich wie in diesem Beispiel anzeigen:

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 Trident verwenden, empfiehlt NetApp, dass Sie DNS-Namen für LIFs anstelle von IP-Adressen angeben.

Beispiel: ONTAP SAN

Dies ist eine Grundkonfiguration 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: MetroCluster

Sie können das Backend konfigurieren, um zu vermeiden, dass die Backend-Definition nach Umschaltung und Switchback während manuell aktualisiert "SVM-Replizierung und Recovery"werden muss.

Geben Sie für ein nahtloses Switchover und Switchback die SVM mit an `managementLIF` und lassen Sie die Parameter weg `svm`. Beispiel:

```
version: 1
storageDriverName: ontap-san
managementLIF: 192.168.1.66
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 für die zertifikatbasierte Authentifizierung

In diesem Beispiel der Grundkonfiguration `clientCertificate` werden , `clientPrivateKey` und `trustedCACertificate` (optional, wenn vertrauenswürdige CA verwendet wird) eingetragen `backend.json` und die base64-kodierten Werte des Clientzertifikats, des privaten Schlüssels und des vertrauenswürdigen CA-Zertifikats verwendet.

```
---
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 erzeugen ein Backend mit `useCHAP` set to `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
```

Beispiel für SCSI over FC (FCP

Auf Ihrem ONTAP-Back-End muss eine SVM mit FC konfiguriert sein. Dies ist eine grundlegende Back-End-Konfiguration für FC.

```
---
version: 1
backendName: fcp-backend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_fc
username: vsadmin
password: password
sanType: fcp
useREST: true
```

Back-End-Konfigurationsbeispiel mit nameTemplate

```
---
version: 1
storageDriverName: ontap-san
backendName: ontap-san-backend
managementLIF: <ip address>
svm: svm0
username: <admin>
password: <password>
defaults:
  nameTemplate:
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.vo\
    lume.RequestName}}"
labels:
  cluster: ClusterA
  PVC: "{{.volume.Namespace}}_{{.volume.RequestName}}"
```

FormatOptions Beispiel für ONTAP-san-Economy-Treiber

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: ""
svm: svm1
username: ""
password: "!"
storagePrefix: whelk_
debugTraceFlags:
  method: true
  api: true
defaults:
  formatOptions: -E nodiscard
```

Beispiele für Back-Ends mit virtuellen Pools

In diesen Beispiel-Back-End-Definitionsdateien werden spezifische Standardwerte für alle Speicherpools festgelegt, z. B. `spaceReserve` bei none, `spaceAllocation` bei false und `encryption` bei false. Die virtuellen Pools werden im Abschnitt Speicher definiert.

Trident legt die Bereitstellungsetiketten im Feld „Kommentare“ fest. Kommentare werden auf die FlexVol volume Trident-Kopien aller Labels, die auf einem virtuellen Pool auf das Speicher-Volume bei der Bereitstellung. Storage-Administratoren können Labels je virtuellen Pool definieren und Volumes nach Label gruppieren.

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

Beispiel: ONTAP SAN

```
---
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 beziehen sich auf Beispiele für Back-Ends mit virtuellen Pools. Mit dem `parameters.selector` Feld ruft jede StorageClass ab, welche virtuellen Pools zum Hosten eines Volumes verwendet werden können. Auf dem Volume werden die Aspekte im ausgewählten virtuellen Pool definiert.

- Die `protection-gold` StorageClass wird dem ersten virtuellen Pool im Backend zugeordnet `ontap-san`. 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"
```

- Die `protection-not-gold` StorageClass wird dem zweiten und dritten virtuellen Pool im Backend zugeordnet `ontap-san`. 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"
```

- Die `app-mysqldb` StorageClass wird dem dritten virtuellen Pool im Backend zugeordnet `ontap-san-economy`. 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"
```

- Die `protection-silver-creditpoints-20k` StorageClass wird dem zweiten virtuellen Pool im Backend zugeordnet `ontap-san`. 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"
```

- Die `creditpoints-5k` StorageClass wird dem dritten virtuellen Pool im Backend und dem vierten virtuellen Pool im Backend `ontap-san-economy` zugeordnet `ontap-san`. 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"
```

- Die `my-test-app-sc` StorageClass wird dem virtuellen Pool im `ontap-san` Treiber mit `sanType: nvme` zugeordnet `testAPP`. Dies ist der einzige Pool, der angeboten `testApp`wird.

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

Trident entscheidet, welcher virtuelle Pool ausgewählt wird, und stellt sicher, dass die Speicheranforderungen erfüllt werden.

ONTAP NAS-Treiber

Übersicht über 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

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

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`

- Verwenden Sie `ontap-san-economy` diese Option nur, wenn die Anzahl der persistenten Volumes voraussichtlich höher ist als "Unterstützte ONTAP-Volume-Größen".
- Verwenden Sie `ontap-nas-economy` diese Option nur, wenn die Anzahl der persistenten Volumes voraussichtlich höher ist als "Unterstützte ONTAP-Volume-Größen" und der `ontap-san-economy` Treiber nicht verwendet werden kann.
- Verwenden Sie diese Option nicht `ontap-nas-economy`, wenn Sie voraussehen, dass Datensicherung, Disaster Recovery oder Mobilität erforderlich sind.
- NetApp empfiehlt nicht die Verwendung von FlexVol Autogrow in allen ONTAP-Treibern außer ONTAP-san. Als Workaround unterstützt Trident die Verwendung von Snapshot-Reserve und skaliert FlexVol-Volumen entsprechend.

Benutzerberechtigungen

Trident geht davon aus, dass es entweder als ONTAP- oder SVM-Administrator ausgeführt wird, wobei der Cluster-Benutzer oder ein `vsadmin` SVM-Benutzer oder ein Benutzer mit einem anderen Namen und derselben Rolle verwendet `admin` wird.

Bei Implementierungen von Amazon FSX for NetApp ONTAP rechnet Trident damit, als ONTAP- oder SVM-Administrator ausgeführt zu werden. Dabei verwendet er den Cluster- `fsxadmin`Benutzer, einen `vsadmin` SVM-Benutzer oder einen Benutzer mit einem anderen Namen mit derselben Rolle. Der `fsxadmin` Benutzer ist ein eingeschränkter Ersatz für den Cluster-Admin-Benutzer.

Wenn Sie den Parameter verwenden `limitAggregateUsage`, sind Administratorberechtigungen für den Cluster erforderlich. Wenn Amazon FSX for NetApp ONTAP mit Trident verwendet wird, funktioniert der `limitAggregateUsage` Parameter nicht mit den `vsadmin` Benutzerkonten und `fsxadmin`. 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-Backends erfordert Trident, dass dem SVM mindestens ein Aggregat zugewiesen wird.
- Sie können mehrere Treiber ausführen und Speicherklassen erstellen, die auf den einen oder den anderen zeigen. Sie können beispielsweise eine Gold-Klasse konfigurieren, die den Treiber verwendet `ontap-nas`, und eine Bronze-Klasse, die den Treiber verwendet `ontap-nas-economy`.
- Alle Kubernetes-Worker-Nodes müssen über die entsprechenden NFS-Tools verfügen. "Hier"Weitere Informationen finden Sie unter.
- Trident unterstützt nur SMB Volumes, die in Pods gemountet sind, die nur auf Windows Nodes ausgeführt werden. Weitere Informationen finden Sie unter Vorbereitung zur Bereitstellung von SMB Volumes .

Authentifizieren Sie das ONTAP-Backend

Trident bietet zwei Arten der Authentifizierung eines ONTAP-Backends.

- Anmeldeinformationsbasiert: Dieser Modus erfordert ausreichende Berechtigungen für das ONTAP-Backend. Es wird empfohlen, ein Konto zu verwenden, das einer vordefinierten Sicherheits-Login-Rolle zugeordnet ist, z. B. `admin` oder `vsadmin`, um maximale Kompatibilität mit ONTAP-Versionen sicherzustellen.
- Zertifikatsbasiert: Für diesen Modus ist ein Zertifikat auf dem Backend installiert, damit Trident mit einem ONTAP-Cluster kommunizieren kann. 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

Für die Kommunikation mit dem ONTAP-Back-End ist die Zugangsdaten an einen Administrator mit SVM-Umfang/Cluster-Umfang erforderlich Trident. Es wird empfohlen, standardmäßige, vordefinierte Rollen wie oder `vsadmin`zu verwenden `admin`. So wird die Kompatibilität mit zukünftigen ONTAP Versionen sichergestellt, die möglicherweise die FunktionAPIs für zukünftige Trident Versionen offenlegen. Eine benutzerdefinierte Sicherheits-Login-Rolle kann erstellt und mit Trident verwendet werden, wird aber nicht empfohlen.

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

YAML

```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

```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. Vergewissern Sie sich, dass die ONTAP-Sicherheits-Anmeldungsrolle die Authentifizierungsmethode unterstützt `cert`.

```
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 für die LIF-Servicerichtlinie 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 vertrauenswürdigem 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...Vaaalllluuuueeee",
"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 Datei Backend.json, 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 an, dass Trident mit dem ONTAP Back-End kommunizieren und zukünftige Volume-Operationen verarbeiten kann.

Benutzerdefinierte ONTAP-Rolle für Trident erstellen

Sie können eine ONTAP-Cluster-Rolle mit minimaler Privileges erstellen, sodass Sie nicht die ONTAP-Administratorrolle verwenden müssen, um Vorgänge in Trident auszuführen. Wenn Sie den Benutzernamen in eine Trident-Back-End-Konfiguration aufnehmen, verwendet Trident die ONTAP-Cluster-Rolle, die Sie für die

Durchführung der Vorgänge erstellt haben.

Weitere Informationen zum Erstellen benutzerdefinierter Trident-Rollen finden Sie unter"Trident Custom-Role Generator".

Verwenden der ONTAP CLI

1. Erstellen Sie eine neue Rolle mit dem folgenden Befehl:

```
security login role create <role_name\> -cmddirname "command" -access all
–vserver <svm_name\>
```

2. Erstellen Sie einen Benutzernamen für den Trident-Benutzer:

```
security login create -username <user_name\> -application ontapi
-authmethod <password\> -role <name_of_role_in_step_1\> –vserver
<svm_name\> -comment "user_description"
```

3. Ordnen Sie die Rolle dem Benutzer zu:

```
security login modify username <user_name\> –vserver <svm_name\> -role
<role_name\> -application ontapi -application console -authmethod
<password\>
```

Verwenden Von System Manager

Führen Sie die folgenden Schritte im ONTAP System Manager durch:

1. **Erstellen Sie eine benutzerdefinierte Rolle**:
   a. Um eine benutzerdefinierte Rolle auf Cluster-Ebene zu erstellen, wählen Sie **Cluster > Einstellungen** aus.

      (Oder) um eine benutzerdefinierte Rolle auf SVM-Ebene zu erstellen, wählen Sie **Storage > Storage VMs > > `required SVM` Einstellungen > Benutzer und Rollen** aus.

   b. Wählen Sie das Pfeilsymbol (→) neben **Users and Roles**.
   c. Wählen Sie unter **Rollen +Hinzufügen** aus.
   d. Definieren Sie die Regeln für die Rolle und klicken Sie auf **Speichern**.
2. **Rolle dem Trident-Benutzer zuordnen**: + Führen Sie auf der Seite **Benutzer und Rollen** folgende Schritte aus:
   a. Wählen Sie unter **Benutzer** das Symbol Hinzufügen **+**.
   b. Wählen Sie den gewünschten Benutzernamen aus, und wählen Sie im Dropdown-Menü für **Rolle** eine Rolle aus.
   c. Klicken Sie Auf **Speichern**.

Weitere Informationen finden Sie auf den folgenden Seiten:

- "Benutzerdefinierte Rollen für die Administration von ONTAP" Oder "Definieren benutzerdefinierter Rollen"
- "Arbeiten Sie mit Rollen und Benutzern"

Management der NFS-Exportrichtlinien

Trident verwendet NFS-Exportrichtlinien, um den Zugriff auf die von ihm bereitstehenden Volumes zu kontrollieren.

Trident bietet zwei Optionen für die Arbeit mit Exportrichtlinien:

- Trident kann die Exportrichtlinie selbst dynamisch managen. In diesem Betriebsmodus gibt der Storage-Administrator eine Liste von CIDR-Blöcken an, die zulässige IP-Adressen darstellen. Trident fügt der Exportrichtlinie automatisch zum Veröffentlichungszeitpunkt anwendbare Node-IPs hinzu, die in diesen Bereichen fallen. Wenn keine CIDRs angegeben werden, werden alternativ alle global scoped Unicast-IPs, die auf dem Knoten gefunden werden, auf dem das Volume veröffentlicht wird, zur Exportrichtlinie hinzugefügt.
- Storage-Administratoren können eine Exportrichtlinie erstellen und Regeln manuell hinzufügen. Trident verwendet die standardmäßige Exportrichtlinie, es sei denn, in der Konfiguration ist ein anderer Name für die Exportrichtlinie angegeben.

Dynamisches Managen von Exportrichtlinien

Trident bietet die Möglichkeit, Richtlinien für den Export für 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. Dies hilft darüber hinaus, den Zugriff auf das Storage-Cluster nur auf Arbeitsknoten zu beschränken, die Volumes mounten und IPs im angegebenen Bereich haben. Dies unterstützt ein granulares und automatisiertes Management.

> ⓘ 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-economy
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 Trident zu zuweisen.

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

- `autoExportPolicy` Ist auf eingestellt `true`. Das zeigt an, dass Trident für jedes mit diesem Backend für die SVM bereitgestellte Volume eine Exportrichtlinie erstellt `svm1` und das Hinzufügen und Löschen von Regeln mithilfe von Adressblöcken handhabt `autoexportCIDRs`. Bis ein Volume mit einem Node verbunden ist, verwendet das Volume eine leere Exportrichtlinie ohne Regeln, um unerwünschten Zugriff auf dieses Volume zu verhindern. Wenn ein Volume auf einem Node veröffentlicht wird, erstellt Trident eine Exportrichtlinie mit demselben Namen wie der zugrunde liegende qtree, der die Node-IP innerhalb des angegebenen CIDR-Blocks enthält. Diese IPs werden auch zu der von der übergeordneten FlexVol volume verwendeten Exportrichtlinie hinzugefügt
  - Beispiel:
    - Back-End UUID 403b5326-8482-40db-96d0-d83fb3f4daec
    - `autoExportPolicy` Stellen Sie auf ein `true`
    - Speicherpräfix `trident`
    - PVC UUID a79bcf5f-7b6d-4a40-9876-e2551f159c1c
    - Qtree namens Trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c erstellt eine Exportrichtlinie für die FlexVol namens `trident-403b5326-8482-40db96d0-d83fb3f4daec`, eine Exportrichtlinie für den genannten qtree `trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c` und eine leere Exportrichtlinie mit dem Namen `trident_empty` auf der SVM. Die Regeln für die FlexVol-Exportrichtlinie stellen eine Überlagerung sämtlicher Regeln dar, die in den qtree Exportrichtlinien enthalten sind. Die leere Exportrichtlinie wird von allen Volumes wiederverwendet, die nicht angehängt sind.
- `autoExportCIDRs` Enthält eine Liste von Adressblöcken. Dieses Feld ist optional und standardmäßig [„0.0.0.0/0“, „:/0“]. Wenn nicht definiert, fügt Trident alle global scoped Unicast-Adressen, die auf den Worker-Knoten mit Publikationen gefunden wurden, hinzu.

In diesem Beispiel wird der `192.168.0.0/24` Adressraum angegeben. Das gibt an, dass Kubernetes-Node-IPs, die mit Publikationen innerhalb dieses Adressbereichs liegen, zur von Trident erstellten Exportrichtlinie hinzugefügt werden. Wenn Trident einen Knoten registriert, auf dem es ausgeführt wird, ruft es die IP-Adressen des Knotens ab und prüft diese anhand der in bereitgestellten Adressblöcke `autoExportCIDRs`. Nach dem Filtern der IPs erstellt Trident zum Zeitpunkt der Veröffentlichung die Exportrichtlinien für die Client-IPs für den Knoten, auf dem er veröffentlicht wird.

Sie können und `autoExportCIDRs` für Back-Ends aktualisieren `autoExportPolicy`, 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 für ein Backend deaktivieren `autoExportPolicy` und auf eine manuell erstellte Exportrichtlinie zurückgreifen. Dazu muss der Parameter in Ihrer Backend-Konfiguration festgelegt `exportPolicy` werden.

Nachdem Trident ein Backend erstellt oder aktualisiert hat, können Sie das Backend mit oder der entsprechenden `tridentbackend` CRD überprüfen `tridentctl`:

```
./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>
      fileSystemType: ext4
```

Wenn ein Node entfernt wird, überprüft Trident alle Exportrichtlinien, um die dem Node entsprechenden Zugriffsregeln zu entfernen. Indem Trident diese Node-IP aus den Exportrichtlinien der Managed Back-Ends entfernt, verhindert es abnormale Mounts, sofern diese IP nicht von einem neuen Node im Cluster wiederverwendet wird.

Bei zuvor vorhandenen Back-Ends wird durch die Aktualisierung des Backend mit `tridentctl update backend` sichergestellt, dass Trident die Exportrichtlinien automatisch verwaltet. Dadurch werden zwei neue Export-Richtlinien erstellt, die nach der UUID und dem qtree-Namen des Backends benannt sind, wenn sie benötigt werden. Volumes, die auf dem Backend vorhanden sind, verwenden die neu erstellten Exportrichtlinien, nachdem sie abgehängt und wieder gemountet wurden.

> 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 Trident Pod auf dem Node neu starten. Trident aktualisiert dann die Exportrichtlinie für Back-Ends, die es verwaltet, um diese IP-Änderung widerzuspiegeln.

Vorbereitung zur Bereitstellung von SMB Volumes

Mit etwas zusätzlicher Vorbereitung können Sie SMB-Volumes mit Treibern bereitstellen `ontap-nas`.

> Sie müssen sowohl NFS- als auch SMB/CIFS-Protokolle auf der SVM konfigurieren, um ein SMB-Volume für On-Premises-ONTAP Cluster zu erstellen `ontap-nas-economy`. Ist eines dieser Protokolle nicht konfiguriert, schlägt die Erstellung von SMB Volumes fehl.

`autoExportPolicy` Wird für SMB-Volumes nicht unterstützt.

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 2022 ausgeführt wird. Trident unterstützt nur SMB Volumes, die in Pods gemountet sind, die nur auf Windows Nodes ausgeführt werden.
- Mindestens ein Trident-Schlüssel, der Ihre Active Directory-Anmeldeinformationen enthält. So generieren Sie ein Geheimnis `smbcreds`:

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

- Ein CSI-Proxy, der als Windows-Dienst konfiguriert ist. Informationen zum Konfigurieren `csi-proxy` von finden Sie unter "GitHub: CSI-Proxy" oder "GitHub: CSI Proxy für Windows" für Kubernetes-Nodes, die unter Windows ausgeführt werden.

Schritte

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

   SMB-Freigaben sind für Amazon FSX for ONTAP erforderlich.

   Sie können die SMB-Administratorfreigaben auf zwei Arten erstellen, entweder mit dem "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` Befehl überprüft den in der Option -path angegebenen Pfad während der Erstellung von Freigaben. 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
```

   Weitere Informationen finden Sie unter"Erstellen Sie eine SMB-Freigabe".

