



# Trident 24.10-Dokumentation

Trident

NetApp  
March 25, 2025

# Inhalt

Trident 24.10-Dokumentation .....	1
Versionshinweise .....	2
Was ist neu .....	2
Was ist neu in 24.10 .....	2
Änderungen in 24.06 .....	3
Änderungen in 24.02 .....	4
Änderungen in 23.10 .....	5
Änderungen in 23.07.1 .....	5
Änderungen in 23.07 .....	6
Änderungen in 23.04 .....	7
Änderungen in 23.01.1 .....	8
Änderungen in 23.01 .....	8
Änderungen in 22.10 .....	9
Änderungen in 22.07 .....	10
Änderungen in 22.04 .....	11
Änderungen in 22.01.1 .....	12
Änderungen in 22.01.0 .....	12
Änderungen in 21.10.1 .....	13
Änderungen in 21.10.0 .....	13
Bekannte Probleme .....	14
Weitere Informationen .....	15
Frühere Versionen der Dokumentation .....	15
Los geht's .....	16
Erfahren Sie mehr über Trident .....	16
Erfahren Sie mehr über Trident .....	16
Architektur von Trident .....	17
Konzepte .....	19
Schnellstart für Trident .....	22
Was kommt als Nächstes? .....	23
Anforderungen .....	23
Wichtige Informationen über Trident .....	24
Unterstützte Frontends (Orchestrators) .....	24
Unterstützte Back-Ends (Storage) .....	25
Anforderungen an die Funktionen .....	25
Getestete Host-Betriebssysteme .....	25
Host-Konfiguration .....	26
Konfiguration des Storage-Systems .....	26
Trident-Ports .....	26
Container-Images und entsprechende Kubernetes-Versionen .....	26
Installation Von Trident .....	27
Installation über den Trident Operator .....	27
Installieren Sie mit tridentctl .....	27
Verwenden Sie Trident .....	28

Bereiten Sie den Knoten „Worker“ vor . . . . .	28
Auswahl der richtigen Werkzeuge . . . . .	28
Ermittlung des Node-Service . . . . .	28
NFS Volumes . . . . .	29
ISCSI-Volumes . . . . .	29
NVMe/TCP-Volumes . . . . .	33
Installieren Sie die FC Tools . . . . .	34
Unterstützung für Fibre Channel (FC) . . . . .	36
Konfiguration und Management von Back-Ends . . . . .	39
Back-Ends konfigurieren . . . . .	40
Azure NetApp Dateien . . . . .	40
Google Cloud NetApp Volumes . . . . .	58
Cloud Volumes Service für Google Cloud-Back-End konfigurieren . . . . .	74
Konfigurieren Sie ein NetApp HCI- oder SolidFire-Backend . . . . .	85
ONTAP SAN-Treiber . . . . .	91
ONTAP NAS-Treiber . . . . .	117
Amazon FSX für NetApp ONTAP . . . . .	148
Back-Ends mit kubectrl erstellen . . . . .	179
Back-Ends managen . . . . .	186
Erstellen und Managen von Storage-Klassen . . . . .	195
Erstellen Sie eine Speicherklasse . . . . .	195
Management von Storage-Klassen . . . . .	198
Provisionierung und Management von Volumes . . . . .	200
Bereitstellen eines Volumes . . . . .	200
Erweitern Sie Volumes . . . . .	204
Volumes importieren . . . . .	212
Passen Sie Volume-Namen und -Beschriftungen an . . . . .	219
Ein NFS-Volume kann über Namespaces hinweg genutzt werden . . . . .	222
Replizieren Sie Volumes mit SnapMirror . . . . .	226
Verwenden Sie die CSI-Topologie . . . . .	233
Arbeiten Sie mit Snapshots . . . . .	240
Management und Monitoring von Trident . . . . .	249
Upgrade von Trident . . . . .	249
Upgrade von Trident . . . . .	249
Upgrade mit dem Bediener . . . . .	250
Upgrade mit tridentctl . . . . .	255
Managen Sie Trident mit tridentctl . . . . .	256
Befehle und globale Alarmmeldungen . . . . .	256
Befehloptionen und -Flags . . . . .	258
Plug-in-Unterstützung . . . . .	264
Monitoring von Trident . . . . .	264
Überblick . . . . .	264
Schritt 1: Definieren Sie ein Prometheus-Ziel . . . . .	264
Schritt: Erstellen Sie einen Prometheus ServiceMonitor . . . . .	265
Schritt 3: Abfrage der Trident-Kennzahlen mit PromQL . . . . .	265

Informationen zur Trident AutoSupport Telemetrie .....	266
Deaktivieren Sie Trident-Kennzahlen .....	267
Deinstallieren Sie Trident .....	268
Bestimmen Sie die ursprüngliche Installationsmethode .....	268
Deinstallieren Sie die Installation eines Trident-Operators .....	268
Deinstallieren Sie eine <code>tridentctl</code> Installation .....	269
Trident für Docker .....	270
Voraussetzungen für die Bereitstellung .....	270
Überprüfen Sie die Anforderungen .....	270
NVMe-Tools .....	272
Implementieren Sie Trident .....	273
Docker Managed Plug-in-Methode (Version 1.13/17.03 und höher) .....	273
Herkömmliche Methode (Version 1.12 oder früher) .....	275
Starten Sie Trident beim Systemstart .....	277
Aktualisieren oder deinstallieren Sie Trident .....	278
Upgrade .....	278
Deinstallieren .....	279
Arbeiten mit Volumes .....	279
Erstellen eines Volumes .....	280
Entfernen Sie ein Volume .....	280
Klonen Sie ein Volume .....	280
Zugriff auf extern erstellte Volumes .....	282
Treiberspezifische Volume-Optionen .....	282
Sammelt Protokolle .....	288
Sammelt Protokolle für die Fehlerbehebung .....	288
Allgemeine Tipps zur Fehlerbehebung .....	288
Management mehrerer Trident Instanzen .....	289
Schritte für Docker Managed Plug-in (Version 1.13/17.03 oder höher) .....	289
Schritte für herkömmliche (Version 1.12 oder früher) .....	289
Optionen für die Storage-Konfiguration .....	290
Globale Konfigurationsoptionen .....	290
ONTAP-Konfiguration .....	291
Konfiguration von Element Software .....	299
Bekannte Probleme und Einschränkungen .....	301
Das Upgrade des Trident Docker Volume Plug-ins auf 20.10 und höher aus älteren Versionen führt zu einem Upgrade-Fehler, ohne dass solche Datei- oder Verzeichnisfehler auftreten. ....	301
Volume-Namen müssen mindestens 2 Zeichen lang sein. ....	302
Docker Swarm verfügt über bestimmte Verhaltensweisen, die verhindern, dass Trident diese bei jeder Storage- und Treiberkombination unterstützen kann. ....	302
Wenn eine FlexGroup bereitgestellt wird, stellt ONTAP keine zweite FlexGroup bereit, wenn die zweite FlexGroup über einen oder mehrere Aggregate verfügt, die mit der bereitgestellten FlexGroup gemeinsam genutzt werden. ....	302
Best Practices und Empfehlungen .....	303
Einsatz .....	303
Implementieren Sie diesen in einem dedizierten Namespace .....	303

Verwenden Sie Kontingente und Bereichsgrenzen, um den Storage-Verbrauch zu kontrollieren . . . . .	303
Storage-Konfiguration . . . . .	303
Plattformübersicht . . . . .	303
Best Practices für ONTAP und Cloud Volumes ONTAP . . . . .	304
SolidFire Best Practices in sich vereint . . . . .	308
Wo finden Sie weitere Informationen? . . . . .	310
Integration von Trident . . . . .	310
Auswahl und Implementierung der Treiber . . . . .	310
Design der Storage-Klasse . . . . .	314
Virtual Pool Design . . . . .	315
Volume-Vorgänge . . . . .	316
OpenShift Services implementieren . . . . .	318
Kennzahlungsservice . . . . .	320
Datensicherung und Disaster Recovery . . . . .	321
Replizierung und Recovery mit Trident . . . . .	321
SVM-Replizierung und Recovery . . . . .	322
Volume-Replizierung und Recovery . . . . .	323
Snapshot Datensicherung . . . . .	323
Sicherheit . . . . .	324
Sicherheit . . . . .	324
Linux Unified Key Setup (LUKS) . . . . .	325
Kerberos Verschlüsselung während der Übertragung . . . . .	330
Sicherung von Applikationen mit Trident-Schutz . . . . .	339
Weitere Informationen zu Trident Protect . . . . .	339
Was kommt als Nächstes? . . . . .	339
Installieren Sie Trident Protect . . . . .	339
Trident Protect – die Anforderungen . . . . .	339
Installieren und konfigurieren Sie Trident Protect . . . . .	342
Installieren Sie das Trident Protect CLI-Plug-in . . . . .	348
Management von Trident Protect . . . . .	352
Managen von Trident Schützen Sie die Autorisierung und Zugriffssteuerung . . . . .	352
Generieren Sie ein Trident Protect Supportpaket . . . . .	358
Upgrade von Trident Protect . . . . .	360
Managen und sichern Sie Applikationen . . . . .	360
Verwenden Sie zum Verwalten von Buckets Trident Protect AppVault-Objekte . . . . .	360
Definieren Sie mit Trident Protect eine Applikation für das Management . . . . .	369
Sicherung von Applikationen mit Trident Protect . . . . .	371
Stellen Sie Applikationen mit Trident Protect wieder her . . . . .	379
Replizieren Sie Applikationen mit NetApp SnapMirror und Trident Protect . . . . .	395
Migrieren Sie Applikationen mit Trident Protect . . . . .	409
Managen von Trident Protect-Ausführungshaken . . . . .	413
Deinstallieren Sie Trident Protect . . . . .	418
Wissen und Support . . . . .	419
Häufig gestellte Fragen . . . . .	419
Allgemeine Fragen . . . . .	419

Installation und Verwendung von Trident auf einem Kubernetes-Cluster . . . . .	419
Fehlerbehebung und Support . . . . .	420
Upgrade von Trident . . . . .	422
Back-Ends und Volumes managen . . . . .	422
Fehlerbehebung . . . . .	426
Allgemeine Fehlerbehebung . . . . .	426
Die Bereitstellung von Trident mit dem Operator ist fehlgeschlagen . . . . .	428
Erfolgreiche Trident-Bereitstellung mit <code>tridentctl</code> . . . . .	430
Entfernen Sie Trident und CRDs vollständig . . . . .	430
Fehler beim Entstopfen des NVMe-Node bei den RWX-RAW-Block-Namespaces o Kubernetes 1.26 . . . . .	431
Support . . . . .	432
Lebenszyklus des Trident Supports . . . . .	432
Self-Support . . . . .	433
Community-Support . . . . .	433
Technischer Support von NetApp . . . . .	433
Finden Sie weitere Informationen . . . . .	433
Referenz . . . . .	434
Trident-Ports . . . . .	434
Trident-Ports . . . . .	434
Trident REST-API . . . . .	434
Wann die REST-API verwendet werden soll . . . . .	434
REST-API wird verwendet . . . . .	434
Befehlszeilenoptionen . . . . .	435
Protokollierung . . . . .	435
Kubernetes . . . . .	435
Docker . . . . .	436
RUHE . . . . .	436
Kubernetes und Trident Objekte . . . . .	436
Wie interagieren die Objekte miteinander? . . . . .	436
`PersistentVolumeClaim` Kubernetes Objekte . . . . .	437
`PersistentVolume` Kubernetes Objekte . . . . .	439
`StorageClass` Kubernetes Objekte . . . . .	439
`VolumeSnapshotClass` Kubernetes Objekte . . . . .	443
`VolumeSnapshot` Kubernetes Objekte . . . . .	443
`VolumeSnapshotContent` Kubernetes Objekte . . . . .	444
`CustomResourceDefinition` Kubernetes Objekte . . . . .	444
Trident-Objekte <code>StorageClass</code> . . . . .	445
Trident Back-End-Objekte . . . . .	445
Trident-Objekte <code>StoragePool</code> . . . . .	445
Trident-Objekte <code>Volume</code> . . . . .	445
Trident-Objekte <code>Snapshot</code> . . . . .	447
Trident <code>ResourceQuota</code> -Objekt . . . . .	448
Pod Security Standards (PSS) und Security Context Constraints (SCC) . . . . .	449
Erforderlicher Kubernetes-Sicherheitskontext und zugehörige Felder . . . . .	449

Pod-Sicherheitsstandards (PSS) . . . . .	450
Pod-Sicherheitsrichtlinien (PSP) . . . . .	450
Sicherheitskontexteinschränkungen (SCC) . . . . .	452
Rechtliche Hinweise . . . . .	454
Urheberrecht . . . . .	454
Marken . . . . .	454
Patente . . . . .	454
Datenschutzrichtlinie . . . . .	454
Open Source . . . . .	454

# Trident 24.10-Dokumentation



# Versionshinweise

## Was ist neu

Versionshinweise enthalten Informationen zu neuen Funktionen, Verbesserungen und Fehlerkorrekturen in der neuesten Version von Trident.



Die `tridentctl` Binärdatei für Linux, die in der ZIP-Datei des Installers enthalten ist, ist die getestete und unterstützte Version. Beachten Sie, dass die `macos` im Teil der ZIP-Datei bereitgestellte Binärdatei `/extras` nicht getestet oder unterstützt wird.

## Was ist neu in 24.10

### Vorgestellt Werden

- Google Cloud NetApp Volumes Treiber ist jetzt für NFS-Volumes allgemein verfügbar und unterstützt das zonenbasierte Provisioning.
- Die GCP Workload-Identität wird mit GKE als NetApp-Identität für Google Cloud Volumes verwendet.
- Konfigurationsparameter zu ONTAP-SAN- und ONTAP-SAN-Economy-Treibern hinzugefügt `formatOptions`, um Benutzern die Angabe von LUN-Formatoptionen zu ermöglichen.
- Verringerte Azure NetApp Files-Mindestgröße für ein Volume auf 50 gib. Azure neue Mindestgröße wird voraussichtlich ab November verfügbar sein.
- Konfigurationsparameter hinzugefügt `denyNewVolumePools`, um ONTAP-NAS-Economy- und ONTAP-SAN-Economy-Treiber auf vorhandene FlexVol-Pools zu beschränken.
- Erkennung für das Hinzufügen, Entfernen oder Umbenennen von Aggregaten aus der SVM über alle ONTAP-Treiber hinweg hinzugefügt.
- LUKS LUNs wurde um 18 MiB Overhead erweitert, um sicherzustellen, dass die gemeldete PVC-Größe verwendet werden kann.
- Verbesserte ONTAP-SAN- und ONTAP-SAN-Economy-Knotenstufe und Entstaunung der Fehlerbehandlung, damit Geräte nach einem Ausfall nicht mehr entfernt werden können.
- Es wurde ein benutzerdefinierter Rollengenerator hinzugefügt, mit dem Kunden eine minimalistische Rolle für Trident in ONTAP erstellen können.
- Zusätzliche Protokollierung für die Fehlerbehebung hinzugefügt `lsscsi` ("[Ausgabe #792](#)").

### Kubernetes

- Neue Trident-Funktionen für Kubernetes-native Workflows hinzugefügt:
  - Datensicherung
  - Datenmigration
  - Disaster Recovery
  - Applikationsmobilität

["Erfahren Sie mehr über Trident Protect"](#).

- Installern wurde ein neues Flag hinzugefügt `--k8s_api_qps`, um den QPS-Wert festzulegen, der von

Trident für die Kommunikation mit dem Kubernetes-API-Server verwendet wird.

- Flag zu Installern für das automatische Management von Speicherprotokollabhängigkeiten auf Kubernetes-Cluster-Nodes hinzugefügt `--node-prep`. Kompatibilität mit Amazon Linux 2023 iSCSI Storage-Protokoll getestet und verifiziert
- Unterstützung für Force-Trennen für ONTAP-NAS-Economy-Volumes bei nicht-graziösen Shutdown-Szenarien für Knoten wurde hinzugefügt.
- Neue ONTAP-NAS-Economy NFS-Volumes verwenden bei der Back-End-Option Exportrichtlinien gemäß `qtree autoExportPolicy`. Qtrees werden zum Zeitpunkt der Veröffentlichung nur den Node-restriktiven Exportrichtlinien zugeordnet, um die Zugriffssteuerung und die Sicherheit zu verbessern. Vorhandene qtrees werden auf das neue Exportrichtlinien-Modell umgestellt, wenn Trident das Volume ohne Beeinträchtigung aktiver Workloads von allen Nodes wieder veröffentlicht.
- Unterstützung für Kubernetes 1.31 hinzugefügt.

### Experimentelle Verbesserungen

- Technische Vorschau für Fibre-Channel-Unterstützung auf ONTAP-SAN-Treiber hinzugefügt Siehe "[Fibre Channel-Unterstützung](#)".

### Korrekturen

- **Kubernetes:**
  - Festancher Aufnahme Webhook verhindert Trident Helm Installationen ("[Ausgabe #839](#)").
  - Fester Affinitätsschlüssel in Ruderkartenwerten ("[Ausgabe #898](#)").
  - Behoben `tridentControllerPluginNodeSelector/tridentNodePluginNodeSelector` funktioniert nicht mit "true" Wert ("[Ausgabe #899](#)").
  - Gelöschte Momentaufnahmen, die während des Klonens erstellt wurden ("[Ausgabe #901](#)").
- Unterstützung für Windows Server 2019 hinzugefügt.
- Behoben ``go mod tidy`` in Trident repo ("[Ausgabe #767](#)").

### Abschreibungen

- **Kubernetes:**
  - Aktualisiertes, mindestens unterstütztes Kubernetes auf 1.25
  - Unterstützung für POD-Sicherheitsrichtlinie wurde entfernt.

### Neubranding von Produkten

Ab Version 24.10 wird Astra Trident unter dem neuen Namen Trident (NetApp Trident) firmiere. Dieses Rebranding hat keine Auswirkungen auf Funktionen, unterstützte Plattformen oder Interoperabilität für Trident.

## Änderungen in 24.06

### Vorgestellt Werden

- **WICHTIG:** Der `limitVolumeSize` Parameter beschränkt jetzt die `qtree/LUN` Größen in den ONTAP Economy Treibern. Verwenden Sie den neuen `limitVolumePoolSize` Parameter, um die FlexVol-Größen in diesen Treibern zu steuern. ("[Ausgabe #341](#)").
- Zusätzliche Möglichkeit für iSCSI Selbstheilung, SCSI-Scans durch exakte LUN-ID zu initiieren, wenn

veraltete Initiatorgruppen verwendet werden ("[Ausgabe #883](#)").

- Zusätzliche Unterstützung für Volume-Klonvorgänge und Größenänderungsvorgänge, die zulässig waren, selbst wenn sich das Backend im unterbrochenen Modus befindet.
- Benutzerdefinierte Protokolleinstellungen für den Trident-Controller, die an Trident-Node-Pods weitergegeben werden sollen, wurden hinzugefügt.
- Unterstützung in Trident hinzugefügt, um REST standardmäßig anstelle von ZAPI für ONTAP-Versionen 9.15.1 und höher zu verwenden.
- Zusätzliche Unterstützung für benutzerdefinierte Volume-Namen und Metadaten auf den ONTAP Storage-Back-Ends für neue persistente Volumes.
- Erweitert den `azure-netapp-files` (ANF)-Treiber, um das Snapshot-Verzeichnis standardmäßig automatisch zu aktivieren, wenn die NFS-Mount-Optionen auf NFS-Version 4.x eingestellt sind
- Bottlerocket-Unterstützung für NFS-Volumes hinzugefügt.
- Unterstützung für die technische Vorschau von Google Cloud NetApp Volumes hinzugefügt.

### Kubernetes

- Unterstützung für Kubernetes 1.30 hinzugefügt.
- Zusätzliche Fähigkeit für Trident DemonSet, Zombie-Mounts und Restverfolgungsdateien beim Start zu reinigen ("[Ausgabe #883](#)").
- PVC-Beschriftung für dynamischen Import von LUKS-Volumes ( ) hinzugefügt ``trident.netapp.io/luksEncryption`` ("[Ausgabe #849](#)").
- ANF-Treiber wurde um Topologiebewusstsein erweitert.
- Unterstützung für Windows Server 2022-Knoten hinzugefügt.

### Korrekturen

- Fehler bei der Trident-Installation aufgrund veralteter Transaktionen behoben.
- `tridentctl` wurde behoben, um Warnmeldungen von Kubernetes ( ) zu ignorieren ("[Ausgabe #892](#)").
- Die Priorität des Trident-Controllers wurde in 0 ("[Ausgabe #887](#)") geändert `SecurityContextConstraint`.
- ONTAP-Treiber akzeptieren jetzt Volumengrößen unter 20MiB ("[Problem\[#885\]](#)").
- Trident wurde behoben, um ein Verkleinern der FlexVols während des Größenänderungsvorgangs für den ONTAP-SAN-Treiber zu verhindern.
- Fehler beim Import von ANF-Volumes mit NFS v4.1 behoben.

## Änderungen in 24.02

### Vorgestellt Werden

- Unterstützung für Cloud Identity wurde zugefügt.
  - AKS mit ANF – Azure Workload Identity wird als Cloud-Identität verwendet.
  - EKS mit FSxN – AWS IAM-Rolle wird als Cloud-Identität verwendet.
- Unterstützung für die Installation von Trident als Add-on auf EKS Cluster von der EKS Konsole hinzugefügt.

- Zusätzliche Möglichkeit zur Konfiguration und Deaktivierung der iSCSI-Selbstheilung ("[Ausgabe #864](#)").
- ONTAP-Treiber wurden um FSX Personality erweitert, um die Integration mit AWS IAM und SecretsManager zu ermöglichen und Trident zu ermöglichen FSX-Volumes mit Backups zu löschen ("[Ausgabe #453](#)").

#### Kubernetes

- Unterstützung für Kubernetes 1.29 hinzugefügt.

#### Korrekturen

- ACP-Warnmeldungen wurden behoben, wenn ACP nicht aktiviert ist ("[Ausgabe #866](#)").
- Es wurde eine Verzögerung von 10 Sekunden hinzugefügt, bevor eine Klonaufteilung während der Snapshot-Löschung für ONTAP-Treiber durchgeführt wird, wenn ein Klon mit dem Snapshot verknüpft ist.

#### Abschreibungen

- In-toto-Teststationen-Framework aus Multi-Plattform-Image-Manifesten entfernt.

## Änderungen in 23.10

#### Korrekturen

- Die Volume-Erweiterung wurde behoben, wenn eine neue angeforderte Größe kleiner ist als die Gesamtgröße für ONTAP-nas- und ONTAP-nas-FlexGroup-Speichertreiber ("[Ausgabe #834](#)").
- Die Volume-Größe wurde festgelegt, um nur die nutzbare Größe des Volumes während des Imports für ONTAP-nas- und ONTAP-nas-FlexGroup-Speichertreiber anzuzeigen ("[Ausgabe #722](#)").
- FlexVol Namenskonvertierung für ONTAP-NAS-Economy wurde korrigiert.
- Fehler bei der Trident-Initialisierung auf einem Windows Node wurde beim Neubooten des Node behoben.

#### Vorgestellt Werden

##### Kubernetes

Unterstützung für Kubernetes 1.28 hinzugefügt.

##### Trident

- Unterstützung für die Nutzung von Azure Managed Identities (AMI) mit Azure-netapp-Files Storage-Treibern hinzugefügt.
- Zusätzliche Unterstützung für NVMe over TCP für den ONTAP-SAN-Treiber.
- Hinzugefügt Möglichkeit, die Bereitstellung eines Volumes anzuhalten, wenn Backend auf suspendiert Zustand von Benutzer () gesetzt wird ("[Ausgabe #558](#)").

## Änderungen in 23.07.1

**Kubernetes:** Festes Dämonenset-Löschen zur Unterstützung von Upgrades ohne Ausfallzeiten ("[Ausgabe #740](#)").

# Änderungen in 23.07

## Korrekturen

### Kubernetes

- Trident Upgrade wurde behoben, um alte Pods, die im Abschlusszustand stecken, zu ignorieren ("[Ausgabe #740](#)").
- Tolerierung zur Definition "transient-Trident-Version-pod" () hinzugefügt "[Ausgabe #795](#)".

### Trident

- ONTAP-ZAPI-Anforderungen wurden behoben, um sicherzustellen, dass die LUN-Seriennummern abgefragt werden, wenn LUN-Attribute zur Identifizierung und Behebung von Ghost-iSCSI-Geräten während der Knotenstagevorgänge abgerufen werden.
- Fehlerbehandlung im Speichertreibercode () behoben "[Ausgabe #816](#)".
- Feste Quota-Größe bei Verwendung von ONTAP-Treibern mit use-Rest=true.
- Erstellung von LUN-Klonen in ontap-san-Economy wurde korrigiert.
- Revert publish info field from to devicePath; added Logic to populate and Recover (in einigen Fällen) devicePath field rawDevicePath.

## Vorgestellt Werden

### Kubernetes

- Unterstützung für den Import vorbereiteter Snapshots wurde hinzugefügt.
- Minimale Bereitstellung und Dämonset linux-Berechtigungen ("[Ausgabe #817](#)").

### Trident

- Es wird kein Statusfeld mehr für „Online“ Volumes und Snapshots gemeldet.
- Aktualisiert den Backend-Zustand, wenn das ONTAP-Backend offline ist ("[Probleme #801](#)", "[#543](#)").
- Die LUN-Seriennummer wird während des Workflows „ControllerVolumePublish“ immer abgerufen und veröffentlicht.
- Zusätzliche Logik zur Überprüfung der Seriennummer und Größe des iSCSI Multipath-Geräts hinzugefügt.
- Zusätzliche Überprüfung für iSCSI-Volumes, um sicherzustellen, dass das richtige Multipath-Gerät nicht bereitgestellt wird.

## Experimentelle Verbesserung

Unterstützung für NVMe over TCP für den ONTAP-SAN-Treiber wurde um eine technische Vorschau erweitert.

## Dokumentation

Viele organisatorische und formatierte Verbesserungen wurden vorgenommen.

## Abschreibungen

## Kubernetes

- Unterstützung für v1beta1-Snapshots wurde entfernt.
- Unterstützung für Pre-CSI-Volumes und Speicherklassen wurde entfernt.
- Aktualisiertes, mindestens unterstütztes Kubernetes auf 1.22

## Änderungen in 23.04



Volume-Trennung für ONTAP-SAN-\* -Volumes erzwingen wird nur bei Kubernetes-Versionen mit aktiviertem Non-Graceful Node Shutdown Feature Gate unterstützt. Die Option zum erzwingen des Abtrennens muss zur Installationszeit mit dem Trident-Installer-Flag aktiviert sein `--enable-force-detach`.

## Korrekturen

- Trident-Operator zur Verwendung von IPv6-localhost für die Installation festgelegt, wenn in Spec angegeben.
- Trident Operator Cluster Rollenberechtigungen wurden korrigiert, um mit den Paketberechtigungen synchronisiert zu sein ("[Ausgabe #799](#)").
- Problem beim Anhängen von RAW-Block-Volumes auf mehreren Knoten im RWX-Modus behoben.
- Unterstützung von FlexGroup-Klonen und Volume-Import für SMB-Volumes wurde korrigiert.
- Problem behoben, bei dem der Trident-Controller nicht sofort heruntergefahren werden konnte ("[Ausgabe #811](#)").
- Es wurde ein Fix zur Auflistung aller igroup-Namen hinzugefügt, die mit einer angegebenen LUN verbunden sind, die mit `ontap-san-*` Treibern bereitgestellt wurde.
- Korrektur hinzugefügt, um die Ausführung externer Prozesse bis zum Abschluss zu ermöglichen.
- Kompilierungsfehler für s390-Architektur () behoben "[Ausgabe #537](#)".
- Falsche Protokollierungsebene bei Volume-Mount-Operationen () behoben "[Ausgabe #781](#)".
- Fehler bei der Assertion des potenziellen Typs () behoben "[Ausgabe #802](#)".

## Vorgestellt Werden

- Kubernetes:
  - Unterstützung für Kubernetes 1.27 hinzugefügt.
  - Unterstützung für den Import von LUKS-Volumes wurde hinzugefügt.
  - Zusätzliche Unterstützung für den ReadWriteOncePod PVC-Zugriffsmodus.
  - Unterstützung für Force-Trennen für ONTAP-SAN-\* -Volumes während nicht-Graceful Node Shutdown-Szenarien hinzugefügt.
  - Alle ONTAP-SAN-\* Volumes verwenden nun Initiatorgruppen pro Node. LUNs werden nur Initiatorgruppen zugeordnet, während sie aktiv auf diesen Nodes veröffentlicht werden, um unsere Sicherheit zu verbessern. Bestehende Volumes werden opportunistisch auf das neue igroup Schema umgestellt, wenn Trident feststellt, dass es sicher ist, dies zu tun, ohne aktive Workloads zu beeinträchtigen ("[Ausgabe #758](#)").
  - Verbesserte die Trident-Sicherheit durch Bereinigung nicht genutzter Trident-gemanagter Initiatorgruppen aus ONTAP-SAN-\* Back-Ends.

- Zusätzliche Unterstützung für SMB Volumes mit Amazon FSX für die ontap-nas-Wirtschaft und ontap-nas-flexgroup-Storage-Treiber.
- Unterstützung von SMB-Freigaben mit ontap-nas, ontap-nas-Economy und ontap-nas-Flexgroup-Storage-Treibern hinzugefügt.
- Unterstützung für arm64 Knoten () hinzugefügt ["Ausgabe #732"](#).
- Verbessertes Trident-Shutdown-Verfahren durch Deaktivierung von API-Servern zuerst (["Ausgabe #811"](#)).
- Cross-Plattform-Build-Unterstützung für Windows- und arm64-Hosts zu Makefile hinzugefügt; siehe BUILD.md.

## Abschreibungen

**Kubernetes:** bei der Konfiguration von ONTAP-san- und ONTAP-san-Economy-Treibern werden keine über Backend scoped igroups mehr erstellt (["Ausgabe #758"](#)).

## Änderungen in 23.01.1

### Korrekturen

- Trident-Operator zur Verwendung von IPv6-localhost für die Installation festgelegt, wenn in Spec angegeben.
- Trident Operator Cluster Rollenberechtigungen wurden korrigiert, um mit den Bundle-Berechtigungen synchronisiert zu sein ["Ausgabe #799"](#).
- Korrektur hinzugefügt, um die Ausführung externer Prozesse bis zum Abschluss zu ermöglichen.
- Problem beim Anhängen von RAW-Block-Volumes auf mehreren Knoten im RWX-Modus behoben.
- Unterstützung von FlexGroup-Klonen und Volume-Import für SMB-Volumes wurde korrigiert.

## Änderungen in 23.01



Kubernetes 1.27 wird jetzt in Trident unterstützt. Führen Sie ein Upgrade von Trident durch, bevor Sie ein Upgrade auf Kubernetes durchführen.

### Korrekturen

- Kubernetes: Hinzufügen von Optionen zum Ausschluss der Pod-Sicherheitsrichtlinie zur Korrektur von Trident-Installationen über Helm (["Ausgaben #783, #794"](#)).

## Vorgestellt Werden

### Kubernetes

- Zusätzliche Unterstützung für Kubernetes 1.26
- Verbesserte allgemeine Nutzung der Trident RBAC-Ressourcen (["Ausgabe #757"](#)).
- Verbesserte Automatisierung zum Erkennen und Beheben defekter oder veralteter iSCSI Sitzungen auf Host Nodes
- Unterstützung für Erweiterung der LUKS-verschlüsselten Volumes hinzugefügt.
- Kubernetes: Unterstützung für die Rotation von Anmeldeinformationen für LUKS-verschlüsselte Volumes hinzugefügt.

## Trident

- Zusätzlicher Support für SMB Volumes mit Amazon FSX für ONTAP für den `ontap-nas-Storage-Treiber`
- Unterstützung für NTFS-Berechtigungen bei der Verwendung von SMB-Volumes hinzugefügt.
- Zusätzlicher Support für Storage Pools für GCP Volumes mit CVS Service Level.
- Unterstützung für optionale Verwendung von `flexgroupAggregateList` bei der Erstellung von FlexGroups mit dem `ontap-nas-flexgroup Storage-Treiber` hinzugefügt.
- Verbesserte Performance für den `ontap-nas-Economy-Storage-Treiber` beim Management mehrerer FlexVols.
- Aktivierte Daten-LIF-Updates für alle ONTAP-NAS-Speichertreiber.
- Aktualisierte die Namenskonvention für Trident Deployment und DemonSet zur Berücksichtigung des Host-Node-Betriebssystems.

## Abschreibungen

- Kubernetes: Aktualisierte die minimal unterstützte Version von Kubernetes auf 1.21.
- Daten-LIFs sollten beim Konfigurieren von oder `ontap-san-economy` Treibern nicht mehr angegeben werden `ontap-san`.

## Änderungen in 22.10

Sie müssen die folgenden wichtigen Informationen lesen, bevor Sie auf Trident 22.10 upgraden.

### **Informationen über Trident 22.10**

- Kubernetes 1.25 wird jetzt in Trident unterstützt. Vor dem Upgrade auf Kubernetes 1.25 müssen Sie Trident auf 22.10 aktualisieren.
- Trident setzt die Verwendung der Multipathing-Konfiguration in SAN-Umgebungen strikt durch, mit einem empfohlenen Wert von `find_multipaths: no` in der `Multipath.conf` Datei.



Die Verwendung einer nicht-Multipathing-Konfiguration oder die Verwendung von `find_multipaths: yes` oder `find_multipaths: smart` Wert in der Datei `Multipath.conf` führt zu Mount-Fehlern. Trident empfiehlt die Verwendung von `find_multipaths: no` seit Version 21.07.

## Korrekturen

- Problem spezifisch für ONTAP-Backend erstellt mit `credentials` Feld nicht online kommen während 22.07.0 Upgrade ("[Ausgabe #759](#)").
- **Docker:** Es wurde ein Problem behoben, dass das Docker Volume Plugin in einigen Umgebungen nicht starten ("[Ausgabe #760](#)" konnte ("[Ausgabe #548](#)" und ).
- Festes SLM-Problem speziell für ONTAP-SAN-Back-Ends, das sicherstellt, dass nur eine Teilmenge von Daten-LIFs, die zu den Berichterstellungs-Nodes gehören, veröffentlicht wird.
- Es wurde ein Performance-Problem behoben, bei dem unnötige Scans für iSCSI-LUNs beim Anschließen eines Volumes aufgetreten sind.
- Granulare Wiederholungen im Trident iSCSI Workflow wurden entfernt, um ein schnelles Fehlschlagen zu ermöglichen und externe Wiederholungsintervalle zu verringern.



- Das Problem wurde behoben, bei dem beim Spülen eines iSCSI-Geräts ein Fehler zurückgegeben wurde, als das entsprechende Multipath-Gerät bereits gespült wurde.

## Vorgestellt Werden

- Kubernetes:
  - Zusätzliche Unterstützung für Kubernetes 1.25 Vor dem Upgrade auf Kubernetes 1.25 müssen Sie Trident auf 22.10 aktualisieren.
  - Hinzufügung eines separaten ServiceAccount, ClusterRole und ClusterBinding für die Trident Deployment und DemonSet, um zukünftige Berechtigungsverbesserungen zu ermöglichen.
  - Unterstützung für hinzugefügt "[Namespace-übergreifende Volume-Freigabe](#)".
- Alle Trident `ontap-*` Storage-Treiber funktionieren jetzt mit der ONTAP REST API.
- Neuer Operator yaml (`bundle_post_1_25.yaml`) ohne `APodSecurityPolicy` zur Unterstützung von Kubernetes 1.25 hinzugefügt.
- Für `ontap-san` und `ontap-san-economy` Speichertreiber hinzugefügt "[Unterstützung für LUKS-verschlüsselte Volumes](#)".
- Unterstützung für Windows Server 2019-Knoten hinzugefügt.
- Über den `azure-netapp-files` Speichertreiber hinzugefügt "[Unterstützung für SMB Volumes auf Windows Nodes](#)".
- Die automatische MetroCluster-Umschalterkennung für ONTAP-Treiber ist jetzt allgemein verfügbar.

## Abschreibungen

- **Kubernetes:** Aktualisiert unterstützt mindestens Kubernetes auf 1.20.
- Astra Data Store (ADS)-Treiber entfernt.
- Unterstützung für und `smart` Optionen für `find_multipaths` das Konfigurieren von Multipathing für Arbeitsknoten für iSCSI wurde entfernt `yes`.

## Änderungen in 22.07

### Korrekturen

#### Kubernetes

- Problem wurde behoben, um boolesche Werte und Zahlenwerte für die Node-Auswahl bei der Konfiguration von Trident mit Helm oder dem Trident Operator zu behandeln. ("[GitHub Ausgabe #700](#)")
- Problem beim Umgang mit Fehlern aus dem nicht-CHAP-Pfad behoben, sodass kubelet erneut versuchen wird, wenn er fehlschlägt. ("[GitHub Ausgabe #736](#)")

## Vorgestellt Werden

- Übergang von `k8s.gcr.io` zu `Registry.k8s.io` als Standard-Registry für CSI-Bilder
- ONTAP-SAN Volumes werden jetzt Initiatorgruppen pro Node verwenden und LUNs nur Initiatorgruppen zuordnen, während diese Nodes aktiv veröffentlicht werden, um unsere Sicherheit zu verbessern. Vorhandene Volumes werden opportun auf das neue `igroup`-Schema umgeschaltet, wenn Trident feststellt, dass der Einsatz sicher ist, ohne dass aktive Workloads beeinträchtigt werden.
- Enthält eine ResourceQuota mit Trident-Installationen, um sicherzustellen, dass Trident DemonSet geplant

ist, wenn die PriorityClass-Nutzung standardmäßig beschränkt ist.

- Unterstützung für Netzwerkfunktionen für den Azure NetApp Files-Treiber hinzugefügt. ("[GitHub Ausgabe #717](#)")
- Technische Vorschau Automatische MetroCluster-Umschalterkennung zu ONTAP-Treibern hinzugefügt. ("[GitHub Ausgabe #228](#)")

## Abschreibungen

- **Kubernetes:** Aktualisiert unterstützt mindestens Kubernetes auf 1.19.
- Back-End-Konfiguration ermöglicht nicht mehr mehrere Authentifizierungstypen in einer einzigen Konfiguration.

## Umzüge

- Der AWS CVS-Treiber (veraltet seit 22.04) wurde entfernt.
- Kubernetes
  - Keine unnötige SYS\_ADMIN-Funktion von Node-Pods entfernt.
  - Verringert die Nodevorbereitung auf einfache Host-Info und aktive Serviceerkennung, um eine Bestätigung für den bestmöglichen Aufwand zu machen, dass NFS/iSCSI-Dienste auf Worker-Knoten verfügbar sind.

## Dokumentation

Ein neuer "[Pod-Sicherheitsstandards](#)" Abschnitt (PSS) wurde hinzugefügt, in dem die von Trident bei der Installation aktivierten Berechtigungen detailliert aufgeführt sind.

## Änderungen in 22.04

NetApp verbessert seine Produkte und Services kontinuierlich. Im Folgenden finden Sie einige der neuesten Funktionen von Trident. Frühere Versionen finden Sie unter "[Frühere Versionen der Dokumentation](#)".



Wenn Sie ein Upgrade von einem früheren Trident-Release durchführen und Azure NetApp Files verwenden, ist der `locationParameter` config jetzt ein obligatorisches Single-Feld.

## Korrekturen

- Verbessertes Analysieren von iSCSI-Initiatornamen. ("[GitHub Ausgabe #681](#)")
- Das Problem wurde behoben, bei dem CSI-Speicherlassenparameter nicht zulässig waren. ("[GitHub Ausgabe #598](#)")
- Doppelte Schlüsseldeklaration im Trident CRD behoben. ("[GitHub Ausgabe #671](#)")
- Fehlerhafte CSI-Snapshot-Protokolle wurden korrigiert. ("[GitHub Ausgabe #629](#)")
- Problem beim Aufheben der Veröffentlichung von Volumes auf gelöschten Nodes behoben. ("[GitHub Ausgabe #691](#)")
- Zusätzliche Bearbeitung von Inkonsistenzen im Dateisystem auf Blockgeräten. ("[GitHub Ausgabe #656](#)")
- Problem beim Abrufen von Auto-Support-Bildern beim Festlegen des Flags während der Installation behoben `imageRegistry`. ("[GitHub Ausgabe #715](#)")
- Es wurde ein Problem behoben, bei dem der Azure NetApp Files-Treiber ein Volume mit mehreren

Exportregeln nicht klonen konnte.

## Vorgestellt Werden

- Eingehende Verbindungen zu den sicheren Endpunkten von Trident erfordern jetzt mindestens TLS 1.3. (["GitHub Ausgabe #698"](#))
- Trident fügt jetzt HSTS-Header zu den Antworten von seinen sicheren Endpunkten hinzu.
- Trident versucht nun, die Azure NetApp Files unix Berechtigungsfunktion automatisch zu aktivieren.
- **Kubernetes:** Trident Demonset wird jetzt in der Klasse mit System-Node-kritischer Priorität ausgeführt. (["GitHub Ausgabe #694"](#))

## Umzüge

E-Series-Treiber (deaktiviert seit 20.07) wurde entfernt.

## Änderungen in 22.01.1

### Korrekturen

- Problem beim Aufheben der Veröffentlichung von Volumes auf gelöschten Nodes behoben. (["GitHub Ausgabe #691"](#))
- Fester Panik beim Zugriff auf Nil-Felder für den aggregierten Speicherplatz in den ONTAP API Antworten.

## Änderungen in 22.01.0

### Korrekturen

- **Kubernetes:** Erhöhung der Neuzulassung der Knotenregistrierung für große Cluster.
- Das Problem wurde behoben, bei dem der Azure-netapp-Files Treiber von mehreren Ressourcen mit demselben Namen verwirrt werden konnte.
- ONTAP SAN IPv6 Daten-LIFs funktionieren jetzt, wenn sie mit Klammern angegeben sind.
- Das Problem wurde behoben, bei dem der Import eines bereits importierten Volumes das EOF zurückgibt, sodass PVC in den ausstehenden Zustand zurückbleibt. (["GitHub Ausgabe #489"](#))
- Das Problem wurde behoben, wenn die Trident Performance langsamer wird, wenn mehr als 32 Snapshots auf einem SolidFire Volume erstellt werden.
- SHA-1 wurde durch SHA-256 bei der Erstellung eines SSL-Zertifikats ersetzt.
- Azure NetApp Files-Treiber wurde behoben, um doppelte Ressourcennamen zu erlauben und Vorgänge auf einen einzelnen Speicherort zu beschränken.
- Azure NetApp Files-Treiber wurde behoben, um doppelte Ressourcennamen zu erlauben und Vorgänge auf einen einzelnen Speicherort zu beschränken.

## Vorgestellt Werden

- Verbesserungen von Kubernetes:
  - Zusätzliche Unterstützung für Kubernetes 1.23
  - Fügen Sie bei der Installation über Trident Operator oder Helm Planungsoptionen für Trident Pods hinzu. (["GitHub Ausgabe #651"](#))

- Erlauben Sie regionenübergreifende Volumes im GCP-Treiber. ("[GitHub Ausgabe #633](#)")
- Unterstützung für die Option „unixPermissions“ für Azure NetApp Files Volumes wurde hinzugefügt. ("[GitHub Ausgabe #666](#)")

## Abschreibungen

Die Trident REST-Schnittstelle kann nur unter 127.0.0.1 oder [: 1] Adressen zuhören und bedient werden

## Änderungen in 21.10.1



In der Version v21.10.0 kann der Trident Controller in den CrashLoopBackOff-Status versetzt werden, wenn ein Node entfernt und dann wieder zum Kubernetes Cluster hinzugefügt wird. Dieses Problem wurde in der Version 21,10,1 behoben (GitHub Ausgabe 669).

## Korrekturen

- Beim Import eines Volumes auf ein GCP CVS Backend wurde eine potenzielle Race-Bedingung behoben, die zu einem Import führt.
- Es wurde ein Problem behoben, durch das der Trident Controller in den CrashLoopBackOff-Status versetzt werden kann, wenn ein Node entfernt und dann wieder zum Kubernetes Cluster hinzugefügt wird (GitHub Ausgabe 669).
- Das Problem wurde behoben, bei dem SVMs nicht mehr erkannt wurden, wenn kein SVM-Name angegeben wurde (GitHub Problem 612).

## Änderungen in 21.10.0

### Korrekturen

- Es wurde ein Problem behoben, bei dem Klone von XFS-Volumes nicht auf demselben Node wie das Quell-Volume gemountet werden konnten (GitHub Ausgabe 514).
- Problem behoben, bei dem Trident beim Herunterfahren einen schwerwiegenden Fehler protokolliert hat (GitHub Problem 597).
- Kubernetes-bezogene Fixes:
  - Geben Sie den belegten Speicherplatz eines Volumes als Mindestgröße für die Wiederherstellung zurück, wenn Sie Snapshots mit und `ontap-nas-flexgroup` Treibern erstellen `ontap-nas` (GitHub Ausgabe 645).
  - Problem behoben, bei dem `Failed to expand filesystem` der Fehler nach der Volume-Größe protokolliert wurde (GitHub Problem 560).
  - Problem behoben, bei dem ein Pod in den Status stecken bleiben konnte `Terminating` (GitHub Ausgabe 572).
  - Es wurde der Fall behoben, dass ein `ontap-san-economy` FlexVol voller Snapshot-LUNs sein konnte (GitHub Ausgabe 533).
  - Problem mit dem benutzerdefinierten YAML-Installationsprogramm mit einem anderen Bild wurde behoben (GitHub Ausgabe 613).
  - Berechnung der Snapshot-Größe wurde korrigiert (GitHub Ausgabe 611).
  - Es wurde ein Problem behoben, bei dem alle Trident Installer einfaches Kubernetes als OpenShift identifizieren konnten (GitHub Ausgabe 639).

- Der Trident-Operator hat den Abgleich behoben, wenn der Kubernetes-API-Server nicht erreichbar ist (GitHub Ausgabe 599).

## Vorgestellt Werden

- Unterstützung für GCP-CVS Performance Volumes wurde zugefügt `unixPermissions`.
- Zusätzliche Unterstützung für für für Skalierung optimierte CVS Volumes in GCP im Bereich von 600 gib bis 1 tib.
- Verbesserungen im Zusammenhang mit Kubernetes:
  - Zusätzliche Unterstützung für Kubernetes 1.22
  - Trident Operator und Helm Chart wurde für die Verwendung mit Kubernetes 1.22 aktiviert (GitHub Ausgabe 628).
  - Befehl 'Operator image' zu images hinzugefügt `tridentctl` (GitHub Ausgabe 570).

## Experimentelle Verbesserungen

- Unterstützung für Volume-Replikation im Treiber hinzugefügt `ontap-san`.
- **Tech Preview** REST-Unterstützung für die, `ontap-san` und `ontap-nas-economy` Treiber hinzugefügt `ontap-nas-flexgroup`.

## Bekannte Probleme

Bekannte Probleme erkennen Probleme, die eine erfolgreiche Verwendung des Produkts verhindern könnten.

- Wenn Sie ein Kubernetes-Cluster von 1.24 auf 1.25 oder höher aktualisieren, auf dem Trident installiert ist, müssen Sie `values.yaml` aktualisieren, um den `helm upgrade` Befehl auf `true` festzulegen `excludePodSecurityPolicy` oder hinzuzufügen `--set excludePodSecurityPolicy=true`, bevor Sie das Cluster aktualisieren können.
- Trident erzwingt jetzt ein Leerzeichen `fsType` (`fsType=""`) für Volumes, die nicht die in ihrer StorageClass angegebene haben `fsType`. Bei der Arbeit mit Kubernetes 1.17 oder höher unterstützt Trident die Bereitstellung eines Leereinschübe `fsType` für NFS-Volumes. Für iSCSI-Volumes müssen Sie die auf Ihrer StorageClass festlegen, wenn Sie `fsType` einen mit einem Sicherheitskontext erzwingen `fsGroup`.
- Wenn Sie ein Back-End über mehrere Trident Instanzen hinweg verwenden, sollte jede Back-End-Konfigurationsdatei einen anderen Wert für ONTAP Back-Ends haben `storagePrefix` oder einen anderen für SolidFire Back-Ends verwenden `TenantName`. Trident kann Volumes nicht erkennen, die von anderen Instanzen von Trident erstellt wurden. Der Versuch, ein vorhandenes Volume auf ONTAP oder SolidFire Back-Ends zu erstellen, ist erfolgreich, da Trident die Volume-Erstellung als einen idempotenten Vorgang behandelt. Wenn `storagePrefix` sich die Volumes unterscheiden oder `TenantName` nicht, kann es zu Namenskollisionen für Volumes kommen, die auf demselben Backend erstellt wurden.
- Bei der Installation von Trident (mit `tridentctl` oder dem Trident Operator) und der Verwendung von `tridentctl` zum Verwalten von Trident sollten Sie sicherstellen, dass die `KUBECONFIG` Umgebungsvariable eingestellt ist. Dies ist notwendig, um den Kubernetes-Cluster anzugeben, der `tridentctl` gegen den eingesetzt werden soll. Wenn Sie mit mehreren Kubernetes-Umgebungen arbeiten, sollten Sie sicherstellen, dass die `KUBECONFIG` Datei korrekt bezogen wird.
- Um Online-Speicherplatzrückgewinnung für iSCSI PVS durchzuführen, muss das zugrunde liegende Betriebssystem auf dem Worker-Node möglicherweise Mount-Optionen an das Volume übergeben werden. Dies gilt für RHEL/RedHat CoreOS Instanzen, die das; verlangen `discard` "[Mount-Option](#)", dass die

discard mountOption in Ihrem^] enthalten ist[StorageClass, um Online-Blockverwerfen zu unterstützen.

- Wenn Sie mehr als eine Instanz von Trident pro Kubernetes-Cluster haben, kann Trident nicht mit anderen Instanzen kommunizieren und keine anderen Volumes erkennen, die sie erstellt haben. Dies führt zu unerwartetem und falschem Verhalten, wenn mehr als eine Instanz in einem Cluster ausgeführt wird. Pro Kubernetes-Cluster sollte es nur eine Instanz von Trident geben.
- Wenn Trident-basierte StorageClass Objekte aus Kubernetes gelöscht werden, während Trident offline ist, entfernt Trident die entsprechenden Storage-Klassen nicht aus seiner Datenbank, wenn sie wieder online geschaltet werden. Sie sollten diese Speicherklassen mit oder der REST-API löschen `tridentctl`.
- Wenn ein Benutzer ein von Trident bereitgestelltes PV löscht, bevor die entsprechende PVC gelöscht wird, löscht Trident nicht automatisch das Back-Volume. Sie sollten das Volume über die REST-API entfernen `tridentctl`.
- ONTAP kann nicht gleichzeitig mehr als ein FlexGroup gleichzeitig bereitstellen, es sei denn, der Satz der Aggregate ist auf jede Bereitstellungsanforderung beschränkt.
- Wenn Sie Trident über IPv6 verwenden, sollten Sie und `dataLIF` in der Backend-Definition in eckigen Klammern angeben `managementLIF`. [`fd20:8b1e:b258:2000:f816:3eff:feec:0`] Beispiel: .



Sie können die Angabe auf einem ONTAP-SAN-Backend nicht `dataLIF` machen. Trident erkennt alle verfügbaren iSCSI LIFs und verwendet diese zur Einrichtung der Multipath-Sitzung.

- Wenn Sie den `solidfire-san` Treiber mit OpenShift 4.5 verwenden, stellen Sie sicher, dass die zugrunde liegenden Arbeitsknoten MD5 als CHAP-Authentifizierungsalgorithmus verwenden. Sichere, FIPS-konforme CHAP-Algorithmen SHA1, SHA-256 und SHA3-256 sind mit Element 12.7 erhältlich.

## Weitere Informationen

- ["Trident GitHub"](#)
- ["Trident Blogs"](#)

## Frühere Versionen der Dokumentation

Wenn Sie Trident 24.10 nicht verwenden, ist die Dokumentation für frühere Versionen auf Basis der verfügbar ["Lebenszyklus des Trident Supports"](#).

- ["Trident 24.06"](#)
- ["Trident 24.02"](#)
- ["Trident 23.10"](#)
- ["Trident 23.07"](#)
- ["Trident 23.04"](#)
- ["Trident 23.01"](#)
- ["Trident 22.10"](#)
- ["Trident 22.07"](#)
- ["Trident 22.04"](#)

# Los geht's

## Erfahren Sie mehr über Trident

### Erfahren Sie mehr über Trident

Trident ist ein vollständig von NetApp unterstütztes Open-Source-Projekt. Es wurde entwickelt, damit Sie die Persistenz-Anforderungen Ihrer Container-Applikation mithilfe von Standardschnittstellen, wie dem Container Storage Interface (CSI), erfüllen können.

#### Was ist Trident?

NetApp Trident ermöglicht die Nutzung und das Management von Storage-Ressourcen über alle gängigen NetApp Storage-Plattformen hinweg, in der Public Cloud oder vor Ort, einschließlich ONTAP (AFF, FAS, Select, Cloud, Amazon FSX for NetApp ONTAP), Element Software (NetApp HCI, SolidFire), Azure NetApp Files Service und Cloud Volumes Service on Google Cloud.

Trident ist ein CSI-konformer dynamischer Storage Orchestrator, der sich nativ in ["Kubernetes"](#)NetApp integrieren lässt. Trident wird als einzelner Controller-Pod plus einen Node Pod auf jedem Worker-Node im Cluster ausgeführt. Weitere Informationen finden Sie unter ["Architektur von Trident"](#) .

Trident ist zudem direkt in das Docker Ecosystem für NetApp Storage-Plattformen integriert. Das NetApp Docker Volume Plug-in (nDVP) unterstützt die Bereitstellung und das Management von Storage-Ressourcen von der Storage-Plattform an Docker Hosts. Weitere Informationen finden Sie unter ["Implementierung von Trident für Docker"](#) .



Wenn Sie Kubernetes zum ersten Mal verwenden, sollten Sie sich mit der vertraut machen ["Kubernetes-Konzepte und -Tools"](#).

### Kubernetes-Integration in NetApp Produkte

Das NetApp Portfolio an Storage-Produkten kann in viele Aspekte eines Kubernetes Clusters integriert werden und bietet erweiterte Datenmanagement-Funktionen, mit denen die Funktionalität, Funktionalität, Performance und Verfügbarkeit der Kubernetes-Implementierung verbessert werden.

#### Amazon FSX für NetApp ONTAP

["Amazon FSX für NetApp ONTAP"](#) Ist ein vollständig gemanagter AWS Service, mit dem Sie Dateisysteme mit dem NetApp ONTAP Storage-Betriebssystem starten und ausführen können.