2. Beim Erstellen des Backend müssen Sie Folgendes konfigurieren, um SMB-Volumes festzulegen. Für alle

FSX für ONTAP Backend-Konfigurationsoptionen, siehe "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 Verwaltungskonsole oder der ONTAP-CLI erstellt wurde, einen Namen, über den Trident die SMB-Freigabe erstellen kann, oder Sie können den Parameter leer lassen, um den Zugriff auf gemeinsame Freigaben auf Volumes zu verhindern. Dieser Parameter ist für On-Premises-ONTAP optional. Dieser Parameter ist für Amazon FSX for ONTAP-Back-Ends erforderlich und darf nicht leer sein.	`smb-share`
`nasType`	**Muss auf.** gesetzt werden `smb` Wenn Null, wird standardmäßig auf `nfs`.	`smb`
`securityStyle`	Sicherheitstyp für neue Volumes. **Muss für SMB Volumes auf oder `mixed` gesetzt werden `ntfs`.**	`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 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` Oder `ontap-nas-flexgroup`
`backendName`	Benutzerdefinierter Name oder das Storage-Backend	Treibername + „_“ + DatenLIF

Parameter	Beschreibung	Standard
`managementLIF`	IP-Adresse eines Clusters oder einer SVM-Management-LIF Ein vollständig qualifizierter Domain-Name (FQDN) kann angegeben werden. Kann so eingestellt werden, dass IPv6-Adressen verwendet werden, wenn 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]`. Informationen über die nahtlose MetroCluster-Umschaltung finden Sie im Beispiel: MetroCluster.	„10.0.0.1“, „[2001:1234:abcd::fefe]“
`dataLIF`	IP-Adresse des LIF-Protokolls. NetApp empfiehlt die Angabe `dataLIF`. Wenn nicht angegeben, ruft Trident die DatenLIFs von der SVM ab. Sie können einen vollständig qualifizierten Domänennamen (FQDN) angeben, der für die NFS-Mount-Vorgänge verwendet werden soll. Dadurch können Sie ein Round-Robin-DNS erstellen, um den Lastausgleich über mehrere DatenLIFs hinweg zu ermöglichen. Kann nach der Anfangseinstellung geändert werden. Siehe . Kann so eingestellt werden, dass IPv6-Adressen verwendet werden, wenn 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]`. **Für MetroCluster weglassen.** Siehe Beispiel: MetroCluster.	Angegebene Adresse oder abgeleitet von SVM, falls nicht angegeben (nicht empfohlen)
`svm`	Zu verwendende virtuelle Speichermaschine **omit für MetroCluster.** Siehe Beispiel: MetroCluster.	Abgeleitet, wenn eine SVM `managementLIF` angegeben wird
`autoExportPolicy`	Aktivieren Sie die automatische Erstellung von Exportrichtlinien und aktualisieren Sie [Boolean]. Mithilfe der `autoExportPolicy` Optionen und `autoExportCIDRs` kann Trident Exportrichtlinien automatisch managen.	Falsch
`autoExportCIDRs`	Liste der CIDRs, nach denen die Node-IPs von Kubernetes gegen gefiltert werden sollen, wenn `autoExportPolicy` aktiviert ist. Mithilfe der `autoExportPolicy` Optionen und `autoExportCIDRs` kann Trident Exportrichtlinien automatisch managen.	[„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 Bei Verwendung von ONTAP-nas-Economy und einem storagePrefix, das aus 24 oder mehr Zeichen besteht, ist das Storage-Präfix für die qtrees nicht eingebettet, obwohl es sich im Volume-Namen befindet.	trident
`aggregate`	Aggregat für die Bereitstellung (optional, wenn eingestellt, muss der SVM zugewiesen werden) Für den `ontap-nas-flexgroup` Treiber wird diese Option ignoriert. Falls nicht, können alle verfügbaren Aggregate verwendet werden, um ein FlexGroup Volume bereitzustellen. Wenn das Aggregat in einer SVM aktualisiert wird, wird es automatisch in Trident aktualisiert, indem es die SVM abfragt, ohne den Trident Controller neu starten zu müssen. Wenn Sie ein bestimmtes Aggregat in Trident für die Bereitstellung von Volumes konfiguriert haben, wird das Back-End Trident bei der Abfrage des SVM-Aggregats in den Status „Fehlgeschlagen“ verschoben. Sie müssen entweder das Aggregat zu einem auf der SVM vorhandenen Aggregat ändern oder es komplett entfernen, um das Back-End wieder online zu schalten.	„“
`limitAggregateUsage`	Bereitstellung fehlgeschlagen, wenn die Nutzung über diesem Prozentsatz liegt. **Gilt nicht für Amazon FSX für ONTAP**	„“ (nicht standardmäßig durchgesetzt)

Parameter	Beschreibung	Standard
FlexgroupAggregateList	Liste der Aggregate für die Bereitstellung (optional, muss dieser SVM zugewiesen werden, falls festgelegt) Zur Bereitstellung eines FlexGroup Volumes werden alle der SVM zugewiesenen Aggregate verwendet. Unterstützt für den **ONTAP-nas-FlexGroup**-Speichertreiber. Bei einer Aktualisierung der Aggregatliste in der SVM wird die Liste automatisch in Trident aktualisiert, indem die SVM abgefragt wird, ohne den Trident Controller neu starten zu müssen. Wenn Sie in Trident eine bestimmte Aggregatliste für die Bereitstellung von Volumes konfiguriert haben und die Aggregatliste umbenannt oder von SVM entfernt wird, wird das Backend in Trident in den Fehlerzustand verschoben, während es das SVM Aggregat abfragt. Sie müssen entweder die Aggregatliste zu einer auf der SVM vorhandenen ändern oder sie komplett entfernen, um das Backend wieder online zu machen.	„“
`limitVolumeSize`	Bereitstellung fehlgeschlagen, wenn die angeforderte Volume-Größe über diesem Wert liegt. Beschränkt darüber hinaus die maximale Größe der Volumes, die es für qtrees managt, und `qtreesPerFlexvol` ermöglicht die Anpassung der maximalen Anzahl an qtrees pro FlexVol volume	„“ (nicht standardmäßig durchgesetzt)
`debugTraceFlags`	Fehler-Flags bei der Fehlerbehebung beheben. Beispiel, {„API“:false, „method“:true} nicht verwenden `debugTraceFlags`, es sei denn, Sie beheben die Fehlerbehebung und benötigen einen detaillierten Log Dump.	Null
`nasType`	Konfiguration der Erstellung von NFS- oder SMB-Volumes Optionen sind `nfs`, `smb` oder Null. Einstellung auf null setzt standardmäßig auf NFS-Volumes.	`nfs`

Parameter	Beschreibung	Standard
`nfsMountOptions`	Kommagetrennte Liste von NFS-Mount-Optionen. Die Mount-Optionen für persistente Kubernetes-Volumes werden normalerweise in Storage-Klassen angegeben. Wenn jedoch keine Mount-Optionen in einer Storage-Klasse angegeben sind, verwendet Trident die Mount-Optionen, die in der Konfigurationsdatei des Storage-Backends angegeben sind. Wenn in der Storage-Klasse oder in der Konfigurationsdatei keine Mount-Optionen angegeben sind, legt Trident keine Mount-Optionen auf einem zugeordneten persistenten Volume fest.	„“
`qtreesPerFlexvol`	Maximale Ques pro FlexVol, muss im Bereich [50, 300] liegen	„200“
`smbShare`	Sie können eine der folgenden Optionen angeben: Den Namen einer SMB-Freigabe, die mit der Microsoft Verwaltungskonsole oder der ONTAP-CLI erstellt wurde, einen Namen, über den Trident die SMB-Freigabe erstellen kann, oder Sie können den Parameter leer lassen, um den Zugriff auf gemeinsame Freigaben auf Volumes zu verhindern. Dieser Parameter ist für On-Premises-ONTAP optional. Dieser Parameter ist für Amazon FSX for ONTAP-Back-Ends erforderlich und darf nicht leer sein.	`smb-share`
`useREST`	Boolescher Parameter zur Verwendung von ONTAP REST-APIs. `useREST` Wenn auf festgelegt `true`, verwendet Trident ONTAP REST APIs, um mit dem Backend zu kommunizieren; wenn auf gesetzt `false`, verwendet Trident ONTAPI (ZAPI) Aufrufe, um mit dem Backend zu kommunizieren. Diese Funktion erfordert ONTAP 9.11.1 und höher. Darüber hinaus muss die verwendete ONTAP-Anmelderolle Zugriff auf die Anwendung haben `ontapi` . Dies wird durch die vordefinierten `vsadmin` Rollen und `cluster-admin` erreicht. Ab Trident 24.06-Version und ONTAP 9.15.1 oder höher `useREST` ist standardmäßig auf gesetzt `true`. Wechseln Sie `useREST` zu `false` ONTAPI (ZAPI)-Aufrufe verwenden.	`true` Für ONTAP 9.15.1 oder höher, andernfalls `false`.
`limitVolumePoolSize`	Maximale anforderbare FlexVol-Größe bei Verwendung von Qtrees im ONTAP-nas-Economy Backend.	„“ (nicht standardmäßig durchgesetzt)
`denyNewVolumePools`	Schränkt das `ontap-nas-economy` Erstellen neuer FlexVol Volumes für Back-Ends ein, um ihre qtrees zu enthalten Zur Bereitstellung neuer PVS werden nur vorbestehende FlexVols verwendet.	

Back-End-Konfigurationsoptionen für die Bereitstellung von Volumes

Mit diesen Optionen können Sie die Standardbereitstellung im Abschnitt der Konfiguration steuern `defaults`.

Ein Beispiel finden Sie unten in den Konfigurationsbeispielen.

Parameter	Beschreibung	Standard
`spaceAllocation`	Platzzuweisung für Qtrees	„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` „keine“ ist, 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, Standardeinstellung ist `false`. NVE muss im Cluster lizenziert und aktiviert sein, damit diese Option verwendet werden kann. Wenn auf dem Backend NAE aktiviert ist, wird jedes in Trident bereitgestellte Volume NAE aktiviert. Weitere Informationen finden Sie unter "Funktionsweise von Trident mit NVE und NAE".	„Falsch“
`tieringPolicy`	Tiering-Richtlinie, die zu „keinen“ verwendet wird	
`unixPermissions`	Modus für neue Volumes	„777“ für NFS Volumes; leer (nicht zutreffend) für SMB Volumes
`snapshotDir`	Steuert den Zugriff auf das `.snapshot` Verzeichnis	„Wahr“ für NFSv4 „falsch“ für NFSv3
`exportPolicy`	Zu verwendende Exportrichtlinie	„Standard“
`securityStyle`	Sicherheitstyp für neue Volumes. NFS-Unterstützung `mixed` und `unix` Sicherheitsstile. SMB-Unterstützung `mixed` und `ntfs` Sicherheitsstile.	NFS-Standard ist `unix`. SMB-Standard ist `ntfs`.
`nameTemplate`	Vorlage zum Erstellen benutzerdefinierter Volume-Namen.	„“

Für die Verwendung von QoS-Richtliniengruppen mit Trident ist ONTAP 9 8 oder höher erforderlich. Sie sollten eine nicht gemeinsam genutzte QoS-Richtliniengruppe verwenden und sicherstellen, dass die Richtliniengruppe auf jede Komponente einzeln angewendet wird. Eine Shared-QoS-Richtliniengruppe erzwingt die 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` verwendet Trident jetzt eine neue Berechnung, um sicherzustellen, dass die FlexVol korrekt mit der Snapshot Reserve Prozentsatz und PVC-Größe ist. Wenn der Benutzer eine PVC anfordert, erstellt Trident mithilfe 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 angefordert hat und einen prozentualen Anteil davon darstellt. `snapshotReserve` Bei Trident 21.07 fordert der Benutzer den beschreibbaren Speicherplatz an, und Trident definiert die `snapshotReserve` Zahl als Prozentsatz des gesamten Volumes. 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 5/.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` Befehl sollte die Ergebnisse ähnlich wie in diesem Beispiel anzeigen:

Vorhandene Back-Ends von vorherigen Installationen stellen Volumes wie oben beschrieben beim Upgrade von Trident bereit. Bei Volumes, die Sie vor dem Upgrade erstellt haben, sollten Sie die Größe ihrer Volumes entsprechend der zu beobachtenden Änderung anpassen. Ein Beispiel: Eine PVC mit 2 gib und einer früheren Version `snapshotReserve=50` führte zu einem Volume, das 1 gib schreibbaren 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 konfigurieren, um zu vermeiden, dass die Backend-Definition nach Umschaltung und Switchback während manuell aktualisiert "SVM-Replizierung und Recovery"werden muss.

Geben Sie für ein nahtloses Switchover und Switchback die SVM mit an `managementLIF` und lassen Sie die Parameter und `svm` weg `dataLIF`. 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 Backend-Konfiguration. `clientCertificate`, `clientPrivateKey` Und `trustedCACertificate` (optional, wenn vertrauenswürdige CA verwendet wird) werden eingetragen `backend.json` und nehmen die base64-kodierten Werte des Clientzertifikats, des privaten Schlüssels und des vertrauenswürdigen CA-Zertifikats an.

```
---
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

Dieses Beispiel zeigt, wie Sie Trident anweisen können, dynamische Exportrichtlinien zu verwenden, um die Exportrichtlinie automatisch zu erstellen und zu verwalten. Dies funktioniert für die und `ontap-nas-flexgroup`-Treiber gleich `ontap-nas-economy`.

```
---
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` die Verwendung 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_
```

Back-End-Konfigurationsbeispiel mit nameTemplate

```
---
version: 1
storageDriverName: ontap-nas
backendName: ontap-nas-backend
managementLIF: <ip address>
svm: svm0
username: <admin>
password: <password>
defaults:
  nameTemplate:
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.vo\
    lume.RequestName}}"
labels:
  cluster: ClusterA
  PVC: "{{.volume.Namespace}}_{{.volume.RequestName}}"
```

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 „none“, `spaceAllocation` „false“ und „false `encryption`“. Die virtuellen Pools werden im Abschnitt Speicher definiert.

Trident legt die Bereitstellungsetiketten im Feld „Kommentare“ fest. Kommentare werden auf FlexVol für oder FlexGroup für `ontap-nas-flexgroup` gesetzt `ontap-nas`. Trident kopiert bei der Bereitstellung alle Labels, die sich in einem virtuellen Pool befinden, auf das Storage-Volume. Storage-Administratoren können Labels je virtuellen Pool definieren und Volumes nach Label gruppieren.

In diesen Beispielen legen einige Speicherpools eigene Werte , `spaceAllocation` und fest `spaceReserve`, und `encryption` 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. Mit dem `parameters.selector` Feld ruft jede StorageClass ab, welche virtuellen Pools zum Hosten eines Volumes verwendet werden können. Auf dem Volume werden die Aspekte im ausgewählten virtuellen Pool definiert.

- Die `protection-gold` StorageClass wird dem ersten und zweiten virtuellen Pool im Backend zugeordnet `ontap-nas-flexgroup`. 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"
```

- Die `protection-not-gold` StorageClass wird dem dritten und vierten virtuellen Pool im Backend zugeordnet `ontap-nas-flexgroup`. 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"
```

- Die `app-mysqldb` StorageClass wird dem vierten virtuellen Pool im Backend zugeordnet `ontap-nas`. 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"
```

- Die `protection-silver-creditpoints-20k` StorageClass wird dem dritten virtuellen Pool im Backend zugeordnet `ontap-nas-flexgroup`. 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"
```

- Die `creditpoints-5k` StorageClass wird dem dritten virtuellen Pool im Backend und dem zweiten virtuellen Pool im Backend `ontap-nas-economy` zugeordnet `ontap-nas`. 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"
```

Trident entscheidet, welcher virtuelle Pool ausgewählt wird, und stellt sicher, dass die Speicheranforderungen erfüllt werden.

Nach der Erstkonfiguration aktualisieren `dataLIF`

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

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

Wenn PVCs an einen oder mehrere Pods angeschlossen sind, müssen Sie alle entsprechenden Pods herunterfahren und sie dann wieder erstellen, damit die neue DataLIF wirksam wird.

Amazon FSX für NetApp ONTAP

Verwenden Sie Trident mit Amazon FSX für NetApp ONTAP

"Amazon FSX für NetApp ONTAP" Ist ein vollständig gemanagter AWS Service, mit dem Kunden Filesysteme mit NetApp ONTAP Storage-Betriebssystem 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.