#### Azure NetApp Dateien

["Azure NetApp Dateien"](#) Ist ein Azure-Dateifreigabeservice der Enterprise-Klasse auf der Basis von NetApp. Sie können anspruchsvollste dateibasierte Workloads nativ in Azure ausführen. So erhalten Sie die Performance und das umfassende Datenmanagement, die Sie von NetApp gewohnt sind.

## Cloud Volumes ONTAP

"[Cloud Volumes ONTAP](#)" Ist eine rein softwarebasierte Storage-Appliance, mit der die ONTAP Datenmanagement-Software in der Cloud ausgeführt wird.

## Google Cloud NetApp Volumes

"[Google Cloud NetApp Volumes](#)" Ist ein vollständig gemanagter File-Storage-Service in Google Cloud mit hochperformantem File-Storage der Enterprise-Klasse.

## Element Software

"[Element](#)" Storage-Administratoren können Workloads konsolidieren, indem sie Performance garantieren und den Storage-Bedarf vereinfachen und optimieren.

## NetApp HCI

"[NetApp HCI](#)" Vereinfacht das Management und die Skalierung des Datacenters durch die Automatisierung von Routineaufgaben und ermöglicht es Infrastruktur-Administratoren, sich auf wichtigere Funktionen zu konzentrieren.

Trident kann Storage-Geräte für Container-Applikationen direkt auf der zugrunde liegenden NetApp HCI Storage-Plattform bereitstellen und managen.

## NetApp ONTAP

"[NetApp ONTAP](#)" Ist das Unified Storage-Betriebssystem NetApp für mehrere Protokolle und bietet für jede Applikation erweiterte Datenmanagementfunktionen.

ONTAP Systeme verfügen über rein Flash-basierte, hybride oder rein HDD-basierte Konfigurationen und bieten eine Vielzahl unterschiedlicher Implementierungsmodelle, darunter speziell entwickelte Hardware (FAS und AFF), White-Box (ONTAP Select) und rein Cloud-basierte Cloud Volumes ONTAP Systeme. Trident unterstützt diese ONTAP Implementierungsmodelle.

## Architektur von Trident

Trident wird als einzelner Controller-Pod plus einen Node Pod auf jedem Worker-Node im Cluster ausgeführt. Der Node Pod muss auf einem beliebigen Host ausgeführt werden, auf dem Sie potenziell ein Trident Volume mounten möchten.

### Allgemeines zu Controller-Pods und Node-Pods

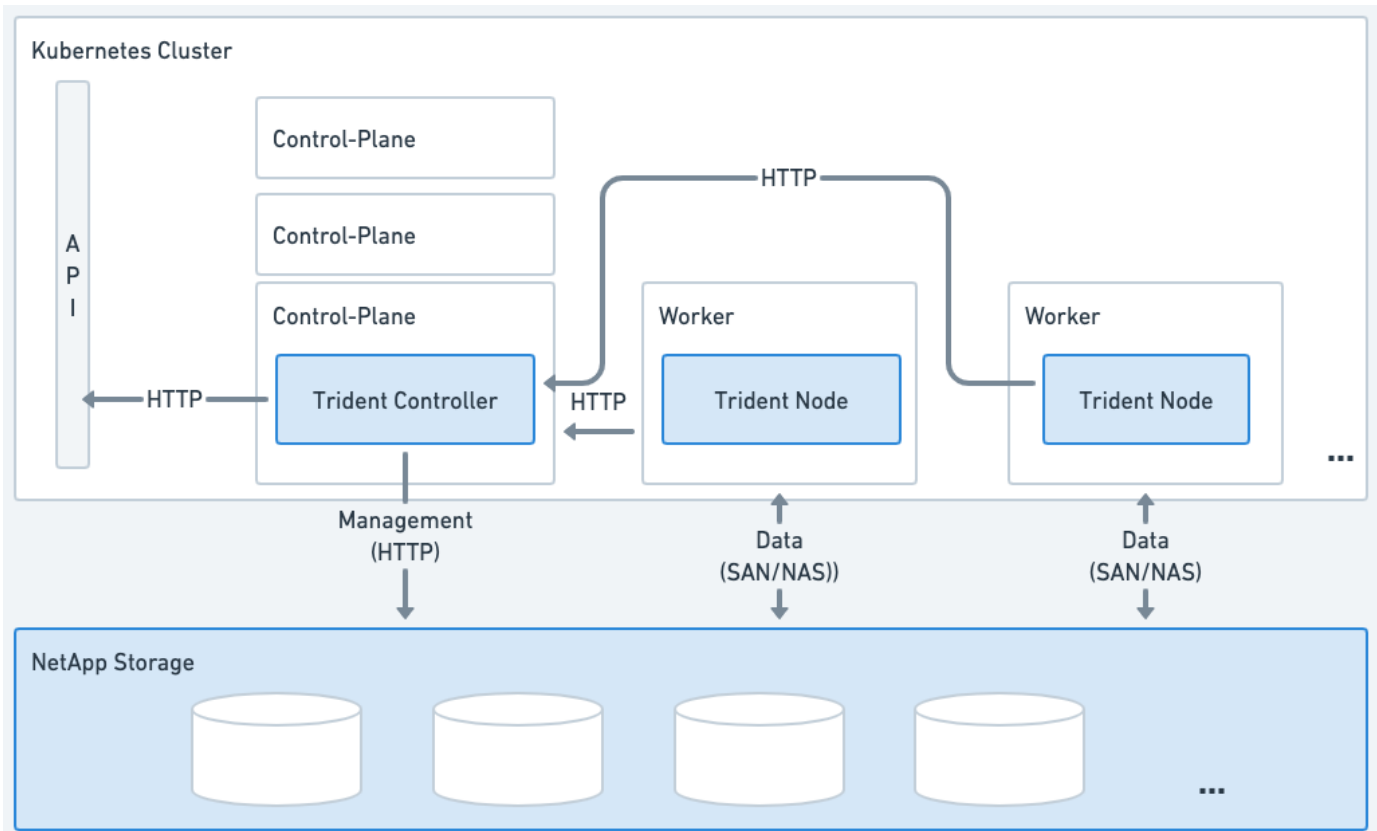
Trident wird als einzelner oder mehrerer [Trident Node Pods](#) Cluster im Kubernetes-Cluster implementiert [Trident Controller Pod](#) und verwendet standardmäßige Kubernetes *CSI Sidecar Container*, um die Implementierung von CSI-Plug-ins zu vereinfachen. "[Kubernetes CSI Sidecar-Container](#)" Werden von der Kubernetes Storage Community unterhalten.

Kubernetes "[Knotenauswahl](#)" und "[Toleranzen und Verfleckungen](#)" schränken die Ausführung eines Pods auf



einem bestimmten oder bevorzugten Node ein. Während der Trident-Installation können Sie Node-Selektoren und Toleranzen für Controller- und Node-Pods konfigurieren.

- Das Controller-Plug-in übernimmt Volume-Bereitstellung und -Management, beispielsweise Snapshots und Größenanpassungen.
- Das Node-Plug-in verarbeitet das Verbinden des Speichers mit dem Node.



**Abbildung 1. Auf dem Kubernetes-Cluster implementierte Trident**

### Trident Controller Pod

Beim Trident Controller Pod handelt es sich um einen einzelnen Pod, auf dem das CSI Controller Plug-in ausgeführt wird.

- Verantwortlich für die Bereitstellung und das Management von Volumes in NetApp Storage
- Management durch eine Kubernetes-Implementierung
- Kann je nach Installationsparameter auf der Steuerebene oder auf den Arbeitsknoten ausgeführt werden.

### Abbildung 2. Trident Controller Pod-Diagramm

#### Trident Node Pods

Trident Node Pods sind privilegierte Pods, auf denen das CSI Node Plug-in ausgeführt wird.

- Verantwortlich für das Mounten und Entmounten von Speicher für Pods, die auf dem Host ausgeführt werden
- Gemanagt von einem Kubernetes DemonSet
- Muss auf jedem Node ausgeführt werden, auf dem NetApp Storage gemountet werden soll

### Abbildung 3. Trident Node Pod-Diagramm

## Unterstützte Kubernetes-Cluster-Architekturen

Trident wird von den folgenden Kubernetes-Architekturen unterstützt:

Kubernetes-Cluster-Architekturen	Unterstützt	Standardinstallation
Ein Master Computing	Ja.	Ja.
Mehrere Master-Computer und Computing-Ressourcen	Ja.	Ja.
Master, etcd, Berechnung	Ja.	Ja.
Master, Infrastruktur, Computing	Ja.	Ja.

## Konzepte

### Bereitstellung

Die Bereitstellung in Trident hat zwei primäre Phasen. In der ersten Phase wird eine Speicherklasse mit einem Satz geeigneter Back-End-Speicherpools verknüpft. Diese werden vor der Bereitstellung als notwendig vorbereitet. Die zweite Phase umfasst die Volume-Erstellung selbst und erfordert die Auswahl eines Speicherpools aus denen, die mit der Storage-Klasse des ausstehenden Volumes verknüpft sind.

### Storage-Klassen-Zuordnung

Die Zuordnung von Back-End-Speicherpools zu einer Storage-Klasse basiert sowohl auf den angeforderten Attributen der Storage-Klasse als auch auf den entsprechenden `storagePools`, `additionalStoragePools` und `excludeStoragePools`-Listen. Wenn Sie eine Storage-Klasse erstellen, vergleicht Trident die von jedem seiner Back-Ends angebotenen Attribute und Pools mit den von der Storage-Klasse angeforderten Attributen. Stimmen Attribute und Name eines Storage-Pools mit allen angeforderten Attributen und Poolnamen überein, fügt Trident diesen Storage-Pool den entsprechenden Storage-Pools für diese Storage-Klasse hinzu. Darüber hinaus fügt Trident allen in der Liste aufgeführten Storage-Pools zu diesem Set hinzu `additionalStoragePools`, selbst wenn ihre Attribute nicht alle angeforderten Attribute der Storage-Klasse erfüllen. Sie sollten die Liste verwenden `excludeStoragePools`, um Speicherpools für eine Speicherklasse zu überschreiben und aus der Verwendung zu entfernen. Trident führt jedes Mal, wenn Sie ein neues Back-End hinzufügen, einen ähnlichen Prozess durch. Dabei wird überprüft, ob seine Storage-Pools die vorhandenen Storage-Klassen erfüllen, und alle als ausgeschlossen markierten werden entfernt.

### Volume-Erstellung

Trident verwendet dann die Zuordnungen zwischen Storage-Klassen und Storage-Pools, um zu bestimmen, wo Volumes bereitgestellt werden sollen. Bei der Erstellung eines Volumes erhält Trident zunächst die Gruppe von Storage-Pools für die Storage-Klasse des Volumes. Wenn Sie ein Protokoll für das Volume angeben, entfernt Trident die Storage-Pools, die das angeforderte Protokoll nicht bereitstellen können (ein NetApp HCI/SolidFire-Backend kann z. B. kein dateibasiertes Volume bereitstellen, während ein ONTAP-NAS-Backend kein blockbasiertes Volume bereitstellen kann). Trident randomisiert die Reihenfolge dieses resultierenden Satzes, um eine gleichmäßige Verteilung der Volumes zu ermöglichen. Anschließend iteriert es durch und versucht, das Volume nacheinander auf den einzelnen Storage-Pools bereitzustellen. Wenn sie erfolgreich ist,

wird sie erfolgreich zurückgegeben, und es werden alle Fehler protokolliert, die im Prozess aufgetreten sind. Trident gibt einen Fehler zurück **nur wenn** es nicht auf **allen** Speicherpools zur Verfügung stellt, die für die angeforderte Speicherklasse und das Protokoll verfügbar sind.

## Volume Snapshots

Erfahren Sie mehr darüber, wie Trident mit der Erstellung von Volume-Snapshots für seine Treiber umgeht.

### Erfahren Sie mehr über die Erstellung von Volume Snapshots

- Für die `ontap-nas`, `ontap-san gcp-cvs` und `azure-netapp-files` Treiber wird jedes Persistent Volume (PV) einem FlexVol zugeordnet. Volume Snapshots werden im Ergebnis als NetApp Snapshots erstellt. NetApp Snapshots liefern weitaus mehr Stabilität, Skalierbarkeit, Wiederherstellbarkeit und Performance als vergleichbare Systeme. Diese Snapshot-Kopien sind äußerst schnell und platzsparend, da sie erstellt und gespeichert werden müssen.
- Für den `ontap-nas-flexgroup` Treiber ist jedes Persistent Volume (PV) einer FlexGroup zugeordnet. Im Ergebnis werden Volume Snapshots als NetApp FlexGroup Snapshots erstellt. NetApp Snapshots liefern weitaus mehr Stabilität, Skalierbarkeit, Wiederherstellbarkeit und Performance als vergleichbare Systeme. Diese Snapshot-Kopien sind äußerst schnell und platzsparend, da sie erstellt und gespeichert werden müssen.
- Für den `ontap-san-economy` Treiber weisen PVS LUNs zu, die auf freigegebenen FlexVols erstellt wurden. VolumeSnapshots von PVS werden durch FlexClones der zugehörigen LUN erreicht. Mit der ONTAP FlexClone Technologie ist es nahezu sofort möglich, Kopien selbst von größten Datensätzen zu erstellen. Kopien nutzen Datenblöcke gemeinsam mit ihren Eltern und verbrauchen somit keinen Storage, außer was für Metadaten erforderlich ist.
- Für den `solidfire-san` Treiber ordnet jedes PV einer LUN zu, die auf dem NetApp Element-Software/NetApp HCI-Cluster erstellt wurde. VolumeSnapshots werden durch Element Snapshots der zugrunde liegenden LUN dargestellt. Diese Snapshots sind zeitpunktgenaue Kopien, die nur eine kleine Menge an Systemressourcen und Platz beanspruchen.
- Bei der Arbeit mit den `ontap-nas` Treibern sind ONTAP Snapshots zeitpunktgenaue Kopien der FlexVol und `ontap-san` belegen Speicherplatz auf der FlexVol selbst. Das kann dazu führen, dass der beschreibbare Speicherplatz auf dem Volume mit der Zeit verkürzt wird, wenn Snapshots erstellt/geplant werden. Eine einfache Möglichkeit dieser Bewältigung ist, das Volumen durch die Anpassung über Kubernetes zu vergrößern. Eine weitere Option ist das Löschen von nicht mehr benötigten Snapshots. Wenn ein über Kubernetes erstellter VolumeSnapshot gelöscht wird, löscht Trident den zugehörigen ONTAP-Snapshot. ONTAP Snapshots, die nicht über Kubernetes erstellt wurden, können auch gelöscht werden.

Mit Trident können Sie VolumeSnapshots verwenden, um daraus neue PVs zu erstellen. Die Erstellung von PVS aus diesen Snapshots wird mithilfe der FlexClone Technologie für unterstützte ONTAP- und CVS-Back-Ends durchgeführt. Wenn ein PV aus einem Snapshot erstellt wird, ist das Back-Volume ein FlexClone des übergeordneten Volume des Snapshots. Der `solidfire-san` Treiber verwendet Volume-Klone der Element Software zur Erstellung von PVS aus Snapshots. Hier erstellt es aus dem Element Snapshot einen Klon.

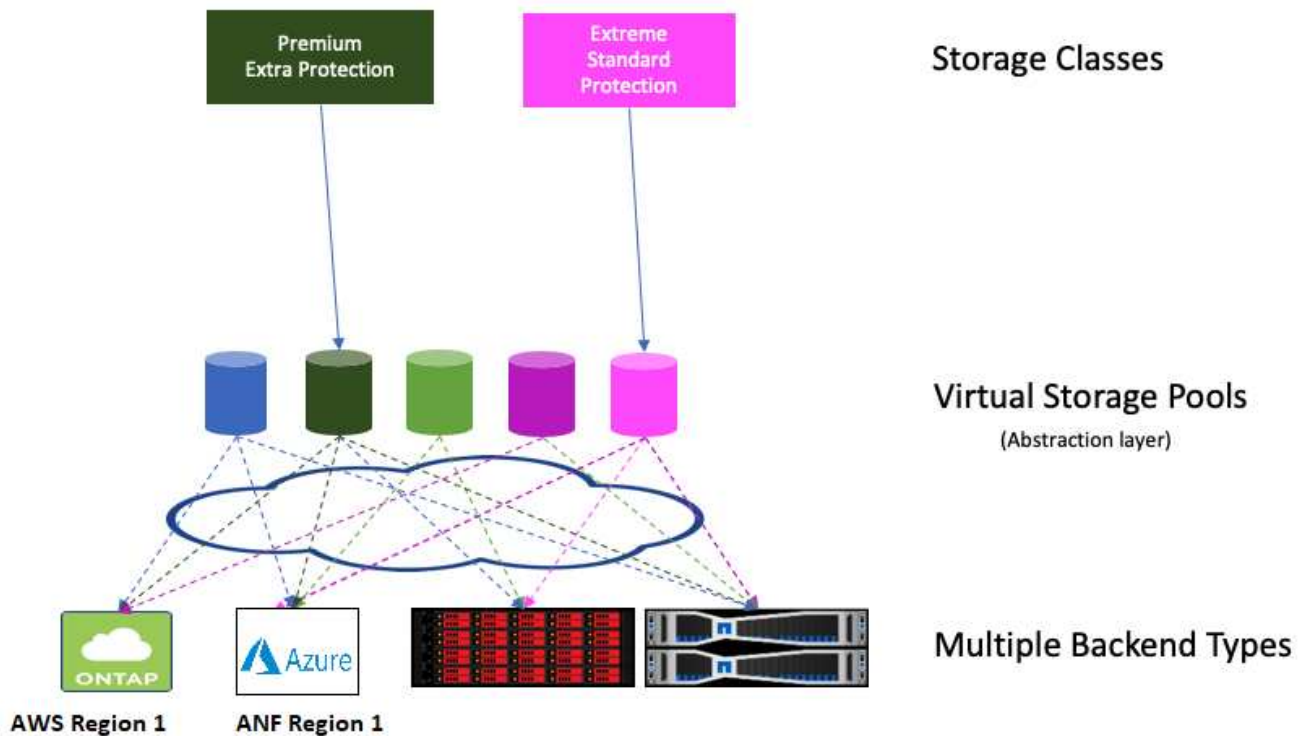
## Virtuelle Pools

Virtuelle Pools bieten eine Abstraktionsebene zwischen Trident Storage Back-Ends und Kubernetes `StorageClasses`. Sie ermöglichen es einem Administrator, Aspekte wie Standort, Performance und Schutz für jedes Back-End auf eine gemeinsame, Backend-unabhängige Weise zu definieren, ohne festzulegen, welches physische Backend

StorageClass, Backend-Pool oder Backend-Typ verwendet werden soll, um die gewünschten Kriterien zu erfüllen.

### Erfahren Sie mehr über virtuelle Pools

Der Storage-Administrator kann virtuelle Pools auf jedem beliebigen Trident Back-End in einer JSON- oder YAML-Definitionsdatei definieren.



Jeder außerhalb der Liste der virtuellen Pools angegebene Aspekt ist global für das Backend und gilt für alle virtuellen Pools, während jeder virtuelle Pool einen oder mehrere Aspekte einzeln angeben kann (alle Backend-globalen Aspekte außer Kraft setzen).



- Versuchen Sie beim Definieren virtueller Pools nicht, die Reihenfolge vorhandener virtueller Pools in einer Backend-Definition neu anzuordnen.
- Wir empfehlen, Attribute für einen vorhandenen virtuellen Pool zu ändern. Sie sollten einen neuen virtuellen Pool definieren, um Änderungen vorzunehmen.

Die meisten Aspekte werden Backend-spezifisch angegeben. Entscheidend ist, dass die Aspect-Werte nicht außerhalb des Backendtreibers angezeigt werden und nicht zum Matching in verfügbar sind StorageClasses. Stattdessen definiert der Administrator für jeden virtuellen Pool ein oder mehrere Labels. Jedes Etikett ist ein Schlüssel:Wert-Paar, und Etiketten können häufig über eindeutige Back-Ends hinweg verwendet werden. Wie Aspekte können auch Labels pro Pool oder global zum Backend angegeben werden. Im Gegensatz zu Aspekten, die vordefinierte Namen und Werte haben, hat der Administrator volle Entscheidungsbefugnis, Beschriftungsschlüssel und -Werte nach Bedarf zu definieren. Storage-Administratoren können Labels je virtuellen Pool definieren und Volumes nach Label gruppieren.

Ein StorageClass gibt an, welcher virtuelle Pool verwendet werden soll, indem die Bezeichnungen innerhalb

eines Auswahlparameters referenziert werden. Virtuelle Pool-Selektoren unterstützen folgende Operatoren:

Operator	Beispiel	Der Wert für die Bezeichnung eines Pools muss:
=	Performance=Premium	Übereinstimmung
!=	Performance!=extrem	Keine Übereinstimmung
in	Lage in (Osten, Westen)	Werden Sie im Satz von Werten
notin	Performance-Dose (Silber, Bronze)	Nicht im Wertungsset sein
<key>	Darstellt	Mit einem beliebigen Wert existieren
!<key>	!Schutz	Nicht vorhanden

## Volume-Zugriffsgruppen

Erfahren Sie mehr über die Verwendung von Trident "[Volume-Zugriffsgruppen](#)".



Ignorieren Sie diesen Abschnitt, wenn Sie CHAP verwenden. Dies wird empfohlen, um die Verwaltung zu vereinfachen und die unten beschriebene Skalierungsgrenze zu vermeiden. Wenn Sie Trident im CSI-Modus verwenden, können Sie diesen Abschnitt ignorieren. Trident verwendet CHAP, wenn es als erweiterte CSI-bereitstellung installiert ist.

### Erfahren Sie mehr über Volume Access Groups

Trident kann mithilfe von Volume-Zugriffsgruppen den Zugriff auf die von ihm bereitstehenden Volumes steuern. Wenn CHAP deaktiviert ist, erwartet es, dass eine Zugriffsgruppe mit dem Namen gefunden `trident` wird, es sei denn, Sie geben eine oder mehrere Zugriffsgruppen-IDs in der Konfiguration an.

Trident ordnet zwar neue Volumes den konfigurierten Zugriffsgruppen zu, erstellt oder verwaltet jedoch keine Zugriffsgruppen selbst. Die Zugriffsgruppen müssen vorhanden sein, bevor das Storage-Back-End zu Trident hinzugefügt wird. Sie müssen die iSCSI-IQNs von jedem Node im Kubernetes-Cluster enthalten, der die über dieses Back-End bereitgestellten Volumes mounten kann. In den meisten Installationen umfasst dies alle Worker Nodes im Cluster.

Bei Kubernetes-Clustern mit mehr als 64 Nodes sollten Sie mehrere Zugriffsgruppen verwenden. Jede Zugriffsgruppe kann bis zu 64 IQNs enthalten, und jedes Volume kann zu vier Zugriffsgruppen gehören. Bei maximal vier Zugriffsgruppen kann jeder Node in einem Cluster mit einer Größe von bis zu 256 Nodes auf beliebige Volumes zugreifen. Die neuesten Grenzwerte für Volume-Zugriffsgruppen finden Sie unter "[Hier](#)".

Wenn Sie die Konfiguration von einer Konfiguration ändern, die die Standardzugriffsgruppe verwendet, zu einer Konfiguration `trident`, die auch andere verwendet, fügen Sie die ID für die `trident` Zugriffsgruppe in die Liste ein.

## Schnellstart für Trident

Sie können Trident installieren und mit dem Management von Storage-Ressourcen in wenigen Schritten beginnen. Bevor Sie beginnen, überprüfen Sie "[Trident-Anforderungen erfüllt](#)".



Informationen zu Docker finden Sie unter "[Trident für Docker](#)".

**1**

### **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.

["Bereiten Sie den Knoten „Worker“ vor"](#)

**2**

### **Installieren Sie Trident**

Trident bietet verschiedene Installationsmethoden und -Modi, die für eine Vielzahl von Umgebungen und Organisationen optimiert sind.

["Installation Von Trident"](#)

**3**

### **Erstellen Sie ein Backend**

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.

["Konfigurieren Sie ein Backend"](#) Für Ihr Speichersystem

**4**

### **Kubernetes StorageClass erstellen**

Das Objekt Kubernetes StorageClass gibt Trident als bereitstellung an und ermöglicht die Erstellung einer Storage-Klasse, um Volumes mit anpassbaren Attributen bereitzustellen. Trident erstellt eine passende Storage-Klasse für Kubernetes-Objekte, die die Trident-bereitstellung angeben.

["Erstellen Sie eine Speicherklasse"](#)

**5**

### **Bereitstellen eines Volumes**

Ein *PersistentVolume* (PV) ist eine physische Speicherressource, die vom Cluster-Administrator auf einem Kubernetes-Cluster bereitgestellt wird. Das *PersistentVolumeClaim* (PVC) ist eine Anforderung für den Zugriff auf das PersistentVolume auf dem Cluster.

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

["Bereitstellen eines Volumes"](#)

## **Was kommt als Nächstes?**

Sie können nun zusätzliche Back-Ends hinzufügen, Storage-Klassen managen, Back-Ends managen und Volume-Operationen durchführen.

## **Anforderungen**

Vor der Installation von Trident sollten Sie die folgenden allgemeinen

Systemanforderungen überprüfen. Spezifische Back-Ends können zusätzliche Anforderungen haben.

## Wichtige Informationen über Trident

Sie müssen die folgenden wichtigen Informationen über Trident lesen.

### **-Informationen über Trident </strong>**

- Kubernetes 1.32 wird jetzt in Trident unterstützt. Upgrade von Trident vor dem Upgrade von Kubernetes.
- Trident setzt die Verwendung der Multipathing-Konfiguration in SAN-Umgebungen strikt durch, wobei der empfohlene Wert `find_multipaths: no` in der `Multipath.conf` Datei verwendet wird.

Die Verwendung einer nicht-Multipathing-Konfiguration oder die Verwendung von `find_multipaths: yes` oder `find_multipaths: smart` Wert in der Datei `Multipath.conf` führt zu Mount-Fehlern. Trident empfiehlt die Verwendung von `find_multipaths: no` seit Version 21.07.

## Unterstützte Frontends (Orchestrators)

Trident unterstützt diverse Container-Engines und -Orchestrierungslösungen, darunter:

- Anthos On-Premises (VMware) und Anthos auf Bare Metal 1.16
- Kubernetes 1.25–1.32
- OpenShift 4.10 - 4.17
- Rancher Kubernetes Engine 2 (RKE2) v1.28.5+rke2r1

Der Trident-Operator wird durch folgende Versionen unterstützt:

- Anthos On-Premises (VMware) und Anthos auf Bare Metal 1.16
- Kubernetes 1.25–1.32
- OpenShift 4.10 - 4.17
- Rancher Kubernetes Engine 2 (RKE2) v1.28.5+rke2r1

Trident kann auch mit einer Vielzahl anderer, vollständig gemanagter und selbst gemanagter Kubernetes-Angebote eingesetzt werden, wie z. B. Google Kubernetes Engine (GKE), Amazon Elastic Kubernetes Services (EKS), Azure Kubernetes Service (AKS), Mirantis Kubernetes Engine (MKE) und VMware Tanzu Portfolio.

Trident und ONTAP können als Speicheranbieter für verwendet werden "[KubeVirt](#)".



Lesen Sie, bevor Sie ein Kubernetes-Cluster von 1.24 auf 1.25 oder höher aktualisieren, auf dem Trident installiert "[Aktualisieren einer Helm-Installation](#)" ist.

## Unterstützte Back-Ends (Storage)

Um Trident verwenden zu können, benötigen Sie eines oder mehrere der folgenden unterstützten Back-Ends:

- Amazon FSX für NetApp ONTAP
- Azure NetApp Dateien
- Cloud Volumes ONTAP
- Google Cloud NetApp Volumes
- Lokale FAS-, AFF- oder ASA r2-Cluster-Versionen unter eingeschränkter Unterstützung von NetApp. Siehe ["Unterstützung Der Softwareversion"](#).
- NetApp All-SAN-Array (ASA)
- NetApp HCI/Element Software 11 oder höher

## Anforderungen an die Funktionen

Die folgende Tabelle bietet einen Überblick über die mit dieser Version von Trident verfügbaren Funktionen und die unterstützten Versionen von Kubernetes.

Funktion	Kubernetes-Version	Funktionstore erforderlich?
Trident	1,25 - 1,32	Nein
Volume Snapshots	1,25 - 1,32	Nein
PVC aus Volume Snapshots	1,25 - 1,32	Nein
ISCSI PV-Größe	1,25 - 1,32	Nein
Bidirektionales ONTAP-CHAP	1,25 - 1,32	Nein
Dynamische Exportrichtlinien	1,25 - 1,32	Nein
Trident Operator	1,25 - 1,32	Nein
CSI-Topologie	1,25 - 1,32	Nein

## Getestete Host-Betriebssysteme

Trident unterstützt zwar offiziell keine bestimmten Betriebssysteme, aber dafür ist bekannt, dass Folgendes funktioniert:

- Redhat CoreOS (RHCOS) Versionen, die von OpenShift Container Platform (AMD64 und ARM64) unterstützt werden
- RHEL 8 ODER HÖHER (AMD64 UND ARM64)



Für NVMe/TCP ist RHEL 9 oder höher erforderlich.



- Ubuntu 22.04 oder höher (AMD64 und ARM64)
- Windows Server 2022

Standardmäßig wird Trident in einem Container ausgeführt und wird daher auf jedem Linux-Worker ausgeführt. Diese Mitarbeiter müssen jedoch in der Lage sein, die Volumes, die Trident bietet, mit dem standardmäßigen NFS-Client oder iSCSI-Initiator zu mounten, je nach den von Ihnen verwendeten Back-Ends.

Das `tridentctl` Dienstprogramm läuft auch auf einer dieser Linux-Distributionen.

## Host-Konfiguration

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

["Bereiten Sie den Knoten „Worker“ vor"](#)

## Konfiguration des Storage-Systems

Trident erfordert möglicherweise Änderungen am Storage-System, bevor es von einer Backend-Konfiguration verwendet werden kann.

["Back-Ends konfigurieren"](#)

## Trident-Ports

Trident erfordert für die Kommunikation den Zugriff auf bestimmte Ports.

["Trident-Ports"](#)

## Container-Images und entsprechende Kubernetes-Versionen

Bei Installationen mit Air-Gap-Technologie ist die folgende Liste eine Referenz für Container-Images, die zur Installation von Trident erforderlich sind. Überprüfen Sie mit dem `tridentctl images` Befehl die Liste der erforderlichen Container-Images.

Kubernetes-Versionen	Container-Image
v1.25.0, v1.26.0, v1.27.0, v1.28.0, v1.29.0 v1.30.0, v1.31.0, v1.32.0	<ul style="list-style-type: none"> <li>• <code>docker.io/netapp/Trident:24.10.0</code></li> <li>• <code>docker.io/netapp/Trident-AutoSupport:24.10</code></li> <li>• <code>Registry.k8s.io/SIG-Storage/csi-provisioner:v5.1.0</code></li> <li>• <code>Registry.k8s.io/SIG-Storage/csi-Attacher:v4.7.0</code></li> <li>• <code>Registry.k8s.io/SIG-Storage/csi-resizer:v1.12.0</code></li> <li>• <code>Registry.k8s.io/SIG-Storage/csi-snapshotter:v8.1.0</code></li> <li>• <code>Registry.k8s.io/SIG-Storage/csi-Node-driver-Registrar:v2.12.0</code></li> <li>• <code>docker.io/netapp/Trident-Operator:24.10.0 (optional)</code></li> </ul>

# **Installation Von Trident**

**Installation über den Trident Operator**

**Installieren Sie mit tridentctl**

# 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 RedHat CoreOS 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`.



Wir empfehlen ONTAP 9.12 oder höher für NVMe/TCP.

#### SCSI-über-FC-Tools

**SCSI over Fibre Channel (FC) ist ein Tech Preview Feature in der Trident 24.10 Version.**

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

Weitere Informationen finden Sie unter "[Möglichkeiten zur Konfiguration von FC- FC-NVMe SAN-Hosts](#)".

## 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, auf denen RHEL/RedHat CoreOS mit iSCSI-PVs ausgeführt wird, die `MountOption` in der StorageClass an `discard`, um Inline-Speicherplatzrückforderung durchzuführen. 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 `iscsiadm` ermitteln, bevor der iSCSI-Daemon `open-iscsi` gestartet 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-100.2410.0.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.

## 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, auf denen RHEL/RedHat CoreOS mit FC PVs ausgeführt wird, die `MountOption` in der `StorageClass` an `discard`, um Inline-Speicherplatz-Rückgewinnung durchzuführen. 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
```

## Unterstützung für Fibre Channel (FC)

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

**SCSI over Fibre Channel (FC) ist ein Tech Preview Feature in der Trident 24.10 Version.**

Fibre Channel ist aufgrund seiner hohen Performance, Zuverlässigkeit und Skalierbarkeit ein weit verbreitetes Protokoll in Enterprise-Storage-Umgebungen. Er bietet einen robusten und effizienten Kommunikationskanal für Speichergeräte, der schnelle und sichere Datenübertragungen ermöglicht. Durch die Verwendung von SCSI über Fibre Channel können Sie ihre vorhandene SCSI-basierte Speicherinfrastruktur nutzen und gleichzeitig von den High-Performance- und Fernfunktionen von Fibre Channel profitieren. Sie unterstützt die Konsolidierung von Speicherressourcen und die Erstellung skalierbarer und effizienter Storage Area Networks (SANs), die große Datenmengen mit geringer Latenz verarbeiten können.

Mithilfe der FC-Funktion mit Trident können Sie folgende Aufgaben ausführen:

- Dynamische Bereitstellung von VES mithilfe einer Implementierungsspezifikation
- Erstellen Sie Volume-Snapshots und ein neues Volume aus dem Snapshot.
- Klonen einer vorhandenen FC-PVC.
- Die Größe eines bereits bereitgestellten Volumes ändern.

### 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"](#)

#### 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 für FC vorzubereiten, müssen Sie die erforderlichen Tools installieren.

#### 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, auf denen RHEL/RedHat CoreOS mit FC PVs ausgeführt wird, die MountOption in der StorageClass an `discard`, um Inline-Speicherplatz-Rückgewinnung durchzuführen. 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
```

## Erstellen Sie eine Backend-Konfiguration

Erstellen Sie ein Trident-Backend für den `ontap-san` Treiber und `fc` als `sanType`.

Siehe:

- ["Vorbereiten der Back-End-Konfiguration mit ONTAP-SAN-Treibern"](#)
- ["ONTAP-SAN-Konfigurationsoptionen und Beispiele"](#)

### Beispiel für eine Back-End-Konfiguration mit FC

```
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
  sanType: fcp
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret
```

## Erstellen Sie eine Speicherklasse

Weitere Informationen finden Sie unter:

- ["Optionen für die Storage-Konfiguration"](#)

### Beispiel für Storage-Klasse

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: fcp-sc
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
  storagePools: "ontap-san-backend:.*"
  fsType: "ext4"
allowVolumeExpansion: True
```

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

## Überlegungen

- Der Azure NetApp Files-Service unterstützt keine Volumes, 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-100.2410.0.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` auf `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-
  xxxxx-xxxx-xxxxxxxxxxx' *
```

## 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-100.2410.0.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-xxxxxxxxxxxx.

### Zusätzliche Anforderungen für SMB Volumes

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

- Active Directory konfiguriert und mit Azure NetApp Files verbunden. Siehe ["Microsoft: Erstellen und Verwalten von Active Directory-Verbindungen für Azure NetApp Files"](#).
- Kubernetes-Cluster mit einem Linux-Controller-Knoten und mindestens einem Windows-Worker-Node, auf dem Windows Server 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)

Parameter	Beschreibung	Standard
location	Name des Azure-Standorts, an dem die neuen Volumes erstellt werden Optional, wenn verwaltete Identitäten auf einem AKS-Cluster aktiviert sind	
resourceGroups	Liste der Ressourcengruppen zum Filtern ermittelter Ressourcen	„[]“ (kein Filter)
netappAccounts	Liste von NetApp Accounts zur Filterung erkannter Ressourcen	„[]“ (kein Filter)
capacityPools	Liste der Kapazitäts-Pools zur Filterung erkannter Ressourcen	„[]“ (kein Filter, zufällig)
virtualNetwork	Name eines virtuellen Netzwerks mit einem delegierten Subnetz	“
subnet	Name eines 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 kommasetrennten Mount-Optionen ein <code>nfsvers=4</code> , 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)

Parameter	Beschreibung	Standard
debugTraceFlags	Fehler-Flags bei der Fehlerbehebung beheben. Beispiel, <code>\{"api": false, "method": true, "discovery": true\}</code> . Verwenden Sie dies nur, wenn Sie Fehler beheben und einen detaillierten Log Dump benötigen.	Null
nasType	Konfiguration der Erstellung von NFS- oder SMB-Volumes Optionen sind <code>nfs</code> , <code>smb</code> oder Null. Einstellung auf null setzt standardmäßig auf NFS-Volumes.	<code>nfs</code>
supportedTopologies	Stellt eine Liste von Regionen und Zonen dar, die von diesem Backend unterstützt werden. Weitere Informationen finden Sie unter " <a href="#">Verwenden Sie die CSI-Topologie</a> ".	



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, sind Ihre App-Registrierung wahrscheinlich nicht über die erforderlichen Berechtigungen und Ressourcen (Subnetz, virtuelles Netzwerk, Kapazitäts-Pool) verbunden. 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	<code>&lt;Ressourcengruppe&gt;</code>
NetApp Konto	<code>&lt;Resource Group&gt;/&lt;netapp Account&gt;</code>
Kapazitäts-Pool	<code>&lt;Resource Group&gt;/&lt;netapp Account&gt;/&lt;Capacity Pool&gt;</code>
Virtuelles Netzwerk	<code>&lt;Ressourcengruppe&gt;/&lt;virtuelles Netzwerk&gt;</code>
Subnetz	<code>&lt;Ressourcengruppe&gt;/&lt;virtuelles Netzwerk&gt;/&lt;Subnetz&gt;</code>



## 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
<code>exportRule</code>	Exportregeln für neue Volumes <code>exportRule</code> 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“
<code>snapshotDir</code>	Steuert die Sichtbarkeit des .Snapshot-Verzeichnisses	„Wahr“ für NFSv4 „falsch“ für NFSv3
<code>size</code>	Die Standardgröße der neuen Volumes	„100 GB“
<code>unixPermissions</code>	die unix-Berechtigungen neuer Volumes (4 Oktal-Ziffern). Für SMB Volumes ignoriert.	„“ (Vorschau-Funktion, erfordert Whitelisting im Abonnement)

### Beispielkonfigurationen

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

## Minimalkonfiguration

Dies ist die absolute minimale Backend-Konfiguration. Mit dieser Konfiguration erkennt 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. Sie können NFS-Volumes über ein Google Cloud NetApp Volumes-Back-End einbinden.

#### 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
<code>google-cloud-netapp-volumes</code>	NFS	Dateisystem	RWO, ROX, RWX, RWOP	<code>nfs</code>

### 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` auf `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-100.2406.0.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 NFS-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 <code>storageDriverName</code> 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
supportedTopologies	Stellt eine Liste von Regionen und Zonen dar, die von diesem Backend unterstützt werden. Weitere Informationen finden Sie unter " <a href="#">Verwenden Sie die CSI-Topologie</a> ". 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
<code>exportRule</code>	Die Exportregeln für neue Volumes. Muss eine kommasetrennte Liste einer beliebigen Kombination von IPv4-Adressen sein.	„0.0.0.0/0“
<code>snapshotDir</code>	Zugriff auf das <code>.snapshot</code> Verzeichnis	„Wahr“ für NFSv4 „falsch“ für NFSv3
<code>snapshotReserve</code>	Prozentsatz des für Snapshots reservierten Volumes	„“ (Standardeinstellung 0 akzeptieren)
<code>unixPermissions</code>	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.



```
XsYg6gyxy4zq70lwWgLwGa==\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
```





```
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
```



```
znHczZsrtrtHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m
znHczZsrtrtHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m
znHczZsrtrtHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/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
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 `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.

### Weitere Beispiele

#### Beispiele für Definitionen von 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"](#).

### 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
<code>gcp-cvs</code>	NFS	Dateisystem	RWO, ROX, RWX, RWOP	<code>nfs</code>

### 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
<code>version</code>		Immer 1
<code>storageDriverName</code>	Name des Speichertreibers	„gcp-cvs“
<code>backendName</code>	Benutzerdefinierter Name oder das Storage-Backend	Treibername + „_“ + Teil des API-Schlüssels
<code>storageClass</code>	Optionaler Parameter zur Angabe des CVS-Servicetyps. Verwenden Sie <code>software</code> , um den CVS-Diensttyp auszuwählen. Andernfalls übernimmt Trident den CVS-Performance Servicetyp ( <code>hardware</code> ).	
<code>storagePools</code>	CVS-Diensttyp nur. Optionaler Parameter zur Angabe von Speicherpools für die Volume-Erstellung.	
<code>projectNumber</code>	Google Cloud Account Projektnummer. Der Wert ist auf der Startseite des Google Cloud Portals zu finden.	
<code>hostProjectNumber</code>	Erforderlich bei Verwendung eines gemeinsamen VPC-Netzwerks. In diesem Szenario <code>projectNumber</code> handelt es sich um das Service-Projekt und <code>hostProjectNumber</code> das Host-Projekt.	
<code>apiRegion</code>	Die Google Cloud-Region, in der Trident Cloud Volumes Service Volumes erstellt. Bei der Erstellung regionsübergreifender Kubernetes-Cluster können in einem erstellte Volumes <code>apiRegion</code> für Workloads verwendet werden, die auf Nodes in mehreren Google Cloud-Regionen geplant sind. Der regionale Verkehr verursacht zusätzliche Kosten.	
<code>apiKey</code>	API-Schlüssel für das Google Cloud-Servicekonto mit der <code>netappcloudvolumes.admin</code> Rolle. Er enthält den JSON-formatierten Inhalt der privaten Schlüsseldatei eines Google Cloud-Dienstkontos (wortgetreu in die Back-End-Konfigurationsdatei kopiert).	

Parameter	Beschreibung	Standard
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 <code>standard</code> , <code>premium</code> oder <code>extreme</code> . CVS-Werte sind <code>standardsw</code> oder <code>zoneredundantstandardsw</code> .	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, <code>\{"api":false, "method":true\}</code> . 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 <code>allowedTopologies</code> alle Regionen umfassen. Beispiel: - <code>key: topology.kubernetes.io/region</code> <code>values:</code> - <code>us-east1</code> - <code>europa-west1</code>	

## Optionen zur Volume-Bereitstellung

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

Parameter	Beschreibung	Standard
exportRule	Die Exportregeln für neue Volumes. Muss eine kommasetrennte Liste beliebiger Kombinationen von IPv4-Adressen oder IPv4-Subnetzen in CIDR-Notation sein.	„0.0.0.0/0“
snapshotDir	Zugriff auf das <code>.snapshot</code> Verzeichnis	„Falsch“
snapshotReserve	Prozentsatz des für Snapshots reservierten Volumes	"" (CVS Standard 0 akzeptieren)

Parameter	Beschreibung	Standard
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-Standardereinstellungen.

```
---
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. Siehe [Definitionen 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
<code>solidfire-san</code>	ISCSI	Block-Storage	RWO, ROX, RWX, RWOP	Kein Dateisystem. Rohes Blockgerät.
<code>solidfire-san</code>	ISCSI	Dateisystem	RWO, RWOP	<code>xfs ext3, , ext4</code>

### 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
<code>version</code>		Immer 1
<code>storageDriverName</code>	Name des Speichertreibers	Immer „solidfire-san“
<code>backendName</code>	Benutzerdefinierter Name oder das Storage-Backend	IP-Adresse „SolidFire_“ + Storage (iSCSI)

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	Findet die ID einer Zugriffsgruppe namens „Dreizack“
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, „Methode“: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 Beispielfdefinitionsdatei 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 (technische Vorschau in Trident 24.10)	Block-Storage	RWO, ROX, RWX, RWOP	Kein Filesystem, rohes Block-Gerät
ontap-san	ISCSI SCSI über FC (technische Vorschau in Trident 24.10)	Dateisystem	RWO, RWOP  ROX und RWX sind im Filesystem-Volume-Modus nicht verfügbar.	xfs ext3, , ext4
ontap-san	NVMe/TCP  Siehe <a href="#">Weitere Überlegungen zu NVMe/TCP</a> .	Block-Storage	RWO, ROX, RWX, RWOP	Kein Filesystem, rohes Block-Gerät
ontap-san	NVMe/TCP  Siehe <a href="#">Weitere Überlegungen zu NVMe/TCP</a> .	Dateisystem	RWO, RWOP  ROX und RWX sind im Filesystem-Volume-Modus nicht verfügbar.	xfs ext3, , ext4
ontap-san-economy	ISCSI	Block-Storage	RWO, ROX, RWX, RWOP	Kein Filesystem, rohes Block-Gerät

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



- 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.

### 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-Back-Ends benötigt Trident der SVM mindestens ein Aggregat zugewiesen.

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 `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

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

#### JSON

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

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

### Aktivieren Sie die zertifikatbasierte Authentifizierung

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

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

Ein typischer Workflow umfasst die folgenden Schritte.

#### Schritte

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

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

2. Fügen Sie dem ONTAP-Cluster ein vertrauenswürdigen 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 vertrauenswürdigen CA-Zertifikat mit Base64.

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

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

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

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

### Aktualisieren Sie Authentifizierungsmethoden, oder drehen Sie die Anmeldedaten

Sie können ein vorhandenes Backend aktualisieren, um eine andere Authentifizierungsmethode zu verwenden oder ihre Anmeldedaten zu drehen. Das funktioniert auf beide Arten: Back-Ends, die einen Benutzernamen/ein Passwort verwenden, können aktualisiert werden, um Zertifikate zu verwenden; Back-Ends, die Zertifikate verwenden, können auf Benutzername/Passwort-basiert aktualisiert werden. Dazu müssen Sie die vorhandene Authentifizierungsmethode entfernen und die neue Authentifizierungsmethode hinzufügen. Verwenden Sie dann die aktualisierte 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 CLI/ONTAP-Benutzeroberfläche, 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-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, 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 <a href="#">[mcc-best]</a> .	„10.0.0.1“, „[2001:1234:abcd::fefe]“
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]. <b>Nicht für iSCSI angeben.</b> Trident verwendet <a href="#">"ONTAP selektive LUN-Zuordnung"</a> , um die für die Einrichtung einer Multi-Path-Sitzung erforderlichen iSCSI LIFs zu ermitteln. Eine Warnung wird erzeugt, wenn dataLIF explizit definiert ist. <b>Für MetroCluster weglassen.</b> Siehe <a href="#">[mcc-best]</a> .	Abgeleitet von SVM
svm	Zu verwendende virtuelle Speichermaschine <b>omit für MetroCluster.</b> Siehe <a href="#">[mcc-best]</a> .	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 <a href="#">"Vorbereiten der Back-End-Konfiguration mit ONTAP-SAN-Treibern"</a> .	false
chapInitiatorSecret	CHAP-Initiatorschlüssel. Erforderlich, wenn useCHAP=true	“

Parameter	Beschreibung	Standard
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.	“ ”
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

Parameter	Beschreibung	Standard
aggregate	<p>Aggregat für die Bereitstellung (optional, wenn eingestellt, muss der SVM zugewiesen werden) Für den <code>ontap-nas-flexgroup</code> Treiber wird diese Option ignoriert. Falls nicht, können alle verfügbaren Aggregate verwendet werden, um ein FlexGroup Volume bereitzustellen.</p> <div style="border: 1px solid gray; padding: 10px; margin-top: 10px;">  <p>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.</p> </div>	“
limitAggregateUsage	Bereitstellung fehlgeschlagen, wenn die Nutzung über diesem Prozentsatz liegt. Wenn Sie ein Amazon FSX für NetApp ONTAP-Backend verwenden, geben Sie nicht an <code>limitAggregateUsage</code> . Die angegebenen <code>fsxadmin</code> und <code>vsadmin</code> enthalten nicht die erforderlichen Berechtigungen, um die aggregierte Nutzung abzurufen und sie mit Trident zu begrenzen.	„ (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)
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

Parameter	Beschreibung	Standard
useREST	<p>Boolescher Parameter zur Verwendung von ONTAP REST-APIs.</p> <p>useREST Bei Einstellung auf <code>true</code> verwendet Trident ONTAP REST APIs zur Kommunikation mit dem Backend; bei Einstellung auf <code>false</code> verwendet Trident ONTAP ZAPI Aufrufe zur Kommunikation mit dem Backend. Diese Funktion erfordert ONTAP 9.11.1 und höher. Darüber hinaus muss die verwendete ONTAP-Anmelderolle Zugriff auf die Anwendung haben <code>ontap</code>. Dies wird durch die vordefinierten <code>vsadmin</code> Rollen und <code>cluster-admin</code> erreicht. Ab Trident 24.06-Version und ONTAP 9.15.1 oder höher useREST ist standardmäßig auf eingestellt <code>true</code>; ändern Sie useREST zu <code>false</code> ONTAP-ZAPI-Aufrufe verwenden. useREST Ist vollständig für NVMe/TCP qualifiziert.</p>	<code>true</code> Für ONTAP 9.15.1 oder höher, andernfalls <code>false</code> .
sanType	<p>Verwenden Sie diese Option, um für iSCSI, <code>nvme</code> für NVMe/TCP oder <code>fc</code> für SCSI über Fibre Channel (FC) auszuwählen <code>iscsi</code>. <b>'fc' (SCSI over FC) ist ein Tech Preview Feature in der Trident 24.10 Version.</b></p>	<code>iscsi</code> Falls leer
formatOptions	<p>Verwenden Sie <code>formatOptions</code> zum Angeben von Befehlszeilenargumenten für den <code>mkfs</code> 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 <code>mkfs</code>-Befehlsoptionen angeben, ohne den Gerätepfad. Beispiel: „-E nodiscard“</p> <ul style="list-style-type: none"> <li>• <code>ontap-san `ontap-san-economy`</code> Nur für und Treiber unterstützt.*</li> </ul>	
limitVolumePoolSize	<p>Maximale anforderbare FlexVol-Größe bei Verwendung von LUNs im ONTAP-san-Economy-Backend.</p>	„“ (nicht standardmäßig durchgesetzt)
denyNewVolumePools	<p>Schränkt das Erstellen neuer FlexVol Volumes für LUNs ein <code>ontap-san-economy</code> Zur Bereitstellung neuer PVS werden nur vorbestehende FlexVols verwendet.</p>	

## 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 (`xf`s, `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
<code>spaceAllocation</code>	Speicherplatzzuweisung für LUNs	„Wahr“
<code>spaceReserve</code>	Modus für Speicherplatzreservierung; „none“ (Thin) oder „Volume“ (Thick)	„Keine“
<code>snapshotPolicy</code>	Die Snapshot-Richtlinie zu verwenden	„Keine“
<code>qosPolicy</code>	QoS-Richtliniengruppe zur Zuweisung für erstellte Volumes Wählen Sie eine der <code>qosPolicy</code> oder <code>adaptiveQosPolicy</code> 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.	“
<code>adaptiveQosPolicy</code>	Adaptive QoS-Richtliniengruppe mit Zuordnung für erstellte Volumes Wählen Sie eine der <code>qosPolicy</code> oder <code>adaptiveQosPolicy</code> pro Storage Pool/Backend	“
<code>snapshotReserve</code>	Prozentsatz des für Snapshots reservierten Volumes	„0“, wenn <code>snapshotPolicy</code> „keine“ ist, andernfalls „“
<code>splitOnClone</code>	Teilen Sie einen Klon bei der Erstellung von seinem übergeordneten Objekt auf	„Falsch“
<code>encryption</code>	Aktivieren Sie NetApp Volume Encryption (NVE) auf dem neuen Volume, Standardeinstellung ist <code>false</code> . 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 <a href="#">"Funktionsweise von Trident mit NVE und NAE"</a> .	„Falsch“
<code>luksEncryption</code>	Aktivieren Sie die LUKS-Verschlüsselung. Siehe <a href="#">"Linux Unified Key Setup (LUKS) verwenden"</a> . LUKS-Verschlüsselung wird für NVMe/TCP nicht unterstützt.	“
<code>securityStyle</code>	Sicherheitstyp für neue Volumes	<code>unix</code>
<code>tieringPolicy</code>	Tiering-Richtlinie, die zu „keinen“ verwendet wird	„Nur snapshot“ für eine SVM-DR-Konfiguration vor ONTAP 9.5
<code>nameTemplate</code>	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:

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

3 entries were displayed.

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

### Minimale Konfigurationsbeispiele

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



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

### 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 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>
```

## 1. Beispiel

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-san
managementLIF: 192.168.1.66
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
```

## 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}}_{{.volume.R
    equestName}}"
  },
  "labels": {"cluster": "ClusterA", "PVC":
    "{{.volume.Namespace}}_{{.volume.RequestName}}"}
}
```

## FormatOptions-Beispiel für den </code>-Treiber <code> ONTAP-san-economr

```
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 dem FlexVol gesetzt. 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 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-nas-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-nas-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.

## 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-Back-Ends benötigt Trident der SVM mindestens ein Aggregat zugewiesen.
- 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 `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

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

## JSON

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

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

### Aktivieren Sie die zertifikatbasierte Authentifizierung

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

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

Ein typischer Workflow umfasst die folgenden Schritte.

### Schritte

1. Erzeugen eines Clientzertifikats und eines Schlüssels. Legen Sie beim Generieren den allgemeinen

Namen (CN) für den ONTAP-Benutzer fest, der sich authentifizieren soll als.

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

2. Fügen Sie dem ONTAP-Cluster ein vertrauenswürdigen 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-Service-Richtlinie auf festgelegt ist default-data-management.

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

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



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

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

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

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

NAME	STORAGE DRIVER	UUID
NasBackend	ontap-nas	98e19b74-aec7-4a3d-8dcf-128e5033b214

```

+-----+-----+-----+
+-----+-----+
| 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.



Im Trident 24.10 `ontap-nas` funktioniert der Speichertreiber weiterhin wie in den früheren Versionen; für den ONTAP-nas-Treiber wurde keine Änderung vorgenommen. Nur der `ontap-nas-economy` Storage-Treiber verfügt in Trident 24.10 über eine Volume-basierte granulare Zugriffssteuerung.

## 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 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 `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>
      fileType: 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 auf der SVM sowohl NFS- als auch SMB/CIFS-Protokolle konfigurieren, um ein SMB-Volume für ONTAP vor Ort 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	<b>Muss auf.</b> gesetzt werden <code>smb</code> Wenn Null, wird standardmäßig auf <code>nfs</code> .	smb
securityStyle	Sicherheitstyp für neue Volumes. <b>Muss für SMB Volumes auf oder <code>mixed</code> gesetzt werden <code>ntfs</code>.</b>	ntfs Oder mixed für SMB Volumes
unixPermissions	Modus für neue Volumes. <b>Muss für SMB Volumes leer gelassen werden.</b>	“ ”

## 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“, „ontap-nas-flexgroup“, „ontap-san“, „ontap-san-Economy“
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 <a href="#">[mcc-best]</a> .	„10.0.0.1“, „[2001:1234:abcd::fefe]“
dataLIF	IP-Adresse des LIF-Protokolls. Wir empfehlen die Angabe dataLIF. Falls nicht bereitgestellt, ruft Trident die Daten-LIFs von der SVM ab. Sie können einen vollständig qualifizierten Domänennamen (FQDN) angeben, der für die NFS-Mount-Vorgänge verwendet werden soll. Damit können Sie ein Round-Robin-DNS zum Load-Balancing über mehrere Daten-LIFs erstellen. Kann nach der Anfangseinstellung geändert werden. Siehe . 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]. <b>Für MetroCluster weglassen.</b> Siehe <a href="#">[mcc-best]</a> .	Angegebene Adresse oder abgeleitet von SVM, falls nicht angegeben (nicht empfohlen)
svm	Zu verwendende virtuelle Speichermaschine <b>omit für MetroCluster.</b> Siehe <a href="#">[mcc-best]</a> .	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	“
trustedCACertificate	Base64-kodierte Wert des vertrauenswürdigen CA-Zertifikats. Optional Wird für zertifikatbasierte Authentifizierung verwendet	“

Parameter	Beschreibung	Standard
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	<p>Das Präfix wird beim Bereitstellen neuer Volumes in der SVM verwendet. Kann nicht aktualisiert werden, nachdem Sie sie festgelegt haben</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;">  <p>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.</p> </div>	trident
aggregate	<p>Aggregat für die Bereitstellung (optional, wenn eingestellt, muss der SVM zugewiesen werden) Für den <code>ontap-nas-flexgroup</code> Treiber wird diese Option ignoriert. Falls nicht, können alle verfügbaren Aggregate verwendet werden, um ein FlexGroup Volume bereitzustellen.</p> <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;">  <p>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.</p> </div>	“
limitAggregateUsage	Bereitstellung fehlgeschlagen, wenn die Nutzung über diesem Prozentsatz liegt. <b>Gilt nicht für Amazon FSX für ONTAP</b>	„ (nicht standardmäßig durchgesetzt)

Parameter	Beschreibung	Standard
FlexgroupAggregateList	<p>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 <b>ONTAP-nas-FlexGroup</b>-Speichertreiber.</p> <div style="border: 1px solid gray; padding: 10px; margin-top: 10px;">  <p>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.</p> </div>	“
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.	“ (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 Bei Einstellung auf true verwendet Trident ONTAP REST APIs zur Kommunikation mit dem Backend; bei Einstellung auf false verwendet Trident ONTAP ZAPI Aufrufe zur Kommunikation mit dem Backend. Diese Funktion erfordert ONTAP 9.11.1 und höher. Darüber hinaus muss die verwendete ONTAP-Anmelderolle Zugriff auf die Anwendung haben <code>ontap</code> . Dies wird durch die vordefinierten <code>vsadmin</code> Rollen und <code>cluster-admin</code> erreicht. Ab Trident 24.06-Version und ONTAP 9.15.1 oder höher useREST ist standardmäßig auf eingestellt true; ändern Sie useREST zu false ONTAP-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 <code>ontap-nas-economy</code> 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 <code>false</code> . 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 " <a href="#">Funktionsweise von Trident mit NVE und NAE</a> ".	„Falsch“
tieringPolicy	Tiering-Richtlinie, die zu „keinen“ verwendet wird	„Nur snapshot“ für eine SVM-DR-Konfiguration vor ONTAP 9.5
unixPermissions	Modus für neue Volumes	„777“ für NFS Volumes; leer (nicht zutreffend) für SMB Volumes
snapshotDir	Steuert den Zugriff auf das <code>.snapshot</code> Verzeichnis	„Wahr“ für NFSv4 „falsch“ für NFSv3
exportPolicy	Zu verwendende Exportrichtlinie	„Standard“
securityStyle	Sicherheitstyp für neue Volumes. NFS-Unterstützung <code>mixed</code> und <code>unix</code> -Sicherheitsstile. SMB-Unterstützung <code>mixed</code> und <code>ntfs</code> Sicherheitsstile.	NFS-Standard ist <code>unix</code> . SMB-Standard ist <code>ntfs</code> .
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  $2/5 = 10$  gib, und die verfügbare Größe beträgt 5 gib. Dies entspricht dem, was der Benutzer in der PVC-Anfrage angefordert hat. Der `volume show` Befehl sollte die Ergebnisse ähnlich wie in diesem Beispiel anzeigen:

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

2 entries were displayed.

Vorhandene Back-Ends 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 `svm` und `dataLIF` weg. 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 `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}}_{{.volume.R
  equestName}}"
},
"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 Daten-LIF nach der Erstkonfiguration ändern, indem Sie den folgenden Befehl ausführen, um die neue Backend-JSON-Datei mit aktualisierten Daten-LIF bereitzustellen.

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



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

## Amazon FSX für NetApp ONTAP

### 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. Amazon FSX für NetApp ONTAP wird Data ONTAP als gemanagtes Dateisystem in der Cloud zur Verfügung stellen. 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 `kubect1` 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, `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_ST ANDARD	Ja.	Ja.	Ja.	Ja.
AL2_x86_64	Ja.	Ja.	Ja**	Ja**
BOTTLEROCKET_x86_64	Ja*	Ja.	K. A.	K. A.

AL2023_ARM_64_S TANDARD	Ja.	Ja.	Ja.	Ja.
AL2_ARM_64	Ja.	Ja.	Ja**	Ja**
BOTTLEROCKET_A RM_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 Secret Manager-Schlüssel

Dieses Beispiel erstellt einen AWS Secret Manager-Schlüssel, 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

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
Secret manager"
```

### Richtlinien-JSON-Datei:

```
policy.json:
{
  "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**

## AWS CLI

```
aws iam create-role --role-name trident-controller \  
--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 OIDC 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
```

Verwenden Sie den folgenden Befehl, um IAM OIDC dem Cluster zuzuordnen:

```
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-Bediener 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/v24.10.0/trident-
installer-24.10.0.tar.gz
tar -xf trident-installer-24.10.0.tar.gz
cd trident-installer/helm
```

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-100.2410.0.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	STATUS	CHART	NAMESPACE	REVISION	UPDATED	APP VERSION
trident-operator			trident	1	2024-10-14 14:31:22.463122	
+0300 IDT	deployed		trident-operator-100.2410.0			24.10.0

### 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 den folgenden Beispielbefehlen 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. Klicken Sie im linken Navigationsbereich auf **Cluster**.
3. Klicken Sie auf den Namen des Clusters, für den Sie das NetApp Trident-CSI-Add-On konfigurieren möchten.
4. Klicken Sie auf **Add-ons** und dann auf **Weitere Add-Ons** erhalten.
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. Klicken Sie Auf **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. Erweitern Sie die **Optionale Konfigurationseinstellungen**, 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: `eks.amazonaws.com/role-arn:arn:aws:iam::464262061435:role/AmazonEKS_FSXN_CSI_DriverRole`). 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:

```
add-on.json
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v24.10.0-eksbuild.1",
  "serviceAccountRoleArn": "<arn:aws:iam::123456:role/astratrident-
role>",
  "configurationValues": "{\"cloudIdentity\":
\"'eks.amazonaws.com/role-arn:
<arn:aws:iam::123456:role/astratrident-role>'\",
  \"cloudProvider\": \"AWS\"}"
}
```

## 2. Trident EKS-Add-On installieren

```
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  v24.10.0-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 v24.10.0-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

- Öffnen Sie die Amazon EKS Konsole <https://console.aws.amazon.com/eks/home#/clusters>.
- Klicken Sie im linken Navigationsbereich auf **Cluster**.
- Klicken Sie auf den Namen des Clusters, für den Sie das NetApp Trident-CSI-Add-On aktualisieren möchten.
- Klicken Sie auf die Registerkarte **Add-ons**.
- Klicken Sie auf **Trident by NetApp** und dann auf **Bearbeiten**.
- Gehen Sie auf der Seite **Configure Trident by NetApp** wie folgt vor:
  - Wählen Sie die **Version** aus, die Sie verwenden möchten.
  - Erweitern Sie die **Optionale Konfigurationseinstellungen** und ändern Sie sie nach Bedarf.
  - Klicken Sie auf **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 v24.6.1-eksbuild.1 \
    --service-account-role-arn arn:aws:iam::111122223333:role/role-name
--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** – Wir empfehlen, 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. Klicken Sie im linken Navigationsbereich auf **Cluster**.
3. Klicken Sie auf den Namen des Clusters, für den Sie das NetApp Trident-CSI-Add-On entfernen möchten.
4. Klicken Sie auf die Registerkarte **Add-ons** und dann auf **Trident by NetApp**.\*
5. Klicken Sie Auf **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. Klicken Sie Auf **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:xxxxxxx: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:xxxxxxx: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
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 für 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 <code>aws</code> Feld angeben <code>fsxFilesystemID</code> , müssen Sie den nicht angeben <code>managementLIF</code> , da Trident die SVM-Informationen von AWS abrufen <code>managementLIF</code> . Daher müssen Sie die Anmeldedaten für einen Benutzer unter der SVM (z. B. <code>vsadmin</code> ) angeben, und der Benutzer muss über die Rolle verfügen <code>vsadmin</code> .	„10.0.0.1“, „[2001:1234:abcd::fefe]“

Parameter	Beschreibung	Beispiel
dataLIF	<p>IP-Adresse des LIF-Protokolls.</p> <p><b>ONTAP NAS drivers:</b> Wir empfehlen die Angabe von dataLIF. Falls nicht bereitgestellt, ruft Trident die Daten-LIFs von der SVM ab. Sie können einen vollständig qualifizierten Domännennamen (FQDN) angeben, der für die NFS-Mount-Vorgänge verwendet werden soll. Damit können Sie ein Round-Robin-DNS zum Load-Balancing über mehrere Daten-LIFs erstellen. Kann nach der Anfangseinstellung geändert werden. Siehe . <b>ONTAP-SAN-Treiber:</b> Geben Sie nicht für iSCSI an. Trident verwendet die selektive LUN-Zuordnung von ONTAP, um die iSCSI LIFs zu ermitteln, die für die Einrichtung einer Multi-Path-Sitzung erforderlich sind. Eine Warnung wird erzeugt, wenn dataLIF explizit definiert ist. 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].</p>	
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	<b>Nicht für Amazon FSX für NetApp ONTAP angeben.</b> Die angegebenen <code>fsxadmin</code> und <code>vsadmin</code> 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.	„ (nicht standardmäßig durchgesetzt)
lunsPerFlexvol	Die maximale Anzahl an LUNs pro FlexVol 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 erfordern 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. <b>Muss für SMB-Volumes auf gesetzt smb werden.</b> Einstellung auf null setzt standardmäßig auf NFS-Volumes.	nfs
qtreesPerFlexvol	Maximale Ques pro FlexVol, 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. <b>Tech Preview</b> useREST wird als <b>Tech Preview</b> bereitgestellt, die für Testumgebungen und nicht für Produktions-Workloads empfohlen wird. Wenn auf festgelegt <code>true</code> , 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 <code>ontap</code> . Dies wird durch die vordefinierten <code>vsadmin</code> Rollen und <code>cluster-admin</code> 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-Anmeldeinformationen an, die in AWS Secret Manager zu speichern sind. - name: Amazon Resource Name (ARN) des Geheimnisses, das die Zugangsdaten von SVM enthält. - type: Gesetzt auf <code>awsarn</code> . Weitere Informationen finden Sie unter " <a href="#">Erstellen Sie einen AWS Secrets Manager-Schlüssel</a> ".	

## 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
<code>spaceAllocation</code>	Speicherplatzzuweisung für LUNs	<code>true</code>
<code>spaceReserve</code>	Space Reservation Mode; „none“ (Thin) oder „Volume“ (Thick)	<code>none</code>
<code>snapshotPolicy</code>	Die Snapshot-Richtlinie zu verwenden	<code>none</code>
<code>qosPolicy</code>	QoS-Richtliniengruppe zur Zuweisung für erstellte Volumes Wählen Sie eine der <code>qosPolicy</code> oder <code>adaptiveQosPolicy</code> 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.	„“
<code>adaptiveQosPolicy</code>	Adaptive QoS-Richtliniengruppe mit Zuordnung für erstellte Volumes Wählen Sie eine der <code>qosPolicy</code> oder <code>adaptiveQosPolicy</code> pro Storage-Pool oder Backend. Nicht unterstützt durch <code>ontap-nas-Ökonomie</code>	„“
<code>snapshotReserve</code>	Prozentsatz des für Snapshots reservierten Volumens „0“	Wenn <code>snapshotPolicy</code> ist <code>none</code> , else „“
<code>splitOnClone</code>	Teilen Sie einen Klon bei der Erstellung von seinem übergeordneten Objekt auf	<code>false</code>



Parameter	Beschreibung	Standard
encryption	Aktivieren Sie NetApp Volume Encryption (NVE) auf dem neuen Volume, Standardeinstellung ist <code>false</code> . 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 <a href="#">"Funktionsweise von Trident mit NVE und NAE"</a> .	<code>false</code>
luksEncryption	Aktivieren Sie die LUKS-Verschlüsselung. Siehe <a href="#">"Linux Unified Key Setup (LUKS) verwenden"</a> . Nur SAN	“
tieringPolicy	Tiering-Richtlinie für die Nutzung <code>none</code>	<code>snapshot-only</code> Für Konfiguration vor ONTAP 9.5 SVM-DR
unixPermissions	Modus für neue Volumes. <b>Leere leer für SMB Volumen.</b>	“
securityStyle	Sicherheitstyp für neue Volumes. NFS-Unterstützung <code>mixed</code> und <code>unix</code> -Sicherheitsstile. SMB-Unterstützung <code>mixed</code> und <code>ntfs</code> Sicherheitsstile.	NFS-Standard ist <code>unix</code> . SMB-Standard ist <code>ntfs</code> .

### 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-Treibern](#) die 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
<code>smbShare</code>	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.	<code>smb-share</code>
<code>nasType</code>	<b>Muss auf.</b> gesetzt werden <code>smb</code> Wenn Null, wird standardmäßig auf <code>nfs</code> .	<code>smb</code>
<code>securityStyle</code>	Sicherheitstyp für neue Volumes. <b>Muss für SMB Volumes auf oder mixed gesetzt werden ntfs.</b>	<code>ntfs</code> Oder <code>mixed</code> für SMB Volumes
<code>unixPermissions</code>	Modus für neue Volumes. <b>Muss für SMB Volumes leer gelassen werden.</b>	“

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

### Erstellen Sie eine Speicherklasse

#### Konfigurieren Sie ein Kubernetes StorageClass-Objekt

Trident wird von als bereitstellung identifiziert, die für diese Klasse verwendet wird. Trident wird darin ["Kubernetes StorageClass-Objekt"](#) angewiesen, ein Volume bereitzustellen. Beispiel:

```
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"
```

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 das PV und die PVC

Ein "*PersistentVolume*" (PV) ist eine physische Speicherressource, die vom Clusteradministrator auf einem Kubernetes-Cluster bereitgestellt wird. Die "*PersistentVolumeClaim*" (PVC) ist eine Anforderung für den Zugriff auf das PersistentVolume auf dem Cluster.

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

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

## Beispielmanifeste

### PersistentVolume-Beispielmanifest

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

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-storage
  labels:
    type: local
spec:
  storageClassName: basic-csi
  capacity:
    storage: 10Gi
  accessModes:
    - 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: 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:
    - ReadWriteMany
  resources:
    requests:
      storage: 300Mi
  storageClassName: protection-gold
```

## Erstellen Sie das PV und die PVC

### Schritte

1. Erstellen Sie das PV.

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

## 2. Überprüfen Sie den PV-Status.

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

## 3. Erstellen Sie das PVC.

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

## 4. Überprüfen Sie den PVC-Status.

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

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 <sup>1</sup>	Zeichenfolge	hdd, Hybrid, ssd	Pool enthält Medien dieser Art. Beides bedeutet Hybrid	Medientyp angeben	ontap-nas, ontap-nas-Economy, ontap-nas-Flexgroup, ontap-san, solidfire-san
Bereitstellungstyp	Zeichenfolge	Dünn, dick	Pool unterstützt diese Bereitstellungsmethode	Bereitstellungsmethode angeben	Thick: All ONTAP; Thin: Alle ONTAP und solidfire-san

Attribut	Typ	Werte	Angebot	Anfrage	Unterstützt von
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 Klone	ontap-nas, ontap-san, solidfire-san, gcp-cvs
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

Beispielanwendung bereitstellen.

### Schritte

1. Mouneten 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. Konfigurieren Sie die optionalen Konfigurationseinstellungen nach Bedarf, und wählen Sie **Weiter**.

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 (Wert sollte im folgenden Format sein: `eks.amazonaws.com/role-arn:arn:aws:iam::464262061435:role/AmazonEKS_FSXN_CSI_DriverRole`). HINWEIS: Wenn Sie für die Konfliktlösungsmethode *Überschreiben* auswählen, können eine oder mehrere Einstellungen für das vorhandene Add-On mit den Amazon EKS-Add-on-Einstellungen 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 Beispielfehl wird das Trident EKS Add-On installiert:

```
eksctl create addon --name aws-ebs-csi-driver --cluster <cluster_name>
--service-account-role-arn arn:aws:iam::<account_id>:role/<role_name>
--force
```

**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, named CRD, das Ihnen ermöglicht, Trident-Backends mit `kubectl` zu verwalten. 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 `tridentctl` verwaltet werden. Durch Festlegen der Löschroutine 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 `spec.backendName` gesetzt. 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 `tridentctl` erstellt wurden, haben kein zugeordnetes `TridentBackendConfig` Objekt. Sie können diese Back-Ends mit `kubectl` verwalten, 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 `kubectl` zu erstellen, 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 `kubectl get tbc <tbc-name> -n <trident-namespace>` beobachten 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
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 <code>ontap-san</code> und <code>ontap-san-economy</code>
ONTAP	ChapInitiatorSecret	CHAP-Initiatorschlüssel. Erforderlich, wenn usCHAP=true verwendet wird. Für <code>ontap-san</code> und <code>ontap-san-economy</code>
ONTAP	ChapTargetBenutzername	Zielbenutzername. Erforderlich, wenn usCHAP=true verwendet wird. Für <code>ontap-san</code> und <code>ontap-san-economy</code>

Beschreibung der geheimen Felder der Speicherplattform	Geheim	Feldbeschreibung
ONTAP	ChapTargetInitiatorSecret	Schlüssel für CHAP-Zielinitiator. Erforderlich, wenn usCHAP=true verwendet wird. Für <code>ontap-san</code> und <code>ontap-san-economy</code>

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 Löschstaus.
  - 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 Löschezustand. 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	PHASE	STATUS	STORAGE DRIVER	BACKEND NAME	DELETION POLICY	BACKEND UUID
backend-tbc-ontap-san-bab2699e6ab8		Bound	Success	ontap-san		8d24fce7-6f60-4d4a-8ef6-
						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 `kubectl describe tbc <tbc-name> -n trident` ausführen oder `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 `tridentctl` durchführen.

### 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 wird weiterhin in einem „Deleting“ Zustand vorhanden sein und Trident wird weiterhin diese Volumes und Snapshots verwalten, bis sie gelöscht werden.

### Zeigen Sie die vorhandenen Back-Ends an

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

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

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

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

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

### Aktualisieren Sie ein Backend

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

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

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

```
tridentctl logs -n trident
```

Nachdem Sie das Problem mit der Konfigurationsdatei identifiziert und behoben haben, können Sie 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'

kubect1 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.

```

kubect1 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.template`, 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 Standard-Speicherklasse 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 PersistentVolume (PV) und ein PersistentVolumeClaim (PVC), das die konfigurierte Kubernetes StorageClass verwendet, um Zugriff auf das PV anzufordern. Anschließend können Sie das PV an einem Pod montieren.

### Überblick

Ein "*PersistentVolume*" (PV) ist eine physische Speicherressource, die vom Clusteradministrator auf einem Kubernetes-Cluster bereitgestellt wird. Die "*PersistentVolumeClaim*" (PVC) ist eine Anforderung für den Zugriff auf das PersistentVolume auf dem Cluster.

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

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

### Beispielmanifeste

#### PersistentVolume-Beispielmanifest

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

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

## PersistentVolumeClaim-Beispielmanifeste

Diese Beispiele zeigen grundlegende PVC-Konfigurationsoptionen.

### PVC mit RWO-Zugang

Dieses Beispiel zeigt 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: pv-storage
      persistentVolumeClaim:
        claimName: basic
  containers:
    - name: pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: pv-storage
```

## Grundlegende NVMe/TCP-Konfiguration

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

## Erstellen Sie das PV und die PVC

### Schritte

1. Erstellen Sie das PV.

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

2. Überprüfen Sie den PV-Status.

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

3. Erstellen Sie das PVC.

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

#### 4. Überprüfen Sie den PVC-Status.

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

#### 5. Mounten Sie das Volume in einem Pod.

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



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

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

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

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

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

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 erforderlichen Konfigurationen zum Erweitern von iSCSI- und NFS-Volumes.

### Erweitern Sie ein iSCSI-Volume

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



Die 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).

```

kubect1 get pvc
NAME          STATUS    VOLUME                                     CAPACITY
ACCESS MODES  STORAGECLASS  AGE
san-pvc      Bound     pvc-8a814d62-bd58-4253-b0d1-82f2885db671  1Gi
RWO          ontap-san    8s

kubect1 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.

```

kubect1 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:

```

kubect1 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
kubect1 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-Volumen

Trident unterstützt Volume-Erweiterung für NFS PVS, die auf, `ontap-nas-economy`, `ontap-nas-flexgroup`, `gcp-cvs` und `azure-netapp-files` Back-Ends bereitgestellt 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 `kubect1 edit storageclass` bearbeiten und die Volume-Erweiterung zulassen.



## 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:

```

kubect1 get pvc ontapnas20mb
NAME                STATUS      VOLUME
CAPACITY    ACCESS MODES   STORAGECLASS   AGE
ontapnas20mb    Bound        pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7    1Gi
RWO                ontapnas                4m44s

kubect1 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 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 volume` 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 im ONTAP Cluster. Das Importieren von FlexVols mit dem `ontap-nas` Treiber funktioniert gleich. Eine FlexVol, die bereits in einem ONTAP-Cluster vorhanden ist, kann 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>
```

PROTOCOL	NAME	BACKEND UUID	SIZE	STATE	STORAGE CLASS	MANAGED
file	pvc-bf5ad463-afbb-11e9-8d9f-5254004dfdb7	c5a6f6a4-b052-423b-80d4-8fb491a14a22	1.0 GiB	online	standard	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
```

PROTOCOL	NAME	BACKEND UUID	SIZE	STATE	STORAGE CLASS	MANAGED
file	pvc-df07d542-afbc-11e9-8d9f-5254004dfdb7	c5a6f6a4-b052-423b-80d4-8fb491a14a22	1.0 GiB	online	standard	false

## ONTAP SAN

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

Trident kann ONTAP SAN FlexVols importieren, die eine einzige LUN enthalten. Dies ist mit dem Treiber konsistent `ontap-san`, der für jede PVC und eine LUN in der FlexVol eine FlexVol erstellt. Trident importiert die FlexVol 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 in das Format und die LUN in der FlexVol in `lun0` um `pvc-<uuid>`.

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

```
tridentctl import volume ontapsan_san_default ontap-san-managed -f pvc-
basic-import.yaml -n trident -d
```

PROTOCOL	NAME	BACKEND UUID	SIZE	STATE	STORAGE CLASS	MANAGED
block	pvc-d6ee4f54-4e40-4454-92fd-d00fc228d74a	cd394786-ddd5-4470-adc3-10c5ce4ca757	20 MiB	online	basic	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
```

PROTOCOL	NAME	BACKEND UUID	SIZE	STATE	STORAGE CLASS	MANAGED
block	pvc-1fc999c9-ce8c-459c-82e4-ed4380a4b228	e3275890-7d80-4af6-90cc-c7a0759f555a	1.0 GiB	online	san-blog	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.

Vserver	Igroup	Protocol	OS Type	Initiators
svm0	k8s-nodename.example.com-fe5d36f2-cded-4f38-9eb0-c7719fc2f9f3	iscsi	linux	iqn.1994-05.com.redhat:4c2e1cf35e0
svm0	unmanaged-example-igroup	mixed	linux	iqn.1994-05.com.redhat:4c2e1cf35e0

### Element

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
```

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

### 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:/importvoll1`, ist der Volume-Pfad `importvoll1`.

### Beispiel: Azure NetApp Files

Das folgende Beispiel importiert ein `azure-netapp-files` Volume auf dem Backend `azurenetafiles_40517` mit dem Volume-Pfad `importvoll1`.

```
tridentctl import volume azurenetappfiles_40517 importvoll -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      |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```

## 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

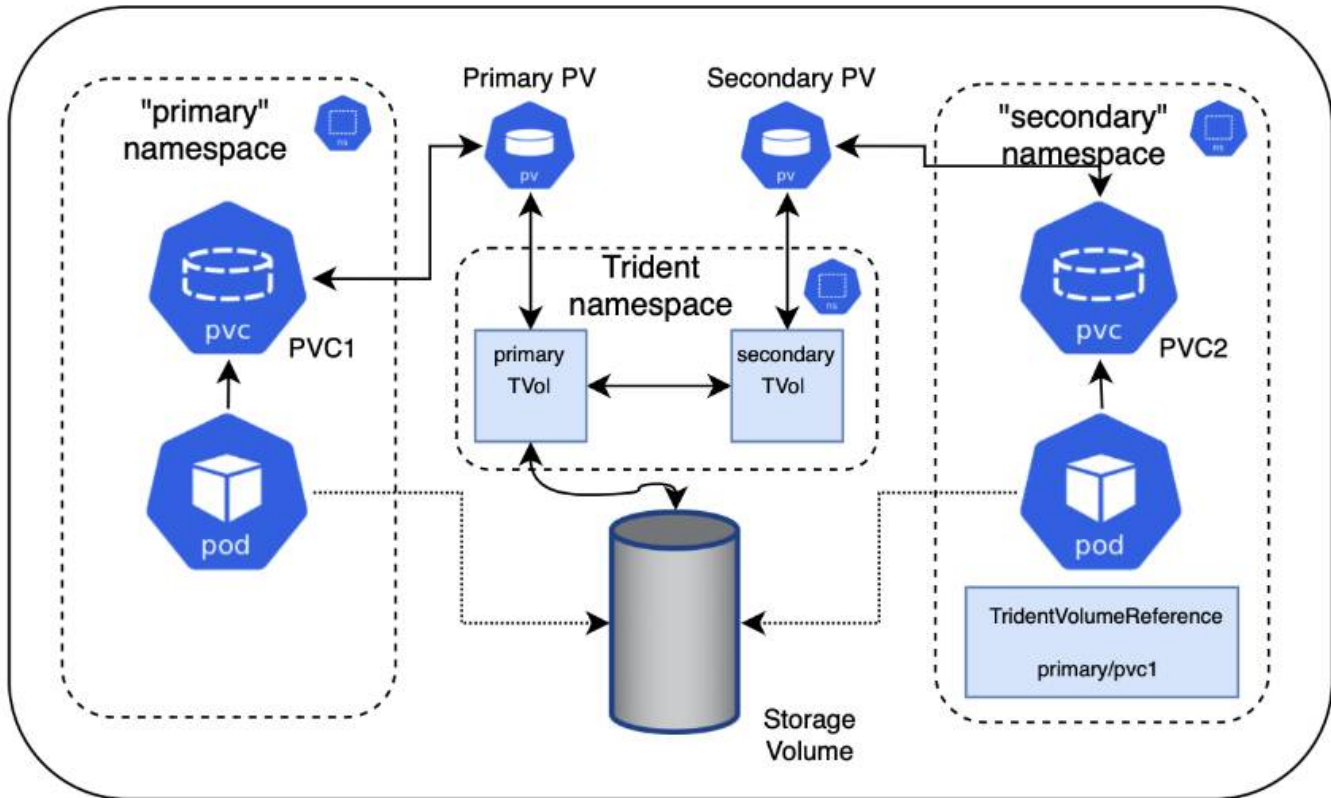
Mit Trident können Sie ein Volume in einem primären Namespace erstellen und es in einem oder mehreren sekundären Namespaces teilen.

### 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 Unterebene 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"](#).

### 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, Named Custom Resource Definition) verwenden, um die folgenden Vorgänge auszuführen:

- Erstellen von Spiegelbeziehungen zwischen Volumes (VES)
- Entfernen Sie Spiegelungsbeziehungen zwischen Volumes
- Brechen Sie die Spiegelbeziehungen auf
- Bewerben des sekundären Volumes bei Ausfällen (Failover)
- Verlustfreie Transition von Applikationen von Cluster zu Cluster (während geplanter Failover oder Migrationen)

## Replikationsvoraussetzungen

Stellen Sie sicher, dass die folgenden Voraussetzungen erfüllt sind, bevor Sie beginnen:

### ONTAP Cluster

- **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:"1e11e4a2108024935ecfcb2912226cedeaafd99df",
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:"1e11e4a2108024935ecfcb2912226cedeaafd99df",
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","kubernetes.io/arch":"amd64","kubernetes.io/hostname":"node1","kubernetes.io/os":"linux","node-role.kubernetes.io/master":"","topology.kubernetes.io/region":"us-east1","topology.kubernetes.io/zone":"us-east1-a"}]
[node2,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kubernetes.io/arch":"amd64","kubernetes.io/hostname":"node2","kubernetes.io/os":"linux","node-role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-east1","topology.kubernetes.io/zone":"us-east1-b"}]
[node3,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kubernetes.io/arch":"amd64","kubernetes.io/hostname":"node3","kubernetes.io/os":"linux","node-role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-east1","topology.kubernetes.io/zone":"us-east1-c"}]
```

## Schritt 1: Erstellen Sie ein Topologieorientiertes Backend

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

```
---
version: 1
storageDriverName: ontap-san
backendName: san-backend-us-east1
managementLIF: 192.168.27.5
svm: iscsi_svm
username: admin
password: password
supportedTopologies:
- topology.kubernetes.io/region: us-east1
  topology.kubernetes.io/zone: us-east1-a
- topology.kubernetes.io/region: us-east1
  topology.kubernetes.io/zone: us-east1-b
```

#### JSON

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "san-backend-us-east1",
  "managementLIF": "192.168.27.5",
  "svm": "iscsi_svm",
  "username": "admin",
  "password": "password",
  "supportedTopologies": [
    {"topology.kubernetes.io/region": "us-east1",
     "topology.kubernetes.io/zone": "us-east1-a"},
    {"topology.kubernetes.io/region": "us-east1",
     "topology.kubernetes.io/zone": "us-east1-b"}
  ]
}
```



`supportedTopologies` Wird verwendet, um eine Liste von Regionen und Zonen pro Backend bereitzustellen. Diese Regionen und Zonen stellen die Liste der zulässigen Werte dar, die in einer StorageClass bereitgestellt werden können. Bei StorageClasses, die 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:
name: netapp-san-us-east1
provisioner: csi.trident.netapp.io
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
- matchLabelExpressions:
- key: topology.kubernetes.io/zone
  values:
  - us-east1-a
  - us-east1-b
- key: topology.kubernetes.io/region
  values:
  - us-east1
parameters:
  fsType: "ext4"

```

In der oben angegebenen StorageClass-Definition `volumeBindingMode` ist auf festgelegt `WaitForFirstConsumer`. VES, die mit dieser StorageClass angefordert werden, werden erst dann gehandelt, wenn sie in einem Pod referenziert werden. Und `allowedTopologies` stellt die 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:
name: pvc-san
spec:
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: netapp-san-us-east1

```

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

```

kubect1 create -f pvc.yaml
persistentvolumeclaim/pvc-san created
kubect1 get pvc
NAME          STATUS      VOLUME      CAPACITY   ACCESS MODES   STORAGECLASS
AGE
pvc-san      Pending
2s
kubect1 describe pvc
Name:          pvc-san
Namespace:    default
StorageClass: netapp-san-us-east1
Status:       Pending
Volume:
Labels:       <none>
Annotations:  <none>
Finalizers:   [kubernetes.io/pvc-protection]
Capacity:
Access Modes:
VolumeMode:   Filesystem
Mounted By:   <none>
Events:
  Type      Reason              Age   From
  ----      -
Normal    WaitForFirstConsumer  6s    persistentvolume-controller
waiting
for first consumer to be created before binding

```

Verwenden Sie für Trident, ein Volume zu erstellen und es an die PVC zu binden, das in einem Pod verwendet wird. Das folgende Beispiel zeigt:

```

apiVersion: v1
kind: Pod
metadata:
  name: app-pod-1
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
          - matchExpressions:
              - key: topology.kubernetes.io/region
                operator: In
                values:
                  - us-east1
            preferredDuringSchedulingIgnoredDuringExecution:
              - weight: 1
                preference:
                  matchExpressions:
                    - key: topology.kubernetes.io/zone
                      operator: In
                      values:
                        - us-east1-a
                        - us-east1-b
      securityContext:
        runAsUser: 1000
        runAsGroup: 3000
        fsGroup: 2000
    volumes:
      - name: voll
        persistentVolumeClaim:
          claimName: pvc-san
    containers:
      - name: sec-ctx-demo
        image: busybox
        command: [ "sh", "-c", "sleep 1h" ]
        volumeMounts:
          - name: voll
            mountPath: /data/demo
        securityContext:
          allowPrivilegeEscalation: false

```

Diese PodSpec 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

Volumen-Snapshot wird unterstützt von `ontap-nas`, `ontap-nas-flexgroup` `ontap-san` `ontap-san-economy` `solidfire-san`, `gcp-cvs` und `azure-netapp-files` Fahrer.

## 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 `Delete` oder `Retain` sein. Wenn auf `Retain` festgelegt wird, 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 `pvc1`, und der Name des Snapshots wird auf `pvc1-snap` festgelegt. 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-Volumen 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: pvcl-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 `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

```

- 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

```

- 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`.
- Jeder Benutzer:** Erstellen Sie ein `PersistentVolumeClaim`, um auf den neu zu 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
```

1. 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 Status der Snapshot-Wiederherstellung 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: pvcl
    volumeSnapshotName: pvcl-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

Wenn Sie ein persistentes Volume mit zugeordneten Snapshots löschen, wird das entsprechende Trident-Volume in einen „Löschzustand“ aktualisiert. Entfernen Sie die Volume-Snapshots, um das 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"](#)

# Management und Monitoring von Trident

## Upgrade von Trident

### Upgrade von Trident

Ab Version 24.02 folgt Trident einem viermonatigen Release-Intervall und liefert drei wichtige Releases pro Kalenderjahr. Jede neue Version baut auf den vorherigen Versionen auf und bietet neue Funktionen, Performance-Verbesserungen, Bug Fixes und Verbesserungen. Wir empfehlen Ihnen, mindestens einmal pro Jahr ein Upgrade durchzuführen, um von den neuen Funktionen von Trident zu profitieren.

### Überlegungen vor dem Upgrade

Beachten Sie beim Upgrade auf die neueste Version von Trident Folgendes:

- In allen Namespaces in einem Kubernetes-Cluster sollte nur eine Trident Instanz installiert werden.
- Trident 23.07 und höher erfordert v1-Volumen-Snapshots und unterstützt keine Alpha- oder Beta-Snapshots mehr.
- Wenn Sie Cloud Volumes Service für Google Cloud im erstellt "[CVS-Diensttyp](#)" haben, müssen Sie die Backend-Konfiguration aktualisieren, um den oder `zoneredundantstandardsw` Service-Level beim Upgrade von Trident 23.01 zu verwenden `standardsw`. Wenn das im Backend nicht aktualisiert `serviceLevel` wird, kann es zu einem Fehlschlagen der Volumes kommen. Weitere Informationen finden Sie unter "[Beispiele für CVS-Diensttypen](#)".
- Beim Upgrade ist es wichtig, dass Sie `StorageClasses` von Trident verwendet angeben `parameter.fsType`. Sie können löschen und neu erstellen `StorageClasses`, ohne bereits vorhandene Volumes zu unterbrechen.
  - Dies ist eine **Anforderung** für die Durchsetzung von "[Sicherheitskontexte](#)" SAN-Volumes.
  - Das Verzeichnis [sample input](#) enthält Beispiele wie `storage-class-basic.yaml.template` und Link:[https://github.com/NetApp/Trident/BLOB/Master/Trident-Installer/sample-input/Storage-class-Samples/default-storage-class-aml-^Bronze\[storage-class-bronze-default.yaml\]](https://github.com/NetApp/Trident/BLOB/Master/Trident-Installer/sample-input/Storage-class-Samples/default-storage-class-aml-^Bronze[storage-class-bronze-default.yaml]).
  - Weitere Informationen finden Sie unter "[Bekannt Probleme](#)".

### Schritt 1: Wählen Sie eine Version

Trident-Versionen folgen einer datumbasierten Namenskonvention `YY.MM`, wobei „YY“ die letzten beiden Ziffern des Jahres und „MM“ der Monat ist. Dot-Releases folgen einer `YY.MM.X` Konvention, wobei „X“ der Patch-Level ist. Sie wählen die Version, auf die Sie aktualisieren möchten, basierend auf der Version aus, von der Sie aktualisieren.

- Sie können ein direktes Upgrade auf jede Zielversion durchführen, die sich innerhalb eines Fensters mit vier Versionen Ihrer installierten Version befindet. Sie können beispielsweise direkt von 23.04 (oder einem beliebigen 23.04-Punkt-Release) auf 24.06 aktualisieren.
- Wenn Sie ein Upgrade von einer Version außerhalb des Fensters mit vier Releases durchführen, führen Sie ein Upgrade in mehreren Schritten durch. Verwenden Sie die Upgrade-Anweisungen für das, von dem "[Frühere Version](#)" Sie aktualisieren, um auf die neueste Version zu aktualisieren, die für das Fenster mit vier Versionen passt. Wenn Sie beispielsweise 22.01 verwenden und ein Upgrade auf 24.06 durchführen möchten:



- a. Erstes Upgrade von 22.07 auf 23.04.
- b. Dann Upgrade von 23.04 auf 24.06.



Wenn Sie ein Upgrade über den Trident-Operator auf der OpenShift Container Platform durchführen, sollten Sie auf Trident 21.01.1 oder höher aktualisieren. Der mit 21.01.0 veröffentlichte Trident-Operator enthält ein bekanntes Problem, das in 21.01.1 behoben wurde. Weitere Informationen finden Sie im ["Details zur Ausgabe auf GitHub"](#).

## Schritt 2: Bestimmen Sie die ursprüngliche Installationsmethode

So bestimmen Sie, welche Version Sie ursprünglich für die Installation von Trident verwendet haben:

1. Verwenden Sie, `kubectl get pods -n trident` um die Pods zu untersuchen.
  - Wenn kein Operator Pod vorhanden ist, wurde Trident mit installiert `tridentctl`.
  - Wenn es einen Operator-Pod gibt, wurde Trident entweder manuell oder über Helm mit dem Trident-Operator installiert.
2. Wenn ein Benutzer-POD vorhanden ist, verwenden Sie `kubectl describe torc`, um zu ermitteln, ob Trident mit Helm installiert wurde.
  - Wenn ein Helm-Label vorhanden ist, wurde Trident mit Helm installiert.
  - Wenn kein Helm-Etikett vorhanden ist, wurde Trident manuell mit dem Trident-Operator installiert.

## Schritt 3: Wählen Sie eine Upgrade-Methode

Im Allgemeinen sollten Sie mit der gleichen Methode aktualisieren, die Sie für die Erstinstallation verwendet haben, jedoch können Sie ["Wechseln Sie zwischen den Installationsmethoden"](#). Es gibt zwei Optionen für ein Upgrade von Trident.

- ["Upgrade über den Trident-Operator"](#)



Wir empfehlen Ihnen, die Überprüfung ["Den Upgrade-Workflow für Bediener verstehen"](#) durchzuführen, bevor Sie mit dem Betreiber ein Upgrade durchführen.

\*

## Upgrade mit dem Bediener

### Den Upgrade-Workflow für Bediener verstehen

Bevor Sie ein Upgrade von Trident mit dem Trident Operator durchführen, sollten Sie sich über die während des Upgrades auftretenden Hintergrundprozesse informieren. Dies umfasst Änderungen am Trident Controller, am Controller Pod und an Node-Pods sowie am Node-DemonSet, die Rolling-Updates ermöglichen.

### Bearbeitung von Trident Upgrades für Betreiber

Eine der vielen ["Vorteile der Verwendung des Trident-Bediener"](#) Installationen und Upgrades von Trident ist die automatische Handhabung von Trident- und Kubernetes-Objekten ohne Unterbrechung vorhandener gemountete Volumes. So kann Trident Upgrades ohne Ausfallzeiten oder ["Rollierende Updates"](#) Insbesondere kommuniziert der Trident Betreiber mit dem Kubernetes-Cluster, um:

- Löschen Sie die Trident Controller-Implementierung und den Node DemonSet und erstellen Sie sie neu.
- Ersetzen Sie den Trident Controller Pod und die Trident Node Pods durch neue Versionen.
  - Wenn ein Node nicht aktualisiert wird, verhindert dies nicht, dass die verbleibenden Nodes aktualisiert werden.
  - Nur Nodes mit einem laufenden Trident Node Pod können Volumes mounten.



Weitere Informationen zur Trident-Architektur auf dem Kubernetes-Cluster finden Sie unter ["Architektur von Trident"](#).

### Arbeitsablauf für die Benutzeraktualisierung

Wenn Sie ein Upgrade mit dem Trident Operator initiieren:

1. Der **Trident-Operator**:
  - a. Erkennt die aktuell installierte Version von Trident (Version  $n$ ).
  - b. Aktualisiert alle Kubernetes-Objekte einschließlich CRDs, RBAC und Trident SVC.
  - c. Löscht die Trident Controller-Bereitstellung für Version  $n$ .
  - d. Erstellt die Trident-Controller-Bereitstellung für Version  $n+1$ .
2. **Kubernetes** erstellt Trident Controller Pod für  $n+1$ .
3. Der **Trident-Operator**:
  - a. Löscht das Trident Node DemonSet für  $n$ . Der Operator wartet nicht auf die Beendigung des Node-Pod.
  - b. Erstellt den Trident Node Demonset für  $n+1$ .
4. **Kubernetes** erstellt Trident Node Pods auf Nodes, auf denen Trident Node Pod  $n$  nicht ausgeführt wird. So wird sichergestellt, dass auf einem Node nie mehr als ein Trident Node Pod einer beliebigen Version vorhanden ist.

### Aktualisieren Sie eine Trident-Installation mit Trident Operator oder Helm

Sie können Trident mit dem Trident-Operator entweder manuell oder mit Helm aktualisieren. Sie können von einer Trident-Bedienerinstallation auf eine andere Trident-Bedienerinstallation aktualisieren oder von einer Installation auf eine Trident-Bedienerversion aktualisieren `tridentctl`. Vor dem Upgrade einer Trident-Bedienerinstallation überprüfen ["Wählen Sie eine Aktualisierungsmethode aus"](#).

#### Aktualisieren einer manuellen Installation

Sie können von einer Installation eines Trident Operators mit Cluster-Umfang auf eine andere Installation eines Trident Operators mit Cluster-Umfang aktualisieren. Alle Trident-Versionen 21.01 und höher verwenden einen Clusteroperator.



Um ein Upgrade von Trident durchzuführen, das mit dem Namespace-Scoped-Operator (Versionen 20.07 bis 20.10) installiert wurde, verwenden Sie die Upgrade-Anweisungen für ["Ihre installierte Version"](#) von Trident.

### Über diese Aufgabe

Trident bietet eine Bundle-Datei, mit der Sie den Operator installieren und zugehörige Objekte für Ihre

Kubernetes-Version erstellen können.

- Verwenden Sie für Cluster mit Kubernetes 1.24 "[Bundle\\_pre\\_1\\_25.yaml](#)".
- Verwenden Sie für Cluster mit Kubernetes 1.25 oder höher "[Bundle\\_Post\\_1\\_25.yaml](#)".

### Bevor Sie beginnen

Stellen Sie sicher, dass Sie ein Kubernetes Cluster verwenden "[Eine unterstützte Kubernetes Version](#)", das ausgeführt wird.

### Schritte

1. Überprüfen Sie Ihre Trident-Version:

```
./tridentctl -n trident version
```

2. Löschen Sie den Trident-Operator, der zur Installation der aktuellen Trident-Instanz verwendet wurde. Wenn Sie beispielsweise ein Upgrade von 23.07 durchführen, führen Sie den folgenden Befehl aus:

```
kubectl delete -f 23.07.0/trident-installer/deploy/<bundle.yaml> -n trident
```

3. Wenn Sie Ihre Erstinstallation mithilfe von Attributen angepasst haben `TridentOrchestrator`, können Sie das Objekt bearbeiten `TridentOrchestrator`, um die Installationsparameter zu ändern. Dies kann auch Änderungen umfassen, die an der Angabe gespiegelter Trident- und CSI-Image-Register für den Offline-Modus vorgenommen wurden, Debug-Protokolle aktivieren oder Geheimnisse für die Bildausziehung angeben.
4. Installieren Sie Trident mit der richtigen YAML-Bundle-Datei für Ihre Umgebung, wobei `<bundle.yaml>` `bundle_pre_1_25.yaml` `bundle_post_1_25.yaml` auf Ihrer Kubernetes-Version basiert. Wenn Sie beispielsweise Trident 24.10 installieren, führen Sie den folgenden Befehl aus:

```
kubectl create -f 24.10.0/trident-installer/deploy/<bundle.yaml> -n trident
```

### Aktualisieren einer Helm-Installation

Sie können eine Trident Helm-Installation aktualisieren.



Wenn Sie ein Kubernetes-Cluster von 1.24 auf 1.25 oder höher aktualisieren, auf dem Trident installiert ist, müssen Sie `values.yaml` aktualisieren, um den `helm upgrade` Befehl auf `true` festzulegen `excludePodSecurityPolicy` oder hinzuzufügen `--set excludePodSecurityPolicy=true`, bevor Sie das Cluster aktualisieren können.

Wenn Sie Ihr Kubernetes-Cluster bereits von 1.24 auf 1.25 aktualisiert haben, ohne das Trident Helm zu aktualisieren, schlägt das Helm Upgrade fehl. Führen Sie die folgenden Schritte aus, damit das Ruder-Upgrade durchgeführt wird:

1. Installieren Sie das Helm-mapkubeapis Plugin von <https://github.com/helm/helm-mapkubeapis>.

2. Führen Sie einen Probelauf für die Trident-Version im Namespace durch, in dem Trident installiert ist. Hier werden die Ressourcen aufgelistet, die bereinigt werden.

```
helm mapkubeapis --dry-run trident --namespace trident
```

3. Führen Sie einen vollständigen Durchlauf mit Ruder durch, um die Bereinigung durchzuführen.

```
helm mapkubeapis trident --namespace trident
```

## Schritte

1. Wenn Sie "[Trident mit Helm installiert](#)", können Sie verwenden `helm upgrade trident netapp-trident/trident-operator --version 100.2410.0`, um ein Upgrade in einem Schritt. Wenn Sie den Helm Repo nicht hinzugefügt haben oder ihn nicht zum Upgrade verwenden können:
  - a. Laden Sie die neueste Trident-Version von "[Die Sektion Assets auf GitHub](#)"herunter.
  - b. Verwenden Sie den `helm upgrade` Befehl `where` zeigt die Version an `trident-operator-24.10.0.tgz`, auf die Sie aktualisieren möchten.

```
helm upgrade <name> trident-operator-24.10.0.tgz
```



Wenn Sie während der Erstinstallation benutzerdefinierte Optionen festlegen (z. B. `private`, gespiegelte Registrierungen für Trident- und CSI-Images angeben), fügen Sie den Befehl mit `--set` an `helm upgrade`, um sicherzustellen, dass diese Optionen im Aktualisierungsbefehl enthalten sind, andernfalls werden die Werte auf die Standardeinstellung zurückgesetzt.

2. Führen Sie aus `helm list`, um zu überprüfen, ob die Karte und die App-Version aktualisiert wurden. Ausführen `tridentctl logs`, um alle Debug-Meldungen zu überprüfen.

## Upgrade von einer `tridentctl` Installation auf einen Trident-Operator

Sie können von einer Installation aus auf die neueste Version des Trident-Bedieners aktualisieren `tridentctl`. Die vorhandenen Back-Ends und VES stehen automatisch zur Verfügung.



Bevor Sie zwischen den Installationsmethoden wechseln, lesen Sie "[Wechseln zwischen den Installationsmethoden](#)".

## Schritte

1. Laden Sie die neueste Trident Version herunter.

```
# Download the release required [24.10.0]
mkdir 24.10.0
cd 24.10.0
wget
https://github.com/NetApp/trident/releases/download/v24.10.0/trident-
installer-24.10.0.tar.gz
tar -xf trident-installer-24.10.0.tar.gz
cd trident-installer
```

2. Erstellen Sie die `tridentorchestrator` CRD aus dem Manifest.

```
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
```

3. Stellen Sie den Clusteroperator im selben Namespace bereit.

```
kubectl create -f deploy/<bundle-name.yaml>

serviceaccount/trident-operator created
clusterrole.rbac.authorization.k8s.io/trident-operator created
clusterrolebinding.rbac.authorization.k8s.io/trident-operator created
deployment.apps/trident-operator created
podsecuritypolicy.policy/tridentoperatorpods created

#Examine the pods in the Trident namespace
NAME                                READY   STATUS    RESTARTS   AGE
trident-controller-79df798bdc-m79dc 6/6     Running   0           150d
trident-node-linux-xrst8             2/2     Running   0           150d
trident-operator-5574dbbc68-nthjv    1/1     Running   0           1m30s
```

4. Erstellen Sie ein `TridentOrchestrator` CR für die Installation von Trident.

```

cat deploy/crds/tridentorchestrator_cr.yaml
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident

kubectl create -f deploy/crds/tridentorchestrator_cr.yaml

#Examine the pods in the Trident namespace
NAME                                READY   STATUS    RESTARTS   AGE
trident-csi-79df798bdc-m79dc        6/6    Running   0           1m
trident-csi-xrst8                    2/2    Running   0           1m
trident-operator-5574dbbc68-nthjv    1/1    Running   0           5m41s

```

5. Bestätigen Sie, dass das Upgrade von Trident auf die beabsichtigte Version durchgeführt wurde.

```

kubectl describe torc trident | grep Message -A 3

Message:          Trident installed
Namespace:        trident
Status:           Installed
Version:          v24.10.0

```

## Upgrade mit tridentctl

Sie können eine vorhandene Trident-Installation ganz einfach mit aktualisieren `tridentctl`.

### Über diese Aufgabe

Die Deinstallation und Neuinstallation von Trident dient als Upgrade. Wenn Sie Trident deinstallieren, werden die Persistent Volume Claim (PVC) und das Persistent Volume (PV), die von der Trident-Bereitstellung verwendet werden, nicht gelöscht. Bereits bereitgestellte PVS bleiben verfügbar, während Trident offline ist, und Trident stellt Volumes für alle PVCs bereit, die in der Zwischenzeit erstellt werden, nachdem sie wieder online sind.

### Bevor Sie beginnen

Überprüfen Sie ["Wählen Sie eine Aktualisierungsmethode aus"](#) vor dem Upgrade mit `tridentctl`.

### Schritte

1. Führen Sie den Deinstallationsbefehl in `tridentctl` aus, um alle mit Trident verbundenen Ressourcen mit Ausnahme der CRDs und zugehörigen Objekte zu entfernen.

```
./tridentctl uninstall -n <namespace>
```

2. Installieren Sie Trident neu. Siehe "[Installieren Sie Trident mit tridentctl](#)".



Unterbrechen Sie den Upgrade-Prozess nicht. Stellen Sie sicher, dass das Installationsprogramm bis zum Abschluss ausgeführt wird.

## Managen Sie Trident mit tridentctl

Im ist das "[Trident Installationspaket](#)" Befehlszeilendienstprogramm für den einfachen Zugriff auf Trident enthalten `tridentctl`. Kubernetes-Benutzer mit genügend Privileges können es verwenden, um Trident zu installieren oder den Namespace zu managen, der den Trident Pod enthält.

### Befehle und globale Alarmmeldungen

Sie können ausführen `tridentctl help`, um eine Liste der verfügbaren Befehle für `tridentctl` oder hängen Sie das Flag an `--help` einen beliebigen Befehl, um eine Liste der Optionen und Flags für diesen bestimmten Befehl zu erhalten.

```
tridentctl [command] [--optional-flag]
```

Das Dienstprogramm Trident `tridentctl` unterstützt die folgenden Befehle und Global Flags.

## Befehle

### **create**

Fügen Sie eine Ressource zu Trident hinzu.

### **delete**

Entfernen Sie eine oder mehrere Ressourcen aus Trident.

### **get**

Holen Sie sich eine oder mehrere Ressourcen von Trident.

### **help**

Hilfe zu jedem Befehl.

### **images**

Drucken Sie eine Tabelle der Container-Bilder, die Trident benötigt.

### **import**

Importieren Sie eine vorhandene Ressource in Trident.

### **install**

Installation Von Trident:

### **logs**

Drucken Sie die Protokolle aus Trident.

### **send**

Senden Sie eine Ressource von Trident.

### **uninstall**

Deinstallieren Sie Trident.

### **update**

Ändern Sie eine Ressource in Trident.

### **update backend state**

Vorübergehende Unterbrechung der Back-End-Vorgänge.

### **upgrade**

Aktualisieren Sie eine Ressource in Trident.

### **version**

Drucken Sie die Version von Trident.



## Globale Alarmmeldungen

### **-d, --debug**

Debug-Ausgabe.

### **-h, --help**

Hilfe für `tridentctl`.

### **-k, --kubeconfig string**

Geben Sie den Pfad an, über den Befehle lokal oder von einem Kubernetes-Cluster zu einem anderen ausgeführt werden `KUBECONFIG` sollen.



Alternativ können Sie die Variable exportieren `KUBECONFIG`, um auf ein bestimmtes Kubernetes-Cluster zu verweisen und Befehle an dieses Cluster auszugeben `tridentctl`.

### **-n, --namespace string**

Namespace der Trident-Implementierung:

### **-o, --output string**

Ausgabeformat. Einer von `json` `yaml`-Namen natürlich `Ärmellos` (Standard).

### **-s, --server string**

Adresse/Port der Trident REST-Schnittstelle.



Die Trident REST-Schnittstelle kann nur für die Wiedergabe unter `127.0.0.1` (für IPv4) oder `[: 1]` (für IPv6) konfiguriert werden.

## Befehloptionen und -Flags

### Erstellen

Verwenden Sie den `create` Befehl, um Trident eine Ressource hinzuzufügen.