Die Integration des Filesystems Amazon FSX for NetApp ONTAP mit Trident stellt sicher, dass Kubernetes-Cluster, die in Amazon Elastic Kubernetes Service (EKS) ausgeführt werden, persistente Block- und dateibasierte Volumes mit ONTAP bereitstellen können.

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. Mit Amazon FSX für NetApp ONTAP wird als gemanagtes Dateisystem in der Cloud zur Verfügung gestellt. Der neue Dateisystemtyp heißt **NetApp ONTAP**.

Durch den Einsatz von Trident mit Amazon FSX for NetApp ONTAP können Sie sicherstellen, dass Kubernetes-Cluster, die im Amazon Elastic Kubernetes Service (EKS) ausgeführt werden, persistente Block- und dateibasierte Volumes bereitstellen können, die von ONTAP unterstützt werden.

Anforderungen

"Trident-Anforderungen erfüllt"Um FSX for ONTAP mit Trident zu integrieren, benötigen Sie zusätzlich:

- Ein vorhandener Amazon EKS Cluster oder selbstverwalteter Kubernetes-Cluster mit `kubectl` installierter Installation.
- Ein vorhandenes Amazon FSX for NetApp ONTAP-Filesystem und eine Storage Virtual Machine (SVM), die über die Worker-Nodes Ihres Clusters erreichbar ist.
- Worker-Knoten, die für vorbereitet sind"NFS oder iSCSI".

Stellen Sie sicher, dass Sie die erforderlichen Schritte zur Knotenvorbereitung für Amazon Linux und Ubuntu (Amis) je nach EKS AMI-Typ befolgen "Amazon Machine Images".

Überlegungen

- SMB Volumes:
  - SMB-Volumes werden nur über den Treiber unterstützt `ontap-nas`.

    - SMB-Volumes werden vom Trident EKS Add-on nicht unterstützt.
    - Trident unterstützt nur SMB Volumes, die in Pods gemountet sind, die nur auf Windows Nodes ausgeführt werden. Weitere Informationen finden Sie unter "Vorbereitung zur Bereitstellung von SMB Volumes" .
- Vor Trident 24.02 konnten auf Amazon FSX-Dateisystemen erstellte Volumes, bei denen automatische Backups aktiviert sind, von Trident nicht gelöscht werden. Um dieses Problem in Trident 24.02 oder höher zu vermeiden, geben Sie `apikey` in der Backend-Konfigurationsdatei für AWS FSX für ONTAP , AWS `apiRegion` und AWS `secretKey` an `fsxFilesystemID`.

> Wenn Sie eine IAM-Rolle als Trident angeben, können Sie die Felder , `apiKey` und `secretKey` explizit als Trident auslassen `apiRegion`. Weitere Informationen finden Sie unter "FSX für ONTAP Konfigurationsoptionen und Beispiele".

Authentifizierung

Trident bietet zwei Authentifizierungsmodi.

- Anmeldeinformationsbasiert (empfohlen): Speichert Anmeldeinformationen sicher in AWS Secrets Manager. Sie können den Benutzer für Ihr Dateisystem oder den für Ihre SVM konfigurierten Benutzer verwenden `fsxadmin` `vsadmin` .

> Trident wird voraussichtlich als SVM-Benutzer oder als Benutzer mit einem anderen Namen, der dieselbe Rolle hat, ausgeführt `vsadmin`. Amazon FSX for NetApp ONTAP hat einen `fsxadmin` Benutzer, der den ONTAP-Cluster-Benutzer nur eingeschränkt ersetzt `admin`. Wir empfehlen die Verwendung `vsadmin` mit Trident.

- Zertifikat-basiert: Trident kommuniziert über ein auf Ihrer SVM installiertes Zertifikat mit der SVM auf Ihrem FSX Filesystem.

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

- "ONTAP NAS-Authentifizierung"
- "ONTAP SAN-Authentifizierung"

Getestete Amazon Machine Images (Amis)

Der EKS Cluster unterstützt zwar verschiedene Betriebssysteme, AWS hat jedoch bestimmte Amazon Machine Images (Amis) für Container und EKS optimiert. Die folgenden Amis wurden mit Trident 24.10 getestet.

AMI	NAS	NAS-Economy	San	SAN-Economy
AL2023_x86_64_STANDARD	Ja.	Ja.	Ja.	Ja.
AL2_x86_64	Ja.	Ja.	Ja**	Ja**
BOTTLEROCKET_x86_64	Ja*	Ja.	K. A.	K. A.

AL2023_ARM_64_STANDARD	Ja.	Ja.	Ja.	Ja.
AL2_ARM_64	Ja.	Ja.	Ja**	Ja**
BOTTLEROCKET_ARM_64	Ja*	Ja.	K. A.	K. A.

- *Muss "nolock" in Mount-Optionen verwenden.
- ** Das PV kann nicht gelöscht werden, ohne den Knoten neu zu starten

> Wenn Ihr gewünschtes AMI hier nicht aufgeführt ist, bedeutet dies nicht, dass es nicht unterstützt wird, sondern dass es einfach nicht getestet wurde. Diese Liste dient als Leitfaden für Amis, die bekannt sind zu arbeiten.

Tests durchgeführt mit:

- EKS-Version: 1.30
- Installationsmethode: Helm und als AWS Add-on
- Für NAS wurden sowohl NFSv3 als auch NFSv4.1 getestet.
- Für SAN wurde nur iSCSI getestet, nicht NVMe-of.

Durchgeführte Tests:

- Erstellen: Storage-Klasse, pvc, POD
- Löschen: Pod, pvc (normal, qtree/lun – Economy, NAS mit AWS Backup)

Weitere Informationen

- "Dokumentation zu Amazon FSX für NetApp ONTAP"
- "Blogbeitrag zu Amazon FSX für NetApp ONTAP"

IAM-Rolle und AWS Secret erstellen

Sie können Kubernetes-Pods für den Zugriff auf AWS-Ressourcen konfigurieren, indem Sie sich als AWS IAM-Rolle authentifizieren anstatt dafür explizite AWS-Anmeldedaten bereitstellen zu müssen.

> Um sich mit einer AWS IAM-Rolle zu authentifizieren, müssen Sie über ein Kubernetes-Cluster mit EKS verfügen.

Erstellen Sie den AWS Secrets Manager Secret

Da Trident APIs gegen einen FSX vserver ausstellen wird, um den Speicher für Sie zu verwalten, benötigt es Anmeldeinformationen, um dies zu tun. Diese Zugangsdaten können Sie sicher über ein AWS Secrets Manager Secret übermitteln. Daher, wenn Sie noch nicht über eine, müssen Sie ein AWS Secrets Manager Secret, die die Anmeldeinformationen für das vsadmin-Konto enthält erstellen.

Dieses Beispiel erstellt einen AWS Secret Manager Secret, um Trident CSI-Anmeldedaten zu speichern:

```
aws secretsmanager create-secret --name trident-secret --description
"Trident CSI credentials"\
  --secret-string
"{\"username\":\"vsadmin\",\"password\":\"<svmpassword>\"}"
```

IAM-Richtlinie erstellen

Für die korrekte Ausführung von Trident-Berechtigungen sind ebenfalls AWS-Berechtigungen erforderlich. Daher müssen Sie eine Richtlinie erstellen, die Trident die erforderlichen Berechtigungen erteilt.

In den folgenden Beispielen wird eine IAM-Richtlinie über die AWS-CLI erstellt:

```
aws iam create-policy --policy-name AmazonFSxNCSIDriverPolicy --policy
-document file://policy.json
  --description "This policy grants access to Trident CSI to FSxN and
Secrets manager"
```

JSON-Beispiel für Richtlinien:

```
{
  "Statement": [
    {
      "Action": [
        "fsx:DescribeFileSystems",
        "fsx:DescribeVolumes",
        "fsx:CreateVolume",
        "fsx:RestoreVolumeFromSnapshot",
        "fsx:DescribeStorageVirtualMachines",
        "fsx:UntagResource",
        "fsx:UpdateVolume",
        "fsx:TagResource",
        "fsx:DeleteVolume"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Action": "secretsmanager:GetSecretValue",
      "Effect": "Allow",
      "Resource": "arn:aws:secretsmanager:<aws-region>:<aws-account-
id>:secret:<aws-secret-manager-name>*"
    }
  ],
  "Version": "2012-10-17"
}
```

Erstellen Sie eine IAM-Rolle für das Dienstkonto

Nachdem Sie die Richtlinie erstellt haben, können Sie sie beim Erstellen der Rolle verwenden, die dem Servicekonto zugewiesen wird, unter dem Trident ausgeführt wird:

AWS CLI

```
aws iam create-role --role-name AmazonEKS_FSxN_CSI_DriverRole \
  --assume-role-policy-document file://trust-relationship.json
```

Trust-Relationship.json-Datei:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::<account_id>:oidc-
provider/<oidc_provider>"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "<oidc_provider>:aud": "sts.amazonaws.com",
          "<oidc_provider>:sub":
"system:serviceaccount:trident:trident-controller"
        }
      }
    }
  ]
}
```

Aktualisieren Sie die folgenden Werte in der `trust-relationship.json` Datei:

- **<account_id>** - Ihre AWS-Konto-ID
- **<oidc_provider>** - das OIDC Ihres EKS-Clusters. Sie können den oidc_Provider erhalten, indem Sie Folgendes ausführen:

  ```
  aws eks describe-cluster --name my-cluster --query
  "cluster.identity.oidc.issuer"\
    --output text | sed -e "s/^https:\/\///"
  ```

Die IAM-Rolle mit der IAM-Richtlinie verknüpfen:

Nachdem die Rolle erstellt wurde, hängen Sie die Richtlinie (die im obigen Schritt erstellt wurde) mit diesem Befehl an die Rolle an:

```
aws iam attach-role-policy --role-name my-role --policy-arn <IAM policy
ARN>
```

Verify OICD Provider is associated:

Vergewissern Sie sich, dass der OIDC-Anbieter dem Cluster zugeordnet ist. Sie können sie mit diesem Befehl überprüfen:

```
aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
```

Wenn die Ausgabe leer ist, weisen Sie IAM OIDC mit dem folgenden Befehl dem Cluster zu:

```
eksctl utils associate-iam-oidc-provider --cluster $cluster_name
--approve
```

Eksctl

Im folgenden Beispiel wird eine IAM-Rolle für das Dienstkonto in EKS erstellt:

```
eksctl create iamserviceaccount --name trident-controller --namespace
trident \
  --cluster <my-cluster> --role-name AmazonEKS_FSxN_CSI_DriverRole
--role-only \
  --attach-policy-arn <IAM-Policy ARN> --approve
```

Installation Von Trident

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.

Sie können Trident mit einer der folgenden Methoden installieren:

- Helm
- EKS-Add-on

Wenn Sie die Snapshot-Funktionalität nutzen möchten, installieren Sie das Add-On für den CSI-Snapshot-Controller. Weitere Informationen finden Sie unter "Snapshot-Funktionalität für CSI-Volumes aktivieren" .

Trident über Helm installieren

1. Laden Sie das Trident-Installationspaket herunter

   Das Trident-Installationspaket enthält alles, was Sie für die Bereitstellung des Trident-Bedieners und die Installation von Trident benötigen. Laden Sie die neueste Version des Trident-Installers herunter und

extrahieren Sie sie aus dem Abschnitt „Assets“ auf GitHub.

```
wget
https://github.com/NetApp/trident/releases/download/v25.02.0/trident-
installer-25.02.0.tar.gz
tar -xf trident-installer-25.02.0.tar.gz
cd trident-installer
```

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

   Das folgende Beispiel installiert Trident und setzt das `cloud-provider` Flag auf `$CP`, und `cloud-identity` auf `$CI`:

```
helm install trident trident-operator-100.2502.0.tgz \
--set cloudProvider="AWS" \
--set cloudIdentity="'eks.amazonaws.com/role-arn:
arn:aws:iam::<accountID>:role/<AmazonEKS_FSxN_CSI_DriverRole>'" \
--namespace trident \
--create-namespace
```

   Mit dem Befehl können `helm list` Sie Installationsdetails wie Name, Namespace, Diagramm, Status, App-Version und Revisionsnummer überprüfen.

```
helm list -n trident
```

```
NAME                NAMESPACE   REVISION    UPDATED
STATUS       CHART                          APP VERSION

trident-operator    trident     1           2024-10-14 14:31:22.463122
+0300 IDT    deployed     trident-operator-100.2502.0    25.02.0
```

Installieren Sie Trident über das EKS-Add-on

Das Trident EKS Add-on enthält die neuesten Sicherheitspatches 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 sicher, dass folgende Voraussetzungen erfüllt sind, bevor Sie das Trident Add-on für AWS EKS konfigurieren:

- 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

Aktivieren Sie das Trident Add-on für AWS

Eksctl

Mit dem folgenden Beispielbefehl wird das Trident EKS Add-On installiert:

```
eksctl create addon --name netapp_trident-operator --cluster
<cluster_name> \
--service-account-role-arn arn:aws:iam::<account_id>:role/<role_name>
--force
```

Management-Konsole

1. Öffnen Sie die Amazon EKS Konsole unter https://console.aws.amazon.com/eks/home#/clusters.
2. Wählen Sie im linken Navigationsbereich **Cluster** aus.
3. Wählen Sie den Namen des Clusters aus, für den Sie das NetApp Trident-CSI-Add-On konfigurieren möchten.
4. Wählen Sie **Add-ons** und dann **Weitere Add-Ons**.
5. Gehen Sie auf der Seite **Add-ons auswählen** wie folgt vor:
   a. Aktivieren Sie im Abschnitt EKS-Addons des AWS Marketplace das Kontrollkästchen **Trident by NetApp**.
   b. Wählen Sie **Weiter**.
6. Gehen Sie auf der Seite **Ausgewählte Add-Ons konfigurieren**-Einstellungen wie folgt vor:
   a. Wählen Sie die **Version** aus, die Sie verwenden möchten.
   b. Für **IAM-Rolle auswählen** lassen Sie bei **nicht gesetzt**.
   c. Folgen Sie dem **Add-on-Konfigurationsschema** und setzen Sie den Parameter configurationValues im Abschnitt **Konfigurationswerte** auf die Rolle-arn, die Sie im vorherigen Schritt erstellt haben (Wert sollte im folgenden Format sein:

      ```
      {

        "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'"

      }
      ```

      Wenn Sie für die Konfliktlösungsmethode Überschreiben auswählen, können eine oder mehrere Einstellungen für das vorhandene Add-On mit den Amazon EKS-Zusatzeinstellungen überschrieben werden. Wenn Sie diese Option nicht aktivieren und es einen Konflikt mit Ihren bestehenden Einstellungen gibt, schlägt der Vorgang fehl. Sie können die resultierende Fehlermeldung verwenden, um den Konflikt zu beheben. Bevor Sie diese Option auswählen, stellen Sie sicher, dass das Amazon EKS-Add-On keine Einstellungen verwaltet, die Sie selbst verwalten müssen.
7. Wählen Sie **Weiter**.
8. Wählen Sie auf der Seite **Überprüfen und Hinzufügen Erstellen**.

   Nachdem die Installation des Add-ons abgeschlossen ist, wird das installierte Add-on angezeigt.

AWS CLI

1. Erstellen Sie die `add-on.json` Datei:

```
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v25.02.1-eksbuild.1",
  "serviceAccountRoleArn": "<role ARN>",
  "configurationValues": {
    "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'",
    "cloudProvider": "AWS"
  }
}
```

Ersetzen Sie `<role ARN>` diese durch die ARN der Rolle, die im vorherigen Schritt erstellt wurde.

2. Installieren Sie das Trident EKS-Add-On.

```
aws eks create-addon --cli-input-json file://add-on.json
```

Aktualisieren Sie das Trident EKS-Add-On

Eksctl

- Überprüfen Sie die aktuelle Version des FSxN Trident CSI-Add-ons. Ersetzen Sie `my-cluster` den Cluster-Namen.

```
eksctl get addon --name netapp_trident-operator --cluster my-cluster
```

Beispielausgabe:

```
NAME                          VERSION              STATUS    ISSUES
IAMROLE    UPDATE AVAILABLE    CONFIGURATION VALUES
netapp_trident-operator    v25.02.1-eksbuild.1    ACTIVE    0
{"cloudIdentity":"'eks.amazonaws.com/role-arn:
arn:aws:iam::139763910815:role/AmazonEKS_FSXN_CSI_DriverRole'"}
```

- Aktualisieren Sie das Add-on auf die Version, DIE unter UPDATE zurückgegeben wurde, DIE in der Ausgabe des vorherigen Schritts VERFÜGBAR ist.