```
tridentctl create [option]
```

### Optionen

`backend`: Fügen Sie ein Backend zu Trident.

### Löschen

Mit dem `delete` Befehl können Sie eine oder mehrere Ressourcen aus Trident entfernen.

```
tridentctl delete [option]
```

### Optionen

`backend`: Löschen Sie eine oder mehrere Speicher-Backends aus Trident.

`snapshot`: Löschen Sie einen oder mehrere Volume-Snapshots aus Trident.

`storageclass`: Löschen Sie eine oder mehrere Speicherklassen aus Trident.  
`volume`: Löschen eines oder mehrerer Speichervolumen aus Trident.

## Get

Mit dem `get` Befehl rufen Sie eine oder mehrere Ressourcen von Trident ab.

```
tridentctl get [option]
```

## Optionen

`backend`: Holen Sie sich ein oder mehrere Speicher-Backends von Trident.  
`snapshot`: Holen Sie sich einen oder mehrere Schnappschüsse von Trident.  
`storageclass`: Holen Sie sich eine oder mehrere Speicherklassen von Trident.  
`volume`: Holen Sie sich einen oder mehrere Bände von Trident.

## Flags

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

## Bilder

Verwenden Sie `images` Markierungen, um eine Tabelle der Container-Bilder zu drucken, die Trident benötigt.

```
tridentctl images [flags]
```

## Flags

`-h, --help`: Hilfe für Bilder.  
`-v, --k8s-version string`: Semantische Version des Kubernetes-Clusters.

## Importvolumen

Importieren Sie ein vorhandenes Volume mit dem `import volume` Befehl in Trident.

```
tridentctl import volume <backendName> <volumeName> [flags]
```

## Aliase

`volume, v`

## Flags

`-f, --filename string`: Pfad zur YAML- oder JSON-PVC-Datei.  
`-h, --help`: Hilfe für Volumen.  
`--no-manage`: Erstellen Sie nur PV/PVC. Nehmen Sie kein Lifecycle Management für Volumen an.

## Installieren

Verwenden Sie die `install` Flags, um Trident zu installieren.

```
tridentctl install [flags]
```

## Flags

`--autosupport-image string`: Das Containerbild für die AutoSupport Telemetrie (Standard "NetApp/Trident AutoSupport:<current-version>").

`--autosupport-proxy string`: Adresse/Port eines Proxys zum Senden von AutoSupport Telemetrie.

`--enable-node-prep`: Versuch, benötigte Pakete auf Knoten zu installieren.

`--generate-custom-yaml`: Generieren Sie YAML-Dateien ohne etwas zu installieren.

`-h, --help`: Hilfe zur Installation.

`--http-request-timeout`: Das HTTP-Anforderungs-Timeout für die REST-API des Trident-Controllers überschreiben (Standard 1m30s).

`--image-registry string`: Adresse/Port einer internen Image-Registry.

`--k8s-timeout duration`: Das Timeout für alle Kubernetes-Operationen (Standard 3m0s).

`--kubelet-dir string`: Der Host-Speicherort des internen Status von kubelet (Default "/var/lib/kubelet").

`--log-format string`: Das Trident-Logging-Format (Text, json) (Standard "Text").

`--node-prep`: Ermöglicht Trident, die Knoten des Kubernetes-Clusters vorzubereiten, um Volumes mit dem angegebenen Datenspeicherprotokoll zu verwalten. **Derzeit `iscsi` wird nur der Wert unterstützt.**

`--pv string`: der Name des von Trident verwendeten Legacy-PV stellt sicher, dass dies nicht existiert (Standard "Trident").

`--pvc string`: Der Name des von Trident verwendeten Legacy-PVC stellt sicher, dass dies nicht existiert (Standard "Trident").

`--silence-autosupport`: Senden Sie AutoSupport-Pakete nicht automatisch an NetApp (Standard TRUE).

`--silent`: Deaktivieren Sie die meisten Ausgaben während der Installation.

`--trident-image string`: Das zu installierende Trident-Image.

`--use-custom-yaml`: Verwenden Sie alle vorhandenen YAML-Dateien, die im Setup-Verzeichnis vorhanden sind.

`--use-ipv6`: Verwenden Sie IPv6 für die Kommunikation von Trident.

## Protokolle

Verwenden Sie `logs` Markierungen, um die Protokolle aus Trident zu drucken.

```
tridentctl logs [flags]
```

## Flags

`-a, --archive`: Erstellen Sie ein Support-Archiv mit allen Protokollen, sofern nicht anders angegeben.

`-h, --help`: Hilfe für Protokolle.

`-l, --log string`: Trident-Protokoll zur Anzeige. Eine von Trident/Trident-Operator/alle (Standard „Auto“).

`--node string`: Der Name des Kubernetes-Knotens, von dem aus die POD-Protokolle des Knotens erfasst werden.

`-p, --previous`: Holen Sie sich die Protokolle für die vorherige Container-Instanz, wenn sie existiert.

`--sidecars`: Holen Sie sich die Protokolle für die Beiwagen-Container.

## Senden

Verwenden Sie den `send` Befehl, um eine Ressource von Trident zu senden.

```
tridentctl send [option]
```

## Optionen

`autosupport`: Senden Sie ein AutoSupport-Archiv an NetApp.

## Deinstallieren

Verwenden Sie `uninstall` Flags, um Trident zu deinstallieren.

```
tridentctl uninstall [flags]
```

## Flags

`-h, --help`: Hilfe zur Deinstallation.

`--silent`: Deaktivieren Sie die meisten Ausgaben während der Deinstallation.

## Aktualisierung

Verwenden Sie den `update` Befehl, um eine Ressource in Trident zu ändern.

```
tridentctl update [option]
```

## Optionen

`backend`: Aktualisieren Sie ein Backend in Trident.

## Back-End-Status aktualisieren

Verwenden Sie den `update backend state` Befehl, um die Back-End-Vorgänge anzuhalten oder fortzusetzen.

```
tridentctl update backend state <backend-name> [flag]
```

## Zu berücksichtigende Aspekte

- Wenn ein Backend mit einem TridentBackendConfig (tbc) erstellt wird, kann das Backend nicht mit einer Datei aktualisiert werden `backend.json`.
- Wenn der `userState` in einem tbc gesetzt wurde, kann er nicht mit dem Befehl geändert werden `tridentctl update backend state <backend-name> --user-state suspended/normal`.
- Um die Möglichkeit, das via `tridentctl` nach dem Setzen über tbc wieder einzustellen `userState`, muss das Feld aus dem tbc `userState` entfernt werden. Dies kann mit dem Befehl erfolgen `kubectl edit tbc`. Nachdem das `userState` Feld entfernt wurde, können Sie mit dem `tridentctl update backend state` Befehl das eines Backends ändern `userState`.
- Verwenden Sie die `tridentctl update backend state`, um die zu ändern `userState`. Sie können auch die Using- oder -Datei aktualisieren `userState TridentBackendConfig backend.json`; dies löst eine vollständige Neuinitialisierung des Backends aus und kann zeitaufwändig sein.

## Flags

`-h, --help`: Hilfe für Backend-Status.

`--user-state`: Auf Pause gesetzt `suspended`. Legen Sie fest `normal`, um die Back-End-Vorgänge fortzusetzen. Wenn eingestellt auf `suspended`:

- `AddVolume` Und `Import Volume` werden angehalten.
- `CloneVolume`, `ResizeVolume`, `PublishVolume`, `UnPublishVolume`, `CreateSnapshot`,

GetSnapshot RestoreSnapshot, , DeleteSnapshot, RemoveVolume, GetVolumeExternal, ReconcileNodeAccess verfügbar bleiben.

Sie können den Backend-Status auch über das Feld in der Backend-Konfigurationsdatei oder aktualisieren `userState TridentBackendConfig backend.json`. Weitere Informationen finden Sie unter ["Optionen für das Management von Back-Ends"](#) und ["Führen Sie das Back-End-Management mit kubectI durch"](#).

**Beispiel:**

## JSON

Führen Sie die folgenden Schritte aus, um die mit der Datei zu aktualisieren `userState backend.json` :

1. Bearbeiten Sie die `backend.json` Datei, um das Feld mit dem Wert „suspendiert“ aufzunehmen `userState`.
2. Aktualisieren Sie das Backend mit dem `tridentctl backend update` Befehl und dem Pfad zur aktualisierten `backend.json` Datei.

**Beispiel:** `tridentctl backend update -f /<path to backend JSON file>/backend.json`

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "<redacted>",
  "svm": "nas-svm",
  "backendName": "customBackend",
  "username": "<redacted>",
  "password": "<redacted>",
  "userState": "suspended",
}
```

## YAML

Sie können den `tbc` bearbeiten, nachdem er angewendet wurde, indem Sie den Befehl verwenden `kubectl edit <tbc-name> -n <namespace>`. Im folgenden Beispiel wird der Back-End-Status mit der Option zum Anhalten aktualisiert `userState: suspended`:

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-ontap-nas
spec:
  version: 1
  backendName: customBackend
  storageDriverName: ontap-nas
  managementLIF: <redacted>
  svm: nas-svm
userState: suspended
credentials:
  name: backend-tbc-ontap-nas-secret
```

## Version

Verwenden Sie `version` Flags, um die Version von und den laufenden Trident-Dienst zu drucken `tridentctl`.

```
tridentctl version [flags]
```

## Flags

- `--client`: Nur Client-Version (kein Server erforderlich).
- `-h`, `--help`: Hilfe zur Version.

## Plug-in-Unterstützung

Tridentctl unterstützt Plugins ähnlich wie kubect. Tridentctl erkennt ein Plugin, wenn der binäre Dateiname des Plugins dem Schema "tridentctl-<plugin>" folgt, und die Binärdatei befindet sich in einem Ordner, der die Umgebungsvariable `PATH` aufführt. Alle erkannten Plugins sind im Plugin-Abschnitt der `tridentctl`-Hilfe aufgeführt. Optional können Sie die Suche auch einschränken, indem Sie in der Environment-Variable `TRIDENTCTL_PLUGIN_PATH` einen `PLUGIN`-Ordner angeben (Beispiel: `TRIDENTCTL_PLUGIN_PATH=~/.tridentctl-plugins/`). Wenn die Variable verwendet wird, sucht `tridentctl` nur im angegebenen Ordner.

# Monitoring von Trident

Trident bietet eine Reihe von Prometheus Kennzahlen-Endpunkten zur Überwachung der Trident-Performance.

## Überblick

Mit den von Trident bereitgestellten Metriken können Sie Folgendes tun:

- Überwachen Sie den Zustand und die Konfiguration von Trident. Sie können prüfen, wie erfolgreich Vorgänge sind und ob sie wie erwartet mit den Back-Ends kommunizieren können.
- Untersuchen Sie die Back-End-Nutzungsinformationen und erfahren Sie, wie viele Volumes auf einem Back-End bereitgestellt werden, sowie den belegten Speicherplatz usw.
- Erstellt eine Zuordnung der Anzahl von Volumes, die über verfügbare Back-Ends bereitgestellt werden.
- Verfolgen Sie die Leistung. Hier sehen Sie, wie lange Trident benötigt, um mit Back-Ends zu kommunizieren und Vorgänge auszuführen.



Standardmäßig sind die Trident-Kennzahlen auf dem Zielport am `/metrics` Endpunkt sichtbar `8001`. Diese Metriken sind bei der Installation von Trident standardmäßig aktiviert.

## Was Sie benötigen

- Ein Kubernetes-Cluster mit installiertem Trident
- Eine Prometheus Instanz. Dies kann ein sein "[Implementierung von Container-Prometheus](#)", oder Sie können wählen, Prometheus als ausführen "[Native Applikation](#)".

## Schritt 1: Definieren Sie ein Prometheus-Ziel

Sie sollten ein Prometheus Ziel definieren, um die Kennzahlen zu erfassen und Informationen über die von

Trident gemanagten Back-Ends, die erstellten Volumes usw. zu erhalten. Dies ["Blog"](#) erklärt, wie Sie Prometheus und Grafana mit Trident verwenden können, um Metriken abzurufen. Im Blog erfahren Sie, wie Sie Prometheus als Betreiber in Ihrem Kubernetes-Cluster ausführen und einen Service Monitor erstellen können, um Trident-Kennzahlen zu erhalten.

## Schritt: Erstellen Sie einen Prometheus ServiceMonitor

Um die Trident-Kennzahlen zu nutzen, sollten Sie einen Prometheus ServiceMonitor erstellen, der den Service überwacht `trident-csi` und den Port abhört `metrics`. Ein Beispiel für ServiceMonitor sieht so aus:

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: trident-sm
  namespace: monitoring
  labels:
    release: prom-operator
spec:
  jobLabel: trident
  selector:
    matchLabels:
      app: controller.csi.trident.netapp.io
  namespaceSelector:
    matchNames:
      - trident
  endpoints:
    - port: metrics
      interval: 15s
```

Diese ServiceMonitor-Definition ruft vom Dienst zurückgegebene Kennzahlen `trident-csi` ab und sucht gezielt nach dem `metrics` Endpunkt des Dienstes. Daher ist Prometheus jetzt so konfiguriert, dass es die Kennzahlen von Trident versteht.

Zusätzlich zu den direkt aus Trident verfügbaren Kennzahlen legt Kubelet viele `kubelet_volume_*` Metriken über seinen eigenen Endpunkt für Kennzahlen dar. Kubelet kann Informationen über verbundene Volumes bereitstellen und Pods und andere interne Vorgänge, die er übernimmt. Siehe ["Hier"](#).

## Schritt 3: Abfrage der Trident-Kennzahlen mit PromQL

PromQL ist gut geeignet, um Ausdrücke zu erstellen, die Zeitreihen- oder tabellarische Daten zurückgeben.

Im Folgenden finden Sie einige PromQL-Abfragen, die Sie verwenden können:

### Abrufen des Integritätsinformationen zu Trident

- **Prozentsatz der HTTP 2XX-Antworten von Trident**



```
(sum (trident_rest_ops_seconds_total_count{status_code=~"2.."} OR on()  
vector(0)) / sum (trident_rest_ops_seconds_total_count)) * 100
```

- **Prozentsatz der REST-Antworten von Trident über Statuscode**

```
(sum (trident_rest_ops_seconds_total_count) by (status_code) / scalar  
(sum (trident_rest_ops_seconds_total_count))) * 100
```

- **Durchschnittliche Dauer in ms der von Trident durchgeführten Operationen**

```
sum by (operation)  
(trident_operation_duration_milliseconds_sum{success="true"}) / sum by  
(operation)  
(trident_operation_duration_milliseconds_count{success="true"})
```

## Holen Sie sich Trident-Nutzungsinformationen

- **Mittlere Volumengröße**

```
trident_volume_allocated_bytes/trident_volume_count
```

- **Gesamter Volume-Speicherplatz, der von jedem Backend bereitgestellt wird**

```
sum (trident_volume_allocated_bytes) by (backend_uuid)
```

## Individuelle Volume-Nutzung



Dies ist nur aktiviert, wenn auch kubelet-Kennzahlen gesammelt werden.

- **Prozentsatz des verwendeten Speicherplatzes für jedes Volumen**

```
kubelet_volume_stats_used_bytes / kubelet_volume_stats_capacity_bytes *  
100
```

## Informationen zur Trident AutoSupport Telemetrie

Standardmäßig sendet Trident im täglichen Rhythmus Prometheus Kennzahlen und grundlegende Back-End-Informationen an NetApp.

- Um zu verhindern, dass Trident Prometheus-Metriken und grundlegende Backend-Informationen an NetApp sendet, übergeben Sie das `--silence-autosupport` Flag während der Trident-Installation.

- Trident kann auch Container-Protokolle an NetApp-Support On-Demand senden über `tridentctl send autosupport`. Sie müssen Trident auslösen, um die Protokolle hochzuladen. Bevor Sie Protokolle senden, sollten Sie NetApp's akzeptieren <https://www.netapp.com/company/legal/privacy-policy/> ["datenschutzrichtlinie"].
- Sofern nicht angegeben, ruft Trident die Protokolle der letzten 24 Stunden ab.
- Sie können den Zeitrahmen für die Protokollaufbewahrung mit dem Flag angeben `--since`. Zum Beispiel: `tridentctl send autosupport --since=1h`. Diese Informationen werden gesammelt und über einen Container gesendet `trident-autosupport`, der zusammen mit Trident installiert wird. Sie können das Container-Bild unter abrufen "[Trident AutoSupport](#)".
- Trident AutoSupport erfasst oder übermittelt keine personenbezogenen Daten oder personenbezogenen Daten. Sie wird mit einem geliefert "[EULA](#)", das sich nicht für das Trident Container-Image selbst eignet. Weitere Informationen zum Engagement von NetApp für Datensicherheit und Vertrauen finden "[Hier](#)" Sie hier.

Ein Beispiel für eine Nutzlast, die von Trident gesendet wird, sieht wie folgt aus:

```
---
items:
- backendUUID: ff3852e1-18a5-4df4-b2d3-f59f829627ed
  protocol: file
  config:
    version: 1
    storageDriverName: ontap-nas
    debug: false
    debugTraceFlags:
    disableDelete: false
    serialNumbers:
    - nwkvzfanek_SN
    limitVolumeSize: ''
  state: online
  online: true
```

- Die AutoSupport Meldungen werden an den AutoSupport Endpunkt von NetApp gesendet. Wenn Sie eine private Registrierung zum Speichern von Container-Images verwenden, können Sie das Flag verwenden `--image-registry`.
- Sie können auch Proxy-URLs konfigurieren, indem Sie die Installation YAML-Dateien erstellen. Dies kann getan werden, indem `tridentctl install --generate-custom-yaml` Sie die YAML-Dateien erstellen und das Argument für den `trident-autosupport` Container in `trident-deployment.yaml` hinzufügen `--proxy-url`.

## Deaktivieren Sie Trident-Kennzahlen

Um **die Meldung von**-Metriken zu deaktivieren, sollten Sie benutzerdefinierte YAMLS (mit dem Flag) generieren `--generate-custom-yaml` und diese bearbeiten, um das Flag für den `trident-main` Container zu entfernen `--metrics`.

# Deinstallieren Sie Trident

Sie sollten dieselbe Methode verwenden, um Trident zu deinstallieren, die Sie bei der Installation von Trident verwendet haben.

## Über diese Aufgabe

- Wenn Sie nach einem Upgrade, Abhängigkeitsproblemen oder einem nicht erfolgreichen oder unvollständigen Upgrade eine Korrektur für Fehler benötigen, sollten Sie Trident deinstallieren und die frühere Version mithilfe der entsprechenden Anweisungen neu installieren "[Version](#)". Dies ist die einzige empfohlene Möglichkeit, *Downgrade* auf eine frühere Version zu übertragen.
- Für eine einfache Aktualisierung und Neuinstallation entfernt die Deinstallation von Trident nicht die von Trident erstellten CRDs oder zugehörigen Objekte. Wenn Sie vollständig entfernen müssen Trident und alle seine Daten, siehe "[Entfernen Sie Trident und CRDs vollständig](#)".

## Bevor Sie beginnen

Wenn Sie Kubernetes-Cluster ausmustern, müssen Sie alle Applikationen löschen, die Volumes verwenden, die von Trident erstellt wurden, bevor Sie sie deinstallieren. Dadurch wird sichergestellt, dass PVCs auf Kubernetes-Nodes nicht veröffentlicht werden, bevor sie gelöscht werden.

## Bestimmen Sie die ursprüngliche Installationsmethode

Sie sollten dieselbe Methode verwenden, um Trident zu deinstallieren, die Sie bei der Installation verwendet haben. Überprüfen Sie vor der Deinstallation, welche Version Sie ursprünglich für die Installation von Trident verwendet haben.

1. Verwenden Sie, `kubectl get pods -n trident` um die Pods zu untersuchen.
  - Wenn kein Operator Pod vorhanden ist, wurde Trident mit installiert `tridentctl`.
  - Wenn es einen Operator-Pod gibt, wurde Trident entweder manuell oder über Helm mit dem Trident-Operator installiert.
2. Wenn ein Benutzer-POD vorhanden ist, verwenden Sie `kubectl describe tproc trident`, um zu ermitteln, ob Trident mit Helm installiert wurde.
  - Wenn ein Helm-Label vorhanden ist, wurde Trident mit Helm installiert.
  - Wenn kein Helm-Etikett vorhanden ist, wurde Trident manuell mit dem Trident-Operator installiert.

## Deinstallieren Sie die Installation eines Trident-Operators

Sie können die Installation eines Dreizack-Bediensers manuell oder mithilfe von Helm deinstallieren.

### Deinstallieren Sie die manuelle Installation

Wenn Sie Trident mit dem Operator installiert haben, können Sie es deinstallieren, indem Sie einen der folgenden Schritte ausführen:

1. **CR bearbeiten `TridentOrchestrator` und das Deinstallationsflag einstellen:**

```
kubectl patch torc <trident-orchestrator-name> --type=merge -p
'{"spec":{"uninstall":true}}'
```

Wenn das `uninstall` Flag auf gesetzt ist `true`, deinstalliert der Trident-Operator Trident, entfernt aber nicht den `TridentOrchestrator` selbst. Sie sollten den `TridentOrchestrator` aufräumen und einen neuen erstellen, wenn Sie Trident erneut installieren möchten.

2. **Löschen `TridentOrchestrator`:** Durch Entfernen des `TridentOrchestrator` CR, der zum Bereitstellen von Trident verwendet wurde, weisen Sie den Bediener an, Trident zu deinstallieren. Der Bediener verarbeitet die Entfernung von `TridentOrchestrator` Trident Deployment und `daemonset` und entfernt die Trident-Pods, die er im Rahmen der Installation erstellt hatte.

```
kubectl delete -f deploy/<bundle.yaml> -n <namespace>
```

## Deinstallieren Sie Helm-Installation

Wenn Sie Trident mit Helm installiert haben, können Sie es mit `helm uninstall` deinstallieren.

```
#List the Helm release corresponding to the Trident install.
helm ls -n trident
NAME                NAMESPACE      REVISION      UPDATED
STATUS              CHART           APP VERSION
trident             trident         1             2021-04-20
00:26:42.417764794 +0000 UTC deployed      trident-operator-21.07.1
21.07.1

#Uninstall Helm release to remove Trident
helm uninstall trident -n trident
release "trident" uninstalled
```

## Deinstallieren Sie eine `tridentctl` Installation

Verwenden Sie den `uninstall` Befehl in `tridentctl`, um alle mit Trident verbundenen Ressourcen mit Ausnahme der CRDs und zugehörigen Objekte zu entfernen:

```
./tridentctl uninstall -n <namespace>
```

# Trident für Docker

## Voraussetzungen für die Bereitstellung

Sie müssen die erforderlichen Protokollvoraussetzungen auf Ihrem Host installieren und konfigurieren, bevor Sie Trident bereitstellen können.

### Überprüfen Sie die Anforderungen

- Stellen Sie sicher, dass Ihre Bereitstellung alle erfüllt "[Anforderungen](#)".
- Vergewissern Sie sich, dass eine unterstützte Version von Docker installiert ist. Wenn Ihre Docker-Version veraltet ist, "[Installieren oder aktualisieren Sie sie](#)".

```
docker --version
```

- Stellen Sie sicher, dass die Protokollvoraussetzungen auf Ihrem Host installiert und konfiguriert sind.

### NFS Tools

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

#### 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-Tools

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

## RHEL 8 ODER HÖHER

1. Installieren Sie die folgenden Systempakete:

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

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

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

3. Scannen auf manuell einstellen:

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

4. Multipathing aktivieren:

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



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

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

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

6. Aktivieren und starten `iscsi`:

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

## Ubuntu

1. Installieren Sie die folgenden Systempakete:

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

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

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

### 3. Scannen auf manuell einstellen:

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

### 4. Multipathing aktivieren:

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



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
```

## NVMe-Tools

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
```

## Implementieren Sie Trident

Trident für Docker bietet eine direkte Integration in das Docker Ecosystem für NetApp Storage-Plattformen. Die Plattform unterstützt auch das Provisioning und Management von Storage-Ressourcen – von der Storage-Plattform bis hin zu Docker Hosts – mit einem Framework für zukünftige zusätzliche Plattformen.

Auf demselben Host können mehrere Instanzen von Trident gleichzeitig ausgeführt werden. Dies ermöglicht simultane Verbindungen zu mehreren Storage-Systemen und Storage-Typen und kann den für die Docker Volumes verwendeten Storage angepasst werden.

### Was Sie benötigen

Siehe "[Voraussetzungen für die Bereitstellung](#)". Nachdem Sie die Voraussetzungen erfüllt haben, können Sie Trident bereitstellen.

## Docker Managed Plug-in-Methode (Version 1.13/17.03 und höher)

### Bevor Sie beginnen



Wenn Sie Trident vor Docker 1.13/17.03 in der herkömmlichen Daemon-Methode verwendet haben, stellen Sie sicher, dass Sie den Trident-Prozess stoppen und den Docker-Daemon neu starten, bevor Sie die Managed Plugin-Methode verwenden.

1. Beenden Sie alle laufenden Instanzen:

```
killall /usr/local/bin/netappdvp
killall /usr/local/bin/trident
```

2. Docker Neu Starten.

```
systemctl restart docker
```



3. Vergewissern Sie sich, dass Docker Engine 17.03 (neu 1.13) oder höher installiert ist.

```
docker --version
```

Wenn Ihre Version veraltet ist, ["Installieren oder aktualisieren Sie Ihre Installation"](#).

## Schritte

1. Erstellen Sie eine Konfigurationsdatei und geben Sie die Optionen wie folgt an:

- `config`: Der Standarddateiname ist `config.json`, Sie können jedoch jeden beliebigen Namen verwenden, indem Sie die Option mit dem Dateinamen angeben `config`. Die Konfigurationsdatei muss sich im Verzeichnis auf dem Hostsystem befinden `/etc/netappdvp`.
- `log-level`: Geben Sie die Protokollierungsebene (`debug`, `info`, `warn` `error` `fatal`) an. Der Standardwert ist `info`.
- `debug`: Geben Sie an, ob Debug-Protokollierung aktiviert ist. Die Standardeinstellung lautet `false`. Überschreibt die Protokollebene, wenn wahr.
  - i. Speicherort für die Konfigurationsdatei erstellen:

```
sudo mkdir -p /etc/netappdvp
```

ii. Konfigurationsdatei erstellen:

```
cat << EOF > /etc/netappdvp/config.json
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1"
}
EOF
```

2. Starten Sie Trident mit dem verwalteten Plugin-System. Ersetzen Sie `<version>` diese durch die von Ihnen verwendete Plugin-Version (`xxx.xx.x`).

```
docker plugin install --grant-all-permissions --alias netapp
netapp/trident-plugin:<version> config=myConfigFile.json
```

3. Beginnen Sie mit der Verwendung von Trident, um Storage aus dem konfigurierten System zu nutzen.

- a. Erstellen Sie ein Volume mit dem Namen „FirstVolume“:

```
docker volume create -d netapp --name firstVolume
```

- b. Erstellen Sie ein Standardvolume beim Starten des Containers:

```
docker run --rm -it --volume-driver netapp --volume  
secondVolume:/my_vol alpine ash
```

- c. Entfernen Sie den Datenträger „FirstVolume“:

```
docker volume rm firstVolume
```

## Herkömmliche Methode (Version 1.12 oder früher)

### Bevor Sie beginnen

1. Stellen Sie sicher, dass Sie Docker Version 1.10 oder höher haben.

```
docker --version
```

Wenn Ihre Version veraltet ist, aktualisieren Sie Ihre Installation.

```
curl -fsSL https://get.docker.com/ | sh
```

Oder, "[Befolgen Sie die Anweisungen für Ihre Distribution](#)".

2. Stellen Sie sicher, dass NFS und/oder iSCSI für Ihr System konfiguriert ist.

### Schritte

1. NetApp Docker Volume Plug-in installieren und konfigurieren:

- a. Laden Sie die Anwendung herunter und entpacken Sie sie:

```
wget  
https://github.com/NetApp/trident/releases/download/v24.10.0/trident-  
installer-24.10.0.tar.gz  
tar xzf trident-installer-24.10.0.tar.gz
```

- b. Verschieben Sie zu einer Position im bin-Pfad:

```
sudo mv trident-installer/extras/bin/trident /usr/local/bin/  
sudo chown root:root /usr/local/bin/trident  
sudo chmod 755 /usr/local/bin/trident
```

c. Speicherort für die Konfigurationsdatei erstellen:

```
sudo mkdir -p /etc/netappdvp
```

d. Konfigurationsdatei erstellen:

```
cat << EOF > /etc/netappdvp/ontap-nas.json  
{  
  "version": 1,  
  "storageDriverName": "ontap-nas",  
  "managementLIF": "10.0.0.1",  
  "dataLIF": "10.0.0.2",  
  "svm": "svm_nfs",  
  "username": "vsadmin",  
  "password": "password",  
  "aggregate": "aggr1"  
}  
EOF
```

2. Nachdem Sie die Binärdatei platziert und die Konfigurationsdatei erstellt haben, starten Sie den Trident-Daemon mit der gewünschten Konfigurationsdatei.

```
sudo trident --config=/etc/netappdvp/ontap-nas.json
```



Sofern nicht angegeben, lautet der Standardname für den Volume-Treiber „NetApp“.

Nachdem der Daemon gestartet wurde, können Sie Volumes mithilfe der Docker CLI-Schnittstelle erstellen und verwalten

3. Volume erstellen:

```
docker volume create -d netapp --name trident_1
```

4. Bereitstellung eines Docker Volumes beim Starten eines Containers:

```
docker run --rm -it --volume-driver netapp --volume trident_2:/my_vol  
alpine ash
```

## 5. Entfernen eines Docker Volumes:

```
docker volume rm trident_1
docker volume rm trident_2
```

## Starten Sie Trident beim Systemstart

Eine Beispieldatei für systemd basierte Systeme finden Sie unter `contrib/trident.service.example` im Git repo. Gehen Sie wie folgt vor, um die Datei mit RHEL zu verwenden:

### 1. Kopieren Sie die Datei an den richtigen Speicherort.

Sie sollten eindeutige Namen für die Einheitendateien verwenden, wenn mehr als eine Instanz ausgeführt wird.

```
cp contrib/trident.service.example
/usr/lib/systemd/system/trident.service
```

### 2. Bearbeiten Sie die Datei, ändern Sie die Beschreibung (Zeile 2) entsprechend dem Treibernamen und dem Konfigurationspfad (Zeile 9), um Ihre Umgebung zu berücksichtigen.

### 3. Systemd neu laden, damit sie Änderungen aufnehmen kann:

```
systemctl daemon-reload
```

### 4. Aktivieren Sie den Service.

Dieser Name hängt davon ab, wie Sie die Datei im Verzeichnis benannt `/usr/lib/systemd/system` haben.

```
systemctl enable trident
```

### 5. Starten Sie den Service.

```
systemctl start trident
```

### 6. Den -Status anzeigen.

```
systemctl status trident
```



Führen Sie jedes Mal, wenn Sie die Einheitendatei ändern, den `systemctl daemon-reload` Befehl für sie aus, um die Änderungen zu beachten.

# Aktualisieren oder deinstallieren Sie Trident

Sie können ein Upgrade von Trident für Docker sicher durchführen, ohne die verwendeten Volumes zu beeinträchtigen. Während des Upgrade-Prozesses wird es einen kurzen Zeitraum geben, in dem `docker volume` Befehle, die auf das Plugin gerichtet sind, nicht erfolgreich sind, und Anwendungen können keine Volumes mounten, bis das Plugin wieder ausgeführt wird. Unter den meisten Umständen dauert das nur wenige Sekunden.

## Upgrade

Führen Sie die folgenden Schritte aus, um Trident für Docker zu aktualisieren.

### Schritte

1. Liste der vorhandenen Volumes:

```
docker volume ls
DRIVER          VOLUME NAME
netapp:latest   my_volume
```

2. Deaktivieren Sie das Plugin:

```
docker plugin disable -f netapp:latest
docker plugin ls
ID                NAME          DESCRIPTION
ENABLED
7067f39a5df5     netapp:latest nDVP - NetApp Docker Volume
Plugin           false
```

3. Upgrade des Plug-ins:

```
docker plugin upgrade --skip-remote-check --grant-all-permissions
netapp:latest netapp/trident-plugin:21.07
```



Die Version 18.01 von Trident ersetzt nDVP. Sie sollten direkt vom Bild auf das `netapp/trident-plugin` Bild upgraden `netapp/ndvp-plugin`.

4. Plug-in aktivieren:

```
docker plugin enable netapp:latest
```

5. Vergewissern Sie sich, dass das Plug-in aktiviert ist:

```
docker plugin ls
ID                NAME                DESCRIPTION
ENABLED
7067f39a5df5     netapp:latest       Trident - NetApp Docker Volume
Plugin    true
```

6. Vergewissern Sie sich, dass die Volumes sichtbar sind:

```
docker volume ls
DRIVER                VOLUME NAME
netapp:latest         my_volume
```



Wenn Sie ein Upgrade von einer alten Version von Trident (vor 20.10) auf Trident 20.10 oder höher durchführen, kann es zu einem Fehler kommen. Weitere Informationen finden Sie unter ["Bekannte Probleme"](#). Wenn Sie in den Fehler laufen, sollten Sie zuerst das Plugin deaktivieren, dann entfernen Sie das Plugin, und installieren Sie dann die erforderliche Trident-Version, indem Sie einen zusätzlichen Konfigurationsparameter übergeben: `docker plugin install netapp/trident-plugin:20.10 --alias netapp --grant-all-permissions config=config.json`

## Deinstallieren

Führen Sie die folgenden Schritte aus, um Trident für Docker zu deinstallieren.

### Schritte

1. Entfernen Sie alle Volumes, die das Plugin erstellt.
2. Deaktivieren Sie das Plugin:

```
docker plugin disable netapp:latest
docker plugin ls
ID                NAME                DESCRIPTION
ENABLED
7067f39a5df5     netapp:latest       nDVP - NetApp Docker Volume
Plugin    false
```

3. Entfernen Sie das Plugin:

```
docker plugin rm netapp:latest
```

## Arbeiten mit Volumes

Volumes können ganz einfach mit den Standardbefehlen erstellt, geklont und entfernt

werden. Der bei Bedarf angegebene Trident-Treibername wird dabei verwendet `docker volume`.

## Erstellen eines Volumes

- Erstellen Sie ein Volume mit einem Treiber unter Verwendung des Standardnamens:

```
docker volume create -d netapp --name firstVolume
```

- Volume mit einer bestimmten Trident-Instanz erstellen:

```
docker volume create -d ntap_bronze --name bronzeVolume
```



Wenn Sie keine angeben "**Optionen**", werden die Standardeinstellungen für den Treiber verwendet.

- Überschreiben Sie die Standard-Volume-Größe. Beachten Sie das folgende Beispiel, um ein 20 gib-Volume mit einem Treiber zu erstellen:

```
docker volume create -d netapp --name my_vol --opt size=20G
```



Die Volume-Größen werden als Strings angegeben, die einen ganzzahligen Wert mit optionalen Einheiten enthalten (Beispiel: 10G, 20GB, 3tib). Wenn keine Einheiten angegeben werden, ist der Standardwert G. Größeneinheiten können entweder als Kräfte von 2 (B, KiB, MiB, gib, tib) oder mit einer Leistung von 10 (B, KB, MB, GB, TB) angegeben werden. Auf Kurzschluss und Einheiten werden 2 Kräfte (G = gib, T = tib, ...) verwendet.

## Entfernen Sie ein Volume

- Entfernen Sie das Volume wie jedes andere Docker Volume:

```
docker volume rm firstVolume
```



Bei der Verwendung des `solidfire-san` Treibers löscht und löscht das obige Beispiel das Volume.

Führen Sie die folgenden Schritte aus, um Trident für Docker zu aktualisieren.

## Klonen Sie ein Volume

Bei Verwendung der `ontap-nas`, `ontap-san` `solidfire-san` und `gcp-cvs storage drivers`, Trident können Volumes klonen. Wenn Sie die oder `ontap-nas-economy`-Treiber verwenden `ontap-nas-flexgroup`, wird das Klonen nicht unterstützt. Wenn Sie ein neues Volume von einem vorhandenen Volume

erstellen, wird ein neuer Snapshot erstellt.

- Überprüfen Sie das Volume, um die Snapshots aufzuzählen:

```
docker volume inspect <volume_name>
```

- Erstellen Sie ein neues Volume von einem vorhandenen Volume aus. Dadurch wird ein neuer Snapshot erstellt:

```
docker volume create -d <driver_name> --name <new_name> -o  
from=<source_docker_volume>
```

- Erstellen Sie ein neues Volume anhand eines vorhandenen Snapshots auf einem Volume. Dadurch wird kein neuer Snapshot erstellt:

```
docker volume create -d <driver_name> --name <new_name> -o  
from=<source_docker_volume> -o fromSnapshot=<source_snap_name>
```

## Beispiel



```

docker volume inspect firstVolume

[
  {
    "Driver": "ontap-nas",
    "Labels": null,
    "Mountpoint": "/var/lib/docker-volumes/ontap-
nas/netappdvp_firstVolume",
    "Name": "firstVolume",
    "Options": {},
    "Scope": "global",
    "Status": {
      "Snapshots": [
        {
          "Created": "2017-02-10T19:05:00Z",
          "Name": "hourly.2017-02-10_1505"
        }
      ]
    }
  }
]

docker volume create -d ontap-nas --name clonedVolume -o from=firstVolume
clonedVolume

docker volume rm clonedVolume
docker volume create -d ontap-nas --name volFromSnap -o from=firstVolume
-o fromSnapshot=hourly.2017-02-10_1505
volFromSnap

docker volume rm volFromSnap

```

## Zugriff auf extern erstellte Volumes

Sie können über Trident \* nur\* auf extern erstellte Blockgeräte (oder deren Clones) zugreifen, wenn sie keine Partitionen haben und wenn ihr Dateisystem von Trident unterstützt wird (z.B.: Ein ext4-formatiertes /dev/sdc1 wird nicht über Trident zugänglich sein).

## Treiberspezifische Volume-Optionen

Jeder Storage-Treiber verfügt über unterschiedliche Optionen, die Sie bei der Volume-Erstellung angeben können, um das Ergebnis anzupassen. Unter finden Sie weitere Optionen, die für Ihr konfiguriertes Storage-System gelten.

Die Verwendung dieser Optionen während der Erstellung des Volumes ist einfach. Geben Sie die Option und den Wert an, den der Operator während des CLI-Betriebs verwendet `-o`. Diese überschreiben alle

gleichwertigen Werte aus der JSON-Konfigurationsdatei.

## ONTAP Volume-Optionen

Bei der Erstellung von Volumes für NFS und iSCSI sind folgende Optionen enthalten:

Option	Beschreibung
size	Die Größe des Volumes beträgt standardmäßig 1 gib.
spaceReserve	Thin oder Thick Provisioning stellen das Volume bereit. Die Standardeinstellung ist „Thin“. Gültige Werte sind <code>none</code> (Thin Provisioning) und (Thick Provisioning) <code>volume</code> .
snapshotPolicy	Dadurch wird die Snapshot-Richtlinie auf den gewünschten Wert eingestellt. Der Standardwert ist <code>none</code> , was bedeutet, dass keine Snapshots automatisch für das Volume erstellt werden. Sofern nicht von Ihrem Speicheradministrator geändert, existiert eine Richtlinie namens „Standard“ auf allen ONTAP Systemen, die sechs stündliche, zwei tägliche und zwei wöchentliche Schnapschüsse erzeugt und speichert. Die Daten, die in einem Snapshot erhalten bleiben, können wiederhergestellt werden, indem Sie in das Verzeichnis in einem beliebigen Verzeichnis auf dem Volume navigieren <code>.snapshot</code> .
snapshotReserve	Dadurch wird die Snapshot-Reserve auf den gewünschten Prozentsatz eingestellt. Der Standardwert ist kein Wert, was bedeutet, dass ONTAP die Snapshot Reserve (in der Regel 5%) auswählen wird, wenn Sie eine Snapshot Policy ausgewählt haben, oder 0%, wenn die Snapshot Policy keine ist. Sie können den Standardwert von <code>snapshotReserve</code> in der Konfigurationsdatei für alle ONTAP-Back-Ends setzen und es als Option zur Erstellung von Volumes für alle ONTAP-Back-Ends außer <code>ontap-nas-Economy</code> verwenden.
splitOnClone	Beim Klonen eines Volume wird dadurch ONTAP den Klon sofort von seinem übergeordneten Volume aufteilen. Der Standardwert ist <code>false</code> . Einige Anwendungsfälle für das Klonen von Volumes werden am besten bedient, indem der Klon unmittelbar nach der Erstellung von seinem übergeordneten Volume aufgeteilt wird, da sich die Storage-Effizienz wahrscheinlich nicht erhöhen wird. Das Klonen einer leeren Datenbank spart beispielsweise viel Zeit, spart aber nur wenig Storage. Daher ist es am besten, den Klon sofort zu teilen.

Option	Beschreibung
encryption	<p>Aktivieren Sie NetApp Volume Encryption (NVE) auf dem neuen Volume, Standardeinstellung ist <code>false</code>. NVE muss im Cluster lizenziert und aktiviert sein, damit diese Option verwendet werden kann.</p> <p>Wenn auf dem Backend NAE aktiviert ist, wird jedes in Trident bereitgestellte Volume NAE aktiviert.</p> <p>Weitere Informationen finden Sie unter <a href="#">"Funktionsweise von Trident mit NVE und NAE"</a>.</p>
tieringPolicy	<p>Legt die Tiering-Richtlinie fest, die für das Volume verwendet werden soll. Damit wird entschieden, ob Daten auf die Cloud-Tier verschoben werden, wenn sie inaktiv sind (kalte).</p>

Die folgenden zusätzlichen Optionen sind nur für NFS\* verfügbar:

Option	Beschreibung
unixPermissions	<p>Dadurch wird der Berechtigungssatz für das Volume selbst festgelegt. Standardmäßig werden die Berechtigungen auf, oder in numerischer Notation <code>0755</code> gesetzt. <code>---</code><code>rw</code><code>xr</code><code>-xr</code><code>-x</code> und <code>root</code> sind der Eigentümer. Das Text- oder Zahlenformat funktioniert.</p>
snapshotDir	<p>Wenn Sie diese Einstellung auf <code>true</code> einstellen, wird das <code>.snapshot</code> Verzeichnis für Clients sichtbar, die auf das Volume zugreifen. Der Standardwert ist <code>false</code>, was bedeutet, dass die Sichtbarkeit des <code>.snapshot</code> Verzeichnisses standardmäßig deaktiviert ist. Einige Bilder, zum Beispiel das offizielle MySQL-Image, funktionieren nicht wie erwartet, wenn das <code>.snapshot</code> Verzeichnis sichtbar ist.</p>
exportPolicy	<p>Legt die Exportrichtlinie fest, die für das Volume verwendet werden soll. Der Standardwert ist <code>default</code>.</p>
securityStyle	<p>Legt den Sicherheitsstil für den Zugriff auf das Volume fest. Der Standardwert ist <code>unix</code>. Gültige Werte sind <code>unix</code> und <code>mixed</code>.</p>

Die folgenden zusätzlichen Optionen sind für iSCSI **nur**:

Option	Beschreibung
fileSystemType	Legt das Dateisystem fest, das zum Formatieren von iSCSI-Volumes verwendet wird. Der Standardwert ist <code>ext4</code> . Gültige Werte sind <code>ext3</code> , <code>ext4</code> und <code>xfs</code> .
spaceAllocation	Wenn Sie diese Einstellung auf <code>false</code> setzen, wird die Funktion zur Speicherplatzzuweisung der LUN deaktiviert. Der Standardwert ist <code>true</code> , was bedeutet, dass ONTAP den Host benachrichtigt, wenn der Speicherplatz des Volume knapp ist und die LUN im Volume keine Schreibvorgänge akzeptieren kann. Mit dieser Option kann ONTAP auch automatisch Speicherplatz freigeben, wenn der Host Daten löscht.

### Beispiele

Sehen Sie sich die folgenden Beispiele an:

- 10 gib-Volume erstellen:

```
docker volume create -d netapp --name demo -o size=10G -o
encryption=true
```

- Erstellen Sie ein 100 gib Volume mit Snapshots:

```
docker volume create -d netapp --name demo -o size=100G -o
snapshotPolicy=default -o snapshotReserve=10
```

- Erstellen Sie ein Volume, bei dem das `setuid`-Bit aktiviert ist:

```
docker volume create -d netapp --name demo -o unixPermissions=4755
```

Die minimale Volume-Größe beträgt 20 MiB.

Wenn die Snapshot-Reserve nicht angegeben wird und die Snapshot-Policy ist `none`, verwenden Trident eine Snapshot-Reserve von 0%.

- Erstellung eines Volumes ohne Snapshot-Richtlinie und ohne Snapshot-Reserve:

```
docker volume create -d netapp --name my_vol --opt snapshotPolicy=none
```

- Erstellen Sie ein Volume ohne Snapshot-Richtlinie und eine individuelle Snapshot-Reserve von 10 %:

```
docker volume create -d netapp --name my_vol --opt snapshotPolicy=none
--opt snapshotReserve=10
```

- Erstellen Sie ein Volume mit einer Snapshot-Richtlinie und einer individuellen Snapshot-Reserve von 10 %:

```
docker volume create -d netapp --name my_vol --opt
snapshotPolicy=myPolicy --opt snapshotReserve=10
```

- Erstellen Sie ein Volume mit einer Snapshot-Richtlinie, und akzeptieren Sie die standardmäßige Snapshot-Reserve von ONTAP (normalerweise 5%):

```
docker volume create -d netapp --name my_vol --opt
snapshotPolicy=myPolicy
```

## Element Software-Volume-Optionen

Die Element Softwareoptionen bieten Zugriff auf die Größe und Quality of Service (QoS)-Richtlinien für das Volume. Beim Erstellen des Volume wird die damit verbundene QoS-Richtlinie mithilfe der Nomenklatur angegeben `-o type=service_level`.

Der erste Schritt bei der Definition eines QoS-Service-Levels mit Element driver besteht darin, mindestens einen Typ zu erstellen und die minimalen, maximalen und Burst-IOPS anzugeben, die mit einem Namen in der Konfigurationsdatei verbunden sind.

Darüber anderem sind bei Volumes für Element Software folgende Optionen verfügbar:

Option	Beschreibung
size	Die Größe des Volumens, standardmäßig 1gib oder Konfigurationseintrag ... "Default": {"size": "5G"}.
blocksize	Verwenden Sie entweder 512 oder 4096, standardmäßig 512 oder den Konfigurationseintrag StandardBlockSize.

## Beispiel

In der folgenden Beispielkonfigurationsdatei finden Sie QoS-Definitionen:

```

{
  "...": "...",
  "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
      }
    }
  ]
}

```

In der obigen Konfiguration haben wir drei Richtliniendefinitionen: Bronze, Silver und Gold. Diese Namen sind frei wählbar.

- Erstellen eines 10 gib Gold-Volumes:

```
docker volume create -d solidfire --name sfGold -o type=Gold -o size=10G
```

- Erstellen eines 100 gib Bronze-Volumens:

```
docker volume create -d solidfire --name sfBronze -o type=Bronze -o
size=100G
```

# Sammelt Protokolle

Sie können Protokolle erfassen, um Hilfe bei der Fehlerbehebung zu erhalten. Die Methode zur Erfassung der Protokolle variiert je nach Ausführung des Docker Plug-ins.

## Sammelt Protokolle für die Fehlerbehebung

### Schritte

1. Wenn Sie Trident mit der empfohlenen Managed-Plugin-Methode ausführen (z. B. mit `docker plugin` Befehlen), zeigen Sie diese wie folgt an:

```
docker plugin ls
ID                NAME                DESCRIPTION
ENABLED
4fb97d2b956b     netapp:latest      nDVP - NetApp Docker Volume
Plugin           false
journalctl -u docker | grep 4fb97d2b956b
```

Die Standardprotokollierungsebene sollte Ihnen die Diagnose der meisten Probleme ermöglichen. Wenn Sie feststellen, dass das nicht genug ist, können Sie Debug-Protokollierung aktivieren.

2. Um die Debug-Protokollierung zu aktivieren, installieren Sie das Plugin mit aktivierter Debug-Protokollierung:

```
docker plugin install netapp/trident-plugin:<version> --alias <alias>
debug=true
```

Oder aktivieren Sie Debug-Protokollierung, wenn das Plugin bereits installiert ist:

```
docker plugin disable <plugin>
docker plugin set <plugin> debug=true
docker plugin enable <plugin>
```

3. Wenn Sie die Binärdatei selbst auf dem Host ausführen, sind Protokolle im Verzeichnis des Hosts verfügbar `/var/log/netappdvp`. Geben Sie zum Aktivieren der Debug-Protokollierung an `-debug`, wann Sie das Plug-in ausführen.

## Allgemeine Tipps zur Fehlerbehebung

- Das häufigste Problem, in dem neue Benutzer auftreten, ist eine fehlerhafte Konfiguration, die verhindert, dass das Plugin initialisiert wird. Wenn dies geschieht, werden Sie wahrscheinlich eine Meldung wie diese sehen, wenn Sie versuchen, das Plugin zu installieren oder zu aktivieren:

```
Error response from daemon: dial unix /run/docker/plugins/<id>/netapp.sock:
connect: no such file or directory
```

Das bedeutet, dass das Plugin nicht gestartet werden konnte. Zum Glück wurde das Plugin mit einer umfassenden Protokollierungsfunktion aufgebaut, die Ihnen bei der Diagnose der meisten Probleme helfen sollte, die Sie wahrscheinlich auftreten.

- Wenn Probleme bei der Montage eines PV an einem Container auftreten, stellen Sie sicher, dass `rpcbind` installiert und ausgeführt wird. Verwenden Sie den erforderlichen Paketmanager für das Host-Betriebssystem, und überprüfen Sie, ob `rpcbind` ausgeführt wird. Sie können den Status des `rpcbind`-Dienstes überprüfen, indem Sie einen oder dessen Äquivalent ausführen `systemctl status rpcbind`.

## Management mehrerer Trident Instanzen

Wenn mehrere Storage-Konfigurationen gleichzeitig verfügbar sind, sind mehrere Instanzen von Trident erforderlich. Der Schlüssel zu mehreren Instanzen besteht darin, ihnen unterschiedliche Namen zu geben, indem sie die Option mit dem Container-Plug-in verwenden `--alias`, oder `--volume-driver` die Option beim Instanzieren von Trident auf dem Host.

### Schritte für Docker Managed Plug-in (Version 1.13/17.03 oder höher)

1. Starten Sie die erste Instanz, die einen Alias und eine Konfigurationsdatei angibt.

```
docker plugin install --grant-all-permissions --alias silver
netapp/trident-plugin:21.07 config=silver.json
```

2. Starten Sie die zweite Instanz, indem Sie einen anderen Alias und eine andere Konfigurationsdatei angeben.

```
docker plugin install --grant-all-permissions --alias gold
netapp/trident-plugin:21.07 config=gold.json
```

3. Erstellen Sie Volumes, die den Alias als Treibernamen angeben.

Beispiel für Gold Volume:

```
docker volume create -d gold --name ntapGold
```

Beispiel für Silbervolumen:

```
docker volume create -d silver --name ntapSilver
```

### Schritte für herkömmliche (Version 1.12 oder früher)

1. Starten Sie das Plug-in mit einer NFS-Konfiguration unter Verwendung einer benutzerdefinierten Treiber-ID:



```
sudo trident --volume-driver=netapp-nas --config=/path/to/config
-nfs.json
```

2. Starten Sie das Plug-in mit einer iSCSI-Konfiguration unter Verwendung einer benutzerdefinierten Treiber-ID:

```
sudo trident --volume-driver=netapp-san --config=/path/to/config
-iscsi.json
```

3. Stellen Sie Docker Volumes für jede Treiberinstanz bereit:

Zum Beispiel für NFS:

```
docker volume create -d netapp-nas --name my_nfs_vol
```

Beispiel für iSCSI:

```
docker volume create -d netapp-san --name my_iscsi_vol
```

## Optionen für die Storage-Konfiguration

Sehen Sie sich die Konfigurationsoptionen an, die für Ihre Trident Konfigurationen verfügbar sind.

### Globale Konfigurationsoptionen

Diese Konfigurationsoptionen sind für alle Trident-Konfigurationen anwendbar, unabhängig davon, welche Storage-Plattform genutzt wird.

Option	Beschreibung	Beispiel
version	Versionsnummer der Konfigurationsdatei	1
storageDriverName	Name des Speichertreibers	ontap-nas, ontap-san, ontap-nas-economy, ontap-nas-flexgroup, solidfire-san
storagePrefix	Optionales Präfix für Volume-Namen Standard: netappdvp_.	staging_

Option	Beschreibung	Beispiel
<code>limitVolumeSize</code>	Optionale Einschränkung von Volume-Größen. Standard: „“ (nicht erzwungen)	10g



Verwenden Sie (einschließlich der Standardeinstellung) keine `storagePrefix` Back-Ends für Elemente. Standardmäßig ignoriert der `solidfire-san` Treiber diese Einstellung und verwendet kein Präfix. Wir empfehlen die Verwendung einer bestimmten TenantID für die Docker Volume-Zuordnung oder die Verwendung der Attributdaten, die mit der Docker-Version, den Treiber-Informationen und dem Raw-Namen aus Docker gefüllt sind, in Fällen, in denen Namensnennung verwendet wurde.

Es stehen Standardoptionen zur Verfügung, damit Sie sie nicht für jedes erstellte Volume angeben müssen. Die `size` Option ist für alle Controller-Typen verfügbar. Im Abschnitt zur ONTAP-Konfiguration finden Sie ein Beispiel dafür, wie Sie die Standard-Volume-Größe festlegen.

Option	Beschreibung	Beispiel
<code>size</code>	Optionale Standardgröße für neue Volumes. Standard: 1G	10G

## ONTAP-Konfiguration

Zusätzlich zu den oben genannten globalen Konfigurationswerten stehen bei Verwendung von ONTAP folgende Optionen auf oberster Ebene zur Verfügung.