```
eksctl update addon --name netapp_trident-operator --version
v25.02.1-eksbuild.1 --cluster my-cluster --force
```

Wenn Sie die Option entfernen `--force` und eine der Amazon EKS-Zusatzeinstellungen mit Ihren vorhandenen Einstellungen in Konflikt steht, schlägt die Aktualisierung des Amazon EKS-Zusatzes fehl. Sie erhalten eine Fehlermeldung, um den Konflikt zu beheben. Bevor Sie diese Option angeben, stellen Sie sicher, dass das Amazon EKS-Add-On keine Einstellungen verwaltet, die Sie verwalten müssen, da diese Einstellungen mit dieser Option überschrieben werden. Weitere Informationen zu anderen Optionen für diese Einstellung finden Sie unter "Add-Ons". Weitere Informationen zum Field Management von Amazon EKS Kubernetes finden Sie unter "Außendienstmanagement von Kubernetes".

Management-Konsole

1. Öffnen Sie die Amazon EKS Konsole https://console.aws.amazon.com/eks/home#/clusters.
2. Wählen Sie im linken Navigationsbereich **Cluster** aus.
3. Wählen Sie den Namen des Clusters aus, für den Sie das NetApp Trident-CSI-Add-On aktualisieren möchten.
4. Wählen Sie die Registerkarte **Add-ons**.
5. Wählen Sie **Trident by NetApp** und dann **Bearbeiten**.
6. Gehen Sie auf der Seite **Configure Trident by NetApp** wie folgt vor:
   a. Wählen Sie die **Version** aus, die Sie verwenden möchten.
   b. Erweitern Sie die **Optionale Konfigurationseinstellungen** und ändern Sie sie nach Bedarf.
   c. Wählen Sie **Änderungen speichern**.

AWS CLI

Im folgenden Beispiel wird das EKS-Add-on aktualisiert:

```
aws eks update-addon --cluster-name my-cluster netapp_trident-operator
vpc-cni --addon-version v25.02.1-eksbuild.1 \
    --service-account-role-arn <role-ARN> --configuration-values '{}'
--resolve-conflicts --preserve
```

Deinstallieren Sie das Trident EKS-Add-On bzw. entfernen Sie es

Sie haben zwei Optionen zum Entfernen eines Amazon EKS-Add-ons:

- **Add-on-Software auf Ihrem Cluster beibehalten** – Diese Option entfernt die Amazon EKS-Verwaltung aller Einstellungen. Amazon EKS kann Sie auch nicht mehr über Updates informieren und das Amazon EKS-Add-On automatisch aktualisieren, nachdem Sie ein Update gestartet haben. Die Add-on-Software auf dem Cluster bleibt jedoch erhalten. Mit dieser Option wird das Add-On zu einer selbstverwalteten Installation anstatt zu einem Amazon EKS-Add-on. Bei dieser Option haben Add-on keine Ausfallzeiten. Behalten Sie die Option im Befehl bei `--preserve` , um das Add-on beizubehalten.
- **Entfernen Sie Add-on-Software komplett aus Ihrem Cluster** – NetApp empfiehlt, das Amazon EKS-Add-on nur dann aus Ihrem Cluster zu entfernen, wenn es keine Ressourcen auf Ihrem Cluster gibt, die davon abhängen. Entfernen Sie die `--preserve` Option aus dem `delete` Befehl, um das Add-On zu entfernen.

Wenn dem Add-On ein IAM-Konto zugeordnet ist, wird das IAM-Konto nicht entfernt.

Eksctl

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

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

Management-Konsole

1. Öffnen Sie die Amazon EKS Konsole unter https://console.aws.amazon.com/eks/home#/clusters.
2. Wählen Sie im linken Navigationsbereich **Cluster** aus.
3. Wählen Sie den Namen des Clusters aus, für den Sie das NetApp Trident CSI-Add-On entfernen möchten.
4. Wählen Sie die Registerkarte **Add-ons** und dann **Trident by NetApp**.*
5. Wählen Sie **Entfernen**.
6. Gehen Sie im Dialogfeld **Remove netapp_Trident-Operator confirmation** wie folgt vor:
   a. Wenn Amazon EKS die Verwaltung der Einstellungen für das Add-On einstellen soll, wählen Sie **auf Cluster beibehalten** aus. Führen Sie diese Option aus, wenn Sie die Add-on-Software auf dem Cluster beibehalten möchten, damit Sie alle Einstellungen des Add-ons selbst verwalten können.
   b. Geben Sie **netapp_Trident-Operator** ein.
   c. Wählen Sie **Entfernen**.

AWS CLI

Ersetzen `my-cluster` Sie den Namen des Clusters, und führen Sie dann den folgenden Befehl aus.

```
aws eks delete-addon --cluster-name my-cluster --addon-name
netapp_trident-operator --preserve
```

Konfigurieren Sie das Speicher-Back-End

Integration von ONTAP-SAN- und NAS-Treibern

Um ein Storage-Backend zu erstellen, müssen Sie eine Konfigurationsdatei im JSON- oder YAML-Format erstellen. Die Datei muss den gewünschten Speichertyp (NAS oder SAN), das Dateisystem und die SVM angeben, von der sie abgerufen werden soll, und wie die Authentifizierung mit ihr durchgeführt werden soll. Im folgenden Beispiel wird gezeigt, wie NAS-basierter Storage definiert wird und wie ein AWS-Schlüssel zum Speichern der Zugangsdaten für die zu verwendende SVM verwendet wird:

YAML

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

JSON

```
{
  "apiVersion": "trident.netapp.io/v1",
  "kind": "TridentBackendConfig",
  "metadata": {
    "name": "backend-tbc-ontap-nas"
    "namespace": "trident"
  },
  "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:xxxxxxxx:secret:secret-
name",
      "type": "awsarn"
    }
  }
}
```

Führen Sie die folgenden Befehle aus, um die Trident-Backend-Konfiguration (TBC) zu erstellen und zu validieren:

- Erstellen Sie die Trident-Backend-Konfiguration (TBC) aus der yaml-Datei, und führen Sie den folgenden Befehl aus:

```
kubectl create -f backendconfig.yaml -n trident
```

```
tridentbackendconfig.trident.netapp.io/backend-tbc-ontap-nas created
```

- Überprüfen Sie, ob die Trident-Backend-Konfiguration (TBC) erfolgreich erstellt wurde:

```
Kubectl get tbc -n trident
```

```
NAME                          BACKEND NAME          BACKEND UUID
PHASE   STATUS

backend-tbc-ontap-nas         tbc-ontap-nas         933e0071-66ce-4324-
b9ff-f96d916ac5e9   Bound   Success
```

FSX für ONTAP-Treiber Details

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

- `ontap-san`: Jedes bereitgestellte PV ist eine LUN innerhalb seines eigenen Amazon FSX für NetApp ONTAP-Volumens. Empfohlen für Blocklagerung.
- `ontap-nas`: Jedes bereitgestellte PV ist ein vollständiges Amazon FSX für NetApp ONTAP Volumen. Für NFS und SMB empfohlen.
- `ontap-san-economy`: Jedes bereitgestellte PV ist eine LUN mit einer konfigurierbaren Anzahl von LUNs pro Amazon FSX für NetApp ONTAP Volumen.
- `ontap-nas-economy`: Jedes bereitgestellte PV ist ein qtree, mit einer konfigurierbaren Anzahl von qtrees pro Amazon FSX für NetApp ONTAP Volumen.
- `ontap-nas-flexgroup`: Jedes bereitgestellte PV ist ein vollständiges Amazon FSX für NetApp ONTAP FlexGroup Volumen.

Informationen zum Treiber finden Sie unter "NAS-Treiber" und "SAN-Treiber".

Nachdem die Konfigurationsdatei erstellt wurde, führen Sie diesen Befehl aus, um sie in Ihrem EKS zu erstellen:

```
kubectl create -f configuration_file
```

Führen Sie den folgenden Befehl aus, um den Status zu überprüfen:

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

```
NAME                      BACKEND NAME            BACKEND UUID
PHASE   STATUS
backend-fsx-ontap-nas   backend-fsx-ontap-nas   7a551921-997c-4c37-a1d1-
f2f4c87fa629   Bound   Success
```

Erweiterte Back-End-Konfiguration und Beispiele

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
`managementLIF`	IP-Adresse eines Clusters oder einer SVM-Management-LIF Ein vollständig qualifizierter Domain-Name (FQDN) kann angegeben werden. Kann so eingestellt werden, dass IPv6-Adressen verwendet werden, wenn 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]. Wenn Sie den im `aws` Feld angeben `fsxFilesystemID`, müssen Sie den nicht angeben `managementLIF`, da Trident die SVM-Informationen von AWS abruft `managementLIF`. Daher müssen Sie die Anmeldedaten für einen Benutzer unter der SVM (z. B. vsadmin) angeben, und der Benutzer muss über die Rolle verfügen `vsadmin` .	„10.0.0.1“, „[2001:1234:abcd::fefe]“

Parameter	Beschreibung	Beispiel
`dataLIF`	IP-Adresse des LIF-Protokolls. **ONTAP NAS-Treiber**: NetApp empfiehlt die Angabe der DatenLIF. Wenn nicht angegeben, ruft Trident die DatenLIFs von der SVM ab. Sie können einen vollständig qualifizierten Domänennamen (FQDN) angeben, der für die NFS-Mount-Vorgänge verwendet werden soll. Dadurch können Sie ein Round-Robin-DNS erstellen, um den Lastausgleich über mehrere DatenLIFs hinweg zu ermöglichen. Kann nach der Anfangseinstellung geändert werden. Siehe . **ONTAP-SAN-Treiber**: Geben Sie nicht für iSCSI an. Trident verwendet die selektive LUN-Zuordnung von ONTAP, um die iSCI LIFs zu ermitteln, die für die Einrichtung einer Multi-Path-Sitzung erforderlich sind. Eine Warnung wird erzeugt, wenn dataLIF explizit definiert ist. Kann so eingestellt werden, dass IPv6-Adressen verwendet werden, wenn 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].	
`autoExportPolicy`	Aktivieren Sie die automatische Erstellung von Exportrichtlinien und aktualisieren Sie [Boolean]. Mithilfe der `autoExportPolicy` Optionen und `autoExportCIDRs` kann Trident Exportrichtlinien automatisch managen.	`false`
`autoExportCIDRs`	Liste der CIDRs, nach denen die Node-IPs von Kubernetes gegen gefiltert werden sollen, wenn `autoExportPolicy` aktiviert ist. Mithilfe der `autoExportPolicy` Optionen und `autoExportCIDRs` kann Trident Exportrichtlinien automatisch managen.	„[„0.0.0.0/0“, „:/0“]“
`labels`	Satz willkürlicher JSON-formatierter Etiketten für Volumes	„“

Parameter	Beschreibung	Beispiel
`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.
`storagePrefix`	Das Präfix wird beim Bereitstellen neuer Volumes in der SVM verwendet. Kann nach der Erstellung nicht geändert werden. Um diesen Parameter zu aktualisieren, müssen Sie ein neues Backend erstellen.	`trident`
`limitAggregateUsage`	**Nicht für Amazon FSX für NetApp ONTAP angeben.** Die angegebenen `fsxadmin` und `vsadmin` enthalten nicht die erforderlichen Berechtigungen, um die aggregierte Nutzung abzurufen und sie mit Trident zu begrenzen.	Verwenden Sie ihn nicht.

Parameter	Beschreibung	Beispiel
`limitVolumeSize`	Bereitstellung fehlgeschlagen, wenn die angeforderte Volume-Größe über diesem Wert liegt. Beschränkt darüber hinaus die maximale Größe der Volumes, die es über qtrees und LUNs verwaltet, und `qtreesPerFlexvol` ermöglicht die Anpassung der maximalen Anzahl von qtrees pro FlexVol volume	„“ (nicht standardmäßig durchgesetzt)
`lunsPerFlexvol`	Die maximale Anzahl an LUNs pro FlexVol volume muss im Bereich [50, 200] liegen. Nur SAN	„`100`“
`debugTraceFlags`	Fehler-Flags bei der Fehlerbehebung beheben. Beispiel, {„API“:false, „method“:true} nicht verwenden `debugTraceFlags`, es sei denn, Sie beheben die Fehlerbehebung und benötigen einen detaillierten Log Dump.	Null
`nfsMountOptions`	Kommagetrennte Liste von NFS-Mount-Optionen. Die Mount-Optionen für persistente Kubernetes-Volumes werden normalerweise in Storage-Klassen angegeben. Wenn jedoch keine Mount-Optionen in einer Storage-Klasse angegeben sind, verwendet Trident die Mount-Optionen, die in der Konfigurationsdatei des Storage-Backends angegeben sind. Wenn in der Storage-Klasse oder in der Konfigurationsdatei keine Mount-Optionen angegeben sind, legt Trident keine Mount-Optionen auf einem zugeordneten persistenten Volume fest.	„“
`nasType`	Konfiguration der Erstellung von NFS- oder SMB-Volumes Optionen sind `nfs`, , `smb` oder Null. **Muss für SMB-Volumes auf gesetzt `smb` werden.** Einstellung auf null setzt standardmäßig auf NFS-Volumes.	`nfs`
`qtreesPerFlexvol`	Maximale Qtrees pro FlexVol volume, muss im Bereich [50, 300] liegen	`"200"`

Parameter	Beschreibung	Beispiel
`smbShare`	Sie können eine der folgenden Optionen angeben: Den Namen einer SMB-Freigabe, die mit der Microsoft Verwaltungskonsole oder der ONTAP-CLI erstellt wurde, oder einen Namen, mit dem Trident die SMB-Freigabe erstellen kann. Dieser Parameter ist für Amazon FSX for ONTAP Back-Ends erforderlich.	`smb-share`
`useREST`	Boolescher Parameter zur Verwendung von ONTAP REST-APIs. Wenn auf festgelegt `true`, verwendet Trident ONTAP REST APIs, um mit dem Backend zu kommunizieren. Diese Funktion erfordert ONTAP 9.11.1 und höher. Darüber hinaus muss die verwendete ONTAP-Anmelderolle Zugriff auf die Anwendung haben `ontap` . Dies wird durch die vordefinierten `vsadmin` Rollen und `cluster-admin` erreicht.	`false`
`aws`	Sie können Folgendes in der Konfigurationsdatei für AWS FSX für ONTAP angeben: - `fsxFilesystemID`: Geben Sie die ID des AWS FSX Dateisystems an. - `apiRegion`: Name der AWS API-Region. - `apikey`: AWS API-Schlüssel. - `secretKey`: AWS Geheimschlüssel.	`""` `""` `""`
`credentials`	Geben Sie die FSX SVM-Zugangsdaten an, die in AWS Secrets Manager gespeichert werden sollen. - `name`: Amazon Resource Name (ARN) des Geheimnisses, das die Zugangsdaten von SVM enthält. - `type`: Gesetzt auf `awsarn`. Weitere Informationen finden Sie unter "Erstellen Sie einen AWS Secrets Manager-Schlüssel" .	

Back-End-Konfigurationsoptionen für die Bereitstellung von Volumes

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

Parameter	Beschreibung	Standard
`spaceAllocation`	Speicherplatzzuweisung für LUNs	`true`
`spaceReserve`	Modus für Speicherplatzreservierung; „none“ (Thin) oder „Volume“ (Thick)	`none`
`snapshotPolicy`	Die Snapshot-Richtlinie zu verwenden	`none`
`qosPolicy`	QoS-Richtliniengruppe zur Zuweisung für erstellte Volumes Wählen Sie eine der qosPolicy oder adaptiveQosPolicy pro Storage-Pool oder Backend. Für die Verwendung von QoS-Richtliniengruppen mit Trident ist ONTAP 9 8 oder höher erforderlich. Sie sollten eine nicht gemeinsam genutzte QoS-Richtliniengruppe verwenden und sicherstellen, dass die Richtliniengruppe auf jede Komponente einzeln angewendet wird. Eine Shared-QoS-Richtliniengruppe erzwingt die Obergrenze für den Gesamtdurchsatz aller Workloads.	„“
`adaptiveQosPolicy`	Adaptive QoS-Richtliniengruppe mit Zuordnung für erstellte Volumes Wählen Sie eine der qosPolicy oder adaptiveQosPolicy pro Storage-Pool oder Backend. Nicht unterstützt durch ontap-nas-Ökonomie	„“
`snapshotReserve`	Prozentsatz des für Snapshots reservierten Volumes „0“	Wenn `snapshotPolicy` `none` , `else` „ “
`splitOnClone`	Teilen Sie einen Klon bei der Erstellung von seinem übergeordneten Objekt auf	`false`
`encryption`	Aktivieren Sie NetApp Volume Encryption (NVE) auf dem neuen Volume, Standardeinstellung ist `false`. NVE muss im Cluster lizenziert und aktiviert sein, damit diese Option verwendet werden kann. Wenn auf dem Backend NAE aktiviert ist, wird jedes in Trident bereitgestellte Volume NAE aktiviert. Weitere Informationen finden Sie unter "Funktionsweise von Trident mit NVE und NAE".	`false`

Parameter	Beschreibung	Standard
`luksEncryption`	Aktivieren Sie die LUKS-Verschlüsselung. Siehe "Linux Unified Key Setup (LUKS) verwenden". Nur SAN	„“
`tieringPolicy`	Tiering-Richtlinie für die Nutzung `none`	
`unixPermissions`	Modus für neue Volumes. **Leere leer für SMB Volumen.**	„“
`securityStyle`	Sicherheitstyp für neue Volumes. NFS-Unterstützung `mixed` und `unix` Sicherheitsstile. SMB-Unterstützung `mixed` und `ntfs` Sicherheitsstile.	NFS-Standard ist `unix`. SMB-Standard ist `ntfs`.

Vorbereitung zur Bereitstellung von SMB Volumes

Sie können SMB-Volumes mit dem Treiber bereitstellen `ontap-nas`. Führen Sie die folgenden Schritte aus, bevor Sie Integration von ONTAP-SAN- und NAS-Treibernddie Schritte ausführen.

Bevor Sie beginnen

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

- Kubernetes-Cluster mit einem Linux-Controller-Knoten und mindestens einem Windows-Worker-Node, auf dem Windows Server 2019 ausgeführt wird. Trident unterstützt nur SMB Volumes, die in Pods gemountet sind, die nur auf Windows Nodes ausgeführt werden.
- Mindestens ein Trident-Schlüssel, der Ihre Active Directory-Anmeldeinformationen enthält. So generieren Sie ein Geheimnis `smbcreds`:

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

- Ein CSI-Proxy, der als Windows-Dienst konfiguriert ist. Informationen zum Konfigurieren `csi-proxy` von finden Sie unter "GitHub: CSI-Proxy" oder "GitHub: CSI Proxy für Windows" für Kubernetes-Nodes, die unter Windows ausgeführt werden.

Schritte

1. Erstellen von SMB-Freigaben Sie können die SMB-Administratorfreigaben auf zwei Arten erstellen, entweder mit dem "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` Befehl überprüft den in der Option -path angegebenen Pfad während der Erstellung von Freigaben. 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
```

Weitere Informationen finden Sie unter"Erstellen Sie eine SMB-Freigabe".

2. Beim Erstellen des Backend müssen Sie Folgendes konfigurieren, um SMB-Volumes festzulegen. Für alle FSX für ONTAP Backend-Konfigurationsoptionen, siehe "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 Verwaltungskonsole oder der ONTAP-CLI erstellt wurde, oder einen Namen, mit dem Trident die SMB-Freigabe erstellen kann. Dieser Parameter ist für Amazon FSX for ONTAP Back-Ends erforderlich.	`smb-share`
`nasType`	**Muss auf.** gesetzt werden `smb` Wenn Null, wird standardmäßig auf `nfs`.	`smb`
`securityStyle`	Sicherheitstyp für neue Volumes. **Muss für SMB Volumes auf oder `mixed` gesetzt werden `ntfs`.**	`ntfs` Oder `mixed` für SMB Volumes
`unixPermissions`	Modus für neue Volumes. **Muss für SMB Volumes leer gelassen werden.**	„“

Konfigurieren Sie eine Storage-Klasse und PVC

Konfigurieren Sie ein Kubernetes StorageClass-Objekt und erstellen Sie die Storage-Klasse, um Trident anzuweisen, wie Volumes bereitgestellt werden. Erstellen Sie ein PersistentVolumeClaim (PVC), das die konfigurierte Kubernetes StorageClass verwendet, um Zugriff auf das PV anzufordern. Anschließend können Sie das PV an einem Pod montieren.

Konfigurieren Sie ein Kubernetes StorageClass-Objekt

Das "Kubernetes StorageClass-Objekt" identifiziert Trident als bereitstellung, die für diese Klasse verwendet wird. Trident erklärt, wie ein Volume bereitgestellt wird. Beispiel:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  provisioningType: "thin"
  snapshots: "true"
```

Fügen Sie zum Bereitstellen von NFSv3 Volumes auf AWS Bottlerocket die erforderliche Storage-Klasse hinzu `mountOptions`:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  media: "ssd"
  provisioningType: "thin"
  snapshots: "true"
mountOptions:
  - nfsvers=3
  - nolock
```

Einzelheiten zur Interaktion von Storage-Klassen mit den `PersistentVolumeClaim` Parametern und zur Steuerung, wie Trident Volumes provisioniert, finden Sie unter"Kubernetes und Trident Objekte".

Erstellen Sie eine Speicherklasse

Schritte

1. Dies ist ein Kubernetes-Objekt. Verwenden Sie es also `kubectl`, um es in Kubernetes zu erstellen.

   ```
   kubectl create -f storage-class-ontapnas.yaml
   ```

2. Sie sollten nun eine **Basic-csi** Storage-Klasse sowohl in Kubernetes als auch in Trident sehen, und 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
```

Erstellen Sie die PVC

A "*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 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: ontap-gold
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteMany
  hostPath:
    path: "/my/host/path"
```

PersistentVolumeClaim-Beispielmanifeste

Diese Beispiele zeigen grundlegende PVC-Konfigurationsoptionen.

PVC mit RWX-Zugang

Dieses Beispiel zeigt ein einfaches PVC mit RWX-Zugriff, das mit einer StorageClass namens verknüpft ist `basic-csi`.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-storage
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-gold
```

PVC mit NVMe/TCP

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

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

Erstellen Sie das PV und die PVC

Schritte

1. Erstellen Sie das PVC.

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

2. Überprüfen Sie den PVC-Status.

```
kubectl get pvc
```

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

Einzelheiten zur Interaktion von Storage-Klassen mit den `PersistentVolumeClaim` Parametern und zur Steuerung, wie Trident Volumes provisioniert, finden Sie unter"Kubernetes und Trident Objekte".

Trident-Attribute

Diese Parameter bestimmen, welche in Trident gemanagten Storage Pools zur Bereitstellung von Volumes eines bestimmten Typs verwendet werden sollten.

Attribut	Typ	Werte	Angebot	Anfrage	Unterstützt von
Medien[1]	Zeichenfolge	hdd, Hybrid, ssd	Pool enthält Medien dieser Art. Beides bedeutet Hybrid	Medientyp angegeben	ontap-nas, ontap-nas-Economy, ontap-nas-Flexgroup, ontap-san, solidfire-san
Bereitstellungstyp	Zeichenfolge	Dünn, dick	Pool unterstützt diese Bereitstellungsmethode	Bereitstellungsmethode angegeben	Thick: All ONTAP; Thin: Alle ONTAP und solidfire-san
BackendType	Zeichenfolge	ontap-nas, ontap-nas-Economy, ontap-nas-Flexgroup, ontap-san, solidfire-san, gcp-cvs, Azure-netapp-Files, ontap-san-Wirtschaftlichkeit	Pool gehört zu dieser Art von Backend	Back-End angegeben	Alle Treiber
Snapshots	bool	Richtig, falsch	Pool unterstützt Volumes mit Snapshots	Volume mit aktivierten Snapshots	ontap-nas, ontap-san, solidfire-san, gcp-cvs
Klone	bool	Richtig, falsch	Pool unterstützt das Klonen von Volumes	Volume mit aktivierten Klonen	ontap-nas, ontap-san, solidfire-san, gcp-cvs

Attribut	Typ	Werte	Angebot	Anfrage	Unterstützt von
Verschlüsselung	bool	Richtig, falsch	Pool unterstützt verschlüsselte Volumes	Volume mit aktivierter Verschlüsselung	ontap-nas, ontap-nas-Economy, ontap-nas-Flexgroups, ontap-san
IOPS	Int	Positive Ganzzahl	Pool kann IOPS in diesem Bereich garantieren	Volume hat diese IOPS garantiert	solidfire-san

[1]: Nicht unterstützt von ONTAP Select-Systemen

Beispielanwendung bereitstellen

Wenn die Storage-Klasse und die PVC erstellt wurden, können Sie das PV an einem Pod mounten. In diesem Abschnitt werden der Beispielbefehl und die Konfiguration zum Anbinden des PV an einen Pod aufgeführt.

Schritte

1. Mounten Sie das Volume in einem Pod.

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

Diese Beispiele zeigen grundlegende Konfigurationen zum Anbringen der PVC an einem POD:
Grundkonfiguration:

```
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
```

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

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

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

```
Filesystem                                                      Size
Used  Avail Use%  Mounted on
192.168.188.78:/trident_pvc_ae45ed05_3ace_4e7c_9080_d2a83ae03d06  1.1G
320K  1.0G   1%   /my/mount/path
```

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

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

Konfigurieren Sie das Trident EKS-Add-on auf einem EKS-Cluster

NetApp 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 NetApp Trident EKS Add-on enthält die neuesten Sicherheitspatches 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 sicher, dass folgende Voraussetzungen erfüllt sind, bevor Sie das Trident Add-on für AWS EKS konfigurieren:

- Ein Amazon EKS-Cluster-Konto mit Berechtigungen zum Arbeiten mit Add-ons. Siehe "Amazon EKS-Add-ons".
- 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. Erstellen Sie unbedingt eine IAM-Rolle und einen AWS Secret, damit EKS-Pods auf AWS Ressourcen zugreifen können. Anweisungen hierzu finden Sie unter "IAM-Rolle und AWS Secret erstellen".
2. Navigieren Sie auf Ihrem EKS Kubernetes-Cluster zur Registerkarte **Add-ons**.
3. Gehen Sie zu **AWS Marketplace Add-ons** und wählen Sie die Kategorie *Storage*.
4. Suchen Sie **NetApp Trident** und aktivieren Sie das Kontrollkästchen für das Trident-Add-on, und klicken Sie auf **Weiter**.
5. Wählen Sie die gewünschte Version des Add-ons aus.
6. Wählen Sie die Option IAM-Rolle aus, die vom Knoten übernommen werden soll.
7. Folgen Sie dem **Add-on-Konfigurationsschema** und setzen Sie den Parameter Konfigurationswerte im Abschnitt **Konfigurationswerte** auf die Rolle-arn, die Sie im vorherigen Schritt (Schritt 1) erstellt haben. Der Wert muss das folgende Format haben:

```
{

  "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'"

}
```

> Wenn Sie für die Konfliktlösungsmethode Überschreiben auswählen, können eine oder mehrere Einstellungen für das vorhandene Add-On mit den Amazon EKS-Zusatzeinstellungen überschrieben werden. Wenn Sie diese Option nicht aktivieren und es einen Konflikt mit Ihren bestehenden Einstellungen gibt, schlägt der Vorgang fehl. Sie können die resultierende Fehlermeldung verwenden, um den Konflikt zu beheben. Bevor Sie diese Option auswählen, stellen Sie sicher, dass das Amazon EKS-Add-On keine Einstellungen verwaltet, die Sie selbst verwalten müssen.

8. Wählen Sie **Erstellen**.
9. Überprüfen Sie, ob der Status des Add-ons *Active* lautet.
10. Führen Sie den folgenden Befehl aus, um zu überprüfen, ob Trident ordnungsgemäß auf dem Cluster installiert ist:

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

11. Setzen Sie die Einrichtung fort und konfigurieren Sie das Storage-Back-End. Weitere Informationen finden Sie unter "Konfigurieren Sie das Speicher-Back-End".

Installieren/deinstallieren Sie das Trident EKS-Add-On über CLI

Installieren Sie das NetApp Trident EKS-Add-On über CLI:

Mit dem folgenden Beispielbefehl wird das Trident EKS Add-On installiert: `eksctl create addon --cluster clusterName --name netapp_trident-operator --version v25.02.1-eksbuild.1` (Mit einer dedizierten Version)

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

Mit dem folgenden Befehl wird das 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 Trident und einem Storage-System. Er erzählt Trident, wie man mit diesem Storage-System kommuniziert und wie Trident Volumes daraus bereitstellen sollte. Nach der Installation von Trident wird im nächsten Schritt ein Backend erstellt. Mit der `TridentBackendConfig` CRD-Definition (Custom Resource Definition) können Sie Trident Back-Ends direkt über die Kubernetes-Schnittstelle erstellen und managen. Sie können dies mit `kubectl` oder mit dem entsprechenden CLI-Tool für Ihre Kubernetes-Distribution tun.

`TridentBackendConfig`

`TridentBackendConfig` (`tbc`, `tbconfig`, `tbackendconfig`) Ist ein Frontend, namested CRD, das Ihnen ermöglicht, Trident-Backends mit verwalten `kubectl`. Kubernetes- und Storage-Administratoren können jetzt Back-Ends direkt über die Kubernetes-CLI erstellen und managen(`tridentctl`, ohne dass ein dediziertes Befehlszeilendienstprogramm erforderlich ist).

Bei der Erstellung eines `TridentBackendConfig` Objekts geschieht Folgendes:

- Basierend auf der von Ihnen bereitgestellten Konfiguration wird von Trident automatisch ein Backend erstellt. Dies wird intern als (`tbe`, `tridentbackend`) CR dargestellt `TridentBackend`.
- Das `TridentBackendConfig` ist eindeutig an ein gebunden `TridentBackend`, das von Trident erstellt wurde.

Jede `TridentBackendConfig` verwaltet ein One-to-One Mapping mit einem `TridentBackend`. ersteres ist die Schnittstelle, die dem Benutzer zur Gestaltung und Konfiguration von Backends zur Verfügung gestellt wird; Letzteres ist, wie Trident das eigentliche Backend-Objekt darstellt.

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

Das folgende Beispiel zeigt das CR-Format `TridentBackendConfig`:

```
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 "trident-Installationsprogramm" Verzeichnis für Beispielkonfigurationen für die gewünschte Speicherplattform/den gewünschten Service ansehen.

Das `spec` übernimmt Backend-spezifische Konfigurationsparameter. In diesem Beispiel verwendet das Backend den `ontap-san` Speichertreiber und verwendet die hier tabellierten 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 im CR eingeführt werden `TridentBackendConfig`:

- `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 das `TridentBackendConfig` gelöscht wird. Es kann einen von zwei möglichen Werten annehmen:
  - `delete`: Dies führt zum Löschen von `TridentBackendConfig` CR und dem zugehörigen Backend. Dies ist der Standardwert.
  - `retain`: Wenn ein `TridentBackendConfig` CR gelöscht wird, ist die Backend-Definition weiterhin vorhanden und kann mit verwaltet werden `tridentctl`. Durch Festlegen der Löschrichtlinie auf `retain` können Benutzer ein Downgrade auf eine frühere Version (vor 21.04) durchführen und die erstellten Back-Ends beibehalten. Der Wert für dieses Feld kann aktualisiert werden, nachdem ein `TridentBackendConfig` erstellt wurde.

> ⓘ Der Name eines Backends wird mit gesetzt `spec.backendName`. Wenn nicht angegeben, wird der Name des Backends auf den Namen des Objekts (metadata.name) gesetzt `TridentBackendConfig`. Es wird empfohlen, Backend-Namen explizit mitzu setzen `spec.backendName`.

Back-Ends, die mit erstellt wurden `tridentctl`, haben kein zugeordnetes `TridentBackendConfig` Objekt. Sie können diese Back-Ends mit verwalten `kubectl`, indem Sie ein CR erstellen `TridentBackendConfig`. Es ist darauf zu achten, identische Konfigurationsparameter anzugeben (z. B. `spec.backendName`, , `spec.storagePrefix` `spec.storageDriverName` und so weiter). Trident bindet das neu erstellte mit dem bereits vorhandenen Backend automatisch `TridentBackendConfig`.

Schritte im Überblick

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

1. Erstellen Sie ein "Kubernetes Secret". das Geheimnis enthält die Anmeldeinformationen, die Trident benötigt, um mit dem Speicher-Cluster/Service zu kommunizieren.
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 dessen Status mithilfe von beobachten `kubectl get tbc <tbc-name> -n <trident-namespace>` und weitere Details erfassen.

Schritt: Ein Kubernetes Secret erstellen

Erstellen Sie einen geheimen Schlüssel, der die Anmeldedaten für den Zugriff für 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

Beschreibung der geheimen Felder der Speicherplattform	Geheim	Feldbeschreibung
Cloud Volumes Service für GCP	Private_Schlüssel_id	ID des privaten Schlüssels. Teil des API-Schlüssels für GCP-Servicekonto mit CVS-Administratorrolle
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 `ontap-san` und `ontap-san-economy`
ONTAP	ChapInitiatorSecret	CHAP-Initiatorschlüssel. Erforderlich, wenn usCHAP=true verwendet wird. Für `ontap-san` und `ontap-san-economy`
ONTAP	ChapTargetBenutzername	Zielbenutzername. Erforderlich, wenn usCHAP=true verwendet wird. Für `ontap-san` und `ontap-san-economy`
ONTAP	ChapTargetInitiatorSecret	Schlüssel für CHAP-Zielinitiator. Erforderlich, wenn usCHAP=true verwendet wird. Für `ontap-san` und `ontap-san-economy`

Der in diesem Schritt erstellte Schlüssel wird im Feld des `TridentBackendConfig` Objekts referenziert `spec.credentials`, das im nächsten Schritt erstellt wird.

Schritt 2: Erstellen Sie den `TridentBackendConfig` CR

Sie können jetzt Ihren CR erstellen `TridentBackendConfig`. In diesem Beispiel wird mithilfe des unten dargestellten Objekts ein Backend erstellt, das den Treiber `TridentBackendConfig` verwendet `ontap-san`:

```
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

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

```
kubectl -n trident get tbc backend-tbc-ontap-san
NAME                     BACKEND NAME          BACKEND UUID
PHASE   STATUS
backend-tbc-ontap-san   ontap-san-backend     8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8   Bound   Success
```

Ein Backend wurde erfolgreich erstellt und an den CR gebunden `TridentBackendConfig`.

Die Phase kann einen der folgenden Werte annehmen:

- `Bound`: Der `TridentBackendConfig` CR ist mit einem Backend verbunden, und das Backend enthält `configRef` gesetzt auf die UID des `TridentBackendConfig` CR.
- `Unbound`: Dargestellt mit `""`. Das `TridentBackendConfig` Objekt ist nicht an ein Backend gebunden. Alle neu erstellten `TridentBackendConfig` CRS befinden sich standardmäßig in dieser Phase. Wenn die Phase sich ändert, kann sie nicht wieder auf Unbound zurückgesetzt werden.

- `Deleting`: Die `TridentBackendConfig` CR's `deletionPolicy` wurden auf Löschen gesetzt. Wenn der `TridentBackendConfig` CR gelöscht wird, wechselt er in den Löschstatus.
  - Wenn auf dem Backend keine Persistent Volume Claims (PVCs) vorhanden sind, führt das Löschen des `TridentBackendConfig` dazu, dass Trident das Backend sowie den CR löscht `TridentBackendConfig`.
  - Wenn ein oder mehrere VES im Backend vorhanden sind, wechselt es in den Löschzustand. Anschließend geht der `TridentBackendConfig` CR auch in die Löschphase über. Das Backend und `TridentBackendConfig` werden erst gelöscht, nachdem alle VES gelöscht wurden.
- `Lost`: Das mit dem CR verknüpfte Backend `TridentBackendConfig` wurde versehentlich oder absichtlich gelöscht und der `TridentBackendConfig` CR hat noch einen Verweis auf das gelöschte Backend. Der `TridentBackendConfig` CR kann unabhängig vom Wert gelöscht werden `deletionPolicy`.
- `Unknown`: Trident kann den Status oder die Existenz des mit dem CR verknüpften Backends nicht bestimmen `TridentBackendConfig`. Beispiel: Wenn der API-Server nicht reagiert oder die `tridentbackends.trident.netapp.io` CRD fehlt. Dies kann Eingriffe erfordern.

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

(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 die `backendName` und die `backendUUID` des Backends, das als Antwort auf den CR erstellt wurde `TridentBackendConfig`. Das `lastOperationStatus` Feld stellt den Status des letzten Vorgangs des CR dar `TridentBackendConfig`, der vom Benutzer ausgelöst werden kann (z. B. Benutzer hat etwas in geändert `spec`) oder durch Trident ausgelöst (z. B. beim Neustart von Trident). Es kann entweder erfolgreich oder fehlgeschlagen sein. `phase` Stellt den Status der Beziehung zwischen dem CR und dem Backend dar `TridentBackendConfig`. Im obigen Beispiel `phase` hat der Wert gebunden, was bedeutet, dass der `TridentBackendConfig` CR mit dem Backend verknüpft ist.

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

Sie können ein Backend, das ein zugeordnetes Objekt enthält, mit `tridentctl` nicht aktualisieren oder löschen `TridentBackendConfig`. Um die Schritte beim Wechsel zwischen und `TridentBackendConfig`zu verstehen `tridentctl`, "Sehen Sie hier".

Back-Ends managen

Führen Sie das Back-End-Management mit kubectl durch

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

Löschen Sie ein Back-End

Durch das Löschen eines `TridentBackendConfig`, weisen Sie Trident an, Back-Ends zu löschen/zu behalten (basierend auf `deletionPolicy`). Um ein Backend zu löschen, stellen Sie sicher, dass `deletionPolicy` es auf „Löschen“ gesetzt ist. Um nur die zu löschen `TridentBackendConfig`, stellen Sie sicher, dass `deletionPolicy` auf beibehalten gesetzt ist. Dadurch wird sichergestellt, dass das Backend noch vorhanden ist und mit verwaltet werden kann `tridentctl`.

Führen Sie den folgenden Befehl aus:

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

Trident löscht die Kubernetes-Geheimnisse nicht, 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` eine Liste aller vorhandenen Back-Ends erhalten. Diese Liste enthält auch Backends, 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. Zum Aktualisieren der Zugangsdaten muss der im Objekt verwendete Kubernetes Secret `TridentBackendConfig` aktualisiert werden. Trident aktualisiert das Backend automatisch mit den neuesten Anmeldeinformationen. 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.
  - Mit dem folgenden Befehl können Sie Objekte direkt über Kubernetes aktualisieren `TridentBackendConfig`:

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

- Alternativ können Sie mit dem folgenden Befehl Änderungen am vorhandenen CR vornehmen `TridentBackendConfig`:

```
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 zu ermitteln, indem Sie oder `kubectl describe tbc <tbc-name> -n trident` ausführen `kubectl get tbc <tbc-name> -o yaml -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 Back-End-Management-Operationen mit durchführen `tridentctl`.