Option	Beschreibung	Beispiel
<code>managementLIF</code>	IP-Adresse des ONTAP Management LIF. Sie können einen vollqualifizierten Domännennamen (FQDN) angeben.	10.0.0.1

Option	Beschreibung	Beispiel
dataLIF	<p>IP-Adresse des LIF-Protokolls.</p> <p><b>ONTAP NAS Treiber:</b> Wir empfehlen die Angabe <code>dataLIF</code>. Falls nicht bereitgestellt, ruft Trident die Daten-LIFs von der SVM ab. Sie können einen vollständig qualifizierten Domännennamen (FQDN) angeben, der für die NFS-Mount-Vorgänge verwendet werden soll. Damit können Sie ein Round-Robin-DNS zum Load-Balancing über mehrere Daten-LIFs erstellen.</p> <p><b>ONTAP-SAN-Treiber:</b> Geben Sie nicht für iSCSI an. Trident verwendet "<b>ONTAP selektive LUN-Zuordnung</b>", um die für die Einrichtung einer Multi-Path-Sitzung erforderlichen iSCSI LIFs zu ermitteln. Eine Warnung wird erzeugt, wenn <code>dataLIF</code> explizit definiert ist.</p>	10.0.0.2
svm	Storage Virtual Machine zu verwenden (erforderlich, falls Management LIF eine Cluster-LIF ist)	svm_nfs
username	Benutzername zur Verbindung mit dem Speichergerät	vsadmin
password	Passwort für die Verbindung mit dem Speichergerät	secret
aggregate	Aggregat für die Bereitstellung (optional, wenn eingestellt, muss der SVM zugewiesen werden) Für den <code>ontap-nas-flexgroup</code> Treiber wird diese Option ignoriert. Zur Bereitstellung eines FlexGroup Volumes werden alle der SVM zugewiesenen Aggregate verwendet.	aggr1
limitAggregateUsage	Optionale, fail-Provisioning-Funktion, wenn die Nutzung über diesem Prozentsatz liegt	75%

Option	Beschreibung	Beispiel
nfsMountOptions	Feingranulare Steuerung der NFS-Mount-Optionen; standardmäßig „-o nfsvers=3“. <b>Nur verfügbar für die ontap-nas und ontap-nas-economy Fahrer.</b> <a href="#">"Siehe Informationen zur NFS-Host-Konfiguration hier"</a> .	-o nfsvers=4
igroupName	Trident erstellt und verwaltet pro Node igroups wie netappdvp.  Dieser Wert kann nicht geändert oder weggelassen werden.  <b>Nur für den Fahrer verfügbar ontap-san.</b>	netappdvp
limitVolumeSize	Maximale anforderbare Volume-Größe.	300g
qtreesPerFlexvol	Maximale Anzahl der qtrees pro FlexVol, die im Bereich [50, 300] liegen müssen, die Standardeinstellung ist 200.  <b>Für den ontap-nas-economy Treiber ermöglicht diese Option die Anpassung der maximalen Anzahl von qtrees pro FlexVol.</b>	300
sanType	<b>Nur für Treiber unterstützt ontap-san.</b> Verwenden Sie diese Option, um für iSCSI, nvme für NVMe/TCP oder fcp für SCSI über Fibre Channel (FC) auszuwählen iscsi. <b>'fcp' (SCSI over FC) ist ein Tech Preview Feature in der Trident 24.10 Version.</b>	iscsi Falls leer
limitVolumePoolSize	* ontap-san-economy `ontap-san-economy` Nur für und Treiber unterstützt.* Begrenzung der FlexVol-Größe bei den wirtschaftlichen Faktoren ONTAP ONTAP-nas und ONTAP-SAN	300g

Es stehen Standardoptionen zur Verfügung, um zu vermeiden, dass sie auf jedem von Ihnen erstellten Volume angegeben werden müssen:

Option	Beschreibung	Beispiel
spaceReserve	Modus für Speicherplatzreservierung <code>none</code> (Thin Provisioning) oder (Thick) <code>volume</code>	<code>none</code>
snapshotPolicy	Zu verwendende Snapshot-Richtlinie, Standard ist <code>none</code>	<code>none</code>
snapshotReserve	Der Prozentsatz der Snapshot-Reserve ist standardmäßig „“, um den ONTAP-Standardwert zu akzeptieren	10
splitOnClone	Teilen Sie einen Klon bei der Erstellung von seinem übergeordneten Element auf. Dies ist standardmäßig der Standardwert <code>false</code>	<code>false</code>
encryption	Aktiviert NetApp Volume Encryption (NVE) auf dem neuen Volume; Standardeinstellung ist <code>false</code> . 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 <a href="#">"Funktionsweise von Trident mit NVE und NAE"</a> .	Richtig
unixPermissions	NAS-Option für bereitgestellte NFS-Volumes, standardmäßig auf <code>777</code>	<code>777</code>
snapshotDir	NAS-Option für den Zugriff auf das <code>.snapshot</code> Verzeichnis.	„Wahr“ für NFSv4 „falsch“ für NFSv3
exportPolicy	NAS-Option für die zu verwendende NFS-Exportrichtlinie, standardmäßig auf <code>default</code>	<code>default</code>
securityStyle	NAS-Option für Zugriff auf das bereitgestellte NFS-Volume.  NFS-Unterstützung <code>mixed</code> und <code>unix</code> -Sicherheitsstile. Der Standardwert ist <code>unix</code> .	<code>unix</code>
fileSystemType	SAN-Option zum Auswählen des Dateisystemtyps, standardmäßig auf <code>ext4</code>	<code>xf</code> s
tieringPolicy	Zu verwendende Tiering Policy, Standard ist <code>none</code> ; <code>snapshot-only</code> für pre-ONTAP 9.5 SVM-DR Konfiguration	<code>none</code>

## Skalierungsoptionen

Die `ontap-nas` Treiber und `ontap-san` erstellen eine ONTAP FlexVol für jedes Docker Volume. ONTAP unterstützt bis zu 1000 FlexVols pro Cluster Node mit einem Cluster maximal 12,000 FlexVols. Wenn die Anforderungen für Docker Volumes dieser Einschränkung entsprechen, ist der `ontap-nas` Treiber aufgrund der zusätzlichen Funktionen von FlexVols wie granulare Docker Volume-Snapshots und Klonen die bevorzugte NAS-Lösung.

Wenn Sie mehr Docker Volumes benötigen, als durch die FlexVol-Beschränkungen unterstützt werden können, wählen Sie den oder den `ontap-san-economy` Treiber aus `ontap-nas-economy`.

Der `ontap-nas-economy` Treiber erstellt Docker Volumes als ONTAP qtrees innerhalb eines Pools von automatisch gemanagten FlexVols. Qtrees bieten eine wesentlich größere Skalierung – bis zu 100,000 pro Cluster-Node und 2,400,000 pro Cluster – zu Lasten einiger Funktionen. `ontap-nas-economy` Snapshots oder Klonen mit granularem Docker Volume werden vom Treiber nicht unterstützt.



Der `ontap-nas-economy` Treiber wird derzeit in Docker Swarm nicht unterstützt, da Swarm die Volume-Erstellung nicht über mehrere Nodes hinweg orchestriert.

Der `ontap-san-economy` Treiber erstellt Docker Volumes als ONTAP LUNs in einem gemeinsamen Pool von automatisch gemanagten FlexVols. Somit ist jede FlexVol nicht auf nur eine LUN beschränkt und bietet eine bessere Skalierbarkeit für SAN-Workloads. Je nach Storage Array unterstützt ONTAP bis zu 16384 LUNs pro Cluster. Da es sich bei den Volumes um LUNs handelt, unterstützt dieser Treiber granulare Docker Snapshots und Klone.

Wählen Sie den `ontap-nas-flexgroup` Treiber, um die Parallelität zu einem einzelnen Volume zu erhöhen, das bis in den Petabyte-Bereich mit Milliarden von Dateien anwachsen kann. Zu den idealen Anwendungsfällen für FlexGroups gehören KI/ML/DL, Big Data und Analysen, Softwareentwicklung, Streaming, Datei-Repositorys und so weiter. Trident verwendet bei der Bereitstellung eines FlexGroup Volumes alle Aggregate, die einer SVM zugewiesen sind. Die Unterstützung von FlexGroup in Trident muss darüber hinaus Folgendes beachtet werden:

- ONTAP Version 9.2 oder höher erforderlich.
- Ab diesem Text unterstützt FlexGroups nur NFS v3.
- Empfohlen, die 64-Bit-NFSv3-IDs für die SVM zu aktivieren.
- Die empfohlene Mindestgröße für FlexGroup-Mitglieder/Volumes beträgt 100 gib.
- Klonen wird für FlexGroup Volumes nicht unterstützt.

Informationen zu FlexGroups und Workloads, die für FlexGroups geeignet sind, finden Sie unter "[Best Practices und Implementierungsleitfaden für NetApp FlexGroup Volumes](#)".

Um erweiterte Funktionen und umfassende Skalierbarkeit in derselben Umgebung zu erhalten, können Sie mehrere Instanzen des Docker Volume Plug-ins ausführen, wobei eine davon und eine andere `ontap-nas-economy` verwendet `ontap-nas` wird.

### Benutzerdefinierte ONTAP-Rolle für Trident

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"  
security login create -username <user_name\> -application http -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](#)"

## Beispiel für ONTAP-Konfigurationsdateien

### NFS-Beispiel für `ONTAP-`-Treiber

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "defaults": {
    "size": "10G",
    "spaceReserve": "none",
    "exportPolicy": "default"
  }
}
```

### NFS-Beispiel für `ONTAP-nas-FlexGroup`-Treiber

```
{
  "version": 1,
  "storageDriverName": "ontap-nas-flexgroup",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "defaults": {
    "size": "100G",
    "spaceReserve": "none",
    "exportPolicy": "default"
  }
}
```



### NFS Beispiel für `-`-Treiber für `ONTAP-nas-economy`

```
{
  "version": 1,
  "storageDriverName": "ontap-nas-economy",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1"
}
```

### ISCSI-Beispiel für `ONTAP-`-Treiber

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.3",
  "svm": "svm_iscsi",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "igroupName": "netappdvp"
}
```

### NFS Beispiel für `-`-Treiber für `ONTAP-san-economy`

```
{
  "version": 1,
  "storageDriverName": "ontap-san-economy",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.3",
  "svm": "svm_iscsi_eco",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "igroupName": "netappdvp"
}
```

## NVMe/TCP – Beispiel für einen `ONTAP`-Treiber

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

## Konfiguration von Element Software

Zusätzlich zu den Werten einer globalen Konfiguration sind bei Verwendung von Element Software (NetApp HCI/SolidFire) diese Optionen verfügbar.

Option	Beschreibung	Beispiel
Endpoint	\Https://<login>:<password>@<mvi p>/json-rpc/<element-version>	https://admin:admin@192.168.160.3/json-rpc/8.0
SVIP	ISCSI-IP-Adresse und -Port	10.0.0.7:3260 Uhr
TenantName	SolidFireF Mandanten zu verwenden (erstellt, falls nicht gefunden)	docker
InitiatorIFace	Geben Sie die Schnittstelle an, wenn der iSCSI-Datenverkehr auf eine nicht-Standardschnittstelle beschränkt wird	default
Types	QoS-Spezifikationen	Siehe das Beispiel unten

Option	Beschreibung	Beispiel
LegacyNamePrefix	Präfix für aktualisierte Trident Installationen. Wenn Sie eine Version von Trident vor 1.3.2 verwendet und ein Upgrade mit vorhandenen Volumes durchführen, müssen Sie diesen Wert einstellen, um auf die alten Volumes zuzugreifen, die über die Volume-Name-Methode zugeordnet wurden.	netappdvp-

Der `solidfire-san` Treiber unterstützt Docker Swarm nicht.

### Beispiel für eine Konfigurationsdatei für die Element Software

```

{
  "version": 1,
  "storageDriverName": "solidfire-san",
  "Endpoint": "https://admin:admin@192.168.160.3/json-rpc/8.0",
  "SVIP": "10.0.0.7:3260",
  "TenantName": "docker",
  "InitiatorIFace": "default",
  "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
      }
    }
  ]
}

```

## Bekannte Probleme und Einschränkungen

Informationen zu bekannten Problemen und Einschränkungen bei der Verwendung von Trident mit Docker

**Das Upgrade des Trident Docker Volume Plug-ins auf 20.10 und höher aus älteren Versionen führt zu einem Upgrade-Fehler, ohne dass solche Datei- oder Verzeichnisfehler auftreten.**

**Behelfslösung**

1. Deaktivieren Sie das Plugin.

```
docker plugin disable -f netapp:latest
```

2. Entfernen Sie das Plug-in.

```
docker plugin rm -f netapp:latest
```

3. Installieren Sie das Plugin neu, indem Sie den zusätzlichen Parameter angeben `config`.

```
docker plugin install netapp/trident-plugin:20.10 --alias netapp --grant  
-all-permissions config=config.json
```

## Volume-Namen müssen mindestens 2 Zeichen lang sein.



Dies ist eine Docker-Client-Einschränkung. Der Client interpretiert einen einzelnen Zeichennamen als Windows-Pfad. "[Siehe Bug 25773](#)".

## Docker Swarm verfügt über bestimmte Verhaltensweisen, die verhindern, dass Trident diese bei jeder Storage- und Treiberkombination unterstützen kann.

- Docker Swarm verwendet derzeit Volume-Namen anstelle der Volume-ID als eindeutige Volume-Kennung.
- Volume-Anforderungen werden gleichzeitig an jeden Node in einem Swarm-Cluster gesendet.
- Volume-Plug-ins (einschließlich Trident) müssen unabhängig auf jedem Node in einem Swarm-Cluster ausgeführt werden. Aufgrund der Funktionsweise von ONTAP und der Funktionsweise der `ontap-nas` und `ontap-san`-Treiber sind sie die einzigen, die innerhalb dieser Einschränkungen arbeiten können.

Der Rest der Fahrer unterliegt Themen wie Rennbedingungen, die dazu führen können, dass eine große Anzahl von Volumes für eine einzelne Anfrage ohne einen klaren „Gewinner“ erstellt werden; zum Beispiel hat Element eine Funktion, die es Volumes erlaubt, den gleichen Namen, aber unterschiedliche IDs zu haben.

NetApp hat das Docker-Team Feedback gegeben, lässt aber keinen Anzeichen für einen zukünftigen Regressanspruch haben.

**Wenn eine FlexGroup bereitgestellt wird, stellt ONTAP keine zweite FlexGroup bereit, wenn die zweite FlexGroup über einen oder mehrere Aggregate verfügt, die mit der bereitgestellten FlexGroup gemeinsam genutzt werden.**

# Best Practices und Empfehlungen

## Einsatz

Verwenden Sie bei der Implementierung von Trident die hier aufgeführten Empfehlungen.

### Implementieren Sie diesen in einem dedizierten Namespace

**"Namespaces"** Trennung von Administratoren zwischen verschiedenen Applikationen und Barriere für die gemeinsame Nutzung von Ressourcen. Beispielsweise kann eine PVC aus einem Namespace nicht von einem anderen genutzt werden. Trident stellt allen Namespaces im Kubernetes-Cluster PV-Ressourcen zur Verfügung und nutzt folglich ein Servicekonto mit erhöhten Privileges.

Außerdem kann der Zugriff auf den Trident Pod dazu führen, dass Benutzer auf die Anmeldedaten des Storage-Systems und andere sensible Informationen zugreifen können. Es ist wichtig, dass Applikationsbenutzer und Management-Applikationen nicht in der Lage sind, auf die Trident Objektdefinitionen oder Pods selbst zuzugreifen.

### Verwenden Sie Kontingente und Bereichsgrenzen, um den Storage-Verbrauch zu kontrollieren

Kubernetes bietet zusammen zwei Funktionen, die einen leistungsstarken Mechanismus zur Begrenzung des Ressourcenverbrauchs durch Applikationen bieten. **"Mechanismus für Storage-Kontingente"** Administratoren können globale Verbrauchsbeschränkungen für Storage-Klassen, Kapazitäten und die Anzahl der Objekte pro Namespace implementieren. Außerdem stellt die Verwendung eines **"Bereichsgrenze"** sicher, dass die PVC-Anforderungen innerhalb eines minimalen und eines maximalen Werts liegen, bevor die Anforderung an die Provisionierung weitergeleitet wird.

Diese Werte werden pro Namespace definiert, was bedeutet, dass jeder Namespace Werte definiert haben sollte, die ihren Ressourcenanforderungen entsprechen. Siehe hier für Informationen über **"Wie man Quoten nutzt"**.

## Storage-Konfiguration

Jede Storage-Plattform im NetApp Portfolio verfügt über einzigartige Funktionen für Applikationen, die in Containern oder nicht unterstützt werden.

### Plattformübersicht

Trident funktioniert mit ONTAP und Element. Es gibt keine Plattform, die besser für alle Anwendungen und Szenarien geeignet ist als die andere, aber bei der Auswahl einer Plattform sollten die Anforderungen der Anwendung und des Teams, das das Gerät verwaltet, berücksichtigt werden.

Sie sollten die Best Practices für das Host-Betriebssystem anhand des von Ihnen verwendeten Protokolls befolgen. Optional können Sie möglicherweise erwägen, falls verfügbar Best Practices für Applikationen mit Back-End-, Storage-Klassen- und PVC-Einstellungen zu integrieren, um den Storage für bestimmte Applikationen zu optimieren.

## Best Practices für ONTAP und Cloud Volumes ONTAP

Best Practices zur Konfiguration von ONTAP und Cloud Volumes ONTAP für Trident enthalten.

Die folgenden Empfehlungen sind Richtlinien zur Konfiguration von ONTAP für Container-Workloads, die Volumes nutzen, die von Trident dynamisch bereitgestellt werden. Jeder sollte in Betracht gezogen und auf Angemessenheit in Ihrer Umgebung überprüft werden.

### Verwenden Sie SVM(s) dediziert für Trident

Storage Virtual Machines (SVMs) sorgen für die Trennung von Mandanten auf einem ONTAP System. Durch die Zuweisung einer SVM für Applikationen können Berechtigungen delegation werden. Zudem lassen sich Best Practices anwenden, um den Ressourcenverbrauch zu begrenzen.

Für das Management der SVM sind verschiedene Optionen verfügbar:

- Stellen Sie die Cluster-Managementoberfläche in der Backend-Konfiguration zusammen mit entsprechenden Zugangsdaten bereit und geben Sie den SVM-Namen an.
- Erstellen Sie mit ONTAP System Manager oder der CLI eine dedizierte Managementoberfläche für die SVM.
- Teilen Sie die Managementrolle mit einer NFS-Datenschnittstelle.

In jedem Fall sollte sich die Schnittstelle im DNS enthalten, und beim Konfigurieren von Trident sollte der DNS-Name verwendet werden. Dadurch lassen sich einige DR-Szenarien, beispielsweise SVM-DR, vereinfachen, ohne die Aufbewahrung der Netzwerkidentität zu nutzen.

Es besteht keine Präferenz zwischen einer dedizierten oder gemeinsam genutzten Management-LIF für die SVM. Sie sollten jedoch sicherstellen, dass Ihre Netzwerksicherheitsrichtlinien mit dem von Ihnen gewählten Ansatz abgestimmt sind. Trotzdem sollte die Management-LIF über DNS zugänglich sein, um maximale Flexibilität zu ermöglichen, sollte **"SVM-DR"** in Verbindung mit Trident verwendet werden.

### Begrenzung der maximalen Volume-Anzahl

ONTAP Storage-Systeme besitzen eine maximale Anzahl an Volumes, die je nach Softwareversion und Hardwareplattform unterschiedlich sind. Informationen zu Ihren spezifischen Plattform- und ONTAP-Versionen finden Sie unter ["NetApp Hardware Universe"](#). Wenn die Anzahl der Volumes erschöpft ist, schlägt die Bereitstellung nicht nur für Trident fehl, sondern für alle Storage-Anforderungen.

Trident `ontap-nas` und `ontap-san` Treiber stellen für jedes erstellte Kubernetes Persistent Volume (PV) ein FlexVolume bereit. Der `ontap-nas-economy` Treiber erstellt ca. ein FlexVolume für je 200 PVs (konfigurierbar zwischen 50 und 300). Der `ontap-san-economy` Treiber erstellt ca. ein FlexVolume für je 100 PVs (konfigurierbar zwischen 50 und 200). Damit Trident nicht alle verfügbaren Volumes im Storage-System verbraucht, sollten Sie ein Limit für die SVM festlegen. Dies können Sie über die Befehlszeile ausführen:

```
vserver modify -vserver <svm_name> -max-volumes <num_of_volumes>
```

Der Wert für `max-volumes` variiert basierend auf verschiedenen Kriterien für Ihre spezifische Umgebung:

- Die Anzahl der vorhandenen Volumes im ONTAP Cluster
- Die Anzahl der Volumes, die für andere Applikationen außerhalb von Trident bereitgestellt werden
- Die Anzahl der persistenten Volumes, die von Kubernetes-Applikationen genutzt werden sollen

Der `max-volumes` Wert ist die Gesamtzahl der Volumes, die über alle Nodes im ONTAP Cluster bereitgestellt werden, nicht aber über einen einzelnen ONTAP Node. Aus diesem Grund treten möglicherweise einige Bedingungen auf, bei denen auf einem ONTAP Cluster-Node mehr oder weniger mit Trident bereitgestellte Volumes als ein anderer Node vorhanden sind.

So kann beispielsweise ein ONTAP Cluster mit zwei Nodes maximal 2000 FlexVols hosten. Eine auf 1250 eingestellte maximale Volumenzahl erscheint sehr vernünftig. Wenn jedoch nur "Aggregate" von einem Node der SVM zugewiesen wird oder die von einem Node zugewiesenen Aggregate nicht bereitgestellt werden können (z. B. aufgrund der Kapazität), dann wird der andere Node das Ziel aller über Trident bereitgestellten Volumes. Das bedeutet, dass das Volume-Limit für diesen Node vor dem Erreichen des Wertes erreicht werden kann `max-volumes`, was sich sowohl auf Trident als auch auf andere Volume-Operationen, die den Node verwenden, auswirkt. **Diese Situation kann vermieden werden, indem sichergestellt wird, dass die Aggregate von jedem Node im Cluster der von Trident verwendeten SVM in gleicher Anzahl zugewiesen werden.**

### **Begrenzung der maximalen Größe der durch Trident erstellten Volumes**

Verwenden Sie den Parameter in Ihrer `backend.json` Definition, um die maximale Größe für Volumes zu konfigurieren, die von Trident erstellt werden `limitVolumeSize` können.

Neben der Kontrolle der Volume-Größe im Storage-Array sollten auch Kubernetes-Funktionen genutzt werden.

### **Beschränkt die maximale Größe von FlexVols, die von Trident erstellt werden**

Um die maximale Größe für FlexVols zu konfigurieren, die als Pools für ONTAP-san-Economy- und ONTAP-nas-Economy-Treiber verwendet werden, verwenden Sie den `limitVolumePoolSize` Parameter in Ihrer `backend.json` Definition.

### **Trident für bidirektionales CHAP konfigurieren**

Sie können in der Back-End-Definition den CHAP-Initiator und die Benutzernamen und Passwörter für das Ziel angeben und Trident CHAP auf der SVM aktivieren. Mithilfe des `useCHAP` Parameters in Ihrer Backend-Konfiguration authentifiziert Trident iSCSI-Verbindungen für ONTAP-Back-Ends mit CHAP.

### **Erstellen und Verwenden einer SVM QoS-Richtlinie**

Die Nutzung einer ONTAP QoS-Richtlinie auf die SVM begrenzt die Anzahl der durch die von Trident bereitgestellten Volumes konsumierbaren IOPS. Dies hilft bei der "Verhindern Sie einen Schläger" Überwachung oder außer Kontrolle geraten durch Container, die Workloads außerhalb der Trident SVM beeinträchtigen.

Sie können in wenigen Schritten eine QoS-Richtlinie für die SVM erstellen. Die genauesten Informationen finden Sie in der Dokumentation Ihrer ONTAP-Version. Das folgende Beispiel erstellt eine QoS-Richtlinie, die die insgesamt für eine SVM verfügbaren IOPS auf 5000 begrenzt.



```
# create the policy group for the SVM
qos policy-group create -policy-group <policy_name> -vserver <svm_name>
-max-throughput 5000iops

# assign the policy group to the SVM, note this will not work
# if volumes or files in the SVM have existing QoS policies
vserver modify -vserver <svm_name> -qos-policy-group <policy_name>
```

Wenn zudem Ihre ONTAP Version sie unterstützt, können Sie den Einsatz eines minimalen QoS-Systems in Erwägung ziehen, um einen hohen Durchsatz für Container-Workloads zu gewährleisten. Die adaptive QoS ist nicht mit einer Richtlinie auf SVM-Ebene kompatibel.

Die Anzahl der für Container-Workloads dedizierten IOPS hängt von vielen Aspekten ab. Dazu zählen unter anderem:

- Anderen Workloads, die das Storage-Array nutzen Bei anderen Workloads, die nicht mit der Kubernetes-Implementierung zusammenhängen und die Storage-Ressourcen nutzen, sollte darauf achten, dass diese Workloads nicht versehentlich beeinträchtigt werden.
- Erwartete Workloads werden in Containern ausgeführt. Wenn Workloads mit hohen IOPS-Anforderungen in Containern ausgeführt werden, führt eine niedrige QoS-Richtlinie zu schlechten Erfahrungen.

Es muss daran erinnert werden, dass eine auf SVM-Ebene zugewiesene QoS-Richtlinie alle Volumes zur Verfügung hat, die der SVM bereitgestellt werden und sich denselben IOPS-Pool teilen. Wenn eine oder nur eine kleine Zahl von Container-Applikationen sehr hohe IOPS-Anforderungen erfüllen, kann dies zu einem problematischer für die anderen Container-Workloads werden. In diesem Fall empfiehlt es sich, QoS-Richtlinien pro Volume mithilfe von externer Automatisierung zuzuweisen.



Sie sollten die QoS Policy Group der SVM **only** zuweisen, wenn Ihre ONTAP Version älter als 9.8 ist.

## Erstellen von QoS-Richtliniengruppen für Trident

Quality of Service (QoS) garantiert, dass die Performance kritischer Workloads nicht durch konkurrierende Workloads beeinträchtigt wird. ONTAP QoS-Richtliniengruppen bieten QoS-Optionen für Volumes und ermöglichen Benutzern, die Durchsatzgrenze für einen oder mehrere Workloads zu definieren. Weitere Informationen zur QoS finden Sie unter "[Garantierter Durchsatz durch QoS](#)". Sie können QoS-Richtliniengruppen im Backend oder im Storage-Pool festlegen und werden auf jedes in diesem Pool oder Backend erstellte Volume angewendet.

ONTAP verfügt über zwei Arten von QoS-Richtliniengruppen: Herkömmliche und anpassungsfähige. Herkömmliche Richtliniengruppen bieten einen flachen maximalen Durchsatz (oder minimalen Durchsatz in späteren Versionen) in IOPS. Adaptive QoS skaliert den Durchsatz automatisch auf die Workload-Größe und erhält das Verhältnis von IOPS zu TB-fähigen GB-Werten, wenn sich die Workload-Größe ändert. Wenn Sie Hunderte oder Tausende Workloads in einer großen Implementierung managen, bietet sich somit ein erheblicher Vorteil.

Beachten Sie beim Erstellen von QoS-Richtliniengruppen Folgendes:

- Sie sollten den Schlüssel im `defaults` Block der Backend-Konfiguration setzen `qosPolicy`. Im folgenden Back-End-Konfigurationsbeispiel:

```

---
version: 1
storageDriverName: ontap-nas
managementLIF: 0.0.0.0
dataLIF: 0.0.0.0
svm: svm0
username: user
password: pass
defaults:
  qosPolicy: standard-pg
storage:
- labels:
  performance: extreme
  defaults:
  adaptiveQosPolicy: extremely-adaptive-pg
- labels:
  performance: premium
  defaults:
  qosPolicy: premium-pg

```

- Sie sollten die Richtliniengruppen pro Volume anwenden, damit jedes Volume den gesamten von der Richtliniengruppe angegebenen Durchsatz erhält. Gemeinsame Richtliniengruppen werden nicht unterstützt.

Weitere Informationen zu QoS-Richtliniengruppen finden Sie unter ["ONTAP 9.8 QoS-Befehle"](#).

### **Beschränken Sie den Zugriff auf die Storage-Ressourcen auf Kubernetes-Cluster-Mitglieder**

Der Zugriff auf die durch Trident erstellten NFS-Volumes und iSCSI-LUNs ist eine entscheidende Komponente der Sicherheit für die Kubernetes-Implementierung. Auf diese Weise wird verhindert, dass Hosts, die nicht zum Kubernetes Cluster gehören, auf die Volumes zugreifen und Daten unerwartet ändern können.

Es ist wichtig zu wissen, dass Namespaces die logische Grenze für Ressourcen in Kubernetes sind. Es wird angenommen, dass Ressourcen im selben Namespace gemeinsam genutzt werden können. Es gibt jedoch keine Cross-Namespace-Funktion. Dies bedeutet, dass PVS zwar globale Objekte sind, aber wenn sie an ein PVC gebunden sind, nur über Pods zugänglich sind, die sich im selben Namespace befinden. **Es ist wichtig sicherzustellen, dass Namensräume verwendet werden, um eine Trennung zu gewährleisten, wenn angemessen.**

Die meisten Unternehmen haben im Zusammenhang mit der Datensicherheit bei Kubernetes die Sorge, dass ein Container-Prozess auf den Storage zugreifen kann, der am Host gemountet ist; dieser ist jedoch nicht für den Container bestimmt. ["Namespaces"](#) sind so konzipiert, dass diese Art von Kompromiss verhindert wird. Allerdings gibt es eine Ausnahme: Privilegierte Container.

Ein privilegierter Container ist ein Container, der mit wesentlich mehr Berechtigungen auf Hostebene als normal ausgeführt wird. Diese werden standardmäßig nicht verweigert. Stellen Sie daher sicher, dass Sie die Funktion mithilfe von deaktivieren ["Pod-Sicherheitsrichtlinien"](#).

Bei Volumes, für die der Zugriff von Kubernetes und externen Hosts gewünscht wird, sollte der Storage auf herkömmliche Weise gemanagt werden. Dabei wird das PV durch den Administrator eingeführt und nicht von

Trident gemanagt. So wird sichergestellt, dass das Storage Volume nur zerstört wird, wenn sowohl Kubernetes als auch externe Hosts getrennt haben und das Volume nicht mehr nutzen. Zusätzlich kann eine benutzerdefinierte Exportrichtlinie angewendet werden, die den Zugriff von den Kubernetes-Cluster-Nodes und Zielserversn außerhalb des Kubernetes-Clusters ermöglicht.

Für Bereitstellungen mit dedizierten Infrastruktur-Nodes (z. B. OpenShift) oder anderen Nodes, die Benutzerapplikationen nicht planen können, sollten separate Exportrichtlinien verwendet werden, um den Zugriff auf Speicherressourcen weiter zu beschränken. Dies umfasst die Erstellung einer Exportrichtlinie für Services, die auf diesen Infrastruktur-Nodes bereitgestellt werden (z. B. OpenShift Metrics and Logging Services), sowie Standardanwendungen, die auf nicht-Infrastruktur-Nodes bereitgestellt werden.

### Verwenden Sie eine dedizierte Exportrichtlinie

Sie sollten sicherstellen, dass für jedes Backend eine Exportrichtlinie vorhanden ist, die nur den Zugriff auf die im Kubernetes-Cluster vorhandenen Nodes erlaubt. Trident kann Richtlinien für den Export automatisch erstellen und managen. So beschränkt Trident den Zugriff auf die Volumes, die ihm im Kubernetes Cluster zur Verfügung stehen, und vereinfacht das Hinzufügen/Löschen von Nodes.

Alternativ können Sie auch eine Exportrichtlinie manuell erstellen und mit einer oder mehreren Exportregeln füllen, die die Zugriffsanforderung für die einzelnen Knoten bearbeiten:

- Erstellen Sie die Exportrichtlinie mit `vserver export-policy create` dem ONTAP-CLI-Befehl.
- Fügen Sie der Exportrichtlinie Regeln mithilfe des ONTAP CLI-Befehls hinzu `vserver export-policy rule create`.

Wenn Sie diese Befehle ausführen, können Sie die Zugriffsrechte der Kubernetes-Nodes auf die Daten beschränken.

### Deaktivieren `showmount` für die Anwendungs-SVM

Die `showmount` Funktion ermöglicht es einem NFS-Client, die SVM nach einer Liste der verfügbaren NFS-Exporte abzufragen. Ein im Kubernetes-Cluster implementierter Pod kann den Befehl für die Daten-LIF ausgeben `showmount -e` und eine Liste der verfügbaren Mounts erhalten, einschließlich derjenigen, auf die er keinen Zugriff hat. Obwohl dies für sich kein Sicherheitskompromiss ist, stellt es keine unnötigen Informationen bereit, die einem nicht autorisierten Benutzer die Verbindung zu einem NFS-Export ermöglichen.

Sie sollten die Deaktivierung `showmount` mit dem ONTAP CLI-Befehl auf SVM-Ebene verwenden:

```
vserver nfs modify -vserver <svm_name> -showmount disabled
```

## SolidFire Best Practices in sich vereint

Lesen Sie Best Practices zur Konfiguration von SolidFire Storage für Trident.

### Erstellen Eines SolidFire-Kontos

Jedes SolidFire-Konto stellt einen eindeutigen Volume-Eigentümer dar und erhält seine eigenen Anmeldeinformationen für das Challenge-Handshake Authentication Protocol (CHAP). Sie können auf Volumes zugreifen, die einem Konto zugewiesen sind, entweder über den Kontonamen und die relativen CHAP-Anmeldeinformationen oder über eine Zugriffsgruppe für Volumes. Einem Konto können bis zu zweitausend Volumes zugewiesen sein, ein Volume kann jedoch nur zu einem Konto gehören.

## Erstellen einer QoS-Richtlinie

Verwenden Sie QoS-Richtlinien (Quality of Service) von SolidFire, um eine standardisierte Quality of Service-Einstellung zu erstellen und zu speichern, die auf viele Volumes angewendet werden kann.

Sie können QoS-Parameter für einzelne Volumes festlegen. Die Performance für jedes Volume kann durch drei konfigurierbare Parameter bestimmt werden, die QoS definieren: Das IOPS-Minimum, das IOPS-Maximum und die Burst-IOPS.

Hier sind die möglichen Minimum-, Maximum- und Burst-IOPS für die 4-KB-Blockgröße.

IOPS-Parameter	Definition	Min. Wert	Standardwert	Max. Wert (4 KB)
IOPS-Minimum	Das garantierte Performance-Level für ein Volume	50	50	15000
IOPS-Maximum	Die Leistung überschreitet dieses Limit nicht.	50	15000	200.000
IOPS-Burst	Maximale IOPS in einem kurzen Burst-Szenario zulässig.	50	15000	200.000



Obwohl die IOPS-Maximum und die Burst-IOPS so hoch wie 200,000 sind, wird die tatsächliche maximale Performance eines Volumes durch die Nutzung von Clustern und die Performance pro Node begrenzt.

Die Blockgröße und die Bandbreite haben einen direkten Einfluss auf die Anzahl der IOPS. Mit zunehmender Blockgröße erhöht das System die Bandbreite auf ein Niveau, das für die Verarbeitung größerer Blockgrößen erforderlich ist. Mit der steigenden Bandbreite sinkt auch die Anzahl an IOPS, die das System erreichen kann. Weitere Informationen zur QoS und Performance finden Sie unter "[SolidFire Quality of Service](#)".

## SolidFire Authentifizierung

Element unterstützt zwei Authentifizierungsmethoden: CHAP und Volume Access Groups (VAG). CHAP verwendet das CHAP-Protokoll, um den Host am Backend zu authentifizieren. Volume Access Groups steuern den Zugriff auf die Volumes, die durch sie bereitgestellt werden. Da die Authentifizierung einfacher ist und über keine Grenzen für die Skalierung verfügt, empfiehlt NetApp die Verwendung von CHAP.



Trident mit dem erweiterten CSI-provisioner unterstützt die Verwendung von CHAP-Authentifizierung. Vags sollten nur im traditionellen nicht-CSI-Betriebsmodus verwendet werden.

CHAP-Authentifizierung (Verifizierung, dass der Initiator der vorgesehene Volume-Benutzer ist) wird nur mit der Account-basierten Zugriffssteuerung unterstützt. Wenn Sie CHAP zur Authentifizierung verwenden, stehen zwei Optionen zur Verfügung: Unidirektionales CHAP und bidirektionales CHAP. Unidirektionales CHAP authentifiziert den Volume-Zugriff mithilfe des SolidFire-Kontonamens und des Initiatorgeheimnisses. Die bidirektionale CHAP-Option bietet die sicherste Möglichkeit zur Authentifizierung des Volumes, da das Volume den Host über den Kontonamen und den Initiatorschlüssel authentifiziert und dann der Host das Volume über den Kontonamen und den Zielschlüssel authentifiziert.

Wenn CHAP jedoch nicht aktiviert werden kann und Vags erforderlich sind, erstellen Sie die Zugriffsgruppe und fügen Sie die Hostinitiatoren und Volumes der Zugriffsgruppe hinzu. Jeder IQN, den Sie einer Zugriffsgruppe hinzufügen, kann mit oder ohne CHAP-Authentifizierung auf jedes Volume in der Gruppe zugreifen. Wenn der iSCSI-Initiator für die Verwendung der CHAP-Authentifizierung konfiguriert ist, wird die kontenbasierte Zugriffssteuerung verwendet. Wenn der iSCSI-Initiator nicht für die Verwendung der CHAP-Authentifizierung konfiguriert ist, wird die Zugriffskontrolle für die Volume Access Group verwendet.

## Wo finden Sie weitere Informationen?

Einige der Best Practices-Dokumentationen sind unten aufgeführt. Suchen Sie im ["NetApp Bibliothek"](#) nach den aktuellsten Versionen.

### ONTAP

- ["NFS Best Practice- und Implementierungsleitfaden"](#)
- ["SAN-Administrationshandbuch"](#) (Für iSCSI)
- ["iSCSI Express-Konfiguration für RHEL"](#)

### Element Software

- ["Konfigurieren von SolidFire für Linux"](#)

### NetApp HCI

- ["Voraussetzungen für die NetApp HCI-Implementierung"](#)
- ["Rufen Sie die NetApp Deployment Engine auf"](#)

### Anwendung Best Practices Informationen

- ["Best Practices für MySQL auf ONTAP"](#)
- ["Best Practices für MySQL auf SolidFire"](#)
- ["NetApp SolidFire und Cassandra"](#)
- ["Best Practices für Oracle auf SolidFire"](#)
- ["Best Practices für PostgreSQL auf SolidFire"](#)

Da nicht alle Applikationen spezifische Richtlinien haben, ist es wichtig, mit Ihrem NetApp Team zusammenzuarbeiten und die aktuellste Dokumentation zu finden. ["NetApp Bibliothek"](#)

## Integration von Trident

Zur Integration von Trident müssen folgende Design- und Architekturelemente integriert werden: Treiberauswahl und -Implementierung, Storage-Klassendesign, Virtual Pool Design, Persistent Volume Claim (PVC) Auswirkungen auf die Storage-Provisionierung, Volume-Betrieb und OpenShift-Services mithilfe von Trident.

### Auswahl und Implementierung der Treiber

Wählen Sie einen Back-End-Treiber für Ihr Speichersystem aus und implementieren Sie ihn.

## Back-End-Treiber für ONTAP

Die Back-End-Treiber für ONTAP unterscheiden sich durch das verwendete Protokoll und die Art und Weise, wie die Volumes im Storage-System bereitgestellt werden. Daher sollten Sie bei der Entscheidung, welchen Treiber eingesetzt werden soll, sorgfältig überlegen.

Auf einer höheren Ebene, wenn Ihre Applikation Komponenten hat, die gemeinsamen Storage benötigen (mehrere Pods, die auf dasselbe PVC zugreifen), sind NAS-basierte Treiber die erste Wahl, während die blockbasierten iSCSI-Treiber die Anforderungen von nicht gemeinsam genutztem Storage erfüllen. Wählen Sie das Protokoll basierend auf den Anforderungen der Applikation und der Komfort-Ebene der Storage- und Infrastrukturtteams. Generell besteht für die meisten Applikationen kein Unterschied zwischen ihnen. Oftmals basiert die Entscheidung darauf, ob gemeinsam genutzter Storage (wo mehr als ein POD den gleichzeitigen Zugriff benötigen) benötigt wird.

Die verfügbaren Back-End-Treiber für ONTAP sind:

- `ontap-nas`: Jedes bereitgestellte PV ist ein vollständiges ONTAP FlexVolume.
- `ontap-nas-economy`: Jedes bereitgestellte PV ist ein qtree, mit einer konfigurierbaren Anzahl von qtrees pro FlexVolume (Standard ist 200).
- `ontap-nas-flexgroup`: Jedes PV, das als vollständiges ONTAP FlexGroup bereitgestellt wird, und alle Aggregate, die einer SVM zugewiesen sind, werden verwendet.
- `ontap-san`: Jedes bereitgestellte PV ist eine LUN innerhalb seines eigenen FlexVolume.
- `ontap-san-economy`: Jedes bereitgestellte PV ist eine LUN, mit einer konfigurierbaren Anzahl von LUNs pro FlexVolume (Standard ist 100).

Die Auswahl zwischen den drei NAS-Treibern hat einige Auswirkungen auf die Funktionen, die der Applikation zur Verfügung gestellt werden.

Beachten Sie, dass in den folgenden Tabellen nicht alle Funktionen über Trident bereitgestellt werden. Einige müssen vom Storage-Administrator nach der Bereitstellung angewendet werden, wenn diese Funktion gewünscht wird. Die Super-Skript-Fußnoten unterscheiden die Funktionalität pro Feature und Treiber.

ONTAP NAS-Treiber	Snapshot s	Klone	Dynamische Exportrichtlinien	Multi-Anschlus s	QoS	Größe Ändern	Replizierung
<code>ontap-nas</code>	Ja.	Ja.	Jafußnote: 5[]	Ja.	Jafußnote: 1[]	Ja.	Jafußnote: 1[]
<code>ontap-nas-economy</code>	NO [3]	NO [3]	Jafußnote: 5[]	Ja.	NO [3]	Ja.	NO [3]
<code>ontap-nas-flexgroup</code>	Jafußnote: 1[]	NEIN	Jafußnote: 5[]	Ja.	Jafußnote: 1[]	Ja.	Jafußnote: 1[]

Trident bietet 2 SAN-Treiber für ONTAP an, deren Funktionen unten dargestellt sind.

ONTAP SAN-Treiber	Snapshots	Klone	Multi-Anschlus s	Bidirektio nales CHAP	QoS	Größe Ändern	Replizieru ng
ontap-san	Ja.	Ja.	Jafußnote: 4[]	Ja.	Jafußnote: 1[]	Ja.	Jafußnote: 1[]
ontap-san-economy	Ja.	Ja.	Jafußnote: 4[]	Ja.	NO [3]	Ja.	NO [3]

Fußnote für die obigen Tabellen: Yes [1]: Nicht von Trident verwaltet Yes [2]: Verwaltet von Trident, aber nicht von PV granular Yes [3]: Nicht von Trident verwaltet und nicht von PV granular Yes [4]: Unterstützt für RAW-Block-Volumes Yes [5]: Unterstützt von Trident

Die Funktionen, die keine PV-Granularität sind, werden auf das gesamte FlexVolume angewendet, und alle PVs (also qtrees oder LUNs in gemeinsam genutzten FlexVols) teilen einen gemeinsamen Zeitplan.

Wie wir in den obigen Tabellen sehen können, ist ein Großteil der Funktionalität zwischen `ontap-nas-economy` die `ontap-nas` gleiche. Da der Treiber jedoch `ontap-nas-economy` die Möglichkeit zur Steuerung des Zeitplans auf PV-Granularität beschränkt, kann dies insbesondere Ihre Disaster Recovery- und Backup-Planung beeinträchtigen. Für Entwicklungsteams, die die PVC-Klonfunktion auf dem ONTAP-Storage nutzen möchten, ist dies nur mit den, `ontap-san` oder `ontap-san-economy`-Treibern möglich `ontap-nas`.



Der `solidfire-san` Treiber kann auch VES klonen.

### Back-End-Treiber für Cloud Volumes ONTAP

Cloud Volumes ONTAP bietet Datenkontrolle und Storage-Funktionen der Enterprise-Klasse für verschiedene Anwendungsfälle, einschließlich Dateifreigaben und Storage-Funktionen auf Blockebene für NAS- und SAN-Protokolle (NFS, SMB/CIFS und iSCSI). Die kompatiblen Treiber für Cloud Volume ONTAP sind `ontap-nas`, `ontap-nas-economy` `ontap-san` und `ontap-san-economy`. Diese gelten für Cloud Volume ONTAP für Azure, Cloud Volume ONTAP für GCP.

### Back-End-Treiber für Amazon FSX for ONTAP

Amazon FSX for NetApp ONTAP ermöglicht Ihnen die Nutzung von NetApp Funktionen, Performance und Administrationsfunktionen, mit denen Sie vertraut sind, und gleichzeitig die Einfachheit, Agilität, Sicherheit und Skalierbarkeit der Speicherung von Daten auf AWS zu nutzen. FSX für ONTAP unterstützt viele ONTAP-Dateisystemfunktionen und Administrations-APIs. Die kompatiblen Treiber für Cloud Volume ONTAP sind `ontap-nas`, `ontap-nas-economy` `ontap-nas-flexgroup` `ontap-san` und `ontap-san-economy`.

### Back-End-Treiber für NetApp HCI/SolidFire

Der `solidfire-san` Treiber, der mit den NetApp HCI/SolidFire-Plattformen verwendet wird, hilft dem Administrator, ein Element Backend für Trident auf der Grundlage von QoS Limits zu konfigurieren. Wenn Sie Ihr Backend so gestalten möchten, dass die spezifischen QoS-Limits für die durch Trident bereitgestellten Volumes festgelegt werden, verwenden Sie den `type` Parameter in der Backend-Datei. Der Admin kann auch die Volume-Größe einschränken, die mit dem Parameter auf dem Storage erstellt werden `limitVolumeSize` kann. Derzeit werden Element Storage-Funktionen wie die Größenanpassung von Volumes und die Volume-Replizierung nicht durch den Treiber unterstützt `solidfire-san`. Diese Vorgänge sollten manuell über die Web-UI von Element Software durchgeführt werden.

SolidFire-Treiber	Snapshots	Klone	Multi-Anschlus s	CHAP	QoS	Größe Ändern	Replizieru ng
solidfire-san	Ja.	Ja.	Jafußnote: 2[]	Ja.	Ja.	Ja.	Jafußnote: 1[]

Fußnote: Ja [1]: Nicht verwaltet von Trident Yes [2]: Unterstützt für RAW-Block-Volumes

### Back-End-Treiber für Azure NetApp Files

Trident verwendet den `azure-netapp-files` Treiber für die Verwaltung des "Azure NetApp Dateien" Dienstes.

Weitere Informationen zu diesem Treiber und zur Konfiguration finden Sie unter "[Trident Back-End-Konfiguration für Azure NetApp Files](#)".

Azure NetApp Files-Treiber	Snapshots	Klone	Multi-Anschluss	QoS	Erweitern	Replizierung
azure-netapp-files	Ja.	Ja.	Ja.	Ja.	Ja.	Jafußnote:1[ ]

Fußnote: JaFußnote:1[]: Nicht von Trident verwaltet

### Cloud Volumes Service auf Google Cloud Backend-Treiber

Trident verwendet den `gcp-cvs` Treiber für die Verbindung mit dem Cloud Volumes Service auf Google Cloud.

Der `gcp-cvs` Treiber verwendet virtuelle Pools, um das Backend zu abstrahieren und Trident die Volume-Platzierung zu bestimmen. Der Administrator definiert die virtuellen Pools in den `backend.json` Dateien. Storage-Klassen verwenden Selektoren, um virtuelle Pools nach Etikett zu identifizieren.

- Wenn im Back-End virtuelle Pools definiert werden, versucht Trident, ein Volume in den Google Cloud Storage-Pools zu erstellen, auf das diese virtuellen Pools begrenzt sind.
- Wenn virtuelle Pools nicht im Backend definiert sind, wählt Trident einen Google Cloud-Speicherpool aus den verfügbaren Speicherpools in der Region aus.

Um das Google Cloud-Backend auf Trident zu konfigurieren, müssen Sie `apiRegion` und `apiKey` in der Backend-Datei angeben `projectNumber`. Die Projektnummer finden Sie in der Google Cloud-Konsole. Der API-Schlüssel wird aus der Datei mit dem privaten Schlüssel des Dienstkontos entnommen, die Sie beim Einrichten des API-Zugriffs für Cloud Volumes Service in der Google Cloud erstellt haben.

Weitere Informationen zu Cloud Volumes Service auf Google Cloud Service-Typen und Service-Leveln finden Sie in "[Erfahren Sie mehr zur Unterstützung von Trident für CVS für GCP](#)".



Cloud Volumes Service für Google Cloud Treiber	Snapshots	Klone	Multi-Anschluss	QoS	Erweitern	Replizierung
gcp-cvs	Ja.	Ja.	Ja.	Ja.	Ja.	Nur für den CVS-Performance-Diensttyp verfügbar.



#### Hinweise zur Replikation

- Die Replizierung wird nicht von Trident gemanagt.
- Der Klon wird im selben Speicherpool erstellt wie das Quell-Volumen.

## Design der Storage-Klasse

Individuelle Storage-Klassen müssen konfiguriert und angewendet werden, um ein Kubernetes Storage Class-Objekt zu erstellen. Dieser Abschnitt erläutert, wie Sie eine Storage-Klasse für Ihre Applikation entwerfen.

### Spezifische Back-End-Auslastung

Die Filterung kann innerhalb eines bestimmten Storage-Klassenobjekts verwendet werden, um festzulegen, welcher Storage-Pool bzw. welche Pools für die jeweilige Storage-Klasse verwendet werden sollen. In der Storage Class können drei Filtersätze eingestellt werden: `storagePools`, `additionalStoragePools` und/oder `excludeStoragePools`.

Mit dem `storagePools` Parameter kann der Speicher auf die Gruppe von Pools beschränkt werden, die mit allen angegebenen Attributen übereinstimmen. Mit dem `additionalStoragePools` Parameter wird der Pool-Satz erweitert, den Trident für das Provisioning verwendet, zusammen mit dem durch die Attribute und Parameter ausgewählten Pool-Satz `storagePools`. Sie können entweder nur einen der Parameter oder beide zusammen verwenden, um sicherzustellen, dass der entsprechende Satz von Speicherpools ausgewählt wird.

Der `excludeStoragePools` Parameter wird verwendet, um speziell die aufgelisteten Pools auszuschließen, die mit den Attributen übereinstimmen.

### QoS-Richtlinien emulieren

Wenn Sie Storage-Klassen so entwerfen möchten, dass sie Quality of Service-Richtlinien emulieren, erstellen Sie eine Storage-Klasse mit dem `media` Attribut `hdd` oder `ssd`. Auf der Grundlage des `media` in der Storage-Klasse bereits erwähnten Attributs wählt Trident das geeignete Back-End mit Servern oder `ssd` Aggregaten aus, das `hdd` dem Medienattribut entspricht, und leitet die Bereitstellung der Volumes dann an das spezifische Aggregat weiter. Daher können wir einen PREMIUM-Storage-Klasse erstellen, für `media` den Attribute festgelegt werden, die als `ssd` PREMIUM-QoS-Richtlinie klassifiziert werden könnten. Wir können einen weiteren STANDARD der Storage-Klasse erstellen, bei dem das Medienattribut auf `hdd` gesetzt wäre. Dieser Standard könnte die QoS-Richtlinie SEIN. Darüber hinaus könnten wir das Attribut ```IOPS'` in der Storage-Klasse verwenden, um die Bereitstellung zu einer Element Appliance umzuleiten, die als QoS-Richtlinie definiert werden kann.

### Nutzung von Backend basierend auf bestimmten Funktionen

Storage-Klassen ermöglichen die direkte Volume-Bereitstellung an einem bestimmten Back-End, bei dem Funktionen wie Thin Provisioning und Thick Provisioning, Snapshots, Klone und Verschlüsselung aktiviert sind.

Um festzulegen, welchen Speicher verwendet werden soll, erstellen Sie Speicherklassen, die das entsprechende Back-End mit aktivierter Funktion angeben.

## Virtuelle Pools

Virtuelle Pools sind für alle Trident Back-Ends verfügbar. Sie können virtuelle Pools für jedes Back-End definieren, indem Sie einen beliebigen Treiber von Trident verwenden.

Mit virtuellen Pools kann ein Administrator eine Abstraktionsebene über Back-Ends erstellen, auf die über Storage-Klassen verwiesen werden kann. So werden Volumes auf Back-Ends flexibler und effizienter platziert. Verschiedene Back-Ends können mit derselben Serviceklasse definiert werden. Darüber hinaus können mehrere Storage Pools auf demselben Backend erstellt werden, jedoch mit unterschiedlichen Eigenschaften. Wenn eine Speicherklasse mit einem Selektor mit den spezifischen Bezeichnungen konfiguriert ist, wählt Trident ein Backend aus, das allen Selektor-Labels entspricht, um das Volume zu platzieren. Wenn die Auswahlbezeichnungen für Speicherklassen mit mehreren Speicherpools übereinstimmen, wählt Trident einen dieser Speicherpools aus, um das Volume bereitzustellen.

## Virtual Pool Design

Beim Erstellen eines Backends können Sie im Allgemeinen eine Reihe von Parametern angeben. Der Administrator konnte kein weiteres Back-End mit denselben Storage Credentials und anderen Parametern erstellen. Mit der Einführung von virtuellen Pools wurde dieses Problem behoben. Virtual Pools ist eine Ebene-Abstraktion, die zwischen dem Backend und der Kubernetes Storage Class eingeführt wird. So kann der Administrator Parameter zusammen mit Labels definieren, die über Kubernetes Storage Klassen als Selektion auf Backend-unabhängige Weise referenziert werden können. Virtuelle Pools können für alle unterstützten NetApp-Back-Ends mit Trident definiert werden. Dazu zählen SolidFire/NetApp HCI, ONTAP, Cloud Volumes Service auf GCP und Azure NetApp Files.



Bei der Definition von virtuellen Pools wird empfohlen, nicht zu versuchen, die Reihenfolge vorhandener virtueller Pools in einer Backend-Definition neu anzuordnen. Es wird auch empfohlen, Attribute für einen vorhandenen virtuellen Pool nicht zu bearbeiten/zu ändern und stattdessen einen neuen virtuellen Pool zu definieren.