Erstellen Sie ein Backend

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

```
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 den Befehl einfach erneut ausführen `create`.

Löschen Sie ein Back-End

Gehen Sie folgendermaßen vor, um ein Backend aus 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 Trident Volumes und Snapshots von diesem Backend bereitgestellt hat, die noch vorhanden sind, verhindert das Löschen des Backends, dass neue Volumes bereitgestellt werden. Das Backend existiert weiterhin im Zustand „Löschen“.

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 den Befehl einfach erneut ausführen `update`.

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, die `tridentctl` für Backend-Objekte ausgegeben wird. Hierbei wird das Dienstprogramm verwendet `jq`, das Sie installieren müssen.

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

Dies gilt auch für Backends, die durch die Verwendung von erstellt wurden `TridentBackendConfig`.

Wechseln Sie zwischen den Back-End-Managementoptionen

Erfahren Sie mehr über die verschiedenen Möglichkeiten für das Management von Back-Ends in Trident.

Optionen für das Management von Back-Ends

Mit der Einführung von `TridentBackendConfig` haben Administratoren nun zwei einzigartige Möglichkeiten, Back-Ends zu managen. Dies stellt die folgenden Fragen:

- Können Back-Ends erstellt `tridentctl` werden mit `TridentBackendConfig`?
- Können Back-Ends erstellt mit `TridentBackendConfig` verwaltet werden `tridentctl` ?

Managen von `tridentctl` Back-Ends mit `TridentBackendConfig`

In diesem Abschnitt werden die Schritte zum Management von Back-Ends behandelt, die durch das Erstellen von Objekten direkt über die Kubernetes-Schnittstelle erstellt `TridentBackendConfig` wurden `tridentctl`.

Dies gilt für die folgenden Szenarien:

- Bereits vorhandene Backends, die nicht über ein verfügen `TridentBackendConfig`, weil sie mit erstellt wurden `tridentctl`.
- Neue Backends, die mit erstellt wurden `tridentctl`, während andere `TridentBackendConfig` Objekte existieren.

In beiden Szenarien sind Back-Ends weiterhin vorhanden, wobei Trident Volumes terminiert und darauf ausgeführt werden. Administratoren können hier eine von zwei Möglichkeiten wählen:

- Verwenden Sie weiter `tridentctl`, um Back-Ends zu verwalten, die mit ihm erstellt wurden.
- Binden von Back-Ends, die mit erstellt `tridentctl` wurden, an ein neues `TridentBackendConfig` Objekt. Dies würde bedeuten, dass die Back-Ends mit und nicht `tridentctl` verwaltet werden `kubectl`.

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

1. Kubernetes Secret erstellen: Das Geheimnis enthält die Anmeldeinformationen, die 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 ist darauf zu achten, identische Konfigurationsparameter anzugeben (z. B. `spec.backendName`, , `spec.storagePrefix` `spec.storageDriverName` und so weiter). `spec.backendName` Muss auf den Namen des vorhandenen Backends gesetzt werden.

Schritt 0: Identifizieren Sie das Backend

Um ein zu erstellen `TridentBackendConfig`, das 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 eines `TridentBackendConfig` CR

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

- Der gleiche Backend-Name ist 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 sieht das `TridentBackendConfig` wie folgt aus:

```
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

Nachdem der `TridentBackendConfig` erstellt wurde, muss seine Phase 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
PHASE   STATUS
tbc-ontap-nas-backend  ontap-nas-backend     52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7   Bound   Success

#confirm that no new backends were created (i.e., TridentBackendConfig did
not end up creating a new backend)
tridentctl get backend -n trident
+---------------------+----------------
+--------------------------------------+--------+---------+
|          NAME       | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+---------------------+----------------
+--------------------------------------+--------+---------+
| ontap-nas-backend   | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |      25 |
+---------------------+----------------
+--------------------------------------+--------+---------+
```

Das Backend wird nun vollständig über das Objekt verwaltet `tbc-ontap-nas-backend TridentBackendConfig`.

Managen von `TridentBackendConfig` Back-Ends mit `tridentctl`

```
`tridentctl` Kann verwendet werden, um Back-Ends aufzulisten, die mit
erstellt wurden `TridentBackendConfig`. Darüber hinaus können
Administratoren auch wählen, um vollständig verwalten solche Back-Ends
durch durch `tridentctl` Löschen `TridentBackendConfig` und sicherstellen,
`spec.deletionPolicy` ist auf gesetzt `retain`.
```

Schritt 0: Identifizieren Sie das Backend

Nehmen wir zum Beispiel an, dass das folgende Backend mit erzeugt wurde `TridentBackendConfig`:

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                    BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        delete

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 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

Aus der Ausgabe wird ersichtlich, dass sie `TridentBackendConfig` erfolgreich erstellt wurde und an ein Backend gebunden ist [Observe the Backend's UUID].

Schritt 1: Bestätigen `deletionPolicy` ist auf eingestellt `retain`

Lassen Sie uns einen Blick auf den Wert von `deletionPolicy`. Dies muss auf eingestellt werden `retain`. Dadurch wird sichergestellt, dass beim Löschen eines `TridentBackendConfig` CR die Backend-Definition weiterhin vorhanden ist und mit verwaltet werden kann `tridentctl`.

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                    BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        delete

# 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
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        retain
```

Fahren Sie nicht mit dem nächsten Schritt fort, es sei denn, es `deletionPolicy` ist auf eingestellt `retain`.

Schritt 2: Löschen Sie den `TridentBackendConfig` CR

Der letzte Schritt besteht darin, den CR zu löschen `TridentBackendConfig`. Nach der Bestätigung, dass der `deletionPolicy` auf gesetzt ist `retain`, können Sie mit dem Löschen 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 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

Beim Löschen des `TridentBackendConfig` Objekts entfernt Trident es einfach, ohne das Backend selbst 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 Trident anzuweisen, wie Volumes bereitgestellt werden.

Konfigurieren Sie ein Kubernetes StorageClass-Objekt

Das "Kubernetes StorageClass-Objekt" identifiziert Trident als bereitstellung, die für diese Klasse verwendet wird. Trident erklärt, wie ein Volume bereitgestellt wird. 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
```

Einzelheiten zur Interaktion von Storage-Klassen mit den `PersistentVolumeClaim` Parametern und zur Steuerung, wie Trident Volumes provisioniert, finden Sie unter"Kubernetes und Trident Objekte".

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. Dies ist ein Kubernetes-Objekt. Verwenden Sie es also `kubectl`, um es in Kubernetes zu erstellen.

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

2. Sie sollten nun eine **Basic-csi** Storage-Klasse sowohl in Kubernetes als auch in Trident sehen, und 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

Trident bietet "Einfache Definitionen von Storage-Klassen für spezifische Back-Ends".

Alternativ können Sie die Datei bearbeiten `sample-input/storage-class-csi.yaml.templ`, die im Lieferumfang des Installationsprogramms enthalten ist, und sie durch den Namen des Speichertreibers ersetzen *`BACKEND_TYPE`*.

```
./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 Trident anzuzeigen:

```
tridentctl get storageclass
```

- Führen Sie den folgenden Befehl aus, um Details zur synchronisierten Storage-Klasse von Trident anzuzeigen:

```
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 in der Definition der Speicherklasse auf true setzen `storageclass.kubernetes.io/is-default-class`. 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, die `tridentctl` für Trident-Backend-Objekte ausgegeben wird. Hierbei wird das Dienstprogramm verwendet `jq`, 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 über diese Storage-Klasse erstellt wurden, bleiben unverändert und Trident managt sie weiterhin.

Trident erzwingt ein Leerzeichen `fsType` für die von ihm erstellten Volumes. Für iSCSI-Back-Ends wird empfohlen, in der StorageClass durchzusetzen `parameters.fsType`. Sie sollten vorhandene StorageClasses löschen und mit den angegebenen neu erstellen `parameters.fsType`.

Provisionierung und Management von Volumes

Bereitstellen eines Volumes

Erstellen Sie 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 "*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 die PVC erstellt haben, können Sie das Volume in einem Pod einbinden.

Erstellen Sie die PVC

Schritte

1. Erstellen Sie das PVC.

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

2. Überprüfen Sie den PVC-Status.

```
kubectl get pvc
```

```
NAME          STATUS   VOLUME     CAPACITY   ACCESS MODES   STORAGECLASS AGE
pvc-storage Bound    pv-name    1Gi        RWO                    5m
```

1. Mounten Sie das Volume in einem Pod.

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

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

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

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

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

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

Beispielmanifeste

PersistentVolumeClaim-Beispielmanifeste

Diese Beispiele zeigen grundlegende PVC-Konfigurationsoptionen.

PVC mit RWO-Zugang

Dieses Beispiel zeigt ein einfaches PVC mit RWO-Zugriff, das mit einer StorageClass namens verknüpft 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 namens 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: storage
      persistentVolumeClaim:
       claimName: pvc-storage
  containers:
    - name: pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: storage
```

Grundlegende NVMe/TCP-Konfiguration

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-nginx
spec:
  volumes:
    - name: basic-pvc
      persistentVolumeClaim:
        claimName: pvc-san-nvme
  containers:
    - name: task-pv-container
      image: nginx
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: basic-pvc
```

Einzelheiten zur Interaktion von Storage-Klassen mit den `PersistentVolumeClaim` Parametern und zur Steuerung, wie Trident Volumes provisioniert, finden Sie unter"Kubernetes und Trident Objekte".

Erweitern Sie Volumes

Trident bietet Kubernetes-Benutzern die Möglichkeit, ihre Volumes nach der Erstellung zu erweitern. Hier finden Sie Informationen zu den Konfigurationen, die für die Erweiterung von iSCSI-, NFS- und FC-Volumes erforderlich sind.

Erweitern Sie ein iSCSI-Volume

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

> Die iSCSI-Volume-Erweiterung wird von den, `ontap-san-economy- solidfire-san`Treibern unterstützt `ontap-san` und erfordert Kubernetes 1.16 und höher.

Schritt: Storage Class für Volume-Erweiterung konfigurieren

Bearbeiten Sie die StorageClass-Definition, um das Feld auf `true` einzustellen `allowVolumeExpansion`.

```
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 diese, um den Parameter einzuschließen `allowVolumeExpansion`.

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

Bearbeiten Sie die PVC-Definition, und aktualisieren Sie den `spec.resources.requests.storage`, um die neu gewünschte Größe wiederzugeben, die größer sein muss als die ursprüngliche Größe.

```
cat pvc-ontapsan.yaml
```

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

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

```
kubectl get pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi
RWO            ontap-san      8s

kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM              STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi        RWO
Delete           Bound    default/san-pvc    ontap-san               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 mit einem Pod verbunden ist, erweitert Trident das Volume im Storage-Back-End, scannt das Gerät erneut und skaliert das Dateisystem.
- Beim Versuch, die Größe eines nicht verbundenen PV zu ändern, erweitert Trident das Volume auf dem Speicher-Back-End. 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`.

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

 kubectl 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, der von 1Gi auf 2Gi erstellt wurde, zu ändern, bearbeiten Sie die PVC-Definition und aktualisieren Sie den `spec.resources.requests.storage` auf 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: Validierung der Erweiterung

Sie können die korrekt bearbeitete Erweiterung validieren, indem Sie die Größe der PVC, des PV und des 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 |
PROTOCOL |             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 FC-Volume

Sie können ein FC Persistent Volume (PV) mit der CSI-provisionierung erweitern.

FC-Volume-Erweiterung wird vom Treiber unterstützt `ontap-san` und erfordert Kubernetes 1.16 und höher.

Schritt: Storage Class für Volume-Erweiterung konfigurieren

Bearbeiten Sie die StorageClass-Definition, um das Feld auf `true` einzustellen `allowVolumeExpansion`.

```
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 diese, um den Parameter einzuschließen `allowVolumeExpansion`.

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

Bearbeiten Sie die PVC-Definition, und aktualisieren Sie den `spec.resources.requests.storage`, um die neu gewünschte Größe wiederzugeben, die größer sein muss als die ursprüngliche Größe.

```
cat pvc-ontapsan.yaml
```

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

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

```
kubectl get pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi
RWO            ontap-san      8s

kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM              STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi        RWO
Delete           Bound    default/san-pvc    ontap-san               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 FC-PV gibt es zwei Szenarien:

- Wenn das PV mit einem Pod verbunden ist, erweitert Trident das Volume im Storage-Back-End, scannt das Gerät erneut und skaliert das Dateisystem.
- Beim Versuch, die Größe eines nicht verbundenen PV zu ändern, erweitert Trident das Volume auf dem Speicher-Back-End. 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`.

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

 kubectl 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, der von 1Gi auf 2Gi erstellt wurde, zu ändern, bearbeiten Sie die PVC-Definition und aktualisieren Sie den `spec.resources.requests.storage` auf 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: Validierung der Erweiterung

Sie können die korrekt bearbeitete Erweiterung validieren, indem Sie die Größe der PVC, des PV und des 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 |
PROTOCOL |             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

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

Schritt: Storage Class für Volume-Erweiterung konfigurieren

Um die Größe eines NFS-PV zu ändern, muss der Administrator zuerst die Speicherklasse konfigurieren, um die Volume-Erweiterung zu ermöglichen, indem er das Feld auf `true`folgende Einstellung setzt `allowVolumeExpansion`:

```
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 bearbeiten und die Volume-Erweiterung zulassen. `kubectl edit storageclass`

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
```

Trident sollte ein 20MiB NFS PV für die folgende 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 setzen Sie `spec.resources.requests.storage` auf 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 Größe der korrekt bearbeiteten Größe validieren, indem Sie die Größe der PVC, des PV und des Trident Volume überprüfen:

```
kubectl get pvc ontapnas20mb
NAME           STATUS   VOLUME
CAPACITY   ACCESS MODES   STORAGECLASS    AGE
ontapnas20mb   Bound    pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   1Gi
RWO            ontapnas        4m44s

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   1Gi        RWO
Delete           Bound    default/ontapnas20mb   ontapnas
5m35s

tridentctl get volume pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 -n trident
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
| 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 importieren `tridentctl import`.

Überblick und Überlegungen

Sie können ein Volume in Trident importieren, 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.

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

- Obwohl `StorageClass` auf einer PVC angegeben werden muss, verwendet Trident diesen Parameter beim Import 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 Volumegröße 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 Zurückgewinnungsrichtlinie ist zunächst im PV auf festgelegt `retain`. Nachdem Kubernetes die PVC und das PV erfolgreich bindet, wird die Zurückgewinnungsrichtlinie aktualisiert und an die Zurückgewinnungsrichtlinie der Storage-Klasse angepasst.
  - Wenn die Zurückgewinnungsrichtlinie der Speicherklasse lautet `delete`, wird das Speichervolume gelöscht, wenn das PV gelöscht wird.
- Standardmäßig verwaltet Trident die PVC und benennt die FlexVol volume und die LUN auf dem Backend um. Sie können das Flag übergeben `--no-manage`, um ein nicht verwaltetes Volume zu importieren. Wenn Sie verwenden `--no-manage`, führt Trident keine zusätzlichen Operationen auf der PVC oder PV für den Lebenszyklus der Objekte aus. 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

Sie können zum Importieren eines Volumes verwenden `tridentctl import`.

Schritte

1. Erstellen Sie die PVC-Datei (Persistent Volume Claim) (z. B. `pvc.yaml`), die zum Erstellen der PVC verwendet wird. Die PVC-Datei sollte , , `namespace` `accessModes` und `storageClassName` enthalten `name`. Optional können Sie in Ihrer PVC-Definition angeben `unixPermissions`.

   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 den `tridentctl import` Befehl, um den Namen des Trident-Backends mit dem Volume sowie den Namen anzugeben, der das Volume auf dem Storage eindeutig identifiziert (z. B. ONTAP FlexVol, Element Volume, Cloud Volumes Service-Pfad). Das `-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

Trident unterstützt den Import von Volumes mit den `ontap-nas` Treibern und `ontap-nas-flexgroup`.

- Der `ontap-nas-economy` Treiber kann qtrees nicht importieren und managen.
- Die `ontap-nas` und `ontap-nas-flexgroup`-Treiber erlauben keine doppelten Volume-Namen.