## Emulation verschiedener Service-Level/QoS

Es ist möglich, virtuelle Pools zur Emulation von Serviceklassen zu entwerfen. Untersuchen wir mit der Implementierung des virtuellen Pools für den Cloud Volume Service für Azure NetApp Files, wie wir verschiedene Serviceklassen einrichten können. Konfigurieren Sie das Azure NetApp Files Back-End mit mehreren Labels, die unterschiedliche Performance-Levels repräsentieren. Stellen Sie `servicelevel` Aspect auf das entsprechende Leistungsniveau ein, und fügen Sie weitere erforderliche Aspekte unter den einzelnen Beschriftungen hinzu. Erstellen Sie nun verschiedene Kubernetes Storage-Klassen, die verschiedenen virtuellen Pools zugeordnet werden würden. Über das `parameters.selector` Feld ruft jede StorageClass ab, welche virtuellen Pools zum Hosten eines Volumes verwendet werden können.

## Zuweisen eines spezifischen Satzes von Aspekten

Mehrere virtuelle Pools mit spezifischen Aspekten können über ein einzelnes Storage-Back-End entwickelt werden. Konfigurieren Sie dazu das Backend mit mehreren Beschriftungen und legen Sie die erforderlichen Aspekte unter jedem Etikett fest. Erstellen Sie nun mithilfe des Felds, das verschiedenen virtuellen Pools zugeordnet wird, verschiedene Kubernetes-Storage-Klassen `parameters.selector`. Die Volumes, die im Backend bereitgestellt werden, werden im ausgewählten virtuellen Pool über die Aspekte definiert.

## PVC-Merkmale, die die Storage-Bereitstellung beeinflussen

Einige Parameter, die über die angeforderte Storage-Klasse hinausgehen, können sich bei der Erstellung einer PVC auf den Entscheidungsprozess für die Bereitstellung von Trident auswirken.

### Zugriffsmodus

Wenn Sie Speicher über ein PVC anfordern, ist eines der Pflichtfelder der Zugriffsmodus. Der gewünschte Modus kann sich auf das ausgewählte Backend auswirken, um die Speichieranforderung zu hosten.

Trident versucht, das verwendete Storage-Protokoll mit der gemäß der folgenden Matrix angegebenen Zugriffsmethode abzustimmen. Dies ist unabhängig von der zugrunde liegenden Storage-Plattform.

	ReadWriteOnce	ReadOnlyManche	ReadWriteViele
ISCSI	Ja.	Ja.	Ja (Raw Block)
NFS	Ja.	Ja.	Ja.

Eine Anfrage nach einem ReadWriteManche PVC, die an eine Trident-Implementierung ohne konfiguriertes NFS-Backend gesendet werden, führt dazu, dass kein Volume bereitgestellt wird. Aus diesem Grund sollte der Anforderer den Zugriffsmodus verwenden, der für seine Anwendung geeignet ist.

## Volume-Vorgänge

### Persistente Volumes ändern

Persistente Volumes sind mit zwei Ausnahmen unveränderliche Objekte in Kubernetes. Sobald die Rückgewinnungsrichtlinie erstellt wurde, kann die Größe geändert werden. Dies hindert jedoch nicht daran, einige Aspekte des Volumes außerhalb von Kubernetes zu ändern. Das kann durchaus wünschenswert sein, wenn das Volume für spezifische Applikationen angepasst werden soll, um sicherzustellen, dass die Kapazität nicht versehentlich verbraucht wird oder das Volume einfach aus irgendeinem Grund auf einen anderen Storage Controller verschoben werden kann.



Kubernetes-in-Tree-Provisioners unterstützen derzeit keine Vorgänge zur Größenanpassung von Volumes für NFS oder iSCSI PVS. Trident unterstützt die Erweiterung von NFS- und iSCSI-Volumes.

Die Verbindungsdetails des PV können nach der Erstellung nicht geändert werden.

### Erstellung von On-Demand-Volume-Snapshots

Trident unterstützt die Erstellung von On-Demand-Volume-Snapshots und die Erstellung von VES aus Snapshots mithilfe des CSI-Frameworks. Snapshots bieten eine bequeme Methode, zeitpunktgenaue Kopien der Daten zu erstellen und haben unabhängig vom Quell-PV in Kubernetes einen Lebenszyklus. Diese Snapshots können zum Klonen von PVCs verwendet werden.

### Volumes-Erstellung aus Snapshots

Trident unterstützt außerdem die Erstellung von PersistentVolumes aus Volume Snapshots. Um dies zu erreichen, erstellen Sie einfach ein PersistentVolumeClaim und erwähnen Sie den `datasource` als den erforderlichen Snapshot, aus dem das Volume erstellt werden muss. Trident wird diese PVC behandeln, indem ein Volume mit den auf dem Snapshot vorhandenen Daten erstellt wird. Mit dieser Funktion können Daten regionsübergreifend dupliziert, Testumgebungen erstellt, ein defektes oder defektes Produktionsvolumen

vollständig ersetzt oder bestimmte Dateien und Verzeichnisse abgerufen und auf ein anderes angeschlossenes Volume übertragen werden.

## Verschieben Sie Volumes im Cluster

Storage-Administratoren können Volumes zwischen Aggregaten und Controllern im ONTAP Cluster unterbrechungsfrei für den Storage-Nutzer verschieben. Dieser Vorgang wirkt sich nicht auf die Trident oder das Kubernetes-Cluster aus, sofern es sich bei dem Zielaggregat um ein Aggregat handelt, auf das die SVM von Trident zugreifen kann. Wichtig: Wenn das Aggregat neu zur SVM hinzugefügt wurde, muss das Backend durch erneutes Hinzufügen zur Trident aktualisiert werden. Dies wird dazu führen, dass Trident die SVM erneut inventarisiert, damit das neue Aggregat erkannt wird.

Das Verschieben von Volumes zwischen Back-Ends wird von Trident jedoch nicht automatisch unterstützt. Dies umfasst auch zwischen SVMs im selben Cluster, zwischen Clustern oder auf eine andere Storage-Plattform (selbst dann, wenn es sich bei dem Storage-System um einen mit Trident verbundenen handelt).

Wenn ein Volume an einen anderen Speicherort kopiert wird, kann die Volume-Importfunktion verwendet werden, um aktuelle Volumes in Trident zu importieren.

## Erweitern Sie Volumes

Trident unterstützt die Anpassung von NFS- und iSCSI-PVS. Dadurch können Benutzer ihre Volumes direkt über die Kubernetes-Ebene skalieren. Eine Volume-Erweiterung ist für alle größeren NetApp Storage-Plattformen möglich, einschließlich ONTAP, SolidFire/NetApp HCI und Cloud Volumes Service Back-Ends. Um später eine mögliche Erweiterung zu ermöglichen, setzen Sie `allowVolumeExpansion` in der mit dem Volume verknüpften StorageClass auf `true`. Wenn die Größe des persistenten Volumes geändert werden muss, bearbeiten Sie die `spec.resources.requests.storage` Anmerkung im Persistent Volume Claim auf die erforderliche Volume-Größe. Trident übernimmt automatisch die Anpassung der Größe des Volumes im Storage-Cluster.

## Importieren eines vorhandenen Volumes in Kubernetes

Mit dem Volume-Import kann ein vorhandenes Storage Volume in eine Kubernetes-Umgebung importiert werden. Dies wird derzeit von den `ontap-nas-flexgroup`, `solidfire-san` `azure-netapp-files` und `gcp-cvs` Treibern unterstützt `ontap-nas`. Diese Funktion ist hilfreich, wenn Sie eine vorhandene Applikation in Kubernetes oder während Disaster-Recovery-Szenarien portieren.

Verwenden Sie bei Verwendung der ONTAP und `solidfire-san` Treiber den Befehl `tridentctl import volume <backend-name> <volume-name> -f /path/pvc.yaml` um ein vorhandenes Volume in Kubernetes zu importieren, das von Trident gemanagt werden soll. Die im Befehl des Import-Volumes verwendete PVC-YAML- oder JSON-Datei verweist auf eine Storage-Klasse, die Trident als bereitstellung identifiziert. Stellen Sie bei Verwendung eines NetApp HCI/SolidFire Backend sicher, dass die Volume-Namen eindeutig sind. Wenn die Volume-Namen dupliziert sind, klonen Sie das Volume auf einen eindeutigen Namen, sodass die Funktion zum Importieren des Volumes zwischen diesen Namen unterscheiden kann.

Verwenden Sie bei Verwendung des `azure-netapp-files` Treibers oder `gcp-cvs` den Befehl `tridentctl import volume <backend-name> <volume path> -f /path/pvc.yaml` um das Volume in Kubernetes zu importieren, das von Trident gemanagt werden soll. Dadurch wird eine eindeutige Volumenreferenz sichergestellt.

Wenn der obige Befehl ausgeführt wird, findet Trident das Volume auf dem Backend und liest seine Größe. Die Volume-Größe der konfigurierten PVC wird automatisch hinzugefügt (und bei Bedarf überschrieben). Trident erstellt dann das neue PV und Kubernetes bindet die PVC an das PV.

Wenn ein Container so eingesetzt wurde, dass er das spezifische importierte PVC benötigt, bleibt er in einem ausstehenden Zustand, bis das PVC/PV-Paar über den Volumenimport gebunden ist. Nachdem das PVC/PV-Paar gebunden ist, sollte der Behälter aufstehen, sofern keine anderen Probleme auftreten.

## OpenShift Services implementieren

Die Cluster-Services OpenShift mit großem Mehrwert bieten Clusteradministratoren und den gehosteten Applikationen wichtige Funktionen. Der Storage, den diese Services nutzen, kann mithilfe der Node-lokalen Ressourcen bereitgestellt werden. Dadurch wird jedoch häufig die Kapazität, Performance, Wiederherstellbarkeit und die Nachhaltigkeit des Service begrenzt. Die Nutzung eines Enterprise-Speicher-Arrays zur Bereitstellung der Kapazität für diese Services kann einen erheblich verbesserten Service ermöglichen. OpenShift und die Speicheradministratoren sollten jedoch eng zusammenarbeiten, um die besten Optionen für die einzelnen zu bestimmen. Die Red hat-Dokumentation sollte intensiv genutzt werden, um die Anforderungen zu ermitteln und sicherzustellen, dass die Anforderungen hinsichtlich Größe und Leistung erfüllt werden.

### Registry-Service

Die Bereitstellung und Verwaltung von Speicher für die Registrierung wurde im dokumentiert ["netapp.io" Blog](#).

### Protokollierungsservice

Wie andere OpenShift-Services wird auch der Protokollierungsservice mithilfe von Ansible implementiert. Die Konfigurationsparameter werden von der Inventardatei, auch als Hosts bekannt, bereitgestellt für das Playbook. Es gibt zwei Installationsmethoden: Die Bereitstellung von Protokollierung während der ersten OpenShift-Installation und die Bereitstellung von Protokollierung nach der Installation von OpenShift.



Ab Red hat OpenShift Version 3.9 empfiehlt die offizielle Dokumentation gegen NFS für den Protokollierungsservice, da sie Bedenken hinsichtlich Datenbeschädigung hat. Dies basiert auf Red hat Tests ihrer Produkte. Der ONTAP NFS-Server weist diese Probleme nicht auf und kann problemlos eine Protokollierungsbereitstellung zurücksichern. Letztendlich liegt die Wahl des Protokolls für den Protokollierungsservice bei Ihnen. Ich weiß nur, dass beide bei der Nutzung von NetApp Plattformen hervorragend funktionieren. Es gibt keinen Grund, NFS zu vermeiden, wenn dies Ihre Präferenz ist.

Wenn Sie NFS mit dem Protokollierungsdienst verwenden, müssen Sie die Ansible-Variable so einstellen `openshift_enable_unsupported_configurations`, dass `true` das Installationsprogramm nicht fehlschlägt.

### Los geht's

Der Protokollierungsservice kann optional sowohl für Applikationen als auch für die Kernvorgänge des OpenShift-Clusters selbst implementiert werden. Wenn Sie sich für die Bereitstellung von Operationen entscheiden, werden durch Angabe der Variable `openshift_logging_use_ops` als `true` zwei Instanzen des Dienstes erstellt. Die Variablen, die die Protokollierungsinstanz für Vorgänge steuern, enthalten darin "OPS", während die Instanz für Anwendungen nicht.

Das Konfigurieren der Ansible-Variablen gemäß der Implementierungsmethode ist wichtig, um sicherzustellen, dass der richtige Storage von den zugrunde liegenden Services verwendet wird. Betrachten wir nun die Optionen für die einzelnen Bereitstellungsmethoden.



Die folgenden Tabellen enthalten nur die für die Speicherkonfiguration relevanten Variablen, die sich auf den Protokollierungsservice beziehen. Sie können andere Optionen finden, in "[Logging-Dokumentation von redhat OpenShift](#)" denen Sie entsprechend Ihrer Bereitstellung prüfen, konfigurieren und verwenden sollten.

Die Variablen in der folgenden Tabelle führen dazu, dass im Ansible-Playbook ein PV und eine PVC für den Protokollierungsservice erstellt werden. Diese Details werden verwendet. Diese Methode ist wesentlich weniger flexibel als nach der Installation von OpenShift das Playbook für die Komponenteninstallation zu verwenden. Wenn Sie jedoch vorhandene Volumes zur Verfügung haben, ist dies eine Option.

Variabel	Details
<code>openshift_logging_storage_kind</code>	Legen Sie fest <code>nfs</code> , dass der Installer ein NFS-PV für den Protokollierungsdienst erstellen soll.
<code>openshift_logging_storage_host</code>	Der Hostname oder die IP-Adresse des NFS-Hosts. Diese Einstellung sollte auf die Daten-LIF für Ihre Virtual Machine eingestellt sein.
<code>openshift_logging_storage_nfs_directory</code>	Der Mount-Pfad für den NFS-Export. Wenn das Volume beispielsweise als verbunden ist, <code>/openshift_logging</code> würden Sie diesen Pfad für diese Variable verwenden.
<code>openshift_logging_storage_volume_name</code>	Der Name, z. B. <code>pv_ose_logs</code> , des zu erstellenden PV.
<code>openshift_logging_storage_volume_size</code>	Die Größe des NFS-Exports, zum Beispiel <code>100Gi</code> .

Wenn Ihr OpenShift-Cluster bereits ausgeführt wird und daher Trident implementiert und konfiguriert wurde, kann das Installationsprogramm die Volumes mithilfe der dynamischen Provisionierung erstellen. Die folgenden Variablen müssen konfiguriert werden.

Variabel	Details
<code>openshift_logging_es_pvc_dynamic</code>	Setzen Sie auf „true“, um dynamisch bereitgestellte Volumes zu verwenden.
<code>openshift_logging_es_pvc_storage_class_name</code>	Der Name der Speicherklasse, die in der PVC verwendet wird.
<code>openshift_logging_es_pvc_size</code>	Die Größe des im PVC angeforderten Volumens.
<code>openshift_logging_es_pvc_prefix</code>	Ein Präfix für die vom Protokollierungsservice verwendeten VES.
<code>openshift_logging_es_ops_pvc_dynamic</code>	Legen Sie fest <code>true</code> , dass dynamisch bereitgestellte Volumes für die OPS-Protokollinstanz verwendet werden sollen.
<code>openshift_logging_es_ops_pvc_storage_class_name</code>	Der Name der Speicherklasse für die OPS-Protokollierungsinstanz.
<code>openshift_logging_es_ops_pvc_size</code>	Die Größe der Volume-Anforderung für die OPS-Instanz.
<code>openshift_logging_es_ops_pvc_prefix</code>	Ein Präfix für die OPS-Instanz VES.

## Bereitstellen des Protokollierungs-Stacks

Wenn Sie die Protokollierung als Teil des ursprünglichen OpenShift-Installationsprozesses bereitstellen, müssen Sie nur den Standardprozess für die Bereitstellung befolgen. Ansible konfiguriert und implementiert die erforderlichen Services und OpenShift-Objekte, sodass der Service sobald Ansible abgeschlossen ist.

Wenn Sie die Implementierung jedoch nach der Erstinstallation durchführen, muss das Komponenten-Playbook von Ansible verwendet werden. Dieser Vorgang kann sich bei verschiedenen Versionen von OpenShift geringfügig ändern. Lesen Sie daher unbedingt ["Dokumentation der redhat OpenShift Container Platform 3.11"](#) die Informationen zu Ihrer Version.

## Kennzahlungsservice

Der Kennzahlungsservice liefert dem Administrator wertvolle Informationen zum Status, zur Ressourcenauslastung und zur Verfügbarkeit des OpenShift-Clusters. Dies ist zudem für die automatische Pod-Funktionalität erforderlich, und viele Unternehmen nutzen die Daten des Kennzahlungsservice für ihre Kostenabrechnung und/oder die Anzeige von Applikationen.

Wie beim Protokollierungsservice und OpenShift als Ganzes wird auch Ansible für die Implementierung des Kennzahlungsservice verwendet. Ebenso wie der Protokollierungsservice kann der Metrikservice während der ersten Einrichtung des Clusters oder nach dessen Betrieb mithilfe der Installationsmethode für Komponenten bereitgestellt werden. Die folgenden Tabellen enthalten die Variablen, die für die Konfiguration von persistentem Storage für den Kennzahlungsservice wichtig sind.



Die nachfolgenden Tabellen enthalten nur die Variablen, die für die Storage-Konfiguration relevant sind, da sie sich auf den Kennzahlenservice beziehen. Es gibt viele andere Optionen in der Dokumentation gefunden, die entsprechend Ihrer Bereitstellung überprüft, konfiguriert und verwendet werden sollten.

Variabel	Details
<code>openshift_metrics_storage_kind</code>	Legen Sie fest <code>nfs</code> , dass der Installer ein NFS-PV für den Protokollierungsdienst erstellen soll.
<code>openshift_metrics_storage_host</code>	Der Hostname oder die IP-Adresse des NFS-Hosts. Diese Einstellung sollte auf die Daten-LIF für Ihre SVM eingestellt sein.
<code>openshift_metrics_storage_nfs_directory</code>	Der Mount-Pfad für den NFS-Export. Wenn das Volume beispielsweise als verbunden ist, <code>/openshift_metrics</code> würden Sie diesen Pfad für diese Variable verwenden.
<code>openshift_metrics_storage_volume_name</code>	Der Name, z. B. <code>pv_ose_metrics</code> , des zu erstellenden PV.
<code>openshift_metrics_storage_volume_size</code>	Die Größe des NFS-Exports, zum Beispiel <code>100Gi</code> .

Wenn Ihr OpenShift-Cluster bereits ausgeführt wird und daher Trident implementiert und konfiguriert wurde, kann das Installationsprogramm die Volumes mithilfe der dynamischen Provisionierung erstellen. Die folgenden Variablen müssen konfiguriert werden.



Variabel	Details
<code>openshift_metrics_cassandra_pvc_prefix</code>	Ein Präfix, das für die PVCs der Kennzahlen verwendet wird.
<code>openshift_metrics_cassandra_pvc_size</code>	Die Größe der Volumes, die angefordert werden sollen.
<code>openshift_metrics_cassandra_storage_type</code>	Der Storage-Typ, der für Metriken verwendet werden soll. Dieser muss für Ansible auf dynamisch festgelegt sein, um PVCs mit der entsprechenden Storage-Klasse zu erstellen.
<code>openshift_metrics_cassandra_pvc_storage_class_name</code>	Der Name der zu verwendenden Speicherklasse.

### Bereitstellen des Kennzahlenservice

Implementieren Sie den Service mithilfe von Ansible, wenn Sie die entsprechenden Ansible-Variablen in der Host-/Inventardatei festlegen. Wenn Sie zur Installationszeit OpenShift bereitstellen, wird das PV automatisch erstellt und verwendet. Wenn Sie nach der Installation von OpenShift mit den Komponenten-Playbooks implementieren, erstellt Ansible alle erforderlichen PVCs. Nachdem Trident Storage für sie bereitgestellt hat, kann der Service implementiert werden.

Die oben genannten Variablen und der Prozess für die Bereitstellung können sich mit jeder Version von OpenShift ändern. Überprüfen und befolgen Sie ["Der OpenShift-Implementierungsleitfaden von Red hat"](#) Ihre Version, damit sie für Ihre Umgebung konfiguriert ist.

## Datensicherung und Disaster Recovery

Informieren Sie sich über Sicherungs- und Recovery-Optionen für Trident und Volumes, die mit Trident erstellt wurden. Für jede Applikation mit einer Persistenzanforderung sollte eine Datensicherungs- und Recovery-Strategie eingesetzt werden.

### Replizierung und Recovery mit Trident

Sie können ein Backup erstellen, um Trident im Notfall wiederherzustellen.

#### Replizierung mit Trident

Trident verwendet Kubernetes CRDs zum Speichern und Managen seines eigenen Zustands sowie des Kubernetes-Clusters und etcd zum Speichern seiner Metadaten.

#### Schritte

1. Backup des Kubernetes-Clusters etcd mit ["Kubernetes: Backup eines uscd-Clusters"](#).
2. Platzieren Sie die Backup-Artefakte auf einer FlexVol.



Wir empfehlen, die SVM, auf der sich die FlexVol befindet, mit einer SnapMirror-Beziehung zu einer anderen SVM zu sichern.



## Recovery von Trident

Mit Kubernetes-CRDs und dem Kubernetes-Cluster und Snapshot können Sie Trident wiederherstellen.

### Schritte

1. Mounten Sie von der Ziel-SVM das Volume, das die Kubernetes usw.-Datendateien und Zertifikate enthält, auf dem Host, der als Master-Node eingerichtet wird.
2. Kopieren Sie alle erforderlichen Zertifikate zum Kubernetes-Cluster unter `/etc/kubernetes/pki` und die etcd-Mitgliedsdateien unter `/var/lib/etcd`.
3. Stellen Sie den Kubernetes-Cluster aus dem etcd "[Kubernetes: Wiederherstellung eines uscd-Clusters](#)" -Backup mit `wieder her`.
4. Führen Sie aus `kubectl get crd`, um zu überprüfen, ob alle benutzerdefinierten Trident-Ressourcen eingerichtet sind, und rufen Sie die Trident-Objekte ab, um zu überprüfen, ob alle Daten verfügbar sind.

## SVM-Replizierung und Recovery

Trident kann keine Replizierungsbeziehungen konfigurieren. Der Storage-Administrator kann jedoch zur Replizierung einer SVM verwenden "[ONTAP SnapMirror](#)".

Bei einem Notfall können Sie die SnapMirror Ziel-SVM aktivieren, um die Datenbereitstellung zu starten. Sie können zurück zum primären System wechseln, wenn die Systeme wiederhergestellt sind.

### Über diese Aufgabe

Bei Verwendung der SnapMirror SVM-Replizierungsfunktion sind die folgenden Überlegungen zu beachten:

- Sie sollten für jede SVM ein eigene Back-End mit aktivierter SVM-DR erstellen.
- Konfigurieren Sie die Storage-Klassen so, dass die replizierten Back-Ends nur bei Bedarf ausgewählt werden, um zu vermeiden, dass Volumes ohne Replizierung auf den Back-Ends bereitgestellt werden, die SVM-DR unterstützen.
- Applikationsadministratoren sollten sich über die zusätzlichen Kosten und die Komplexität der Replizierung informieren und ihren Recovery-Plan vor Beginn des Prozesses sorgfältig prüfen.

### SVM-Replizierung

Sie können zum Erstellen der SVM-Replizierungsbeziehung verwenden "[ONTAP: SnapMirror SVM-Replizierung](#)".

Mit SnapMirror können Sie festlegen, was repliziert werden soll. Sie müssen wissen, welche Optionen Sie beim Performing ausgewählt [SVM-Recovery mit Trident](#) haben.

- "[-Identität-bewahren wahr](#)" Replizierung der gesamten SVM-Konfiguration
- "[-Discard-configs Netzwerk](#)" Davon sind LIFs und zugehörige Netzwerkeinstellungen nicht enthalten.
- "[-Identity-preserve false](#)" Repliziert nur die Volumes und die Sicherheitskonfiguration.

### SVM-Recovery mit Trident

Trident erkennt SVM-Fehler nicht automatisch. Bei einem Notfall kann der Administrator das Trident Failover manuell auf die neue SVM initialisieren.

### Schritte

1. Abbrechen geplanter und laufender SnapMirror Übertragungen, Abbrechen der Replizierungsbeziehung, stoppen Sie die Quell-SVM und aktivieren Sie dann die SnapMirror Ziel-SVM.
2. Wenn Sie die SVM-Replikation angegeben `-identity-preserve false` oder `-discard-config network` konfiguriert haben, aktualisieren Sie `managementLIF` und `dataLIF` in der Trident-Backend-Definitionsdatei.
3. Bestätigen `storagePrefix` ist in der Trident-Backend-Definitionsdatei vorhanden. Dieser Parameter kann nicht geändert werden. Wenn Sie das Backend nicht `storagePrefix` mehr verwenden, schlägt das Update fehl.
4. Aktualisieren Sie alle erforderlichen Back-Ends, um den neuen Ziel-SVM-Namen widerzuspiegeln. Verwenden Sie dazu Folgendes:

```
./tridentctl update backend <backend-name> -f <backend-json-file> -n
<namespace>
```

5. Wenn Sie oder `discard-config network` angegeben `-identity-preserve false` haben, müssen Sie alle Anwendungspads zurückspringen.



Wenn Sie angegeben haben `-identity-preserve true`, beginnen alle von Trident bereitgestellten Volumes mit der Bereitstellung von Daten, wenn die Ziel-SVM aktiviert ist.

## Volume-Replizierung und Recovery

Trident kann keine SnapMirror-Replizierungsbeziehungen konfigurieren. Der Storage-Administrator kann jedoch zur Replizierung von Volumes verwenden "[Replizierung und Recovery mit ONTAP SnapMirror](#)", die von Trident erstellt wurden.

Sie können dann importieren Sie die wiederhergestellten Volumes in Trident mit "[Tridentctl-Volumenimport](#)".



Import wird auf, `,ontap-san-economy`` oder ``ontap-flexgroup-economy` Treibern nicht unterstützt `ontap-nas-economy`.

## Snapshot Datensicherung

Sie können Daten schützen und wiederherstellen mit:

- Ein externer Snapshot-Controller und CRDs zum Erstellen von Kubernetes-Volume-Snapshots von persistenten Volumes (PVs).

["Volume Snapshots"](#)

- ONTAP Snapshots zur Wiederherstellung der gesamten Inhalte eines Volumes oder zur Wiederherstellung einzelner Dateien oder LUNs.

["ONTAP Snapshots"](#)

# Sicherheit

## Sicherheit

Mit den hier aufgeführten Empfehlungen können Sie sicherstellen, dass Ihre Trident-Installation sicher ist.

### Ausführen von Trident in seinem eigenen Namespace

Es ist wichtig, zu verhindern, dass Applikationen, Applikationsadministratoren, Benutzer und Managementapplikationen auf Trident-Objektdefinitionen oder Pods zugreifen, um zuverlässigen Storage sicherzustellen und potenzielle böswillige Aktivitäten zu blockieren.

Um die anderen Anwendungen und Benutzer von Trident (`trident``) zu trennen, installieren Sie Trident immer in seinem eigenen Kubernetes Namespace ). Wenn Trident in seinen eigenen Namespace gelegt wird, wird sichergestellt, dass nur das Kubernetes-Administratorpersonal Zugriff auf den Trident Pod hat und auf die Artefakte (z. B. Backend- und CHAP-Geheimnisse, falls zutreffend), die in den `nameschritt-CRD`-Objekten gespeichert sind. Sie sollten sicherstellen, dass nur Administratoren Zugriff auf den Trident-Namespace und damit auf die ``tridentctl` Anwendung haben.

### Verwenden Sie CHAP-Authentifizierung mit ONTAP SAN Back-Ends

Trident unterstützt die CHAP-basierte Authentifizierung für ONTAP-SAN-Workloads (unter Verwendung der `ontap-san` Treiber und `ontap-san-economy`). NetApp empfiehlt zur Authentifizierung zwischen einem Host und dem Storage-Back-End die Verwendung von bidirektionalem CHAP mit Trident.

Für ONTAP-Back-Ends, die die SAN-Speichertreiber verwenden, kann Trident bidirektionales CHAP einrichten und CHAP-Benutzernamen und -Schlüssel über `tridentctl` verwalten. Weitere Informationen zur Trident Konfiguration von CHAP auf ONTAP-Back-Ends finden Sie unter "[Vorbereiten der Back-End-Konfiguration mit ONTAP-SAN-Treibern](#)".

### Verwenden Sie CHAP-Authentifizierung mit NetApp HCI und SolidFire Back-Ends

NetApp empfiehlt die Implementierung von bidirektionalem CHAP, um die Authentifizierung zwischen einem Host und den NetApp HCI und SolidFire Back-Ends zu gewährleisten. Trident verwendet ein geheimes Objekt, das zwei CHAP-Passwörter pro Mandant enthält. Wenn Trident installiert ist, verwaltet es die CHAP-Schlüssel und speichert sie in einem `tridentvolume` CR-Objekt für das jeweilige PV. Wenn Sie ein PV erstellen, verwendet Trident die CHAP-Schlüssel, um eine iSCSI-Sitzung zu initiieren und über CHAP mit dem NetApp HCI- und SolidFire-System zu kommunizieren.



Die Volumes, die von Trident erstellt werden, sind keiner Volume Access Group zugeordnet.

### Verwenden Sie Trident mit NVE und NAE

NetApp ONTAP bietet Verschlüsselung ruhender Daten zum Schutz sensibler Daten, wenn eine Festplatte gestohlen, zurückgegeben oder einer neuen Verwendung zugewiesen wird. Weitere Informationen finden Sie unter "[NetApp Volume Encryption Übersicht konfigurieren](#)".

- Wenn auf dem Backend NAE aktiviert ist, wird jedes in Trident bereitgestellte Volume NAE-aktiviert.
- Wenn NAE im Back-End nicht aktiviert ist, wird jedes in Trident bereitgestellte Volume NVE-aktiviert, es sei denn, Sie setzen in der Backend-Konfiguration das NVE-Verschlüsselungsflag auf `false`.

Volumes, die in Trident auf einem NAE-fähigen Back-End erstellt wurden, müssen mit NVE oder NAE verschlüsselt werden.



- Sie können in der Trident-Backend-Konfiguration das NVE-Verschlüsselungsflag auf `true` setzen, um die NAE-Verschlüsselung außer Kraft zu setzen und für jedes Volume einen bestimmten Verschlüsselungsschlüssel zu verwenden.
- Wenn Sie das NVE-Verschlüsselungsflag auf ein NAE-fähiges Back-End setzen `false`, wird ein NAE-fähiges Volume erstellt. Sie können die NAE-Verschlüsselung nicht deaktivieren, indem Sie das NVE-Verschlüsselungsflag auf `false`.

- Sie können ein NVE Volume manuell in Trident erstellen, indem Sie das NVE Verschlüsselungs-Flag explizit auf setzen `true`.

Weitere Informationen zu Back-End-Konfigurationsoptionen finden Sie unter:

- ["SAN-Konfigurationsoptionen von ONTAP"](#)
- ["NAS-Konfigurationsoptionen von ONTAP"](#)

## Linux Unified Key Setup (LUKS)

Sie können Linux Unified Key Setup (LUKS) aktivieren, um ONTAP SAN und ONTAP SAN ECONOMY Volumes auf Trident zu verschlüsseln. Trident unterstützt die Rotation der Passphrase und die Volume-Erweiterung für LUKS-verschlüsselte Volumes.

In Trident verwenden LUKS-verschlüsselte Volumes den `aes-xts-plain64`-Cypher und `-Modus`, wie von empfohlen ["NIST"](#).

### Bevor Sie beginnen

- Worker Nodes müssen `cryptsetup 2.1` oder höher (aber unter `3.0`) installiert sein. Weitere Informationen finden Sie unter ["Gitlab: Cryptsetup"](#).
- Aus Performance-Gründen wird empfohlen, dass Arbeiterknoten Advanced Encryption Standard New Instructions (AES-NI) unterstützen. Führen Sie den folgenden Befehl aus, um die Unterstützung von AES-NI zu überprüfen:

```
grep "aes" /proc/cpuinfo
```

Wenn nichts zurückgegeben wird, unterstützt Ihr Prozessor nicht AES-NI. Weitere Informationen zu AES-NI finden Sie unter ["Intel: Advanced Encryption Standard Instructions \(AES-NI\)"](#).

### Aktivieren Sie die LUKS-Verschlüsselung

Sie können die Verschlüsselung auf Host-Seite pro Volume mithilfe von Linux Unified Key Setup (LUKS) für ONTAP SAN und ONTAP SAN ECONOMY Volumes aktivieren.

#### Schritte

1. Definieren Sie LUKS-Verschlüsselungsattribute in der Backend-Konfiguration. Weitere Informationen zu den Back-End-Konfigurationsoptionen für ONTAP-SAN finden Sie unter ["SAN-Konfigurationsoptionen von ONTAP"](#).

```

"storage": [
  {
    "labels":{"luks": "true"},
    "zone":"us_east_1a",
    "defaults": {
      "luksEncryption": "true"
    }
  },
  {
    "labels":{"luks": "false"},
    "zone":"us_east_1a",
    "defaults": {
      "luksEncryption": "false"
    }
  },
]

```

2. `parameters.selector` Zum Definieren der Speicherpools mit LUKS-Verschlüsselung. Beispiel:

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: csi.trident.netapp.io
parameters:
  selector: "luks=true"
  csi.storage.k8s.io/node-stage-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}

```

3. Erstellen Sie ein Geheimnis, das die LUKS-Passphrase enthält. Beispiel:

```

kubectl -n trident create -f luks-pvc1.yaml
apiVersion: v1
kind: Secret
metadata:
  name: luks-pvc1
stringData:
  luks-passphrase-name: A
  luks-passphrase: secretA

```

### Einschränkungen

LUKS-verschlüsselte Volumes können die ONTAP Deduplizierung und Komprimierung nicht nutzen.

## Back-End-Konfiguration zum Importieren von LUKS-Volumes

Um ein LUKS-Volume zu importieren, müssen Sie auf dem Backend festlegen `luksEncryption`. Die `luksEncryption` Option zeigt Trident an, ob das Volume LUKS-konform ist (`true`) oder nicht LUKS-konform (`false`), wie im folgenden Beispiel gezeigt.

```
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: trident_svm
username: admin
password: password
defaults:
  luksEncryption: 'true'
  spaceAllocation: 'false'
  snapshotPolicy: default
  snapshotReserve: '10'
```

## PVC-Konfiguration für den Import von LUKS-Volumes

Um LUKS-Volumes dynamisch zu importieren, setzen Sie die Beschriftung `trident.netapp.io/luksEncryption` auf `true` und fügen Sie eine LUKS-fähige Storage-Klasse in die PVC ein, wie in diesem Beispiel gezeigt.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: luks-pvc
  namespace: trident
  annotations:
    trident.netapp.io/luksEncryption: "true"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: luks-sc
```

## Eine LUKS-Passphrase drehen

Sie können die LUKS-Passphrase drehen und die Drehung bestätigen.



Vergessen Sie keine Passphrase, bis Sie überprüft haben, dass sie nicht mehr von einem Volume, einem Snapshot oder einem geheimen Schlüssel referenziert wird. Wenn eine referenzierte Passphrase verloren geht, können Sie das Volume möglicherweise nicht mounten und die Daten bleiben verschlüsselt und unzugänglich.

### Über diese Aufgabe

DIE Drehung der LUKS-Passphrase erfolgt, wenn ein Pod, das das Volume bindet, nach der Angabe einer neuen LUKS-Passphrase erstellt wird. Wenn ein neuer Pod erstellt wird, vergleicht Trident die LUKS-Passphrase auf dem Volume mit der aktiven Passphrase im Secret.

- Wenn die Passphrase auf dem Volume nicht mit der aktiven Passphrase im Geheimnis übereinstimmt, erfolgt die Drehung.
- Wenn die Passphrase auf dem Volume mit der aktiven Passphrase im Secret übereinstimmt, wird der `previous-luks-passphrase` Parameter ignoriert.

### Schritte

1. Fügen Sie die Parameter `node-publish-secret-namespace` StorageClass hinzu `node-publish-secret-name`. Beispiel:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-san
provisioner: csi.trident.netapp.io
parameters:
  trident.netapp.io/backendType: "ontap-san"
  csi.storage.k8s.io/node-stage-secret-name: luks
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
  csi.storage.k8s.io/node-publish-secret-name: luks
  csi.storage.k8s.io/node-publish-secret-namespace: ${pvc.namespace}
```

2. Identifizieren Sie vorhandene Passphrases auf dem Volume oder Snapshot.

### Datenmenge

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames: ["A"]
```

### Snapshot

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames: ["A"]
```

3. Aktualisieren Sie das LUKS-Geheimnis für das Volume, um die neuen und vorherigen Passphrases anzugeben. Stellen Sie sicher, dass `previous-luks-passphrase-name`previous-luks-passphrase`` die vorherige Passphrase übereinstimmt.

```
apiVersion: v1
kind: Secret
metadata:
  name: luks-pvc1
stringData:
  luks-passphrase-name: B
  luks-passphrase: secretB
  previous-luks-passphrase-name: A
  previous-luks-passphrase: secretA
```

4. Erstellen Sie einen neuen Pod, der das Volume montiert. Dies ist erforderlich, um die Rotation zu initiieren.
5. Überprüfen Sie, ob die Passphrase gedreht wurde.

#### Datenmenge

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames: ["B"]
```

#### Snapshot

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames: ["B"]
```

#### Ergebnisse

Die Passphrase wurde gedreht, wenn nur die neue Passphrase auf dem Volume und dem Snapshot zurückgegeben wird.



Werden beispielsweise zwei Passphrasen zurückgegeben, `luksPassphraseNames: ["B", "A"]` ist die Rotation unvollständig. Sie können einen neuen Pod auslösen, um zu versuchen, die Rotation abzuschließen.

#### Aktivieren Sie die Volume-Erweiterung

Sie können Volume-Erweiterung auf einem LUKS-verschlüsselten Volume aktivieren.

#### Schritte

1. Aktivieren Sie das `CSINodeExpandSecret` Feature Gate (Beta 1.25+). Weitere Informationen finden Sie unter ["Kubernetes 1.25: Verwenden Sie Secrets zur Node-gesteuerten Erweiterung von CSI Volumes"](#).



2. Fügen Sie die Parameter `node-expand-secret-namespace` StorageClass hinzu `node-expand-secret-name`. Beispiel:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: csi.trident.netapp.io
parameters:
  selector: "luks=true"
  csi.storage.k8s.io/node-stage-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
  csi.storage.k8s.io/node-expand-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-expand-secret-namespace: ${pvc.namespace}
allowVolumeExpansion: true
```

## Ergebnisse

Wenn Sie die Online-Speichererweiterung initiieren, gibt das Kubelet die entsprechenden Zugangsdaten an den Treiber weiter.

## Kerberos Verschlüsselung während der Übertragung

Mit der Kerberos-Verschlüsselung während der Übertragung können Sie die Datensicherheit verbessern, indem Sie die Verschlüsselung für den Datenverkehr zwischen dem verwalteten Cluster und dem Storage-Back-End aktivieren.

Trident unterstützt Kerberos Verschlüsselung für ONTAP als Storage-Back-End:

- **On-Premise-ONTAP** – Trident unterstützt Kerberos-Verschlüsselung über NFSv3- und NFSv4-Verbindungen von Red hat OpenShift und Upstream-Kubernetes-Clustern zu lokalen ONTAP-Volumes.

Sie können Snapshots, Klone, schreibgeschütztes Klonen und Importieren von Volumes mit NFS-Verschlüsselung.

## Konfiguration der in-Flight-Kerberos-Verschlüsselung mit lokalen ONTAP-Volumes

Sie können die Kerberos-Verschlüsselung auf dem Storage-Datenverkehr zwischen dem verwalteten Cluster und einem lokalen ONTAP-Storage-Back-End aktivieren.



Kerberos-Verschlüsselung für NFS-Datenverkehr mit On-Premise ONTAP-Storage-Back-Ends wird nur mithilfe des Speichertreibers unterstützt `ontap-nas`.

## Bevor Sie beginnen

- Stellen Sie sicher, dass Sie Zugriff auf das Dienstprogramm haben `tridentctl`.
- Stellen Sie sicher, dass Sie Administratorzugriff auf das ONTAP Storage Back-End haben.
- Stellen Sie sicher, dass Sie den Namen des Volumes oder der Volumes kennen, die Sie über das ONTAP-Speicher-Back-End freigeben werden.

- Stellen Sie sicher, dass Sie die ONTAP-Storage-VM auf die Unterstützung der Kerberos-Verschlüsselung für NFS-Volumes vorbereitet haben. Anweisungen hierzu finden Sie unter ["Aktivieren Sie Kerberos auf einer Daten-LIF"](#) .
- Stellen Sie sicher, dass alle NFSv4-Volumes, die Sie mit Kerberos-Verschlüsselung verwenden, korrekt konfiguriert sind. Weitere Informationen finden Sie im Abschnitt NetApp NFSv4-Domänenkonfiguration (Seite 13) der ["NetApp Leitfaden zu NFSv4-Verbesserungen und Best Practices"](#).

### ONTAP-Exportrichtlinien hinzufügen oder ändern

Sie müssen bestehenden ONTAP-Exportrichtlinien Regeln hinzufügen oder neue Exportrichtlinien erstellen, die Kerberos-Verschlüsselung für das ONTAP Storage-VM-Root-Volume sowie alle mit dem Upstream-Kubernetes-Cluster gemeinsam genutzten ONTAP-Volumes unterstützen. Die von Ihnen hinzugefügten Regeln für die Exportrichtlinie oder neu erstellte Richtlinien für den Export müssen die folgenden Zugriffsprotokolle und Zugriffsberechtigungen unterstützen:

#### Zugriffsprotokolle

Konfigurieren Sie die Exportrichtlinie mit NFS-, NFSv3- und NFSv4-Zugriffsprotokollen.

#### Zugriffsdetails

Sie können eine von drei verschiedenen Versionen der Kerberos-Verschlüsselung konfigurieren, je nach Ihren Anforderungen für das Volume:

- **Kerberos 5** - (Authentifizierung und Verschlüsselung)
- **Kerberos 5i** - (Authentifizierung und Verschlüsselung mit Identitätsschutz)
- **Kerberos 5p** - (Authentifizierung und Verschlüsselung mit Identitäts- und Datenschutz)

Konfigurieren Sie die ONTAP-Exportrichtlinie mit den entsprechenden Zugriffsberechtigungen. Wenn beispielsweise Cluster die NFS-Volumes mit einer Mischung aus Kerberos 5i- und Kerberos 5p-Verschlüsselung mounten, verwenden Sie die folgenden Zugriffseinstellungen:

Typ	Schreibgeschützter Zugriff	Lese-/Schreibzugriff	Superuser-Zugriff
UNIX	Aktiviert	Aktiviert	Aktiviert
Kerberos 5i	Aktiviert	Aktiviert	Aktiviert
Kerberos 5p	Aktiviert	Aktiviert	Aktiviert

Informationen zum Erstellen von ONTAP Exportrichtlinien und Exportrichtlinienregeln finden Sie in der folgenden Dokumentation:

- ["Erstellen Sie eine Exportrichtlinie"](#)
- ["Fügen Sie eine Regel zu einer Exportrichtlinie hinzu"](#)

### Erstellen eines Storage-Backends

Sie können eine Trident-Storage-Back-End-Konfiguration mit Kerberos Verschlüsselungsfunktion erstellen.

#### Über diese Aufgabe

Wenn Sie eine Speicher-Back-End-Konfigurationsdatei erstellen, die die Kerberos-Verschlüsselung konfiguriert, können Sie eine von drei verschiedenen Versionen der Kerberos-Verschlüsselung mithilfe des Parameters angeben `spec.nfsMountOptions` :

- spec.nfsMountOptions: sec=krb5 (Authentifizierung und Verschlüsselung)
- spec.nfsMountOptions: sec=krb5i (Authentifizierung und Verschlüsselung mit Identitätsschutz)
- spec.nfsMountOptions: sec=krb5p (Authentifizierung und Verschlüsselung mit Identitäts- und Datenschutz)

Geben Sie nur eine Kerberos-Ebene an. Wenn Sie in der Parameterliste mehr als eine Kerberos-Verschlüsselungsebene angeben, wird nur die erste Option verwendet.

## Schritte

1. Erstellen Sie auf dem verwalteten Cluster mithilfe des folgenden Beispiels eine Speicher-Back-End-Konfigurationsdatei. Ersetzen Sie Werte in Klammern <> durch Informationen aus Ihrer Umgebung:

```

apiVersion: v1
kind: Secret
metadata:
  name: backend-ontap-nas-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-ontap-nas
spec:
  version: 1
  storageDriverName: "ontap-nas"
  managementLIF: <STORAGE_VM_MGMT_LIF_IP_ADDRESS>
  dataLIF: <PROTOCOL_LIF_FQDN_OR_IP_ADDRESS>
  svm: <STORAGE_VM_NAME>
  username: <STORAGE_VM_USERNAME_CREDENTIAL>
  password: <STORAGE_VM_PASSWORD_CREDENTIAL>
  nasType: nfs
  nfsMountOptions: ["sec=krb5i"] #can be krb5, krb5i, or krb5p
  qtreesPerFlexvol:
  credentials:
    name: backend-ontap-nas-secret

```

2. Verwenden Sie die Konfigurationsdatei, die Sie im vorherigen Schritt erstellt haben, um das Backend zu erstellen:

```
tridentctl create backend -f <backend-configuration-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.

### Erstellen Sie eine Speicherklasse

Sie können eine Storage-Klasse für die Bereitstellung von Volumes mit Kerberos-Verschlüsselung erstellen.

### Über diese Aufgabe

Wenn Sie ein Storage-Klasse-Objekt erstellen, können Sie mit dem Parameter eine von drei verschiedenen Versionen der Kerberos-Verschlüsselung angeben `mountOptions` :

- `mountOptions: sec=krb5` (Authentifizierung und Verschlüsselung)
- `mountOptions: sec=krb5i` (Authentifizierung und Verschlüsselung mit Identitätsschutz)
- `mountOptions: sec=krb5p` (Authentifizierung und Verschlüsselung mit Identitäts- und Datenschutz)

Geben Sie nur eine Kerberos-Ebene an. Wenn Sie in der Parameterliste mehr als eine Kerberos-Verschlüsselungsebene angeben, wird nur die erste Option verwendet. Wenn die in der Storage-Backend-Konfiguration angegebene Verschlüsselungsebene von der Ebene abweicht, die Sie im Storage-Klasse-Objekt angeben, hat das Storage-Klasse-Objekt Vorrang.

### Schritte

1. Erstellen Sie mithilfe des folgenden Beispiels ein StorageClass-Kubernetes-Objekt:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-nas-sc
provisioner: csi.trident.netapp.io
mountOptions: ["sec=krb5i"] #can be krb5, krb5i, or krb5p
parameters:
  backendType: "ontap-nas"
  storagePools: "ontapnas_pool"
  trident.netapp.io/nasType: "nfs"
allowVolumeExpansion: True
```

2. Speicherklasse erstellen:

```
kubectl create -f sample-input/storage-class-ontap-nas-sc.yaml
```

3. Stellen Sie sicher, dass die Storage-Klasse erstellt wurde:

```
kubectl get sc ontap-nas-sc
```

Sie sollten eine Ausgabe wie die folgende sehen:

```
NAME          PROVISIONER          AGE
ontap-nas-sc  csi.trident.netapp.io  15h
```

### Bereitstellen von Volumes

Nachdem Sie ein Storage-Back-End und eine Storage-Klasse erstellt haben, können Sie nun ein Volume bereitstellen. Anweisungen hierzu finden Sie unter ["Bereitstellen eines Volumes"](#).

### Konfiguration der Verschlüsselung von Kerberos während der Übertragung mit Azure NetApp Files Volumes

Sie können die Kerberos-Verschlüsselung für den Storage-Datenverkehr zwischen dem gemanagten Cluster und einem einzelnen Azure NetApp Files Storage-Back-End oder einem virtuellen Pool von Azure NetApp Files Storage-Back-Ends aktivieren.

#### Bevor Sie beginnen

- Stellen Sie sicher, dass Sie Trident auf dem verwalteten Red hat OpenShift-Cluster aktiviert haben.
- Stellen Sie sicher, dass Sie Zugriff auf das Dienstprogramm haben `tridentctl`.
- Stellen Sie sicher, dass Sie das Azure NetApp Files-Speicher-Back-End für die Kerberos-Verschlüsselung vorbereitet haben, indem Sie die Anforderungen beachten und die Anweisungen in befolgen ["Azure NetApp Files-Dokumentation"](#).
- Stellen Sie sicher, dass alle NFSv4-Volumes, die Sie mit Kerberos-Verschlüsselung verwenden, korrekt konfiguriert sind. Weitere Informationen finden Sie im Abschnitt NetApp NFSv4-Domänenkonfiguration (Seite 13) der ["NetApp Leitfaden zu NFSv4-Verbesserungen und Best Practices"](#).

#### Erstellen eines Storage-Backends

Sie können eine Azure NetApp Files-Storage-Back-End-Konfiguration mit Kerberos Verschlüsselungsfunktionen erstellen.

#### Über diese Aufgabe

Wenn Sie eine Speicher-Backend-Konfigurationsdatei erstellen, die die Kerberos-Verschlüsselung konfiguriert, können Sie sie so definieren, dass sie auf einer der zwei möglichen Ebenen angewendet werden sollte:

- Die **Speicher-Backend-Ebene** mit dem `spec.kerberos` Feld
- Die **virtuelle Pool-Ebene** mit dem `spec.storage.kerberos` Feld

Wenn Sie die Konfiguration auf der Ebene des virtuellen Pools definieren, wird der Pool mithilfe der Beschriftung in der Speicherklasse ausgewählt.

Auf beiden Ebenen können Sie eine von drei verschiedenen Versionen der Kerberos-Verschlüsselung angeben:

- `kerberos: sec=krb5` (Authentifizierung und Verschlüsselung)

- kerberos: sec=krb5i (Authentifizierung und Verschlüsselung mit Identitätsschutz)
- kerberos: sec=krb5p (Authentifizierung und Verschlüsselung mit Identitäts- und Datenschutz)

### Schritte

1. Erstellen Sie auf dem verwalteten Cluster eine Speicher-Backend-Konfigurationsdatei mit einem der folgenden Beispiele, je nachdem, wo Sie das Speicher-Back-End definieren müssen (Speicher-Back-End-Ebene oder virtuelle Pool-Ebene). Ersetzen Sie Werte in Klammern <> durch Informationen aus Ihrer Umgebung:

### Beispiel auf Storage-Back-End-Ebene

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-secret
```

### Beispiel auf Ebene des virtuellen Pools

```

apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  storage:
    - labels:
      type: encryption
      kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-secret

```

2. Verwenden Sie die Konfigurationsdatei, die Sie im vorherigen Schritt erstellt haben, um das Backend zu erstellen:

```
tridentctl create backend -f <backend-configuration-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.

### Erstellen Sie eine Speicherklasse

Sie können eine Storage-Klasse für die Bereitstellung von Volumes mit Kerberos-Verschlüsselung erstellen.

### Schritte

1. Erstellen Sie mithilfe des folgenden Beispiels ein StorageClass-Kubernetes-Objekt:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-nfs
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "nfs"
  selector: "type=encryption"
```

2. Speicherklasse erstellen:

```
kubectl create -f sample-input/storage-class-sc-nfs.yaml
```

3. Stellen Sie sicher, dass die Storage-Klasse erstellt wurde:

```
kubectl get sc -sc-nfs
```

Sie sollten eine Ausgabe wie die folgende sehen:

NAME	PROVISIONER	AGE
sc-nfs	csi.trident.netapp.io	15h

### Bereitstellen von Volumes

Nachdem Sie ein Storage-Back-End und eine Storage-Klasse erstellt haben, können Sie nun ein Volume bereitstellen. Anweisungen hierzu finden Sie unter "[Bereitstellen eines Volumes](#)".

# Sicherung von Applikationen mit Trident-Schutz

## Weitere Informationen zu Trident Protect

NetApp Trident Protect bietet erweiterte Funktionen für das Applikations-Datenmanagement, mit denen die Funktionalität und Verfügbarkeit zustandsbehafteter Kubernetes-Applikationen auf der Basis von NetApp ONTAP Storage-Systemen und der NetApp Trident CSI Storage-bereitstellung verbessert werden. Trident Protect vereinfacht das Management, die Sicherung und die Verschiebung von Container-Workloads zwischen Public Clouds und On-Premises-Umgebungen. Außerdem bietet es Automatisierungsfunktionen über seine API und CLI.

Sie können Anwendungen mit Trident Protect schützen, indem Sie benutzerdefinierte Ressourcen (CRS) erstellen oder die Trident Protect CLI verwenden.

### Was kommt als Nächstes?

Informieren Sie sich vor der Installation über die Trident Protect-Anforderungen:

- ["Trident Protect – die Anforderungen"](#)

## Installieren Sie Trident Protect

### Trident Protect – die Anforderungen

Prüfen Sie zunächst die Bereitschaft Ihrer Betriebsumgebung, Applikations-Cluster, Applikationen und Lizenzen. Stellen Sie sicher, dass Ihre Umgebung diese Anforderungen für den Einsatz und Betrieb von Trident Protect erfüllt.

### Trident schützen die Kompatibilität von Kubernetes-Clustern

Trident Protect ist mit einer Vielzahl vollständig gemanagter und selbst gemanagter Kubernetes-Angebote kompatibel, darunter:

- Amazon Elastic Kubernetes Service (EKS)
- Google Kubernetes Engine (GKE)
- Microsoft Azure Kubernetes Service (AKS)
- Red hat OpenShift
- SUSE Rancher
- VMware Tanzu Portfolio
- Vorgelagerte Kubernetes-Systeme



Stellen Sie sicher, dass der Cluster, auf dem Sie Trident Protect installieren, mit einem laufenden Snapshot-Controller und den zugehörigen CRDs konfiguriert ist. Informationen zum Installieren eines Snapshot-Controllers finden Sie unter ["Diese Anweisungen"](#).

## Trident schützen die Kompatibilität mit dem Storage-Back-End

Trident Protect unterstützt die folgenden Storage-Back-Ends:

- Amazon FSX für NetApp ONTAP
- Cloud Volumes ONTAP
- ONTAP Storage-Arrays
- Google Cloud NetApp Volumes
- Azure NetApp Dateien

Stellen Sie sicher, dass Ihr Storage-Back-End die folgenden Anforderungen erfüllt:

- Vergewissern Sie sich, dass mit dem Cluster verbundener NetApp-Storage Astra Trident 24.02 oder neuer verwendet (Trident 24.10 wird empfohlen).
  - Wenn Astra Trident älter als die Version 24.06.1 ist und Sie die Disaster-Recovery-Funktion von NetApp SnapMirror nutzen möchten, müssen Sie die Astra Control Provisioner manuell aktivieren.
- Stellen Sie sicher, dass Sie über die neueste Astra Control Provisioner-Version verfügen (standardmäßig installiert und aktiviert ab Astra Trident 24.06.1).
- Stellen Sie sicher, dass Sie über ein NetApp ONTAP Storage-Back-End verfügen.
- Stellen Sie sicher, dass Sie einen Objekt-Storage-Bucket zum Speichern von Backups konfiguriert haben.
- Erstellen Sie alle Anwendungsnamepaces, die Sie für Anwendungen oder Verwaltungsvorgänge von Anwendungsdaten verwenden möchten. Trident Protect erstellt diese Namespaces nicht für Sie. Wenn Sie einen nicht vorhandenen Namespace in einer benutzerdefinierten Ressource angeben, schlägt der Vorgang fehl.

### Anforderungen für nas-Economy-Volumen

Trident Protect unterstützt Backup- und Restore-Vorgänge auf Volumes mit nas-Wirtschaft. Snapshots, Klone und SnapMirror-Replizierung auf nas-Economy Volumes werden derzeit nicht unterstützt. Sie müssen ein Snapshot-Verzeichnis für jedes nas-Economy-Volumen aktivieren, das Sie mit Trident Protect verwenden möchten.



Einige Anwendungen sind nicht mit Volumes kompatibel, die ein Snapshot-Verzeichnis verwenden. Für diese Anwendungen müssen Sie das Snapshot-Verzeichnis ausblenden, indem Sie den folgenden Befehl auf dem ONTAP-Speichersystem ausführen:

```
nfs modify -vserver <svm> -v3-hide-snapshot enabled
```

Sie können das Snapshot-Verzeichnis aktivieren, indem Sie den folgenden Befehl für jedes nas-Economy-Volume ausführen und durch die UUID des Volumes ersetzen <volume-UUID>, das Sie ändern möchten:

```
tridentctl update volume <volume-UUID> --snapshot-dir=true --pool-level  
=true -n trident
```



Sie können Snapshot-Verzeichnisse standardmäßig für neue Volumes aktivieren, indem Sie die Trident-Backend-Konfigurationsoption auf `true` setzen `snapshotDir`. Vorhandene Volumes werden nicht beeinträchtigt.

## Datensicherung mit KubeVirt VMs

Trident Protect 24.10 und 24.10.1 und neuere Versionen weisen beim Schutz von Applikationen auf KubeVirt VMs unterschiedliche Verhaltensweisen auf. Bei beiden Versionen können Sie das Einfrieren und Entsperrn des Dateisystems während der Datenschutzvorgänge aktivieren oder deaktivieren.

Für alle Trident Protect-Versionen müssen Sie möglicherweise dem Application Namespace privilegierte Berechtigungen erteilen, um die Funktion zum automatischen Einfrieren in OpenShift-Umgebungen zu aktivieren oder zu deaktivieren. Beispiel:



```
oc adm policy add-scc-to-user privileged -z default -n  
<application-namespace>
```

### Trident Protect 24.10

Trident Protect 24.10 stellt nicht automatisch einen konsistenten Zustand für KubeVirt VM-Dateisysteme während Datensicherheitsvorgängen sicher. Wenn Sie Ihre KubeVirt VM-Daten mit Trident Protect 24.10 schützen möchten, müssen Sie die Freeze/Unfreeze-Funktion für die Dateisysteme vor dem Datenschutzvorgang manuell aktivieren. Dadurch wird sichergestellt, dass sich die Dateisysteme in einem konsistenten Zustand befinden.

Sie können Trident Protect 24.10 so konfigurieren, dass das Einfrieren und Entfrieren des VM-Dateisystems während der Datenschutzvorgänge verwaltet wird, indem ["Konfiguration der Virtualisierung"](#) Sie dann den folgenden Befehl verwenden:

```
kubectl set env deployment/trident-protect-controller-manager  
NEPTUNE_VM_FREEZE=true -n trident-protect
```

### Trident Protect 24.10.1 und neuer

Ab Trident Protect 24.10.1 friert Trident Protect KubeVirt-Dateisysteme während der Datensicherheitsvorgänge automatisch ein und entfriert diese. Optional können Sie dieses automatische Verhalten mit dem folgenden Befehl deaktivieren:

```
kubectl set env deployment/trident-protect-controller-manager  
NEPTUNE_VM_FREEZE=false -n trident-protect
```

## Anforderungen für die SnapMirror-Replizierung

NetApp SnapMirror steht zusammen mit Trident Protect für die folgenden ONTAP Lösungen zur Verfügung:

- NetApp ASA
- NetApp AFF
- NetApp FAS

- NetApp ONTAP Select
- NetApp Cloud Volumes ONTAP
- Amazon FSX für NetApp ONTAP

### ONTAP-Cluster-Anforderungen für die SnapMirror-Replizierung

Stellen Sie sicher, dass Ihr ONTAP Cluster die folgenden Anforderungen erfüllt, wenn Sie SnapMirror Replizierung nutzen möchten:

- **Astra Control Provisioner oder Trident:** Astra Control Provisioner oder Trident müssen sowohl auf den Quell- als auch auf den Ziel-Kubernetes-Clustern vorhanden sein, die ONTAP als Backend verwenden. Trident Protect unterstützt die Replikation mit NetApp SnapMirror-Technologie unter Verwendung von Storage-Klassen, die von den folgenden Treibern gesichert werden:
  - `ontap-nas`
  - `ontap-san`
- **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-Überlegungen für die SnapMirror-Replizierung

Stellen Sie sicher, dass Ihre Umgebung die folgenden Anforderungen erfüllt, wenn Sie Storage-Back-End-Peering verwenden möchten:

- **Cluster und SVM:** Die ONTAP Speicher-Back-Ends müssen aktiviert werden. Weitere Informationen finden Sie unter "[Übersicht über Cluster- und SVM-Peering](#)".



Vergewissern Sie sich, dass die in der Replizierungsbeziehung zwischen zwei ONTAP-Clustern verwendeten SVM-Namen eindeutig sind.

- **Astra Control Provisioner oder Trident und SVM:** Die Remote-SVMs müssen für die Astra Control Bereitstellung oder die Trident im Ziel-Cluster verfügbar sein.
- **Managed Back-Ends:** Sie müssen ONTAP-Speicher-Back-Ends in Trident Protect hinzufügen und managen, um eine Replikationsbeziehung zu erstellen.
- **NVMe über TCP:** Trident Protect unterstützt keine NetApp SnapMirror-Replizierung für Storage-Back-Ends, die das NVMe-over-TCP-Protokoll verwenden.

### Trident/ONTAP-Konfiguration für SnapMirror-Replikation

Trident Protect setzt voraus, dass Sie mindestens ein Storage-Back-End konfigurieren, das die Replizierung sowohl für die Quell- als auch für Ziel-Cluster unterstützt. Wenn die Quell- und Ziel-Cluster identisch sind, sollte die Zielanwendung ein anderes Speicher-Back-End als die Quellenanwendung verwenden, um die beste Ausfallsicherheit zu erreichen.

## Installieren und konfigurieren Sie Trident Protect

Wenn Ihre Umgebung die Anforderungen für Trident Protect erfüllt, können Sie mit den folgenden Schritten Trident Protect auf Ihrem Cluster installieren. Sie können Trident Protect von NetApp beziehen oder es von Ihrer eigenen privaten Registrierung installieren. Die Installation von einer privaten Registrierung ist hilfreich, wenn Ihr Cluster

nicht auf das Internet zugreifen kann.



Standardmäßig erfasst Trident Protect Support-Informationen, die bei allen NetApp-Supportfällen helfen, die Sie öffnen können, einschließlich Protokollen, Kennzahlen und Topologieinformationen zu Clustern und gemanagten Applikationen. Trident Protect sendet diese Support-Pakete nach einem täglichen Zeitplan an NetApp. Sie können diese Support-Bundle-Sammlung optional deaktivieren, wenn Sie Trident Protect installieren. Sie können jederzeit manuell ["Generieren Sie ein Support-Bundle"](#)wählen.

### **Installieren Sie Trident Protect**

## Installieren Sie Trident Protect von NetApp

### Schritte

1. Trident Helm Repository hinzufügen:

```
helm repo add netapp-trident-protect
https://netapp.github.io/trident-protect-helm-chart
```

2. Installieren Sie die Trident Protect CRDs:

```
helm install trident-protect-crds netapp-trident-protect/trident-
protect-crds --version 100.2410.1 --create-namespace --namespace
trident-protect
```

3. Verwenden Sie Helm, um Trident Protect mit einem der folgenden Befehle zu installieren. Ersetzen Sie `<name_of_cluster>` ihn durch einen Cluster-Namen, der dem Cluster zugewiesen wird und zum Identifizieren der Backups und Snapshots des Clusters verwendet wird:

- Trident Protect normal installieren:

```
helm install trident-protect netapp-trident-protect/trident-
protect --set clusterName=<name_of_cluster> --version 100.2410.1
--create-namespace --namespace trident-protect
```

- Installieren Sie Trident Protect und deaktivieren Sie die geplanten täglichen Uploads von Trident Protect AutoSupport Support-Paketen:

```
helm install trident-protect netapp-trident-protect/trident-
protect --set autoSupport.enabled=false --set
clusterName=<name_of_cluster> --version 100.2410.1 --create
-namespace --namespace trident-protect
```

## Installieren Sie Trident Protect aus einer privaten Registrierung

Sie können Trident Protect aus einer privaten Image-Registrierung installieren, wenn Ihr Kubernetes-Cluster nicht auf das Internet zugreifen kann. Ersetzen Sie in diesen Beispielen Werte in Klammern durch Informationen aus Ihrer Umgebung:

### Schritte

1. Ziehen Sie die folgenden Bilder auf Ihren lokalen Computer, aktualisieren Sie die Tags und schieben Sie sie dann in Ihre private Registrierung:

```
netapp/controller:24.10.1
netapp/restic:24.10.1
netapp/kopia:24.10.1
netapp/trident-autosupport:24.10.0
netapp/exehook:24.10.1
netapp/resourcebackup:24.10.1
netapp/resourcerestore:24.10.1
netapp/resourcedelete:24.10.1
bitnami/kubectl:1.30.2
kubebuilder/kube-rbac-proxy:v0.16.0
```

### Beispiel:

```
docker pull netapp/controller:24.10.1
```

```
docker tag netapp/controller:24.10.1 <private-registry-
url>/controller:24.10.1
```

```
docker push <private-registry-url>/controller:24.10.1
```

### 2. Trident Protect System Namespace erstellen:

```
kubectl create ns trident-protect
```

### 3. Melden Sie sich bei der Registrierung an:

```
helm registry login <private-registry-url> -u <account-id> -p <api-
token>
```

### 4. Erstellen Sie einen Pull-Schlüssel, der für die Authentifizierung der privaten Registrierung verwendet werden soll:

```
kubectl create secret docker-registry regcred --docker
-username=<registry-username> --docker-password=<api-token> -n
trident-protect --docker-server=<private-registry-url>
```

### 5. Trident Helm Repository hinzufügen:



```
helm repo add netapp-trident-protect
https://netapp.github.io/trident-protect-helm-chart
```

6. Erstellen Sie eine Datei mit dem Namen `protectValues.yaml`. Stellen Sie sicher, dass es die folgenden Trident Protect-Einstellungen enthält:

```
---
image:
  registry: <private-registry-url>
imagePullSecrets:
  - name: regcred
controller:
  image:
    registry: <private-registry-url>
rbacProxy:
  image:
    registry: <private-registry-url>
crCleanup:
  imagePullSecrets:
    - name: regcred
webhooksCleanup:
  imagePullSecrets:
    - name: regcred
```

7. Installieren Sie die Trident Protect CRDs:

```
helm install trident-protect-crds netapp-trident-protect/trident-
protect-crds --version 100.2410.1 --create-namespace --namespace
trident-protect
```

8. Verwenden Sie Helm, um Trident Protect mit einem der folgenden Befehle zu installieren. Ersetzen Sie `<name_of_cluster>` ihn durch einen Cluster-Namen, der dem Cluster zugewiesen wird und zum Identifizieren der Backups und Snapshots des Clusters verwendet wird:

- Trident Protect normal installieren:

```
helm install trident-protect netapp-trident-protect/trident-
protect --set clusterName=<name_of_cluster> --version 100.2410.1
--create-namespace --namespace trident-protect -f
protectValues.yaml
```

- Installieren Sie Trident Protect und deaktivieren Sie die geplanten täglichen Uploads von Trident Protect AutoSupport Support-Paketen:

```
helm install trident-protect netapp-trident-protect/trident-protect --set autoSupport.enabled=false --set clusterName=<name_of_cluster> --version 100.2410.1 --create --namespace --namespace trident-protect -f protectValues.yaml
```

## Geben Sie die Höchstwerte für die Trident Protect Container-Ressourcen an

Nach der Installation von Trident Protect können Sie eine Konfigurationsdatei verwenden, um Ressourcenbeschränkungen für Trident Protect Container festzulegen. Durch das Festlegen der Grenzwerte für die Ressourcen können Sie steuern, wie viele Ressourcen des Clusters von Trident Protect-Vorgängen verbraucht werden.

### Schritte

1. Erstellen Sie eine Datei mit dem Namen `resourceLimits.yaml`.
2. Füllen Sie die Datei entsprechend den Anforderungen Ihrer Umgebung mit Optionen für die Ressourcenbegrenzung für Trident Container.

Die folgende Beispielkonfigurationsdatei zeigt die verfügbaren Einstellungen und enthält die Standardvaules für jedes Ressourcenlimit:

```
---
jobResources:
  defaults:
    limits:
      cpu: 8000m
      memory: 10000Mi
      ephemeralStorage: ""
    requests:
      cpu: 100m
      memory: 100Mi
      ephemeralStorage: ""
  resticVolumeBackup:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  resticVolumeRestore:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
```

```

requests:
  cpu: ""
  memory: ""
  ephemeralStorage: ""
kopiaVolumeBackup:
  limits:
    cpu: ""
    memory: ""
    ephemeralStorage: ""
  requests:
    cpu: ""
    memory: ""
    ephemeralStorage: ""
kopiaVolumeRestore:
  limits:
    cpu: ""
    memory: ""
    ephemeralStorage: ""
  requests:
    cpu: ""
    memory: ""
    ephemeralStorage: ""

```

3. Anwenden der Werte aus der `resourceLimits.yaml` Datei:

```

helm upgrade trident-protect -n trident-protect -f <resourceLimits.yaml>
--reuse-values

```

## Installieren Sie das Trident Protect CLI-Plug-in

Sie können das Befehlszeilenplugin Trident Protect verwenden, das eine Erweiterung des Dienstprogramms Trident ist `tridentctl`, um benutzerdefinierte Ressourcen (Trident Protect Custom Resources, CRS) zu erstellen und mit ihnen zu interagieren.

### Installieren Sie das Trident Protect CLI-Plug-in

Bevor Sie das Befehlszeilendienstprogramm verwenden, müssen Sie es auf dem Computer installieren, mit dem Sie auf das Cluster zugreifen. Befolgen Sie diese Schritte, je nachdem, ob Ihr Computer eine x64- oder ARM-CPU verwendet.

## Download Plugin für Linux AMD64 CPUs

### Schritte

1. Laden Sie das Trident Protect CLI-Plug-in herunter:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/24.10.1/tridentctl-protect-linux-amd64
```

## Download Plugin für Linux ARM64 CPUs

### Schritte

1. Laden Sie das Trident Protect CLI-Plug-in herunter:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/24.10.1/tridentctl-protect-linux-arm64
```

## Download Plugin für Mac AMD64 CPUs

### Schritte

1. Laden Sie das Trident Protect CLI-Plug-in herunter:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/24.10.1/tridentctl-protect-macos-amd64
```

## Download Plugin für Mac ARM64 CPUs

### Schritte

1. Laden Sie das Trident Protect CLI-Plug-in herunter:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/24.10.1/tridentctl-protect-macos-arm64
```

1. Ausführungsberechtigungen für die Plug-in-Binärdatei aktivieren:

```
chmod +x tridentctl-protect
```

2. Kopieren Sie die Plug-in-Binärdatei an einen Speicherort, der in Ihrer PFADVARIABLE definiert ist. Beispiel /usr/bin: Oder /usr/local/bin (Sie benötigen möglicherweise erhöhte Privileges):

```
cp ./tridentctl-protect /usr/local/bin/
```

- Optional können Sie die Plugin-Binärdatei an einen Speicherort in Ihrem Home-Verzeichnis kopieren. In diesem Fall wird empfohlen, sicherzustellen, dass der Speicherort Teil Ihrer PFADVARIABLE ist:

```
cp ./tridentctl-protect ~/bin/
```



Das Kopieren des Plugins an einen Ort in Ihrer PFADVARIABLE ermöglicht es Ihnen, das Plugin durch Eingabe oder `tridentctl protect` von einem beliebigen Ort zu verwenden `tridentctl-protect`.

### Trident CLI Plug-in-Hilfe ansehen

Sie können die integrierten Plug-in-Hilfefunktionen nutzen, um detaillierte Hilfe zu den Funktionen des Plug-ins zu erhalten:

#### Schritte

1. Verwenden Sie die Hilfefunktion, um die Verwendungshilfe anzuzeigen:

```
tridentctl-protect help
```

### Aktivieren Sie die automatische Vervollständigung von Befehlen

Nachdem Sie das Trident Protect CLI-Plugin installiert haben, können Sie die automatische Vervollständigung für bestimmte Befehle aktivieren.

## Aktivieren Sie die automatische Vervollständigung für die Bash-Shell

### Schritte

1. Download des Abschlusskripts:

```
curl -L -O https://github.com/NetApp/tridentctl-protect/releases/download/24.10.1/tridentctl-completion.bash
```

2. Erstellen Sie ein neues Verzeichnis in Ihrem Home-Verzeichnis, um das Skript zu enthalten:

```
mkdir -p ~/.bash/completions
```

3. Das heruntergeladene Skript in das Verzeichnis verschieben ~/.bash/completions:

```
mv tridentctl-completion.bash ~/.bash/completions/
```

4. Fügen Sie der Datei im Home-Verzeichnis folgende Zeile hinzu ~/.bashrc:

```
source ~/.bash/completions/tridentctl-completion.bash
```

## Aktivieren Sie die automatische Vervollständigung für die Z-Shell

### Schritte

1. Download des Abschlusskripts:

```
curl -L -O https://github.com/NetApp/tridentctl-protect/releases/download/24.10.1/tridentctl-completion.zsh
```

2. Erstellen Sie ein neues Verzeichnis in Ihrem Home-Verzeichnis, um das Skript zu enthalten:

```
mkdir -p ~/.zsh/completions
```

3. Das heruntergeladene Skript in das Verzeichnis verschieben ~/.zsh/completions:

```
mv tridentctl-completion.zsh ~/.zsh/completions/
```

4. Fügen Sie der Datei im Home-Verzeichnis folgende Zeile hinzu ~/.zprofile:

```
source ~/.zsh/completions/tridentctl-completion.zsh
```

## Ergebnis

Bei Ihrer nächsten Shell-Anmeldung können Sie den Befehl Auto-Vervollständigung mit dem `tridentctl-Protect-Plugin` verwenden.

# Management von Trident Protect

## Managen von Trident Schützen Sie die Autorisierung und Zugriffssteuerung

Trident Protect nutzt das Kubernetes-Modell der rollenbasierten Zugriffssteuerung (Role Based Access Control, RBAC). Standardmäßig stellt Trident Protect einen einzelnen System-Namespace und sein dazugehöriges Standarddienstkonto bereit. Wenn Ihr Unternehmen über eine Vielzahl von Benutzern oder spezifische Sicherheitsanforderungen verfügt, können Sie die RBAC-Funktionen von Trident Protect verwenden, um eine granularere Kontrolle über den Zugriff auf Ressourcen und Namespaces zu erlangen.

Der Clusteradministrator hat immer Zugriff auf Ressourcen im Standard- `trident-protect` Namespace und kann auch auf Ressourcen in allen anderen Namespaces zugreifen. Um den Zugriff auf Ressourcen und Anwendungen zu kontrollieren, müssen Sie zusätzliche Namespaces erstellen und diesen Namespaces Ressourcen und Anwendungen hinzufügen.

Beachten Sie, dass keine Benutzer Anwendungsdatenmanagement-CRS im Standard-Namespace erstellen können `trident-protect`. Sie müssen Anwendungsdatenmanagement-CRS in einem Anwendungs-Namespace erstellen (als Best Practice erstellen Sie Anwendungsdatenmanagement-CRS im gleichen Namespace wie ihre zugehörige Anwendung).

Nur Administratoren sollten Zugriff auf privilegierte Trident haben, die benutzerdefinierte Ressourcenobjekte schützen, darunter:



- **AppVault**: Erfordert Bucket-Zugangsdaten
- **AutoSupportBundle**: Sammelt Kennzahlen, Protokolle und andere sensible Trident schützen Daten
- **AutoSupportBundleSchedule**: Verwaltet Zeitpläne für die Protokollsammlung

Verwenden Sie als Best Practice RBAC, um den Zugriff auf privilegierte Objekte auf Administratoren zu beschränken.

Weitere Informationen darüber, wie RBAC den Zugriff auf Ressourcen und Namespaces regelt, finden Sie im ["RBAC-Dokumentation für Kubernetes"](#).

Informationen zu Servicekonten finden Sie im ["Dokumentation des Kubernetes Service-Kontos"](#).

### Beispiel: Zugriff für zwei Benutzergruppen verwalten

Ein Unternehmen verfügt beispielsweise über einen Cluster-Administrator, eine Gruppe von Engineering-Benutzern und eine Gruppe von Marketing-Benutzern. Der Clusteradministrator führt die folgenden Aufgaben aus, um eine Umgebung zu erstellen, in der die Engineering-Gruppe und die Marketing-Gruppe jeweils nur auf die Ressourcen zugreifen können, die ihren jeweiligen Namespaces zugewiesen sind.

### Schritt 1: Erstellen Sie einen Namespace, der Ressourcen für jede Gruppe enthält

Durch das Erstellen eines Namespace können Sie Ressourcen logisch trennen und besser kontrollieren, wer Zugriff auf diese Ressourcen hat.

#### Schritte

1. Erstellen Sie einen Namespace für die Engineering-Gruppe:

```
kubectl create ns engineering-ns
```

2. Erstellen Sie einen Namespace für die Marketinggruppe:

```
kubectl create ns marketing-ns
```

### Schritt 2: Erstellen Sie neue Dienstkonten, um mit Ressourcen in jedem Namespace zu interagieren

Jeder neue Namespace, den Sie erstellen, verfügt über ein Standard-Dienstkonto. Sie sollten jedoch für jede Benutzergruppe ein Dienstkonto erstellen, damit Sie Privileges bei Bedarf in Zukunft weiter zwischen Gruppen aufteilen können.

#### Schritte

1. Erstellen Sie ein Servicekonto für die Engineering-Gruppe:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: eng-user
  namespace: engineering-ns
```

2. Erstellen Sie ein Service-Konto für die Marketinggruppe:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: mkt-user
  namespace: marketing-ns
```

### Schritt 3: Erstellen Sie ein Geheimnis für jedes neue Service-Konto

Ein Dienstkontogeheimnis wird verwendet, um sich beim Dienstkonto zu authentifizieren. Er kann bei einer Kompromittierung einfach gelöscht und neu erstellt werden.

#### Schritte

1. Einen Schlüssel für das Engineering-Servicekonto erstellen:



```
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: eng-user
  name: eng-user-secret
  namespace: engineering-ns
  type: kubernetes.io/service-account-token
```

## 2. Erstellen Sie ein Geheimnis für das Marketingservicekonto:

```
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: mkt-user
  name: mkt-user-secret
  namespace: marketing-ns
  type: kubernetes.io/service-account-token
```

### Schritt 4: Erstellen Sie ein RoleBinding-Objekt, um das ClusterRole-Objekt an jedes neue Servicekonto zu binden

Bei der Installation von Trident Protect wird ein Standardobjekt für ClusterRole erstellt. Sie können diese ClusterRole an das Dienstkonto binden, indem Sie ein RoleBinding-Objekt erstellen und anwenden.

#### Schritte

### 1. Binden Sie die ClusterRole an das Engineering-Servicekonto:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: engineering-ns-tenant-rolebinding
  namespace: engineering-ns
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: trident-protect-tenant-cluster-role
subjects:
- kind: ServiceAccount
  name: eng-user
  namespace: engineering-ns
```

### 2. Binden Sie den ClusterRole an das Marketingservicekonto:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: marketing-ns-tenant-rolebinding
  namespace: marketing-ns
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: trident-protect-tenant-cluster-role
subjects:
- kind: ServiceAccount
  name: mkt-user
  namespace: marketing-ns
```

### Schritt 5: Testberechtigungen

Überprüfen Sie, ob die Berechtigungen korrekt sind.

#### Schritte

1. Bestätigung, dass Engineering-Benutzer auf Engineering-Ressourcen zugreifen können:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get applications.protect.trident.netapp.io -n engineering-ns
```

2. Bestätigen Sie, dass Engineering-Benutzer nicht auf Marketing-Ressourcen zugreifen können:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get applications.protect.trident.netapp.io -n marketing-ns
```

### Schritt 6: Zugriff auf AppVault-Objekte gewähren

Um Datenmanagementaufgaben wie Backups und Snapshots auszuführen, muss der Clusteradministrator einzelnen Benutzern Zugriff auf AppVault-Objekte gewähren.

#### Schritte

1. Erstellen und Anwenden einer AppVault- und geheimen YAML-Kombinationsdatei, die einem Benutzer Zugriff auf einen AppVault gewährt. Der folgende CR gewährt dem Benutzer beispielsweise Zugriff auf einen AppVault `eng-user`:

```

apiVersion: v1
data:
  accessKeyID: <ID_value>
  secretAccessKey: <key_value>
kind: Secret
metadata:
  name: appvault-for-eng-user-only-secret
  namespace: trident-protect
type: Opaque
---
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: appvault-for-eng-user-only
  namespace: trident-protect # Trident protect system namespace
spec:
  providerConfig:
    azure:
      accountName: ""
      bucketName: ""
      endpoint: ""
    gcp:
      bucketName: ""
      projectID: ""
    s3:
      bucketName: testbucket
      endpoint: 192.168.0.1:30000
      secure: "false"
      skipCertValidation: "true"
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: appvault-for-eng-user-only-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: appvault-for-eng-user-only-secret
  providerType: GenericS3

```

2. Erstellen und Anwenden eines Rollen-CR, damit Clusteradministratoren Zugriff auf bestimmte Ressourcen in einem Namespace gewähren können. Beispiel:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: eng-user-appvault-reader
  namespace: trident-protect
rules:
- apiGroups:
  - protect.trident.netapp.io
  resourceNames:
  - appvault-for-enguser-only
  resources:
  - appvaults
  verbs:
  - get
```

3. Erstellen und wenden Sie einen RoleBinding CR an, um die Berechtigungen an den Benutzer eng-user zu binden. Beispiel:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: eng-user-read-appvault-binding
  namespace: trident-protect
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: eng-user-appvault-reader
subjects:
- kind: ServiceAccount
  name: eng-user
  namespace: engineering-ns
```

4. Überprüfen Sie, ob die Berechtigungen korrekt sind.

- a. Es wird versucht, die AppVault-Objektinformationen für alle Namespaces abzurufen:

```
kubectl get appvaults -n trident-protect
--as=system:serviceaccount:engineering-ns:eng-user
```

Sie sollten eine Ausgabe wie die folgende sehen:

```
Error from server (Forbidden): appvaults.protect.trident.netapp.io is
forbidden: User "system:serviceaccount:engineering-ns:eng-user"
cannot list resource "appvaults" in API group
"protect.trident.netapp.io" in the namespace "trident-protect"
```

b. Testen Sie, ob der Benutzer die AppVault-Informationen erhalten kann, auf die er jetzt Zugriff hat:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get appvaults.protect.trident.netapp.io/appvault-for-eng-user-only -n
trident-protect
```

Sie sollten eine Ausgabe wie die folgende sehen:

```
yes
```

### Ergebnis

Die Benutzer, denen Sie AppVault-Berechtigungen erteilt haben, sollten autorisierte AppVault-Objekte für Anwendungsdatenverwaltungsvorgänge verwenden können und nicht in der Lage sein, auf Ressourcen außerhalb der zugewiesenen Namespaces zuzugreifen oder neue Ressourcen zu erstellen, auf die sie keinen Zugriff haben.

### Generieren Sie ein Trident Protect Supportpaket

Mit Trident Protect können Administratoren Bundles erstellen, die für die Unterstützung von NetApp nützliche Informationen enthalten, einschließlich Protokollen, Kennzahlen und Topologieinformationen zu den zu managenden Clustern und Apps. Wenn Sie mit dem Internet verbunden sind, können Sie Supportpakete mithilfe einer benutzerdefinierten Ressourcendatei (CR) auf die NetApp-Support-Website (NSS) hochladen.

## Erstellen Sie mithilfe eines CR-Systems ein Supportpaket

### Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie (z. B. `trident-protect-support-bundle.yaml`).
2. Konfigurieren Sie die folgenden Attribute:
  - **metadata.name:** (*required*) der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und sinnvollen Namen für Ihre Umgebung.
  - **Spec.triggerType:** (*required*) legt fest, ob das Support-Bundle sofort generiert oder geplant wird. Die geplante Bundle-Generierung findet um 12:00 UHR UTC statt. Mögliche Werte:
    - Geplant
    - Manuell
  - **Spec.UploadEnabled:** (*Optional*) steuert, ob das Supportpaket nach der Generierung auf die NetApp-Support-Website hochgeladen werden soll. Wenn nicht angegeben, wird standardmäßig auf `false`. Mögliche Werte:
    - Richtig
    - False (Standard)
  - **Spec.dataWindowStart:** (*Optional*) Eine Datumstring im RFC 3339-Format, die das Datum und die Uhrzeit angibt, zu der das Fenster der im Support-Bundle enthaltenen Daten beginnen soll. Wenn nicht angegeben, ist die Standardeinstellung vor 24 Stunden. Das früheste Fensterdatum, das Sie angeben können, ist vor 7 Tagen.

Beispiel YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: AutoSupportBundle
metadata:
  name: trident-protect-support-bundle
spec:
  triggerType: Manual
  uploadEnabled: true
  dataWindowStart: 2024-05-05T12:30:00Z
```

3. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt `astra-support-bundle.yaml` haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-support-bundle.yaml
```

## Erstellen Sie ein Support-Bundle mithilfe der CLI

### Schritte

1. Erstellen Sie das Supportpaket, und ersetzen Sie Werte in Klammern durch Informationen aus Ihrer Umgebung. Der `trigger-type` legt fest, ob das Bündel sofort erstellt wird oder ob die

Erstellungszeit vom Zeitplan vorgegeben ist, und kann oder Scheduled sein Manual. Die Standardeinstellung ist Manual.

Beispiel:

```
tridentctl-protect create autosupportbundle <my_bundle_name>  
--trigger-type <trigger_type>
```

## Upgrade von Trident Protect

Sie können ein Upgrade von Trident Protect auf die neueste Version durchführen, um von neuen Funktionen oder Fehlerkorrekturen zu profitieren.

Führen Sie zum Upgrade von Trident Protect die folgenden Schritte aus.

### Schritte

1. Aktualisieren Sie das Trident Helm-Repository:

```
helm repo update
```

2. Aktualisieren Sie die Trident-Schutz-CRDs:

```
helm upgrade trident-protect-crds netapp-trident-protect/trident-  
protect-crds --version 100.2410.1 --namespace trident-protect
```

3. Upgrade-Trident-Schutz:

```
helm upgrade trident-protect netapp-trident-protect/trident-protect  
--version 100.2410.1 --namespace trident-protect
```

## Managen und sichern Sie Applikationen

### Verwenden Sie zum Verwalten von Buckets Trident Protect AppVault-Objekte

Die benutzerdefinierte Bucket-Ressource (CR) für Trident Protect wird als AppVault bezeichnet. AppVault-Objekte sind die deklarative Kubernetes-Workflow-Darstellung eines Storage-Buckets. Ein AppVault CR enthält die Konfigurationen, die für einen Bucket erforderlich sind, der für Schutzvorgänge verwendet werden kann, z. B. Backups, Snapshots, Wiederherstellungsvorgänge und SnapMirror-Replikation. Nur Administratoren können AppVaults erstellen.

## **Beispiele für die Schlüsselgenerierung und AppVault-Definitionen**

Beim Definieren eines AppVault CR müssen Sie Anmeldeinformationen eingeben, um auf die vom Anbieter gehosteten Ressourcen zugreifen zu können. Die Art und Weise, wie Sie die Schlüssel für die Anmeldeinformationen generieren, hängt vom Anbieter ab. Im Folgenden finden Sie Beispiele für die Schlüsselgenerierung über die Befehlszeile für mehrere Anbieter, gefolgt von AppVault-Beispieldefinitionen für jeden Anbieter.

### **Beispiele für die Schlüsselgenerierung**

In den folgenden Beispielen können Sie Schlüssel für die Anmeldedaten der einzelnen Cloud-Provider erstellen.



## Google Cloud

```
kubectl create secret generic <secret-name> --from-file=credentials  
=<mycreds-file.json> -n trident-protect
```

## Amazon S3 (AWS)

```
kubectl create secret generic <secret-name> --from-literal=accessKeyID  
=<objectstorage-accesskey> --from-literal=secretAccessKey=<generic-s3-  
trident-protect-src-bucket-secret> -n trident-protect
```

## Microsoft Azure

```
kubectl create secret generic <secret-name> --from-literal=accountKey  
=<secret-name> -n trident-protect
```

## Allgemein S3

```
kubectl create secret generic <secret-name> --from-literal=accessKeyID  
=<objectstorage-accesskey> --from-literal=secretAccessKey=<generic-s3-  
trident-protect-src-bucket-secret> -n trident-protect
```

## ONTAP S3

```
kubectl create secret generic <secret-name> --from-literal=accessKeyID  
=<objectstorage-accesskey> --from-literal=secretAccessKey=<generic-s3-  
trident-protect-src-bucket-secret> -n trident-protect
```

## StorageGRID S3

```
kubectl create secret generic <secret-name> --from-literal=  
accessKeyID=<objectstorage-accesskey> --from-literal=secretAccessKey  
=<generic-s3-trident-protect-src-bucket-secret> -n trident-protect
```

## AppVault CR-Beispiele

Sie können die folgenden CR-Beispiele verwenden, um AppVault-Objekte für jeden Cloud-Provider zu erstellen.

## Google Cloud

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: gcp-trident-protect-src-bucket-b643cc50-0429-4ad5-971f-
ac4a83621922
  namespace: trident-protect
spec:
  providerType: GCP
  providerConfig:
    gcp:
      bucketName: trident-protect-src-bucket
      projectID: project-id
  providerCredentials:
    credentials:
      valueFromSecret:
        key: credentials
        name: gcp-trident-protect-src-bucket-secret
```

## Amazon S3 (AWS)

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: amazon-s3-trident-protect-src-bucket-b643cc50-0429-4ad5-971f-
ac4a83621922
  namespace: trident-protect
spec:
  providerType: AWS
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3
```

## Microsoft Azure

```

apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: azure-trident-protect-src-bucket-b643cc50-0429-4ad5-971f-
ac4a83621922
  namespace: trident-protect
spec:
  providerType: Azure
  providerConfig:
    azure:
      accountName: account-name
      bucketName: trident-protect-src-bucket
  providerCredentials:
    accountKey:
      valueFromSecret:
        key: accountKey
        name: azure-trident-protect-src-bucket-secret

```

### Allgemein S3

```

apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: generic-s3-trident-protect-src-bucket-b643cc50-0429-4ad5-971f-
ac4a83621922
  namespace: trident-protect
spec:
  providerType: GenericS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3

```

### ONTAP S3

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: ontap-s3-trident-protect-src-bucket-b643cc50-0429-4ad5-971f-
ac4a83621922
  namespace: trident-protect
spec:
  providerType: OntapS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3
```

### StorageGRID S3

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: storagegrid-s3-trident-protect-src-bucket-b643cc50-0429-4ad5-
  971f-ac4a83621922
  namespace: trident-protect
spec:
  providerType: StorageGridS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3
```

### Beispiele für die Erstellung von AppVault mithilfe der Trident Protect CLI

Sie können die folgenden CLI-Befehlsbeispiele verwenden, um AppVault CRS für jeden Anbieter zu erstellen.

## Google Cloud

```
tridentctl-protect create vault GCP my-new-vault --bucket mybucket  
--project my-gcp-project --secret <gcp-creds>/<credentials>
```

## Amazon S3 (AWS)

```
tridentctl-protect create vault AWS <vault-name> --bucket <bucket-name>  
--secret <secret-name> --endpoint <s3-endpoint>
```

## Microsoft Azure

```
tridentctl-protect create vault Azure <vault-name> --account <account-  
name> --bucket <bucket-name> --secret <secret-name>
```

## Allgemein S3

```
tridentctl-protect create vault GenericS3 <vault-name> --bucket  
<bucket-name> --secret <secret-name> --endpoint <s3-endpoint>
```

## ONTAP S3

```
tridentctl-protect create vault OntapS3 <vault-name> --bucket <bucket-  
name> --secret <secret-name> --endpoint <s3-endpoint>
```

## StorageGRID S3

```
tridentctl-protect create vault StorageGridS3 s3vault --bucket <bucket-  
name> --secret <secret-name> --endpoint <s3-endpoint>
```

## Verwenden Sie den AppVault-Browser, um Informationen zu AppVault anzuzeigen

Sie können das Trident Protect CLI-Plugin verwenden, um Informationen über AppVault-Objekte anzuzeigen, die auf dem Cluster erstellt wurden.

### Schritte

1. Inhalt eines AppVault-Objekts anzeigen:

```
tridentctl-protect get appvaultcontent gcp-vault --show-resources all
```

### Beispielausgabe:

```

+-----+-----+-----+-----+
+-----+
| CLUSTER | APP | TYPE | NAME |
|-----|-----|-----|-----|
| TIMESTAMP | | | |
+-----+-----+-----+-----+
+-----+
| | mysql | snapshot | mysnap | 2024-
08-09 21:02:11 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815180300 | 2024-
08-15 18:03:06 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815190300 | 2024-
08-15 19:03:06 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815200300 | 2024-
08-15 20:03:06 (UTC) |
| production1 | mysql | backup | hourly-e7db6-20240815180300 | 2024-
08-15 18:04:25 (UTC) |
| production1 | mysql | backup | hourly-e7db6-20240815190300 | 2024-
08-15 19:03:30 (UTC) |
| production1 | mysql | backup | hourly-e7db6-20240815200300 | 2024-
08-15 20:04:21 (UTC) |
| production1 | mysql | backup | mybackup5 | 2024-
08-09 22:25:13 (UTC) |
| | mysql | backup | mybackup | 2024-
08-09 21:02:52 (UTC) |
+-----+-----+-----+-----+
+-----+

```

2. Um den AppVaultPath für jede Ressource anzuzeigen, verwenden Sie optional das Flag `--show-paths`.

Der Cluster-Name in der ersten Spalte der Tabelle ist nur verfügbar, wenn in der Installation Trident Protect Helm ein Cluster-Name angegeben wurde. Zum Beispiel: `--set clusterName=production1`.

### Entfernen Sie einen AppVault

Sie können ein AppVault-Objekt jederzeit entfernen.



Entfernen Sie den Schlüssel im AppVault CR nicht `finalizers`, bevor Sie das AppVault-Objekt löschen. Wenn Sie dies tun, kann dies zu Restdaten im AppVault-Bucket und verwaisten Ressourcen im Cluster führen.

### Bevor Sie beginnen

Stellen Sie sicher, dass Sie alle Snapshots und Backups gelöscht haben, die im zugehörigen Bucket gespeichert sind.

### Entfernen Sie einen AppVault mithilfe der Kubernetes-CLI

1. Entfernen Sie das AppVault-Objekt und ersetzen Sie `appvault_name` es durch den Namen des zu entfernenden AppVault-Objekts:

```
kubectl delete appvault <appvault_name> -n trident-protect
```

### Entfernen Sie einen AppVault mithilfe der Trident Protect-CLI

1. Entfernen Sie das AppVault-Objekt und ersetzen Sie `appvault_name` es durch den Namen des zu entfernenden AppVault-Objekts:

```
tridentctl-protect delete appvault <appvault_name> -n trident-protect
```

## Definieren Sie mit Trident Protect eine Applikation für das Management

Sie können eine Anwendung definieren, die Sie mit Trident Protect verwalten möchten, indem Sie eine Anwendungs-CR und einen zugehörigen AppVault CR erstellen.

### Erstellen Sie ein AppVault CR

Sie müssen einen AppVault CR erstellen, der bei der Durchführung von Datenschutzvorgängen auf der Anwendung verwendet wird. Der AppVault CR muss sich auf dem Cluster befinden, auf dem Trident Protect installiert ist. Der AppVault CR ist spezifisch für Ihre Umgebung. Beispiele für AppVault CRS finden Sie unter ["Benutzerdefinierte Ressourcen von AppVault."](#)

### Definieren Sie eine Anwendung

Sie müssen jede Applikation definieren, die Sie mit Trident Protect managen möchten. Sie können eine Anwendung für die Verwaltung definieren, indem Sie entweder manuell eine Anwendungs-CR erstellen oder die Trident Protect-CLI verwenden.



## Fügen Sie eine Anwendung mit einem CR hinzu

### Schritte

1. Erstellen Sie die CR-Datei der Zielanwendung:
  - a. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie (z. B. `maria-app.yaml`).
  - b. Konfigurieren Sie die folgenden Attribute:
    - **metadata.name:** (*required*) der Name der benutzerdefinierten Ressource der Anwendung. Beachten Sie den von Ihnen ausgewählten Namen, da sich andere CR-Dateien, die für Schutzvorgänge benötigt werden, auf diesen Wert beziehen.
    - **spec.includedNamespaces:** (*required*) Verwenden Sie Namespace-Labels oder einen Namespace-Namen, um Namespaces anzugeben, in denen die Anwendungsressourcen vorhanden sind. Der Application Namespace muss Teil dieser Liste sein.

Beispiel YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  name: maria
  namespace: my-app-namespace
spec:
  includedNamespaces:
    - namespace: my-app-namespace
```

2. Nachdem Sie die CR-Anwendung erstellt haben, die Ihrer Umgebung entspricht, wenden Sie den CR an. Beispiel:

```
kubectl apply -f maria-app.yaml
```

## Fügen Sie eine Anwendung mithilfe der CLI hinzu

### Schritte

1. Erstellen und wenden Sie die Anwendungsdefinition an, indem Sie Werte in Klammern durch Informationen aus Ihrer Umgebung ersetzen. Sie können Namespaces und Ressourcen in die Anwendungsdefinition mit kommagetrennten Listen mit den im folgenden Beispiel gezeigten Argumenten aufnehmen:

```
tridentctl-protect create application <my_new_app_cr_name>
--namespaces <namespaces_to_include> --csr
<cluster_scoped_resources_to_include> --namespace <my-app-namespace>
```

## Sicherung von Applikationen mit Trident Protect

Sie können alle von Trident Protect gemanagten Applikationen schützen, indem Sie Snapshots und Backups mithilfe einer automatisierten Sicherungsrichtlinie oder ad hoc erstellen.



Sie können Trident Protect so konfigurieren, dass Dateisysteme während des Datenschutzes eingefroren und wieder eingefroren werden. ["Erfahren Sie mehr über das Konfigurieren des Dateisystemfrierens mit Trident Protect"](#).

### Erstellen Sie einen On-Demand Snapshot

Sie können jederzeit einen On-Demand-Snapshot erstellen.

## Erstellen Sie mithilfe eines CR-Systems einen Snapshot

### Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie `trident-protect-snapshot-cr.yaml`.
2. Konfigurieren Sie in der erstellten Datei die folgenden Attribute:
  - **metadata.name:** (*required*) der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und sinnvollen Namen für Ihre Umgebung.
  - **Spec.applicationRef:** Der Kubernetes-Name der zu Snapshot enden Anwendung.
  - **Spec.appVaultRef:** (*required*) der Name des AppVault, in dem der Snapshot-Inhalt (Metadaten) gespeichert werden soll.
  - **Spec.reclaimPolicy:** (*Optional*) definiert, was mit dem AppArchiv eines Snapshots geschieht, wenn der Snapshot CR gelöscht wird. Das bedeutet, dass der Snapshot auch dann gelöscht wird, wenn er auf gesetzt `Retain` ist. Gültige Optionen:
    - `Retain` (Standard)
    - `Delete`

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Snapshot
metadata:
  namespace: my-app-namespace
  name: my-cr-name
spec:
  applicationRef: my-application
  appVaultRef: appvault-name
  reclaimPolicy: Delete
```

3. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt `trident-protect-snapshot-cr.yaml` haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-snapshot-cr.yaml
```

## Erstellen Sie einen Snapshot mithilfe der CLI

### Schritte

1. Erstellen Sie den Snapshot, und ersetzen Sie Werte in Klammern durch Informationen aus Ihrer Umgebung. Beispiel:

```
tridentctl-protect create snapshot <my_snapshot_name> --appvault
<my_appvault_name> --app <name_of_app_to_snapshot> -n
<application_namespace>
```

## **Erstellen Sie ein On-Demand-Backup**

Sie können eine App jederzeit sichern.

## Erstellen Sie ein Backup mit einem CR

### Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie `trident-protect-backup-cr.yaml`.
2. Konfigurieren Sie in der erstellten Datei die folgenden Attribute:
  - **metadata.name:** (*required*) der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und sinnvollen Namen für Ihre Umgebung.
  - **Spec.applicationRef:** (*required*) der Kubernetes-Name der zu Back-up-Applikation.
  - **Spec.appVaultRef:** (*required*) der Name des AppVault, in dem der Backup-Inhalt gespeichert werden soll.
  - **Spec.DataMover:** (*Optional*) Eine Zeichenfolge, die angibt, welches Backup-Tool für den Backup-Vorgang verwendet werden soll. Mögliche Werte (Groß-/Kleinschreibung beachten):
    - Restic
    - Kopia (Standard)
  - **Spec.reclaimPolicy:** (*Optional*) definiert, was mit einem Backup geschieht, wenn es von seinem Anspruch freigegeben wird. Mögliche Werte:
    - Delete
    - Retain (Standard)
  - **Spec.snapshotRef:** (*Optional*): Name des als Quelle des Backups zu verwendenden Snapshot. Falls nicht angegeben, wird ein temporärer Snapshot erstellt und gesichert.

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Backup
metadata:
  namespace: my-app-namespace
  name: my-cr-name
spec:
  applicationRef: my-application
  appVaultRef: appvault-name
  dataMover: Kopia
```

3. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt `trident-protect-backup-cr.yaml` haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-backup-cr.yaml
```

## Erstellen Sie mithilfe der CLI ein Backup

### Schritte

1. Erstellen Sie das Backup, und ersetzen Sie Werte in Klammern durch Informationen aus Ihrer Umgebung. Beispiel:

```
tridentctl-protect create backup <my_backup_name> --appvault <my-  
vault-name> --app <name_of_app_to_back_up> -n  
<application_namespace>
```

## Erstellen Sie einen Zeitplan für die Datensicherung

Eine Sicherungsrichtlinie sichert eine Applikation, indem Snapshots, Backups oder beides nach einem definierten Zeitplan erstellt werden. Sie können Snapshots und Backups stündlich, täglich, wöchentlich und monatlich erstellen. Außerdem können Sie die Anzahl der beizubehaltenden Kopien festlegen.

## Erstellen Sie einen Zeitplan mit einem CR

### Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie `trident-protect-schedule-cr.yaml`.
2. Konfigurieren Sie in der erstellten Datei die folgenden Attribute:
  - **metadata.name:** (*required*) der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und sinnvollen Namen für Ihre Umgebung.
  - **Spec.DataMover:** (*Optional*) Eine Zeichenfolge, die angibt, welches Backup-Tool für den Backup-Vorgang verwendet werden soll. Mögliche Werte (Groß-/Kleinschreibung beachten):
    - `Restic`
    - `Kopia` (Standard)
  - **Spec.applicationRef:** Der Kubernetes-Name der zu Back-up Applikation.
  - **Spec.appVaultRef:** (*required*) der Name des AppVault, in dem der Backup-Inhalt gespeichert werden soll.
  - **Spec.backupRetention:** Die Anzahl der zu behaltenden Backups. Null bedeutet, dass keine Backups erstellt werden sollen.
  - **Spec.snapshotRetention:** Die Anzahl der zu behaltenden Snapshots. Null bedeutet, dass keine Snapshots erstellt werden sollen.
  - **spec.granularity:** die Häufigkeit, mit der der Zeitplan ausgeführt werden soll. Mögliche Werte, zusammen mit den erforderlichen zugeordneten Feldern:
    - `hourly` (Erfordert, dass Sie angeben `spec.minute`)
    - `daily` (Erfordert, dass Sie und angeben `spec.minute` `spec.hour`)
    - `weekly` (Erfordert, dass Sie , und `spec.dayOfWeek` angeben `spec.minute` , `spec.hour`)
    - `monthly` (Erfordert, dass Sie , und `spec.dayOfMonth` angeben `spec.minute` , `spec.hour`)
  - **Spec.dayOfMonth:** (*Optional*) der Tag des Monats (1 - 31), an dem der Zeitplan ausgeführt werden soll. Dieses Feld ist erforderlich, wenn die Granularität auf eingestellt ist `monthly`.
  - **Spec.dayOfWeek:** (*Optional*) der Wochentag (0 - 7), an dem der Zeitplan ausgeführt werden soll. Werte von 0 oder 7 zeigen Sonntag an. Dieses Feld ist erforderlich, wenn die Granularität auf eingestellt ist `weekly`.
  - **Spec.hour:** (*Optional*) die Stunde des Tages (0 - 23), die der Zeitplan ausführen soll. Dieses Feld ist erforderlich, wenn die Granularität auf , , oder eingestellt ist `daily` `weekly` `monthly`.
  - **Spec.minute:** (*Optional*) die Minute der Stunde (0 - 59), die der Zeitplan ausführen soll. Dieses Feld ist erforderlich, wenn die Granularität auf , , , oder eingestellt ist `hourly` `daily` `weekly` `monthly`.

```

---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  namespace: my-app-namespace
  name: my-cr-name
spec:
  dataMover: Kopia
  applicationRef: my-application
  appVaultRef: appvault-name
  backupRetention: "15"
  snapshotRetention: "15"
  granularity: <monthly>
  dayOfMonth: "1"
  dayOfWeek: "0"
  hour: "0"
  minute: "0"

```

3. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt `trident-protect-schedule-cr.yaml` haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-schedule-cr.yaml
```

## Erstellen Sie einen Zeitplan über die CLI

### Schritte

1. Erstellen Sie den Schutzplan und ersetzen Sie Werte in Klammern durch Informationen aus Ihrer Umgebung. Beispiel:



Mit können `tridentctl-protect create schedule --help` Sie detaillierte Hilfeinformationen für diesen Befehl anzeigen.

```

tridentctl-protect create schedule <my_schedule_name> --appvault
<my_appvault_name> --app <name_of_app_to_snapshot> --backup
--retention <how_many_backups_to_retain> --data-mover
<kopia_or_restic> --day-of-month <day_of_month_to_run_schedule>
--day-of-week <day_of_month_to_run_schedule> --granularity
<frequency_to_run> --hour <hour_of_day_to_run> --minute
<minute_of_hour_to_run> --recurrence-rule <recurrence> --snapshot
--retention <how_many_snapshots_to_retain> -n <application_namespace>

```



## Löschen Sie einen Snapshot

Löschen Sie die geplanten oder On-Demand Snapshots, die Sie nicht mehr benötigen.

### Schritte

1. Entfernen Sie den Snapshot CR, der dem Snapshot zugeordnet ist:

```
kubectl delete snapshot <snapshot_name> -n my-app-namespace
```

## Löschen Sie ein Backup

Löschen Sie die geplanten oder On-Demand-Backups, die Sie nicht mehr benötigen.

### Schritte

1. Entfernen Sie den Backup-CR, der dem Backup zugeordnet ist:

```
kubectl delete backup <backup_name> -n my-app-namespace
```

## Überprüfen Sie den Status eines Sicherungsvorgangs

Sie können die Befehlszeile verwenden, um den Status eines laufenden, abgeschlossenen oder fehlgeschlagenen Sicherungsvorgangs zu überprüfen.

### Schritte

1. Verwenden Sie den folgenden Befehl, um den Status des Sicherungsvorgangs abzurufen und Werte in Brackern durch Informationen aus Ihrer Umgebung zu ersetzen:

```
kubectl get backup -n <namespace_name> <my_backup_cr_name> -o jsonpath  
='{.status}'
```

## Backup und Restore für Azure-NetApp-Files (ANF)-Vorgänge

Falls Sie Trident Protect installiert haben, können Sie die platzsparenden Backup- und Restore-Funktionen für Storage-Back-Ends aktivieren, die die Azure-NetApp-Files Storage-Klasse verwenden und vor Trident 24.06 erstellt wurden. Diese Funktion arbeitet mit NFSv4-Volumes zusammen und verbraucht keinen zusätzlichen Speicherplatz aus dem Kapazitäts-Pool.

### Bevor Sie beginnen

Stellen Sie Folgendes sicher:

- Sie haben Trident Protect installiert.
- Sie haben eine Anwendung in Trident Protect definiert. Diese Anwendung verfügt nur über begrenzte Schutzfunktionen, bis Sie diesen Vorgang abgeschlossen haben.
- Sie haben `azure-netapp-files` als Standard-Storage-Klasse für Ihr Storage-Back-End ausgewählt.

## Erweitern Sie für Konfigurationsschritte

1. Gehen Sie in Trident folgendermaßen vor, wenn das ANF-Volume vor dem Upgrade auf Trident 24.10 erstellt wurde:
  - a. Aktivieren Sie das Snapshot-Verzeichnis für jedes PV, das auf Azure-NetApp-Dateien basiert und der Anwendung zugeordnet ist:

```
tridentctl update volume <pv name> --snapshot-dir=true -n trident
```

- b. Vergewissern Sie sich, dass das Snapshot-Verzeichnis für jedes zugeordnete PV aktiviert wurde:

```
tridentctl get volume <pv name> -n trident -o yaml | grep  
snapshotDir
```

Antwort:

```
snapshotDirectory: "true"
```

+

Wenn das Snapshot-Verzeichnis nicht aktiviert ist, wählt Trident Protect die regelmäßige Backup-Funktion aus, die während des Backup-Prozesses vorübergehend Speicherplatz im Kapazitäts-Pool verbraucht. Stellen Sie in diesem Fall sicher, dass im Kapazitätspool ausreichend Speicherplatz verfügbar ist, um ein temporäres Volume der Größe des zu sichernden Volumes zu erstellen.