Jedes mit dem Treiber erstellte Volume `ontap-nas` ist eine FlexVol volume im ONTAP Cluster. Der Import von FlexVol-Volumes mit dem `ontap-nas` Treiber funktioniert gleich. FlexVol Volumes, die bereits in einem ONTAP-Cluster vorhanden sind, können als PVC importiert werden `ontap-nas`. Ebenso können FlexGroup-Volumes als PVCs importiert werden `ontap-nas-flexgroup`.

Beispiele für ONTAP NAS

Die folgende Darstellung zeigt ein Beispiel für ein verwaltetes Volume und einen nicht verwalteten Volume-Import.

Gemanagtes Volume

Das folgende Beispiel importiert ein Volume mit dem Namen `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` Arguments benennt Trident das Volume nicht um.

Im folgenden Beispiel werden Importe auf das `ontap_nas` Backend importiert `unmanaged_volume`:

```
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

Trident unterstützt den Import von Volumes mit den `ontap-san` Treibern und `ontap-san-economy`.

Trident kann ONTAP-SAN-FlexVol-Volumes importieren, die eine einzelne LUN enthalten. Dies ist mit dem Treiber konsistent `ontap-san`, der für jede PVC und eine LUN in der FlexVol volume eine FlexVol volume erstellt. Trident importiert die FlexVol volume und ordnet sie der PVC-Definition zu.

Beispiele für ONTAP SAN

Die folgende Darstellung zeigt ein Beispiel für ein verwaltetes Volume und einen nicht verwalteten Volume-Import.

Gemanagtes Volume

Für verwaltete Volumes benennt Trident die FlexVol volume in das Format und die LUN in der FlexVol volume in `lun0` um `pvc-<uuid>`.

Im folgenden Beispiel werden die auf dem Backend vorhandenen FlexVol volume `ontap_san_default` importiert `ontap-san-managed`:

```
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    |
+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
```

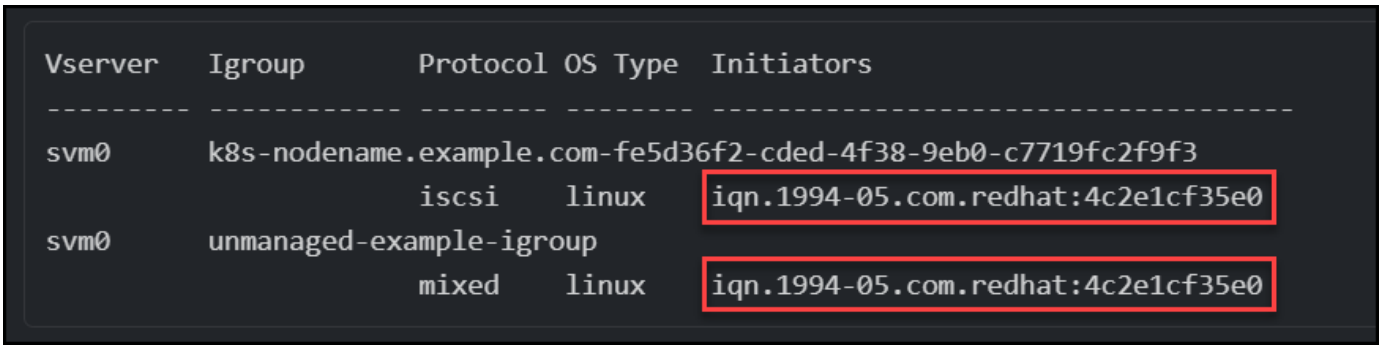
Nicht verwaltetes Volume

Im folgenden Beispiel werden Importe auf das `ontap_san` Backend importiert `unmanaged_example_volume`:

```
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   |
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
```

Wenn LUNS Initiatorgruppen zugeordnet sind, die einen IQN mit einem Kubernetes-Node-IQN teilen, wie im folgenden Beispiel dargestellt, erhalten Sie die Fehlermeldung: `LUN already mapped to initiator(s)`

`in this group`. Sie müssen den Initiator entfernen oder die Zuordnung der LUN aufheben, um das Volume zu importieren.



Element

Trident unterstützt NetApp Element-Software und NetApp HCI-Volume-Import mit dem `solidfire-san` Treiber.

> Der Elementtreiber unterstützt doppelte Volume-Namen. 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

Das folgende Beispiel importiert ein `element-managed` Volume auf dem 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

Trident unterstützt den Import von Volumes mithilfe des `gcp-cvs` Treibers.

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 Volumenpfad ist der Teil des Exportpfades des Volumes nach dem `:/`. Wenn der Exportpfad beispielsweise lautet `10.0.0.1:/adroit-jolly-swift`, ist der Volumenpfad `adroit-jolly-swift` .

Beispiel für die Google Cloud Platform

Im folgenden Beispiel wird ein Volume auf dem Backend `gcpcvs_YEppr` mit dem Volume-Pfad von `adroit-jolly-swift` importiert `gcp-cvs`.

```
tridentctl import volume gcpcvs_YEppr adroit-jolly-swift -f <path-to-pvc-
file> -n trident

+------------------------------------------+--------+---------------
+----------+------------------------------------------+--------+---------+
|                   NAME                   |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+--------+---------------
+----------+------------------------------------------+--------+---------+
| pvc-a46ccab7-44aa-4433-94b1-e47fc8c0fa55 | 93 GiB | gcp-storage   | file
| e1a6e65b-299e-4568-ad05-4f0a105c888f | online | true    |
+------------------------------------------+--------+---------------
+----------+------------------------------------------+--------+---------+
```

Azure NetApp Dateien

Trident unterstützt den Import von Volumes mithilfe des `azure-netapp-files` Treibers.

Um ein Azure NetApp Files-Volume zu importieren, identifizieren Sie das Volume anhand seines Volume-Pfads. Der Volumenpfad ist der Teil des Exportpfades des Volumes nach dem `:/`. Wenn der Mount-Pfad beispielsweise lautet `10.0.0.2:/importvol1`, ist der Volume-Pfad `importvol1` .

Beispiel: Azure NetApp Files

Das folgende Beispiel importiert ein `azure-netapp-files` Volume auf dem Backend `azurenetappfiles_40517` mit dem Volume-Pfad `importvol1`.

```
tridentctl import volume azurenetappfiles_40517 importvol1 -f <path-to-
pvc-file> -n trident

+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
| pvc-0ee95d60-fd5c-448d-b505-b72901b3a4ab | 100 GiB | anf-storage   |
file     | 1c01274f-d94b-44a3-98a3-04c953c9a51e | online | true    |
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
```

Google Cloud NetApp Volumes

Trident unterstützt den Import von Volumes mithilfe des `google-cloud-netapp-volumes` Treibers.

Beispiel: Google Cloud NetApp Volumes

Das folgende Beispiel importiert ein `google-cloud-netapp-volumes` Volume auf dem Backend `backend-tbc-gcnv1` mit dem Volume `testvoleasiaeast1`.

```
tridentctl import volume backend-tbc-gcnv1 "testvoleasiaeast1" -f < path-
to-pvc> -n trident

+-----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS
| PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
| pvc-a69cda19-218c-4ca9-a941-aea05dd13dc0 |  10 GiB | gcnv-nfs-sc-
identity | file     | 8c18cdf1-0770-4bc0-bcc5-c6295fe6d837 | online | true
|
+-----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

Im folgenden Beispiel wird ein Volume importiert `google-cloud-netapp-volumes`, wenn zwei Volumes in derselben Region vorhanden sind:

```
tridentctl import volume backend-tbc-gcnv1
"projects/123456789100/locations/asia-east1-a/volumes/testvoleasiaeast1"
-f <path-to-pvc> -n trident

+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS
| PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
| pvc-a69cda19-218c-4ca9-a941-aea05dd13dc0 |  10 GiB | gcnv-nfs-sc-
identity | file     | 8c18cdf1-0770-4bc0-bcc5-c6295fe6d837 | online | true
|
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

Passen Sie Volume-Namen und -Beschriftungen an

Mit Trident können Sie Volumes, die Sie erstellen, aussagekräftige Namen und Labels zuweisen. So können Sie Volumes leichter identifizieren und ihren jeweiligen Kubernetes-Ressourcen (PVCs) zuweisen. Sie können auch Vorlagen auf Backend-Ebene definieren, um benutzerdefinierte Volume-Namen und benutzerdefinierte Labels zu erstellen. Alle Volumes, die Sie erstellen, importieren oder klonen, werden an die Vorlagen angepasst.

Bevor Sie beginnen

Anpassbare Volumennamen und Beschriftungen unterstützen:

1. Volume-Erstellung, -Import und -Klonen
2. Im Fall des ontap-nas-Economy-Treibers entspricht nur der Name des Qtree-Volumes der Namensvorlage.
3. Im Fall des ontap-san-Economy-Treibers entspricht nur der LUN-Name der Namensvorlage.

Einschränkungen

1. Anpassbare Volume-Namen sind nur mit ONTAP On-Premises-Treibern kompatibel.
2. Anpassbare Volume-Namen gelten nicht für vorhandene Volumes.

Wichtige Verhaltensweisen anpassbarer Volumennamen

1. Wenn ein Fehler aufgrund einer ungültigen Syntax in einer Namensvorlage auftritt, schlägt die Back-End-Erstellung fehl. Wenn jedoch die Vorlagenapplikation fehlschlägt, wird das Volume gemäß der bestehenden Namenskonvention benannt.
2. Storage-Präfix ist nicht anwendbar, wenn ein Volume mit einer Namensvorlage aus der Back-End-

Konfiguration benannt wird. Jeder gewünschte Präfixwert kann direkt zur Vorlage hinzugefügt werden.

Beispiele für die Backend-Konfiguration mit Namensvorlage und Beschriftungen

Benutzerdefinierte Namensvorlagen können auf Root- und/oder Poolebene definiert werden.

Beispiel für die Stammebene

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "ontap-nfs-backend",
  "managementLIF": "<ip address>",
  "svm": "svm0",
  "username": "<admin>",
  "password": "<password>",
  "defaults": {
    "nameTemplate":
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.volume.Requ
estName}}"
  },
  "labels": {
    "cluster": "ClusterA",
    "PVC": "{{.volume.Namespace}}_{{.volume.RequestName}}"
  }
}
```

Beispiel auf Poolebene

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "ontap-nfs-backend",
  "managementLIF": "<ip address>",
  "svm": "svm0",
  "username": "<admin>",
  "password": "<password>",
  "useREST": true,
  "storage": [
    {
      "labels": {
        "labelname": "label1",
        "name": "{{ .volume.Name }}"
      },
      "defaults": {
        "nameTemplate": "pool01_{{ .volume.Name }}_{{ .labels.cluster
}}_{{ .volume.Namespace }}_{{ .volume.RequestName }}"
      }
    },
    {
      "labels": {
        "cluster": "label2",
        "name": "{{ .volume.Name }}"
      },
      "defaults": {
        "nameTemplate": "pool02_{{ .volume.Name }}_{{ .labels.cluster
}}_{{ .volume.Namespace }}_{{ .volume.RequestName }}"
      }
    }
  ]
}
```

Beispiele für Namensvorlagen

Beispiel 1:

```
"nameTemplate": "{{ .config.StoragePrefix }}_{{ .volume.Name }}_{{
.config.BackendName }}"
```

Beispiel 2:

```
"nameTemplate": "pool_{{ .config.StoragePrefix }}_{{ .volume.Name }}_{{
slice .volume.RequestName 1 5 }}""
```

Zu berücksichtigende Aspekte

1. Bei Volumenimporten werden die Etiketten nur aktualisiert, wenn das vorhandene Volume über Etiketten in einem bestimmten Format verfügt. Zum Beispiel: `{"provisioning":{"Cluster":"ClusterA", "PVC": "pvcname"}}`.
2. Im Fall des Imports von verwalteten Volumes folgt der Name des Volumes der Namensvorlage, die in der Backend-Definition auf Root-Ebene definiert wurde.
3. Trident unterstützt die Verwendung eines Slice-Operators mit dem Speicherpräfix nicht.
4. Wenn die Vorlagen nicht zu eindeutigen Volume-Namen führen, fügt Trident einige zufällige Zeichen an, um eindeutige Volume-Namen zu erstellen.
5. Wenn der benutzerdefinierte Name für ein NAS-Economy-Volume 64 Zeichen lang ist, benennt Trident die Volumes entsprechend der bestehenden Namenskonvention. Bei allen anderen ONTAP-Treibern schlägt die Erstellung des Volumes fehl, wenn der Datenträgername das Limit für den Namen überschreitet.

Ein NFS-Volume kann über Namespaces hinweg genutzt werden

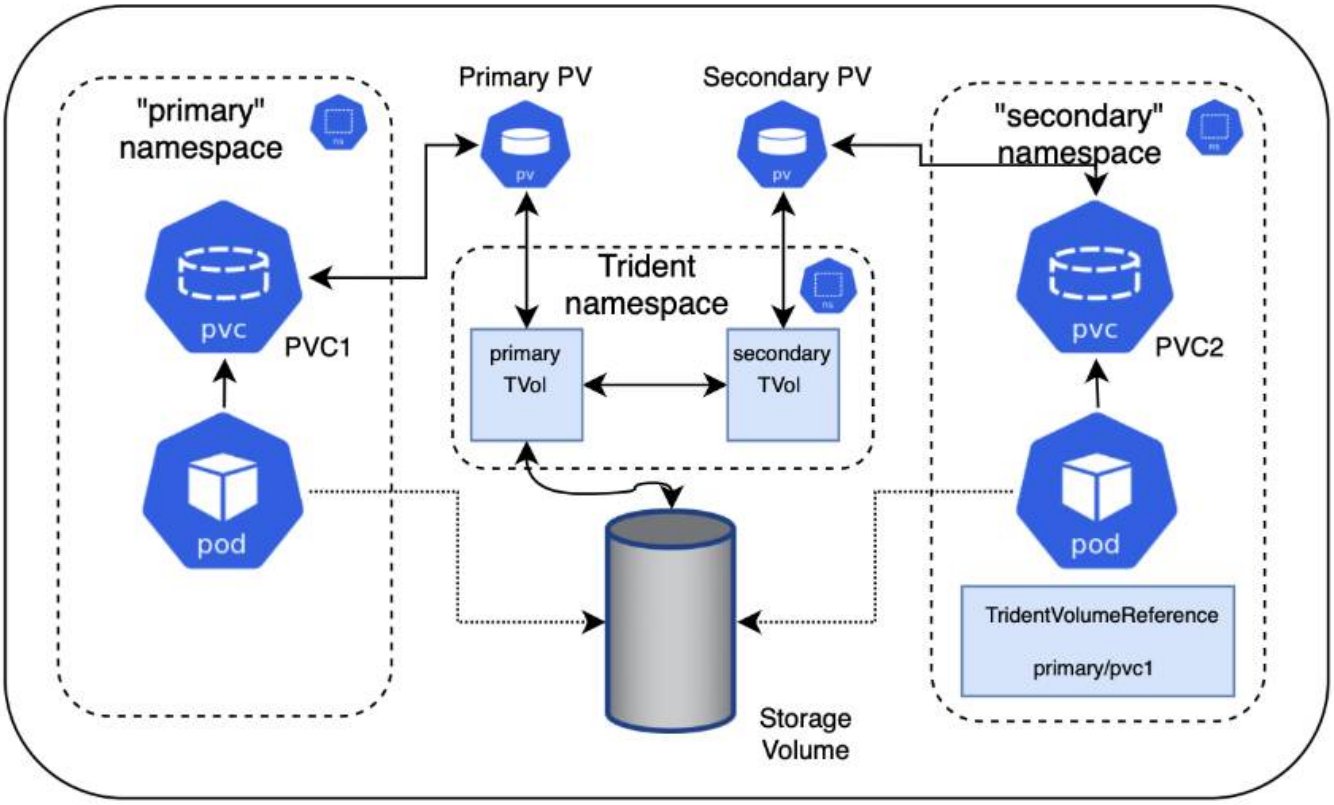
Mit Trident können Sie ein Volume in einem primären Namespace erstellen und es in einem oder mehreren sekundären Namespaces teilen.

Funktionen

Mit dem TridentVolumeReference CR können Sie ReadWriteMany (RWX) NFS-Volumes sicher über einen oder mehrere Kubernetes-Namespaces freigeben. 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 Quell-PVC für die gemeinsame Nutzung des Volumes**

Der Eigentümer des Quell-Namespace erteilt die Berechtigung, auf die Daten im Quell-PVC zuzugreifen.

2 **Erteilen Sie die Berechtigung zum Erstellen eines CR im Zielspeicherort**

Der Clusteradministrator erteilt dem Eigentümer des Ziel-Namespace die Berechtigung, das TridentVolumeReference CR zu erstellen.

3 **Erstellen Sie TridentVolumeReference im Ziel-Namespace**

Der Eigentümer des Ziel-Namespace erstellt das TridentVolumeReference CR, um sich auf das Quell-PVC zu beziehen.

4 **Erstellen Sie die 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 die PVC (`pvc1`) im Source Namespace, der die Erlaubnis erteilt, mit dem Ziel-Namespace zu teilen (`namespace2`) mit der `shareToNamespace` Annotation.

```
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
```

   Trident erstellt das PV und das dazugehörige Backend-NFS-Storage-Volume.

   - Sie können das PVC über eine durch Kommas getrennte Liste mehreren Namespaces freigeben. `trident.netapp.io/shareToNamespace: namespace2,namespace3,namespace4`Beispiel: .
   - Mit können Sie alle Namespaces teilen `*`. Beispiel: `trident.netapp.io/shareToNamespace: *`
   - Sie können die PVC so aktualisieren, dass die Anmerkung jederzeit enthalten `shareToNamespace` ist.

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. **Destination Namespace Owner:** 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. **Destination Namespace Owner:** Erstellen Sie eine PVC (`pvc2`) im Destination Namespace (`namespace2`) mit der `shareFromPVC` Anmerkung die Quell-PVC zu bestimmen.

```
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

Trident liest die `shareFromPVC` Annotation auf der Ziel-PVC und erstellt das Ziel-PV als ein untergeordnetes Volume ohne eigene Speicherressource, die auf das Quell-PV verweist und die Quell-PV-Speicherressource gemeinsam nutzt. 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. Trident entfernt den Zugriff auf das Volume im Quell-Namespace und hat auch Zugriff auf andere Namespaces, die das Volume gemeinsam nutzen. Wenn alle Namespaces, die auf das Volume verweisen, entfernt werden, löscht Trident das Volume.

Zum Abfragen untergeordneter Volumes verwenden `tridentctl get`

Mit dem[`tridentctl` Dienstprogramm können Sie den Befehl ausführen `get`, um untergeordnete Volumes zu erhalten. Weitere Informationen finden Sie unter Link:../Trident-reference/tridentctl.html[`tridentctl`

Commands and options].

```
Usage:
  tridentctl get [option]
```

Markierungen:

- `-h, --help`: Hilfe für Bände.
- `--parentOfSubordinate string`: Abfrage auf untergeordneten Quellvolume beschränken.
- `--subordinateOf string`: Abfrage auf Untergebene des Volumens beschränken.

Einschränkungen

- Trident kann nicht verhindern, dass Zielnamepaces auf das gemeinsam genutzte 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 Quell-PVC nicht aufheben, indem Sie die Anmerkungen oder `shareFromNamespace` entfernen `shareToNamespace` oder den CR löschen `TridentVolumeReference`. 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".
- Sehen Sie sich die Demo an "NetAppTV".

Volumes können in Namespaces geklont werden

Mit Trident können Sie neue Volumes unter Verwendung vorhandener Volumes oder Volume-Snapshots aus einem anderen Namespace im selben Kubernetes-Cluster erstellen.

Voraussetzungen

Stellen Sie vor dem Klonen von Volumes sicher, dass die Quell- und Ziel-Back-Ends vom gleichen Typ sind und dieselbe Storage-Klasse aufweisen.

Schnellstart

Die Einrichtung von Volume-Klonen ist in wenigen Schritten möglich.

Konfigurieren Sie die Quell-PVC zum Klonen des Volume

Der Eigentümer des Quell-Namespace erteilt die Berechtigung, auf die Daten im Quell-PVC zuzugreifen.

2 Erteilen Sie die Berechtigung zum Erstellen eines CR im Zielspeicherort

Der Clusteradministrator erteilt dem Eigentümer des Ziel-Namespace die Berechtigung, das TridentVolumeReference CR zu erstellen.

3 Erstellen Sie TridentVolumeReference im Ziel-Namespace

Der Eigentümer des Ziel-Namespace erstellt das TridentVolumeReference CR, um sich auf das Quell-PVC zu beziehen.

4 Erstellen Sie die Klon-PVC im Ziel-Namespace

Der Eigentümer des Ziel-Namespace erstellt die PVC zum Klonen der PVC aus dem Quell-Namespace.

Konfigurieren Sie die Namensräume für Quelle und Ziel

Um die Sicherheit zu gewährleisten, müssen Volumes über Namespaces hinweg gemeinsam genutzt werden. Der Eigentümer des Quell-Namespace, der Cluster-Administrator und der Eigentümer des Ziel-Namespaces müssen entsprechende Maßnahmen ergreifen. In jedem Schritt wird die Benutzerrolle festgelegt.

Schritte

1. **Source Namespace Owner:** Erstellen Sie die PVC (`pvc1`) im source Namespace (`namespace1`), die die Erlaubnis erteilt, mit dem Ziel-Namespace zu teilen (`namespace2`) mit der `cloneToNamespace` Annotation.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1
  namespace: namespace1
  annotations:
    trident.netapp.io/cloneToNamespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

Trident erstellt das PV und das zugehörige Back-End Storage Volume.

- Sie können das PVC über eine durch Kommas getrennte Liste mehreren Namespaces freigeben. `trident.netapp.io/cloneToNamespace: namespace2,namespace3,namespace4`Beispiel: .
- Mit können Sie alle Namespaces teilen `*`. Beispiel: `trident.netapp.io/cloneToNamespace: *`
- Sie können die PVC so aktualisieren, dass die Anmerkung jederzeit enthalten `cloneToNamespace` ist.

2. **Cluster admin:** Erstellen Sie die benutzerdefinierte Rolle und kubeconfig, um dem Ziel-Namespace-Eigentümer die Berechtigung zu erteilen, den TridentVolume Reference CR im Ziel-Namespace zu erstellen(`namespace2`).
3. **Destination Namespace Owner:** 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. **Destination Namespace Owner:** Erstellen Sie eine PVC (`pvc2`) im Destination Namespace (`namespace2`) Verwenden Sie `cloneFromPVC` die oder `cloneFromSnapshot`, und `cloneFromNamespace` Anmerkungen, um die Quell-PVC zu kennzeichnen.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  annotations:
    trident.netapp.io/cloneFromPVC: pvc1
    trident.netapp.io/cloneFromNamespace: namespace1
  name: pvc2
  namespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

Einschränkungen

- Für PVCs, die über ONTAP-nas-Economy-Treiber bereitgestellt werden, werden schreibgeschützte Klone nicht unterstützt.

Replizieren Sie Volumes mit SnapMirror

Trident unterstützt Spiegelungsbeziehungen zwischen einem Quell-Volume auf einem Cluster und dem Ziel-Volume auf dem Peering-Cluster, damit Daten für Disaster Recovery repliziert werden. Sie können eine benutzerdefinierte Ressourcendefinition (CRD, Namested 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

- **Trident**: Trident Version 22.10 oder höher muss sowohl auf den Quell- als auch auf den Ziel-Kubernetes-Clustern existieren, die ONTAP als Backend nutzen.
- **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.

- **Trident und SVM**: Die Peered Remote SVMs müssen Trident auf dem Ziel-Cluster zur Verfügung stehen.

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
      ```

      Trident ruft die internen Informationen für das Volume und den aktuellen DP-Status des Volumes ab

und füllt dann das Statusfeld der MirrorRelation 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"
```

Trident erstellt eine SnapMirror Beziehung mit dem Namen der konfigurierten Beziehungsrichtlinie (oder dem Standard für ONTAP) und initialisiert diesen.

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
```

Trident überprüft die CRD der tridentMirrorRelationship und erstellt das Volume nicht, wenn die Beziehung nicht vorhanden ist. Wenn die Beziehung besteht, stellt Trident sicher, dass die neue FlexVol volume auf einer SVM platziert wird, die mit der Remote-SVM, die in MirrorRelation definiert ist, 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 Trident den gewünschten Status angibt. 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

Trident unterstützt folgende Vorgänge auf primären und sekundären 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)

Trident prüft, ob replizierte VES vorhanden sind, und gibt die Replizierungsbeziehung frei, bevor das Volume gelöscht werden soll.

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 Trident den Löschvorgang abgeschlossen hat. Wenn der für den Löschvorgang ausgewählte TMR bereits den Status *heraufgestuft* hat, gibt es keine bestehende Spiegelbeziehung und der TMR wird entfernt und Trident 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 Trident über eine direkte Verbindung zum ONTAP-Cluster verfügt. 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.

Verwenden Sie die CSI-Topologie

Trident kann selektiv Volumes erstellen und an Nodes in einem Kubernetes-Cluster anhängen, indem Sie die verwenden "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. Um die Bereitstellung von Volumes für Workloads in einer Architektur mit mehreren Zonen zu vereinfachen, verwendet Trident die CSI-Topologie.

Erfahren Sie mehr über die Funktion „CSI-Topologie "Hier"“.

Kubernetes bietet zwei unterschiedliche Modi für die Volume-Bindung:

- Mit `VolumeBindingMode` Set to `Immediate` erzeugt Trident das Volumen ohne jegliche Topologiewahrnehmung. Die Volume-Bindung und die dynamische Bereitstellung werden bei der Erstellung des PVC behandelt. Dies ist die Standardeinstellung `VolumeBindingMode` und eignet sich für Cluster, die keine Topologieeinschränkungen erzwingen. Persistente Volumes werden erstellt, ohne von den Planungsanforderungen des anfragenden Pods abhängig zu sein.
- Mit der `VolumeBindingMode` Einstellung auf `WaitForFirstConsumer` wird die Erstellung und Bindung eines persistenten Volumes für eine PVC 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.

Für den `WaitForFirstConsumer` 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:

- Ein 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 Topologiebewusstsein und `topology.kubernetes.io/zone`) einführen(`topology.kubernetes.io/region`. Diese Bezeichnungen **sollten auf Knoten im Cluster** vorhanden sein, bevor Trident installiert ist, damit Trident topologiefähig 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","kube
rnetes.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","kube
rnetes.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","kube
rnetes.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

Trident Storage-Back-Ends können so entworfen werden, dass sie Volumes selektiv basierend auf Verfügbarkeitszonen bereitstellen. Jedes Backend kann einen optionalen Block enthalten `supportedTopologies` , der eine Liste der unterstützten 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

```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

```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 eine Teilmenge der Regionen und Zonen enthalten, die in einem Back-End bereitgestellt werden, erstellt Trident auf dem Back-End ein Volume.

Sie können auch pro Speicherpool definieren `supportedTopologies`. 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
    supportedTopologies:
      - topology.kubernetes.io/region: us-central1
        topology.kubernetes.io/zone: us-central1-a
  - labels:
      workload: dev
    supportedTopologies:
      - topology.kubernetes.io/region: us-central1
        topology.kubernetes.io/zone: us-central1-b
```

In diesem Beispiel stehen die `region` Etiketten und `zone` für den Speicherort des Speicherpools. `topology.kubernetes.io/region` Und `topology.kubernetes.io/zone` legen Sie fest, wo die Speicherpools genutzt 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: null
name: netapp-san-us-east1
provisioner: csi.trident.netapp.io
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
  - matchLabelExpressions: null
  - 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 zu verwendenden Zonen und Regionen bereit. Die `netapp-san-us-east1` StorageClass erstellt VES auf dem `san-backend-us-east1` oben definierten Back-End.

Schritt 3: Erstellen und verwenden Sie ein PVC

Wenn die StorageClass erstellt und einem Backend zugeordnet wird, können Sie jetzt PVCs erstellen.

Siehe das folgende Beispiel `spec`:

```
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata: null
name: pvc-san
spec: null
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: netapp-san-us-east1
```

Das Erstellen eines PVC mithilfe dieses Manifests würde Folgendes zur Folge haben:

```
kubectl create -f pvc.yaml
persistentvolumeclaim/pvc-san created
kubectl get pvc
NAME      STATUS    VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS
AGE
pvc-san   Pending                                      netapp-san-us-east1
2s
kubectl 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                         Message
  ----    ------                ----  ----                         -------
  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:

```yaml
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: vol1
    persistentVolumeClaim:
      claimName: pvc-san
  containers:
  - name: sec-ctx-demo
    image: busybox
    command: [ "sh", "-c", "sleep 1h" ]
    volumeMounts:
    - name: vol1
      mountPath: /data/demo
    securityContext:
      allowPrivilegeEscalation: false
```

Diese PodSpec weist Kubernetes an, den Pod auf Nodes zu planen, die in der Region vorhanden sind `us-east1`, und aus jedem Node, der in der Zone oder `us-east1-b` vorhanden ist, auszuwählen `us-east1-a`.

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
```

Back-Ends aktualisieren, um sie einzuschließen `supportedTopologies`

Bereits vorhandene Back-Ends können aktualisiert werden, um eine Liste der Verwendung `tridentctl backend update` aufzunehmen `supportedTopologies`. 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 Trident erstellten Volumes erstellen, einen außerhalb von Trident erstellten Snapshot importieren, ein neues Volume aus einem vorhandenen Snapshot erstellen und Volume-Daten aus Snapshots wiederherstellen.

Überblick

Volume-Snapshot wird unterstützt von `ontap-nas` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy` , `solidfire-san` , `gcp-cvs` , `azure-netapp-files` , Und `google-cloud-netapp-volumes` 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 Ihre Kubernetes-Distribution den Snapshot Controller und CRDs nicht enthält, finden Sie weitere Informationen unter 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 eine `VolumeSnapshotClass`. Weitere Informationen finden Sie unter "VolumeSnapshotKlasse".
   - Der `driver` verweist auf den Trident-CSI-Treiber.
   - `deletionPolicy` Kann oder `Retain` sein `Delete`. Wenn auf festgelegt `Retain`, wird der zugrunde liegende physische Snapshot auf dem Speicher-Cluster auch dann beibehalten, wenn das `VolumeSnapshot` Objekt gelöscht wird.

     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`, und der Name des Snapshots wird auf festgelegt `pvc1-snap`. Ein VolumeSnapshot ist analog zu einer PVC und einem Objekt zugeordnet `VolumeSnapshotContent`, 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 das Objekt für den `pvc1-snap` VolumeSnapshot identifizieren `VolumeSnapshotContent`, indem Sie es beschreiben. Das `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

Sie können verwenden `dataSource`, um eine PVC mit einem VolumeSnapshot zu erstellen, der als Datenquelle benannt `<pvc-name>` ist. 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 als Datenquelle erstellt `pvc1-snap`.

```
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: pvc1-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

Importieren Sie einen Volume-Snapshot

Trident unterstützt das, damit der "Vorab bereitgestellter Snapshot-Prozess von Kubernetes"Clusteradministrator ein Objekt erstellen und Snapshots importieren kann `VolumeSnapshotContent`, die außerhalb von Trident erstellt wurden.

Bevor Sie beginnen

Trident muss das übergeordnete Volume des Snapshots erstellt oder importiert haben.

Schritte

1. **Cluster admin:** Erstellen Sie ein `VolumeSnapshotContent` Objekt, das auf den Back-End-Snapshot verweist. Dadurch wird der Snapshot Workflow in Trident gestartet.
   - Geben Sie den Namen des Back-End-Snapshots in `annotations` als `trident.netapp.io/internalSnapshotName: <"backend-snapshot-name">`an.
   - Geben Sie `<name-of-parent-volume-in-trident>/<volume-snapshot-content-name>` in an `snapshotHandle`. Dies ist die einzige Information, die Trident vom externen Snapshotter im Aufruf zur Verfügung gestellt `ListSnapshots` wird.

     Der `<volumeSnapshotContentName>` kann aufgrund von Einschränkungen bei der CR-Benennung nicht immer mit dem Namen des Back-End-Snapshots übereinstimmen.

     Beispiel

     Im folgenden Beispiel wird ein Objekt erstellt `VolumeSnapshotContent`, das auf einen 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>
  volumeSnapshotRef:
    name: import-snap
    namespace: default
```

2. **Cluster admin:** Erstellen Sie den `VolumeSnapshot` CR, der das Objekt referenziert `VolumeSnapshotContent`. Damit wird der Zugriff auf die Verwendung des in einem bestimmten Namespace benötigt `VolumeSnapshot`.

   Beispiel

   Im folgenden Beispiel wird ein CR mit dem `import-snap` Namen erstellt `VolumeSnapshot`, der auf den Namen `import-snap-content` verweist `VolumeSnapshotContent`.

   ```
   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 Schnapper erkennt das neu erstellte `VolumeSnapshotContent` und führt den `ListSnapshots` Aufruf aus. Trident erstellt die `TridentSnapshot`.
   - Der externe Schnapper setzt den `VolumeSnapshotContent` auf `readyToUse` und den `VolumeSnapshot` auf `true`.
   - Trident kehrt zurück `readyToUse=true`.
4. **Jeder Benutzer:** Erstellen Sie ein `PersistentVolumeClaim`, um auf den neuenzu verweisen `VolumeSnapshot`, wobei der `spec.dataSource` (oder `spec.dataSourceRef`) Name der Name ist

VolumeSnapshot.

Beispiel

Im folgenden Beispiel wird eine PVC erstellt, die auf den Namen import-snap verweist VolumeSnapshot.

```
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 der mit den Treibern und ontap-nas-economy bereitgestellten Volumes zu ermöglichen ontap-nas. Aktivieren Sie das .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.

In-Place-Volume-Wiederherstellung aus einem Snapshot

Trident ermöglicht mithilfe des CR-Systems (TASR) 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.

Trident unterstützt die Wiederherstellung von Snapshots auf dem ontap-san, , ontap-san-economy ontap-nas ontap-nas-flexgroup azure-netapp-files , , , gcp-cvs, google-cloud-netapp-

`volumes` und `solidfire-san` 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`.

   Der TASR CR muss sich in einem Namensraum befinden, in dem PVC und VS vorhanden sind.

```
cat tasr-pvc1-snapshot.yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentActionSnapshotRestore
metadata:
  name: trident-snap
  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/trident-snap created
```

Ergebnisse

Trident stellt die Daten aus dem Snapshot wieder her. Sie können den Wiederherstellungsstatus von

Snapshots überprüfen:

```
kubectl get tasr -o yaml
```

```
apiVersion: trident.netapp.io/v1
items:
- apiVersion: trident.netapp.io/v1
  kind: TridentActionSnapshotRestore
  metadata:
    creationTimestamp: "2023-04-14T00:20:33Z"
    generation: 3
    name: trident-snap
    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 Trident den Vorgang bei einem Ausfall nicht automatisch erneut. Sie müssen den Vorgang erneut ausführen.
- Kubernetes-Benutzer ohne Administratorzugriff müssen möglicherweise vom Administrator zum Erstellen eines TASR CR in ihrem Applikations-Namespace erhalten.

Löschen Sie ein PV mit den zugehörigen Snapshots

Beim Löschen eines persistenten Volumes mit zugeordneten Snapshots wird das entsprechende Trident-Volume auf den „Löschstatus“ aktualisiert. Entfernen Sie die Volume-Snapshots, um das 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.yam
l
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 ggf. `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` Ihren Namespace und aktualisieren Sie `namespace` ihn.

Weiterführende Links

- "Volume Snapshots"
- "VolumeSnapshotKlasse"