### Ergebnis

Die Applikation ist mit Trident Protect für die Sicherung und Wiederherstellung bereit. Jede PVC kann auch von anderen Anwendungen für Backups und Wiederherstellungen verwendet werden.

## Stellen Sie Applikationen mit Trident Protect wieder her

Sie können Trident Protect verwenden, um Ihre Anwendung aus einem Snapshot oder einem Backup wiederherzustellen. Das Wiederherstellen aus einem vorhandenen Snapshot erfolgt schneller, wenn die Anwendung auf dasselbe Cluster wiederhergestellt wird.



Wenn Sie eine Anwendung wiederherstellen, werden alle für die Anwendung konfigurierten Ausführungshaken mit der App wiederhergestellt. Wenn ein Hook für die Ausführung nach der Wiederherstellung vorhanden ist, wird er automatisch als Teil des Wiederherstellungsvorgangs ausgeführt.

### Namespace-Annotationen und -Labels bei Restore- und Failover-Vorgängen

Während von Restore- und Failover-Vorgängen werden Labels und Annotationen im Ziel-Namespace so angefertigt, dass sie den Labels und Annotationen im Quell-Namespace entsprechen. Beschriftungen oder

Anmerkungen aus dem Quell-Namespace, die nicht im Ziel-Namespace vorhanden sind, werden hinzugefügt. Alle bereits vorhandenen Beschriftungen oder Anmerkungen werden überschrieben, um dem Wert aus dem Quell-Namespace zu entsprechen. Beschriftungen oder Anmerkungen, die nur im Ziel-Namespace vorhanden sind, bleiben unverändert.



Wenn Sie RedHat OpenShift verwenden, ist es wichtig, die wichtige Rolle von Namespace-Annotationen in OpenShift-Umgebungen zu beachten. Namespace-Anmerkungen stellen sicher, dass wiederhergestellte Pods den durch OpenShift Security Context Constraints (SCCs) definierten Berechtigungen und Sicherheitskonfigurationen entsprechen und ohne Berechtigungsprobleme auf Volumes zugreifen können. Weitere Informationen finden Sie im ["Dokumentation zu den Einschränkungen für den Sicherheitskontext von OpenShift"](#).

Sie können verhindern, dass bestimmte Annotationen im Ziel-Namespace überschrieben werden, indem Sie die Kubernetes-Umgebungsvariable vor der Wiederherstellung oder dem Failover einstellen `RESTORE_SKIP_NAMESPACE_ANNOTATIONS`. Beispiel:

```
kubectl set env -n trident-protect deploy/trident-protect-controller-
manager
RESTORE_SKIP_NAMESPACE_ANNOTATIONS=<annotation_key_to_skip_1>,<annotation_
key_to_skip_2>
```

Wenn Sie die Quellanwendung mit Helm mit dem Flag installiert haben, wird der Label-Schlüssel mit `--create-namespace` einer speziellen Bestrahlung `name` versehen. Während des Wiederherstellungs- oder Failover-Prozesses kopiert Trident Protect dieses Label in den Ziel-Namespace, aktualisiert jedoch den Wert auf den Ziel-Namespace-Wert, wenn der Wert von Quell-Namespace mit dem Quell-Namespace übereinstimmt. Wenn dieser Wert nicht mit dem Quell-Namespace übereinstimmt, wird er ohne Änderungen in den Ziel-Namespace kopiert.

**Beispiel**

Im folgenden Beispiel wird ein Quell- und Ziel-Namespace präsentiert, der jeweils unterschiedliche Annotationen und Labels enthält. Sie können den Status des Ziel-Namespace vor und nach dem Vorgang anzeigen und sehen, wie die Annotationen und Labels im Ziel-Namespace kombiniert oder überschrieben werden.

**Vor der Wiederherstellung oder dem Failover**

Die folgende Tabelle zeigt den Status der Beispiel-Namespace für Quelle und Ziel vor dem Restore- oder Failover-Vorgang:

Namespace	Anmerkungen	Etiketten
Namespace ns-1 (Quelle)	<ul style="list-style-type: none"> <li>• Annotation.one/key: „Aktualisiertwert“</li> <li>• Anmerkung.zwei/Taste: "Wahr"</li> </ul>	<ul style="list-style-type: none"> <li>• Umgebung = Produktion</li> <li>• Compliance=hipaa</li> <li>• Name=ns-1</li> </ul>
Namespace ns-2 (Ziel)	<ul style="list-style-type: none"> <li>• Anmerkung.One/key: „True“</li> <li>• Anmerkung.drei/Taste: "False"</li> </ul>	<ul style="list-style-type: none"> <li>• Rolle=Datenbank</li> </ul>

## Nach dem Wiederherstellungsvorgang

In der folgenden Tabelle wird der Status des Beispiel-Ziel-Namespace nach der Wiederherstellung oder dem Failover-Vorgang dargestellt. Einige Schlüssel wurden hinzugefügt, einige wurden überschrieben und das Label wurde aktualisiert, `name` um dem Ziel-Namespace zu entsprechen:

Namespace	Anmerkungen	Etiketten
Namespace ns-2 (Ziel)	<ul style="list-style-type: none"><li>• Annotation.one/key: „Aktualisiertwert“</li><li>• Anmerkung.zwei/Taste: "Wahr"</li><li>• Anmerkung.drei/Taste: "False"</li></ul>	<ul style="list-style-type: none"><li>• Name=ns-2</li><li>• Compliance=hipaa</li><li>• Umgebung = Produktion</li><li>• Rolle=Datenbank</li></ul>

## Wiederherstellung aus einem Backup in einem anderen Namespace

Wenn Sie ein Backup mit einem Backup Restore CR in einem anderen Namespace wiederherstellen, stellt Trident Protect die Anwendung in einem neuen Namespace wieder her und erstellt eine Anwendungs-CR für die wiederhergestellte Anwendung. Erstellen Sie zum Schutz der wiederhergestellten Applikation On-Demand-Backups oder Snapshots oder erstellen Sie einen Sicherungszeitplan.



Wenn Sie ein Backup in einem anderen Namespace mit vorhandenen Ressourcen wiederherstellen, werden keine Ressourcen verändert, die Namen mit denen im Backup teilen. Um alle Ressourcen im Backup wiederherzustellen, löschen Sie entweder den Ziel-Namespace und erstellen Sie ihn neu, oder stellen Sie das Backup in einem neuen Namespace wieder her.

## CR verwenden

### Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie `trident-protect-backup-restore-cr.yaml`.
2. Konfigurieren Sie in der erstellten Datei die folgenden Attribute:
  - **metadata.name:** (*required*) der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und sinnvollen Namen für Ihre Umgebung.
  - **Spec.appArchivePath:** Der Pfad innerhalb von AppVault, in dem die Backup-Inhalte gespeichert werden. Sie können den folgenden Befehl verwenden, um diesen Pfad zu finden:

```
kubectl get backups <BACKUP_NAME> -n my-app-namespace -o  
jsonpath='{.status.appArchivePath}'
```

- **Spec.appVaultRef:** (*required*) der Name des AppVault, in dem der Backup-Inhalt gespeichert ist.
- **spec.namespaceMapping:** die Zuordnung des Quell-Namespaces des Wiederherstellungsvorgangs zum Ziel-Namespace. Ersetzen `my-source-namespace` Sie und `my-destination-namespace` mit Informationen aus Ihrer Umgebung.
- **Spec.storageClassMapping:** Das Mapping der Quellspeicherklasse des Wiederherstellungsvorgangs an die Zielspeicherklasse. Ersetzen `destinationStorageClass` Sie und `sourceStorageClass` mit Informationen aus Ihrer Umgebung.

```
---  
apiVersion: protect.trident.netapp.io/v1  
kind: BackupRestore  
metadata:  
  name: my-cr-name  
  namespace: my-destination-namespace  
spec:  
  appArchivePath: my-backup-path  
  appVaultRef: appvault-name  
  namespaceMapping: [{"source": "my-source-namespace",  
"destination": "my-destination-namespace"}]  
  storageClassMapping:  
    destination: "${destinationStorageClass}"  
    source: "${sourceStorageClass}"
```

3. (*Optional*) Wenn Sie nur bestimmte Ressourcen der wiederherzustellenden Anwendung auswählen müssen, fügen Sie eine Filterung hinzu, die Ressourcen mit bestimmten Bezeichnungen enthält oder ausschließt:
  - **ResourceFilter.resourceSelectionCriteria:** (Erforderlich für die Filterung) Verwenden `Include` oder `Exclude` um eine in `resourceMatchers` definierte Ressource ein- oder auszuschließen. Fügen Sie die folgenden `resourceMatchers`-Parameter hinzu, um die einzuschließenden oder auszuschließenden Ressourcen zu definieren:

- **RefindeFilter.refindeMatchers:** Eine Reihe von refindeMatcher-Objekten. Wenn Sie mehrere Elemente in diesem Array definieren, stimmen sie als OP-Operation überein, und die Felder innerhalb jedes Elements (Gruppe, Typ, Version) stimmen mit einer UND-Operation überein.
  - **ResourceMatchers[].Group:** (*Optional*) Gruppe der zu filternden Ressource.
  - **ResourceMatchers[].Kind:** (*Optional*) Art der zu filternden Ressource.
  - **ResourceMatchers[].Version:** (*Optional*) Version der zu filternden Ressource.
  - **ResourceMatchers[].Namen:** (*Optional*) Namen im Kubernetes metadata.name-Feld der zu filternden Ressource.
  - **ResourceMatchers[].Namespaces:** (*Optional*) Namespaces im Kubernetes metadata.name-Feld der zu filternden Ressource.
  - **ResourceMatchers[].labelSelectors:** (*Optional*) Label selector string im Feld Kubernetes metadata.name der Ressource, wie im definiert "[Kubernetes-Dokumentation](#)". Zum Beispiel: "trident.netapp.io/os=linux".

Beispiel:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt trident-protect-backup-restore-cr.yaml haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-backup-restore-cr.yaml
```

## Verwenden Sie die CLI

### Schritte

1. Stellen Sie das Backup in einem anderen Namespace wieder her und ersetzen Sie die Werte in Klammern durch Informationen aus Ihrer Umgebung. Das namespace-mapping Argument verwendet durch Doppelpunkte getrennte Namespaces, um Quellnamespaces im Format den richtigen Zielnamespaces zuzuordnen source1:dest1, source2:dest2. Beispiel:

```
tridentctl-protect create backuprestore <my_restore_name> --backup  
<backup_namespace>/<backup_to_restore> --namespace-mapping  
<source_to_destination_namespace_mapping> -n <application_namespace>
```

### Wiederherstellung von einem Backup in den ursprünglichen Namespace

Sie können ein Backup im ursprünglichen Namespace jederzeit wiederherstellen.

## CR verwenden

### Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie `trident-protect-backup-ipr-cr.yaml`.
2. Konfigurieren Sie in der erstellten Datei die folgenden Attribute:
  - **metadata.name:** (*required*) der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und sinnvollen Namen für Ihre Umgebung.
  - **Spec.appArchivePath:** Der Pfad innerhalb von AppVault, in dem die Backup-Inhalte gespeichert werden. Sie können den folgenden Befehl verwenden, um diesen Pfad zu finden:

```
kubectl get backups <BACKUP_NAME> -n my-app-namespace -o  
jsonpath='{.status.appArchivePath}'
```

- **Spec.appVaultRef:** (*required*) der Name des AppVault, in dem der Backup-Inhalt gespeichert ist.

Beispiel:

```
---  
apiVersion: protect.trident.netapp.io/v1  
kind: BackupInplaceRestore  
metadata:  
  name: my-cr-name  
  namespace: my-app-namespace  
spec:  
  appArchivePath: my-backup-path  
  appVaultRef: appvault-name
```

3. (*Optional*) Wenn Sie nur bestimmte Ressourcen der wiederherzustellenden Anwendung auswählen müssen, fügen Sie eine Filterung hinzu, die Ressourcen mit bestimmten Bezeichnungen enthält oder ausschließt:
  - **ResourceFilter.resourceSelectionCriteria:** (Erforderlich für die Filterung) Verwenden `Include` oder `Exclude` um eine in `resourceMatchers` definierte Ressource ein- oder auszuschließen. Fügen Sie die folgenden `resourceMatchers`-Parameter hinzu, um die einzuschließenden oder auszuschließenden Ressourcen zu definieren:
    - **RefindeFilter.refindeMatchers:** Eine Reihe von `refindeMatcher`-Objekten. Wenn Sie mehrere Elemente in diesem Array definieren, stimmen sie als OP-Operation überein, und die Felder innerhalb jedes Elements (Gruppe, Typ, Version) stimmen mit einer UND-Operation überein.
      - **ResourceMatchers[].Group:** (*Optional*) Gruppe der zu filternden Ressource.
      - **ResourceMatchers[].Kind:** (*Optional*) Art der zu filternden Ressource.
      - **ResourceMatchers[].Version:** (*Optional*) Version der zu filternden Ressource.
      - **ResourceMatchers[].Namen:** (*Optional*) Namen im Kubernetes `metadata.name`-Feld der zu filternden Ressource.



- **ResourceMatchers[].Namespaces:** (*Optional*) Namespaces im Kubernetes metadata.name-Feld der zu filternden Ressource.
- **ResourceMatchers[].labelSelectors:** (*Optional*) Label selector string im Feld Kubernetes metadata.name der Ressource, wie im definiert "[Kubernetes-Dokumentation](#)". Zum Beispiel: "trident.netapp.io/os=linux".

Beispiel:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt trident-protect-backup-ipr-cr.yaml haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-backup-ipr-cr.yaml
```

## Verwenden Sie die CLI

### Schritte

1. Stellen Sie das Backup auf den ursprünglichen Namespace wieder her, und ersetzen Sie die Werte in Klammern durch Informationen aus Ihrer Umgebung. Das backup Argument verwendet einen Namespace und einen Backup-Namen im Format <namespace>/<name>. Beispiel:

```
tridentctl-protect create backupinplacerestore <my_restore_name>
--backup <namespace/backup_to_restore> -n <application_namespace>
```

## Wiederherstellung von einem Backup in einem anderen Cluster

Sie können ein Backup in einem anderen Cluster wiederherstellen, wenn ein Problem mit dem ursprünglichen Cluster auftritt.

## Bevor Sie beginnen

Stellen Sie sicher, dass die folgenden Voraussetzungen erfüllt sind:

- Auf dem Zielcluster ist Trident Protect installiert.
- Der Zielcluster hat Zugriff auf den Bucket-Pfad desselben AppVault wie das Quellcluster, in dem das Backup gespeichert ist.

## Schritte

1. Überprüfen Sie die Verfügbarkeit von AppVault CR auf dem Zielcluster mithilfe des Trident Protect CLI-Plug-ins:

```
tridentctl-protect get appvault --context <destination_cluster_name>
```



Stellen Sie sicher, dass der für die Anwendungswiederherstellung vorgesehene Namespace auf dem Zielcluster vorhanden ist.

2. Zeigen Sie die Backup-Inhalte des verfügbaren AppVault vom Zielcluster an:

```
tridentctl-protect get appvaultcontent <appvault_name> --show-resources  
backup --show-paths --context <destination_cluster_name>
```

Mit diesem Befehl werden die verfügbaren Backups im AppVault angezeigt, einschließlich der ursprünglichen Cluster, der entsprechenden Anwendungsnamen, Zeitstempel und Archivpfade.

### Beispielausgabe:

```
+-----+-----+-----+-----+  
+-----+-----+-----+-----+  
| CLUSTER | APP | TYPE | NAME | TIMESTAMP  
| PATH |  
+-----+-----+-----+-----+  
+-----+-----+-----+-----+  
| production1 | wordpress | backup | wordpress-bkup-1 | 2024-10-30  
08:37:40 (UTC) | backuppath1 |  
| production1 | wordpress | backup | wordpress-bkup-2 | 2024-10-30  
08:37:40 (UTC) | backuppath2 |  
+-----+-----+-----+-----+  
+-----+-----+-----+-----+
```

3. Stellen Sie die Anwendung mithilfe des AppVault-Namens und Archivpfads auf dem Zielcluster wieder her:

## CR verwenden

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie `trident-protect-backup-restore-cr.yaml`.
2. Konfigurieren Sie in der erstellten Datei die folgenden Attribute:
  - **metadata.name:** (*required*) der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und sinnvollen Namen für Ihre Umgebung.
  - **Spec.appVaultRef:** (*required*) der Name des AppVault, in dem der Backup-Inhalt gespeichert ist.
  - **Spec.appArchivePath:** Der Pfad innerhalb von AppVault, in dem die Backup-Inhalte gespeichert werden. Sie können den folgenden Befehl verwenden, um diesen Pfad zu finden:

```
kubectl get backups <BACKUP_NAME> -n my-app-namespace -o jsonpath='{.status.appArchivePath}'
```



Wenn BackupRestore CR nicht verfügbar ist, können Sie den in Schritt 2 genannten Befehl verwenden, um den Inhalt des Backups anzuzeigen.

- **spec.namespaceMapping:** die Zuordnung des Quell-Namespaces des Wiederherstellungsvorgangs zum Ziel-Namespace. Ersetzen `my-source-namespace` Sie und `my-destination-namespace` mit Informationen aus Ihrer Umgebung.

Beispiel:

```
apiVersion: protect.trident.netapp.io/v1
kind: BackupRestore
metadata:
  name: my-cr-name
  namespace: my-destination-namespace
spec:
  appVaultRef: appvault-name
  appArchivePath: my-backup-path
  namespaceMapping: [{"source": "my-source-namespace", "destination": "my-destination-namespace"}]
```

3. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt `trident-protect-backup-restore-cr.yaml` haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-backup-restore-cr.yaml
```

## Verwenden Sie die CLI

1. Verwenden Sie den folgenden Befehl, um die Anwendung wiederherzustellen und Werte in Klammern durch Informationen aus Ihrer Umgebung zu ersetzen. Das Namespace-Mapping-Argument verwendet durch Doppelpunkte getrennte Namespaces, um Quellnamespaces im Format

source1:dest1,source2:dest2 den korrekten Zielnamespaces zuzuordnen. Beispiel:

```
tridentctl-protect create backuprestore <restore_name> --namespace  
-mapping <source_to_destination_namespace_mapping> --appvault  
<appvault_name> --path <backup_path> -n <application_namespace>  
--context <destination_cluster_name>
```

## Wiederherstellung von einem Snapshot in einem anderen Namespace

Sie können Daten aus einem Snapshot mithilfe einer benutzerdefinierten Ressourcendatei (CR) entweder in einem anderen Namespace oder im ursprünglichen QuellNamespace wiederherstellen. Wenn Sie einen Snapshot mithilfe eines SnapshotRestore CR in einem anderen Namespace wiederherstellen, stellt Trident Protect die Anwendung in einem neuen Namespace wieder her und erstellt eine Anwendungs-CR für die wiederhergestellte Anwendung. Erstellen Sie zum Schutz der wiederhergestellten Applikation On-Demand-Backups oder Snapshots oder erstellen Sie einen Sicherungszeitplan.

## CR verwenden

### Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie `trident-protect-snapshot-restore-cr.yaml`.
2. Konfigurieren Sie in der erstellten Datei die folgenden Attribute:
  - **metadata.name:** (*required*) der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und sinnvollen Namen für Ihre Umgebung.
  - **Spec.appVaultRef:** (*required*) der Name des AppVault, in dem der Snapshot-Inhalt gespeichert ist.
  - **Spec.appArchivePath:** Der Pfad innerhalb von AppVault, wo der Snapshot-Inhalt gespeichert wird. Sie können den folgenden Befehl verwenden, um diesen Pfad zu finden:

```
kubectl get snapshots <SNAPSHOT_NAME> -n my-app-namespace -o  
jsonpath='{.status.appArchivePath}'
```

- **spec.namespaceMapping:** die Zuordnung des Quell-Namespaces des Wiederherstellungsvorgangs zum Ziel-Namespace. Ersetzen `my-source-namespace` Sie und `my-destination-namespace` mit Informationen aus Ihrer Umgebung.
- **Spec.storageClassMapping:** Das Mapping der Quellspeicherklasse des Wiederherstellungsvorgangs an die Zielspeicherklasse. Ersetzen `destinationStorageClass` Sie und `sourceStorageClass` mit Informationen aus Ihrer Umgebung.

```
---  
apiVersion: protect.trident.netapp.io/v1  
kind: SnapshotRestore  
metadata:  
  name: my-cr-name  
  namespace: my-app-namespace  
spec:  
  appVaultRef: appvault-name  
  appArchivePath: my-snapshot-path  
  namespaceMapping: [{"source": "my-source-namespace",  
"destination": "my-destination-namespace"}]  
  storageClassMapping:  
    destination: "${destinationStorageClass}"  
    source: "${sourceStorageClass}"
```

3. (*Optional*) Wenn Sie nur bestimmte Ressourcen der wiederherzustellenden Anwendung auswählen müssen, fügen Sie eine Filterung hinzu, die Ressourcen mit bestimmten Bezeichnungen enthält oder ausschließt:
  - **ResourceFilter.resourceSelectionCriteria:** (Erforderlich für die Filterung) Verwenden `Include` oder `Exclude` um eine in `resourceMatchers` definierte Ressource ein- oder auszuschließen. Fügen Sie die folgenden `resourceMatchers`-Parameter hinzu, um die einzuschließenden oder

auszuschließenden Ressourcen zu definieren:

- **RefindeFilter.refindeMatchers:** Eine Reihe von refindeMatcher-Objekten. Wenn Sie mehrere Elemente in diesem Array definieren, stimmen sie als OP-Operation überein, und die Felder innerhalb jedes Elements (Gruppe, Typ, Version) stimmen mit einer UND-Operation überein.
  - **ResourceMatchers[].Group:** (*Optional*) Gruppe der zu filternden Ressource.
  - **ResourceMatchers[].Kind:** (*Optional*) Art der zu filternden Ressource.
  - **ResourceMatchers[].Version:** (*Optional*) Version der zu filternden Ressource.
  - **ResourceMatchers[].Namen:** (*Optional*) Namen im Kubernetes metadata.name-Feld der zu filternden Ressource.
  - **ResourceMatchers[].Namespaces:** (*Optional*) Namespaces im Kubernetes metadata.name-Feld der zu filternden Ressource.
  - **ResourceMatchers[].labelSelectors:** (*Optional*) Label selector string im Feld Kubernetes metadata.name der Ressource, wie im definiert "[Kubernetes-Dokumentation](#)". Zum Beispiel: "trident.netapp.io/os=linux".

Beispiel:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt trident-protect-snapshot-restore-cr.yaml haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-snapshot-restore-cr.yaml
```

## Verwenden Sie die CLI

### Schritte

1. Stellen Sie den Snapshot in einem anderen Namespace wieder her und ersetzen Sie Werte in Klammern durch Informationen aus Ihrer Umgebung.

- Das `snapshot` Argument verwendet einen Namespace und Snapshot-Namen im Format `<namespace>/<name>`.
- Das `namespace-mapping` Argument verwendet durch Doppelpunkte getrennte Namespaces, um Quellnamespaces im Format den richtigen Zielnamespaces zuzuordnen `source1:dest1,source2:dest2`.

Beispiel:

```
tridentctl-protect create snapshotrestore <my_restore_name>  
--snapshot <namespace/snapshot_to_restore> --namespace-mapping  
<source_to_destination_namespace_mapping> -n <application_namespace>
```

### Wiederherstellung von einem Snapshot im ursprünglichen Namespace

Sie können einen Snapshot jederzeit im ursprünglichen Namespace wiederherstellen.

## CR verwenden

### Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie `trident-protect-snapshot-ipr-cr.yaml`.
2. Konfigurieren Sie in der erstellten Datei die folgenden Attribute:
  - **metadata.name:** (*required*) der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und sinnvollen Namen für Ihre Umgebung.
  - **Spec.appVaultRef:** (*required*) der Name des AppVault, in dem der Snapshot-Inhalt gespeichert ist.
  - **Spec.appArchivePath:** Der Pfad innerhalb von AppVault, wo der Snapshot-Inhalt gespeichert wird. Sie können den folgenden Befehl verwenden, um diesen Pfad zu finden:

```
kubectl get snapshots <SNAPSHOT_NAME> -n my-app-namespace -o  
jsonpath='{.status.appArchivePath}'
```

```
---  
apiVersion: protect.trident.netapp.io/v1  
kind: SnapshotInplaceRestore  
metadata:  
  name: my-cr-name  
  namespace: my-app-namespace  
spec:  
  appVaultRef: appvault-name  
  appArchivePath: my-snapshot-path
```

3. (*Optional*) Wenn Sie nur bestimmte Ressourcen der wiederherzustellenden Anwendung auswählen müssen, fügen Sie eine Filterung hinzu, die Ressourcen mit bestimmten Bezeichnungen enthält oder ausschließt:
  - **ResourceFilter.resourceSelectionCriteria:** (Erforderlich für die Filterung) Verwenden `Include` oder `Exclude` um eine in `resourceMatchers` definierte Ressource ein- oder auszuschließen. Fügen Sie die folgenden `resourceMatchers`-Parameter hinzu, um die einzuschließenden oder auszuschließenden Ressourcen zu definieren:
    - **RefindeFilter.refindeMatchers:** Eine Reihe von `refindeMatcher`-Objekten. Wenn Sie mehrere Elemente in diesem Array definieren, stimmen sie als OP-Operation überein, und die Felder innerhalb jedes Elements (Gruppe, Typ, Version) stimmen mit einer UND-Operation überein.
      - **ResourceMatchers[].Group:** (*Optional*) Gruppe der zu filternden Ressource.
      - **ResourceMatchers[].Kind:** (*Optional*) Art der zu filternden Ressource.
      - **ResourceMatchers[].Version:** (*Optional*) Version der zu filternden Ressource.
      - **ResourceMatchers[].Namen:** (*Optional*) Namen im Kubernetes `metadata.name`-Feld der zu filternden Ressource.
      - **ResourceMatchers[].Namespaces:** (*Optional*) Namespaces im Kubernetes `metadata.name`-Feld der zu filternden Ressource.



- **ResourceMatchers[].labelSelectors:** (*Optional*) Label selector string im Feld Kubernetes metadata.name der Ressource, wie im definiert "[Kubernetes-Dokumentation](#)". Zum Beispiel: "trident.netapp.io/os=linux".

Beispiel:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt trident-protect-snapshot-ipr-cr.yaml haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-snapshot-ipr-cr.yaml
```

## Verwenden Sie die CLI

### Schritte

1. Stellen Sie den Snapshot auf den ursprünglichen Namespace wieder her, und ersetzen Sie Werte in Klammern durch Informationen aus Ihrer Umgebung. Beispiel:

```
tridentctl-protect create snapshotinplacerestore <my_restore_name>
--snapshot <snapshot_to_restore> -n <application_namespace>
```

## Überprüfen Sie den Status eines Wiederherstellungsvorgangs

Sie können die Befehlszeile verwenden, um den Status eines Wiederherstellungsvorgangs zu überprüfen, der gerade ausgeführt wird, abgeschlossen wurde oder fehlgeschlagen ist.

### Schritte

1. Verwenden Sie den folgenden Befehl, um den Status des Wiederherstellungsvorgangs abzurufen und

Werte in Bracken durch Informationen aus Ihrer Umgebung zu ersetzen:

```
kubectl get backuprestore -n <namespace_name> <my_restore_cr_name> -o  
jsonpath='{.status}'
```

## Replizieren Sie Applikationen mit NetApp SnapMirror und Trident Protect

Mit Trident Protect können Sie die asynchronen Replizierungsfunktionen der NetApp SnapMirror Technologie verwenden, um Daten und Applikationsänderungen von einem Storage-Back-End auf ein anderes zu replizieren, im selben Cluster oder zwischen verschiedenen Clustern.

### Namespace-Annotationen und -Labels bei Restore- und Failover-Vorgängen

Während von Restore- und Failover-Vorgängen werden Labels und Annotationen im Ziel-Namespace so angefertigt, dass sie den Labels und Annotationen im Quell-Namespace entsprechen. Beschriftungen oder Anmerkungen aus dem Quell-Namespace, die nicht im Ziel-Namespace vorhanden sind, werden hinzugefügt. Alle bereits vorhandenen Beschriftungen oder Anmerkungen werden überschrieben, um dem Wert aus dem Quell-Namespace zu entsprechen. Beschriftungen oder Anmerkungen, die nur im Ziel-Namespace vorhanden sind, bleiben unverändert.



Wenn Sie RedHat OpenShift verwenden, ist es wichtig, die wichtige Rolle von Namespace-Annotationen in OpenShift-Umgebungen zu beachten. Namespace-Anmerkungen stellen sicher, dass wiederhergestellte Pods den durch OpenShift Security Context Constraints (SCCs) definierten Berechtigungen und Sicherheitskonfigurationen entsprechen und ohne Berechtigungsprobleme auf Volumes zugreifen können. Weitere Informationen finden Sie im ["Dokumentation zu den Einschränkungen für den Sicherheitskontext von OpenShift"](#).

Sie können verhindern, dass bestimmte Annotationen im Ziel-Namespace überschrieben werden, indem Sie die Kubernetes-Umgebungsvariable vor der Wiederherstellung oder dem Failover einstellen `RESTORE_SKIP_NAMESPACE_ANNOTATIONS`. Beispiel:

```
kubectl set env -n trident-protect deploy/trident-protect-controller-  
manager  
RESTORE_SKIP_NAMESPACE_ANNOTATIONS=<annotation_key_to_skip_1>,<annotation_  
key_to_skip_2>
```

Wenn Sie die Quellanwendung mit Helm mit dem Flag installiert haben, wird der Label-Schlüssel mit `--create-namespace` einer speziellen Bestrahlung `name` versehen. Während des Wiederherstellungs- oder Failover-Prozesses kopiert Trident Protect dieses Label in den Ziel-Namespace, aktualisiert jedoch den Wert auf den Ziel-Namespace-Wert, wenn der Wert von Quell-Namespace mit dem Quell-Namespace übereinstimmt. Wenn dieser Wert nicht mit dem Quell-Namespace übereinstimmt, wird er ohne Änderungen in den Ziel-Namespace kopiert.

### Beispiel

Im folgenden Beispiel wird ein Quell- und Ziel-Namespace präsentiert, der jeweils unterschiedliche Annotationen und Labels enthält. Sie können den Status des Ziel-Namespace vor und nach dem Vorgang

anzeigen und sehen, wie die Annotationen und Labels im Ziel-Namespaces kombiniert oder überschrieben werden.

### Vor der Wiederherstellung oder dem Failover

Die folgende Tabelle zeigt den Status der Beispiel-Namespaces für Quelle und Ziel vor dem Restore- oder Failover-Vorgang:

Namespace	Anmerkungen	Etiketten
Namespace ns-1 (Quelle)	<ul style="list-style-type: none"><li>• Annotation.one/key: „Aktualisiertwert“</li><li>• Anmerkung.zwei/Taste: "Wahr"</li></ul>	<ul style="list-style-type: none"><li>• Umgebung = Produktion</li><li>• Compliance=hipaa</li><li>• Name=ns-1</li></ul>
Namespace ns-2 (Ziel)	<ul style="list-style-type: none"><li>• Anmerkung.One/key: „True“</li><li>• Anmerkung.drei/Taste: "False"</li></ul>	<ul style="list-style-type: none"><li>• Rolle=Datenbank</li></ul>

### Nach dem Wiederherstellungsvorgang

In der folgenden Tabelle wird der Status des Beispiel-Ziel-Namespaces nach der Wiederherstellung oder dem Failover-Vorgang dargestellt. Einige Schlüssel wurden hinzugefügt, einige wurden überschrieben und das Label wurde aktualisiert, `name` um dem Ziel-Namespaces zu entsprechen:

Namespace	Anmerkungen	Etiketten
Namespace ns-2 (Ziel)	<ul style="list-style-type: none"><li>• Annotation.one/key: „Aktualisiertwert“</li><li>• Anmerkung.zwei/Taste: "Wahr"</li><li>• Anmerkung.drei/Taste: "False"</li></ul>	<ul style="list-style-type: none"><li>• Name=ns-2</li><li>• Compliance=hipaa</li><li>• Umgebung = Produktion</li><li>• Rolle=Datenbank</li></ul>



Sie können Trident Protect so konfigurieren, dass Dateisysteme während des Datenschutzes eingefroren und wieder eingefroren werden. ["Erfahren Sie mehr über das Konfigurieren des Dateisystemfrierens mit Trident Protect"](#).

### Richten Sie eine Replikationsbeziehung ein

Die Einrichtung einer Replikationsbeziehung umfasst Folgendes:

- Festlegen der Häufigkeit, die Trident schützen soll, um einen App-Snapshot zu erstellen (was die Kubernetes-Ressourcen der Applikation sowie die Volume-Snapshots für die Volumes der Applikation umfasst)
- Auswahl des Replizierungszeitplans (einschließlich Kubernetes-Ressourcen und persistenter Volume-Daten)
- Einstellen der Uhrzeit für die Erstellung des Snapshots

### Schritte

1. Erstellen Sie einen AppVault für die Quellanwendung auf dem Quellcluster. Ändern Sie abhängig von Ihrem Storage-Anbieter ein Beispiel in ["Benutzerdefinierte Ressourcen von AppVault"](#) entsprechend Ihrer

Umgebung:

## Erstellen Sie einen AppVault mit einem CR

- a. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie (z. B. `trident-protect-appvault-primary-source.yaml`).
- b. Konfigurieren Sie die folgenden Attribute:
  - **metadata.name:** (*required*) der Name der benutzerdefinierten AppVault-Ressource. Notieren Sie sich den ausgewählten Namen, da sich andere CR-Dateien, die für eine Replikationsbeziehung benötigt werden, auf diesen Wert beziehen.
  - **spec.providerConfig:** (*required*) speichert die Konfiguration, die für den Zugriff auf den AppVault unter Verwendung des angegebenen Providers erforderlich ist. Wählen Sie einen BucketName und alle anderen notwendigen Details für Ihren Anbieter. Notieren Sie sich die ausgewählten Werte, da sich andere CR-Dateien, die für eine Replikationsbeziehung benötigt werden, auf diese Werte beziehen. Beispiele für AppVault CRS mit anderen Anbietern finden Sie unter "[Benutzerdefinierte Ressourcen von AppVault](#)".
  - **spec.providerCredentials:** (*required*) speichert Verweise auf alle Anmeldeinformationen, die für den Zugriff auf AppVault unter Verwendung des angegebenen Anbieters erforderlich sind.
    - **spec.providerCredentials.valueFromSecret:** (*required*) gibt an, dass der Wert der Zugangsdaten von einem Secret stammen soll.
      - **Key:** (*required*) der gültige Schlüssel des zu wählenden Geheimnisses.
      - **Name:** (*required*) Name des Geheimnisses, das den Wert für dieses Feld enthält. Muss sich im gleichen Namespace befinden.
    - **spec.providerCredentials.secretAccessKey:** (*required*) der Zugriffsschlüssel, der für den Zugriff auf den Provider verwendet wird. Der **Name** sollte mit **spec.providerCredentials.valueFromSecret.name** übereinstimmen.
  - **spec.providerType:** (*required*) legt fest, was das Backup bereitstellt, z. B. NetApp ONTAP S3, generisches S3, Google Cloud oder Microsoft Azure. Mögliche Werte:
    - AWS
    - Azure
    - GCP
    - Generisch-s3
    - ONTAP-s3
    - StorageGRID-s3
- c. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt `trident-protect-appvault-primary-source.yaml` haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-appvault-primary-source.yaml -n trident-protect
```

## Erstellen Sie einen AppVault mithilfe der CLI

- a. Erstellen Sie AppVault und ersetzen Sie Werte in Klammern durch Informationen aus Ihrer Umgebung:

```
tridentctl-protect create vault Azure <vault-name> --account  
<account-name> --bucket <bucket-name> --secret <secret-name>
```

2. Erstellen Sie die CR-Quellanwendung:

### Erstellen Sie die Quellenanwendung mit einem CR

- a. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie (z. B. `trident-protect-app-source.yaml`).
- b. Konfigurieren Sie die folgenden Attribute:
  - **metadata.name:** (*required*) der Name der benutzerdefinierten Ressource der Anwendung. Notieren Sie sich den ausgewählten Namen, da sich andere CR-Dateien, die für eine Replikationsbeziehung benötigt werden, auf diesen Wert beziehen.
  - **spec.includedNamespaces:** (*required*) ein Array von Namespaces und zugehörigen Labels. Verwenden Sie Namespace-Namen und Grenzen Sie optional den Umfang der Namespaces mit Beschriftungen ein, um Ressourcen anzugeben, die in den hier aufgeführten Namespaces vorhanden sind. Der Application Namespace muss Teil dieses Arrays sein.

#### Beispiel YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  name: my-app-name
  namespace: my-app-namespace
spec:
  includedNamespaces:
    - namespace: my-app-namespace
      labelSelector: {}
```

- c. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt `trident-protect-app-source.yaml` haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-app-source.yaml -n my-app-namespace
```

### Erstellen Sie die Quellenanwendung mithilfe der CLI

- a. Erstellen Sie die Quellenanwendung. Beispiel:

```
tridentctl-protect create app <my-app-name> --namespaces
<namespaces-to-be-included> -n <my-app-namespace>
```

3. Optional können Sie einen Snapshot der Quellenanwendung erstellen. Dieser Snapshot wird als Basis für die Anwendung auf dem Zielcluster verwendet. Wenn Sie diesen Schritt überspringen, müssen Sie warten, bis der nächste geplante Snapshot ausgeführt wird, damit Sie über einen aktuellen Snapshot verfügen.

## Erstellen Sie einen Schnappschuss mit einem CR-System

a. Erstellen Sie einen Replikationszeitplan für die Quellanwendung:

- i. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie (z. B. `trident-protect-schedule.yaml`).
- ii. Konfigurieren Sie die folgenden Attribute:
  - **metadata.name:** (*required*) der Name der benutzerdefinierten Ressource für den Zeitplan.
  - **Spec.AppVaultRef:** (*required*) dieser Wert muss mit dem Feld `metadata.name` des AppVault für die Quellanwendung übereinstimmen.
  - **Spec.ApplicationRef:** (*required*) dieser Wert muss mit dem Feld `metadata.name` der Quellanwendung CR übereinstimmen.
  - **Spec.backupRetention:** (*required*) Dieses Feld ist erforderlich und der Wert muss auf 0 gesetzt werden.
  - **Spec.enabled:** Muss auf `true` gesetzt werden.
  - **spec.granularity:** muss auf eingestellt sein `Custom`.
  - **Spec.recurrenceRule:** Definieren Sie ein Startdatum in UTC-Zeit und ein Wiederholungsintervall.
  - **Spec.snapshotRetention:** Muss auf 2 gesetzt werden.

Beispiel YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  name: appmirror-schedule-0e1f88ab-f013-4bce-8ae9-6afed9df59a1
  namespace: my-app-namespace
spec:
  appVaultRef: generic-s3-trident-protect-src-bucket-04b6b4ec-46a3-420a-b351-45795e1b5e34
  applicationRef: my-app-name
  backupRetention: "0"
  enabled: true
  granularity: custom
  recurrenceRule: |-
    DTSTART:20220101T000200Z
    RRULE:FREQ=MINUTELY;INTERVAL=5
  snapshotRetention: "2"
```

- i. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt `trident-protect-schedule.yaml` haben, wenden Sie den CR an:



```
kubectl apply -f trident-protect-schedule.yaml -n my-app-namespace
```

### Erstellen Sie einen Snapshot mit der CLI

- a. Erstellen Sie den Snapshot, und ersetzen Sie Werte in Klammern durch Informationen aus Ihrer Umgebung. Beispiel:

```
tridentctl-protect create snapshot <my_snapshot_name> --appvault  
<my_appvault_name> --app <name_of_app_to_snapshot> -n  
<application_namespace>
```

4. Erstellen Sie eine Quellenanwendung AppVault CR auf dem Zielcluster, die mit dem AppVault CR identisch ist, den Sie auf das Quellcluster angewendet haben, und benennen Sie es (z. B. `trident-protect-appvault-primary-destination.yaml`).

5. CR anwenden:

```
kubectl apply -f trident-protect-appvault-primary-destination.yaml -n  
my-app-namespace
```

6. Erstellen Sie einen AppVault für die Zielanwendung auf dem Zielcluster. Ändern Sie abhängig von Ihrem Storage-Anbieter ein Beispiel in "[Benutzerdefinierte Ressourcen von AppVault](#)" entsprechend Ihrer Umgebung:

- a. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie (z. B. `trident-protect-appvault-secondary-destination.yaml`).

- b. Konfigurieren Sie die folgenden Attribute:

- **metadata.name:** (*required*) der Name der benutzerdefinierten AppVault-Ressource. Notieren Sie sich den ausgewählten Namen, da sich andere CR-Dateien, die für eine Replikationsbeziehung benötigt werden, auf diesen Wert beziehen.
- **spec.providerConfig:** (*required*) speichert die Konfiguration, die für den Zugriff auf den AppVault unter Verwendung des angegebenen Providers erforderlich ist. Wählen Sie eine `bucketName` und alle anderen notwendigen Details für Ihren Provider. Notieren Sie sich die ausgewählten Werte, da sich andere CR-Dateien, die für eine Replikationsbeziehung benötigt werden, auf diese Werte beziehen. Beispiele für AppVault CRS mit anderen Anbietern finden Sie unter "[Benutzerdefinierte Ressourcen von AppVault](#)".
- **spec.providerCredentials:** (*required*) speichert Verweise auf alle Anmeldeinformationen, die für den Zugriff auf AppVault unter Verwendung des angegebenen Anbieters erforderlich sind.
  - **spec.providerCredentials.valueFromSecret:** (*required*) gibt an, dass der Wert der Zugangsdaten von einem Secret stammen soll.
    - **Key:** (*required*) der gültige Schlüssel des zu wählenden Geheimnisses.
    - **Name:** (*required*) Name des Geheimnisses, das den Wert für dieses Feld enthält. Muss sich im gleichen Namespace befinden.

- **spec.providerCredentials.secretAccessKey**: (*required*) der Zugriffsschlüssel, der für den Zugriff auf den Provider verwendet wird. Der **Name** sollte mit **spec.providerCredentials.valueFromSecret.name** übereinstimmen.
  - **spec.providerType**: (*required*) legt fest, was das Backup bereitstellt, z. B. NetApp ONTAP S3, generisches S3, Google Cloud oder Microsoft Azure. Mögliche Werte:
    - AWS
    - Azure
    - GCP
    - Generisch-s3
    - ONTAP-s3
    - StorageGRID-s3
- c. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt `trident-protect-appvault-secondary-destination.yaml` haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-appvault-secondary-destination.yaml
-n my-app-namespace
```

7. Erstellen Sie eine AppMirrorRelationship CR-Datei:

## Erstellen Sie eine AppMirrorRelationship mithilfe eines CR

- a. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie (z. B. `trident-protect-relationship.yaml`).
- b. Konfigurieren Sie die folgenden Attribute:
  - **metadata.name:** (erforderlich) der Name der benutzerdefinierten AppMirrorRelationship-Ressource.
  - **spec.destinationAppVaultRef:** (*required*) dieser Wert muss mit dem Namen des AppVault für die Zielanwendung auf dem Zielcluster übereinstimmen.
  - **spec.namespaceMapping:** (*required*) der Ziel- und Quellnamespaces muss mit dem im jeweiligen Anwendungs-CR definierten AnwendungsNamespace übereinstimmen.
  - **Spec.sourceAppVaultRef:** (*required*) dieser Wert muss mit dem Namen des AppVault für die Quellanwendung übereinstimmen.
  - **Spec.sourceApplicationName:** (*required*) dieser Wert muss mit dem Namen der Quellanwendung übereinstimmen, die Sie in der Quellanwendung CR definiert haben.
  - **Spec.storageClassName:** (*required*) Wählen Sie den Namen einer gültigen Storage-Klasse auf dem Cluster. Die Storage-Klasse muss mit einer ONTAP Storage-VM verknüpft werden, die mit der Quellumgebung peert wird.
  - **Spec.recurrenceRule:** Definieren Sie ein Startdatum in UTC-Zeit und ein Wiederholungsintervall.

Beispiel YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: AppMirrorRelationship
metadata:
  name: amr-16061e80-1b05-4e80-9d26-d326dc1953d8
  namespace: my-app-namespace
spec:
  desiredState: Established
  destinationAppVaultRef: generic-s3-trident-protect-dst-bucket-
8fe0b902-f369-4317-93d1-ad7f2edc02b5
  namespaceMapping:
    - destination: my-app-namespace
      source: my-app-namespace
  recurrenceRule: |-
    DTSTART:20220101T000200Z
    RRULE:FREQ=MINUTELY;INTERVAL=5
  sourceAppVaultRef: generic-s3-trident-protect-src-bucket-
b643cc50-0429-4ad5-971f-ac4a83621922
  sourceApplicationName: my-app-name
  sourceApplicationUID: 7498d32c-328e-4ddd-9029-122540866aeb
  storageClassName: sc-vsimg-2
```

- c. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt `trident-protect-relationship.yaml` haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-namespace
```

### Erstellen Sie eine AppMirrorRelationship mithilfe der CLI

- a. Erstellen und wenden Sie das AppMirrorRelationship-Objekt an, und ersetzen Sie Werte in Klammern durch Informationen aus Ihrer Umgebung. Beispiel:

```
tridentctl-protect create appmirrorrelationship  
<name_of_appmirrorrelationship> --destination-app-vault  
<my_vault_name> --recurrence-rule <rule> --source-app  
<my_source_app> --source-app-vault <my_source_app_vault> -n  
<application_namespace>
```

8. (Optional) Status und Status der Replikationsbeziehung prüfen:

```
kubectl get amr -n my-app-namespace <relationship name> -o=jsonpath  
='{.status}' | jq
```

### Failover zum Ziel-Cluster

Mit Trident Protect können Sie ein Failover replizierter Applikationen auf ein Ziel-Cluster durchführen. Durch dieses Verfahren wird die Replikationsbeziehung angehalten und die App wird auf dem Ziel-Cluster online geschaltet. Trident Protect stoppt die App auf dem Quell-Cluster nicht, wenn sie betriebsbereit war.

### Schritte

1. Öffnen Sie die AppMirrorRelationship CR-Datei (z.B. `trident-protect-relationship.yaml`) und ändern Sie den Wert von **spec.desiredState** in `Promoted`.
2. Speichern Sie die CR-Datei.
3. CR anwenden:

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-namespace
```

4. (Optional) Erstellen Sie alle Schutzzeitpläne, die Sie für die Failover-Anwendung benötigen.
5. (Optional) Status und Status der Replikationsbeziehung prüfen:

```
kubectl get amr -n my-app-namespace <relationship name> -o=jsonpath  
='{.status}' | jq
```

## Neusynchronisierung einer fehlgeschlagenen Replikationsbeziehung

Durch den Neusynchronisierung wird die Replikationsbeziehung wiederhergestellt. Nach einer Neusynchronisierung wird die ursprüngliche Quellanwendung zur laufenden Anwendung, und alle Änderungen, die an der laufenden Anwendung auf dem Zielcluster vorgenommen werden, werden verworfen.

Der Prozess stoppt die App auf dem Zielcluster, bevor die Replikation wiederhergestellt wird.



Alle Daten, die während des Failovers auf die Zielapplikation geschrieben werden, gehen verloren.

### Schritte

1. Erstellen Sie einen Snapshot der Quellanwendung.
2. Öffnen Sie die AppMirrorRelationship CR-Datei (z. B. `trident-protect-relationship.yaml`) und ändern Sie den Wert von `spec.desiredState` in `Established`.
3. Speichern Sie die CR-Datei.
4. CR anwenden:

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-namespace
```

5. Wenn Sie Schutzzeitpläne auf dem Zielcluster erstellt haben, um die Failover-Anwendung zu schützen, entfernen Sie sie. Alle Zeitpläne, die weiterhin zu Fehlern bei Volume-Snapshots führen.

## Reverse Resynchronisierung einer Failover-Replizierungsbeziehung

Wenn Sie eine Failover-Replikationsbeziehung umkehren, wird die Zielanwendung zur Quellanwendung, und die Quelle wird zum Ziel. Änderungen, die während des Failovers an der Zielapplikation vorgenommen werden, werden beibehalten.

### Schritte

1. Löschen Sie die AppMirrorRelationship-CR auf dem ursprünglichen Ziel-Cluster. Dadurch wird das Ziel zur Quelle. Wenn auf dem neuen Ziel-Cluster noch Schutzpläne vorhanden sind, entfernen Sie sie.
2. Richten Sie eine Replikationsbeziehung ein, indem Sie die CR-Dateien anwenden, die Sie ursprünglich zum Einrichten der Beziehung zu den anderen Clustern verwendet haben.
3. Stellen Sie sicher, dass die AppVault CRS auf jedem Cluster bereit sind.
4. Richten Sie eine Replikationsbeziehung auf dem anderen Cluster ein, und konfigurieren Sie Werte für die umgekehrte Richtung.

## Richtung der Anwendungsreplikation umkehren

Wenn Sie die Replizierungsrichtung umkehren, verschiebt Trident Protect die Applikation auf das Ziel-Storage-Back-End, während die Replikation zurück zum ursprünglichen Quell-Storage-Back-End fortgesetzt wird. Trident Protect stoppt die Quellapplikation und repliziert die Daten zum Ziel, bevor ein Failover zur Zielapplikation erfolgt.

In dieser Situation tauschen Sie Quelle und Ziel aus.

### Schritte

1. Erstellen Sie einen Snapshot zum Herunterfahren:

## Erstellen Sie einen Snapshot zum Herunterfahren mit einem CR

- a. Deaktivieren Sie die Schutzrichtlinienpläne für die Quellenanwendung.
- b. Erstellen Sie eine CR-Datei für den ShutdownSnapshot:
  - i. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie (z. B. `trident-protect-shutdownsnapshot.yaml`).
  - ii. Konfigurieren Sie die folgenden Attribute:
    - **metadata.name:** (*required*) der Name der benutzerdefinierten Ressource.
    - **Spec.AppVaultRef:** (*required*) dieser Wert muss mit dem Feld `metadata.name` des AppVault für die Quellenanwendung übereinstimmen.
    - **Spec.ApplicationRef:** (*required*) dieser Wert muss mit dem Feld `metadata.name` der CR-Datei der Quellenanwendung übereinstimmen.

Beispiel YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: ShutdownSnapshot
metadata:
  name: replication-shutdown-snapshot-afc4c564-e700-4b72-86c3-
c08a5dbe844e
  namespace: my-app-namespace
spec:
  appVaultRef: generic-s3-trident-protect-src-bucket-04b6b4ec-
46a3-420a-b351-45795e1b5e34
  applicationRef: my-app-name
```

- c. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt `trident-protect-shutdownsnapshot.yaml` haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-shutdownsnapshot.yaml -n my-app-namespace
```

## Erstellen Sie einen Snapshot zum Herunterfahren über die CLI

- a. Erstellen Sie den Snapshot zum Herunterfahren, und ersetzen Sie Werte in Klammern durch Informationen aus Ihrer Umgebung. Beispiel:

```
tridentctl-protect create shutdownsnapshot <my_shutdown_snapshot>
--appvault <my_vault> --app <app_to_snapshot> -n
<application_namespace>
```

2. Nachdem der Snapshot abgeschlossen ist, rufen Sie den Status des Snapshots ab:

```
kubectl get shutdownsnapshot -n my-app-namespace  
<shutdown_snapshot_name> -o yaml
```

3. Suchen Sie den Wert von **shutdownSnapshot.Status.appArchivePath** mit dem folgenden Befehl und notieren Sie den letzten Teil des Dateipfads (auch Basisname genannt; dies ist alles nach dem letzten Schrägstrich):

```
k get shutdownsnapshot -n my-app-namespace <shutdown_snapshot_name> -o  
jsonpath='{.status.appArchivePath}'
```

4. Führen Sie mit der folgenden Änderung ein Failover vom Ziel-Cluster zum Quell-Cluster durch:



Fügen Sie in Schritt 2 des Failover-Verfahrens das Feld in die AppMirrorRelationship CR-Datei ein `spec.promotedSnapshot`, und setzen Sie den Wert auf den Basisnamen, den Sie oben in Schritt 3 aufgezeichnet haben.

5. Führen Sie die Schritte für die umgekehrte Resynchronisierung in [Reverse Resynchronisierung einer Failover-Replizierungsbeziehung](#) aus.

6. Aktivieren Sie Schutzzeitpläne auf dem neuen Quellcluster.

## Ergebnis

Die folgenden Aktionen werden aufgrund der umgekehrten Replikation durchgeführt:

- Von den Kubernetes-Ressourcen der ursprünglichen Quell-Applikation wird ein Snapshot erstellt.
- Die PODs der ursprünglichen Quell-App werden mit sanfter Weise gestoppt, indem die Kubernetes-Ressourcen der App gelöscht werden (wodurch PVCs und PVS aktiviert bleiben).
- Nach dem Herunterfahren der Pods werden Snapshots der Volumes der App erstellt und repliziert.
- Die SnapMirror Beziehungen sind beschädigt, wodurch die Zieldatenträger für Lese-/Schreibvorgänge bereit sind.
- Die Kubernetes-Ressourcen der App werden aus dem Snapshot vor dem Herunterfahren wiederhergestellt. Dabei werden die Volume-Daten verwendet, die nach dem Herunterfahren der ursprünglichen Quell-App repliziert wurden.
- Die Replizierung wird in umgekehrter Richtung wieder hergestellt.

## Führen Sie ein Failback von Anwendungen auf das ursprüngliche Quellcluster durch

Mithilfe von Trident Protect können Sie nach einem Failover-Vorgang ein Failback durchführen, indem Sie die folgende Sequenz von Vorgängen verwenden. In diesem Workflow zur Wiederherstellung der ursprünglichen Replikationsrichtung repliziert Trident Protect alle Anwendungsänderungen zurück zur ursprünglichen Quellanwendung, bevor die Replikationsrichtung rückgängig gemacht wird.

Dieser Prozess beginnt mit einer Beziehung, bei der ein Failover zu einem Ziel durchgeführt wurde, und umfasst die folgenden Schritte:

- Starten Sie mit einem Failover-Status fehlgeschlagen.
- Umgekehrte Neusynchronisierung der Replikationsbeziehung.



Führen Sie keine normale Neusynchronisierung durch, da dadurch während des Failover Daten, die auf das Ziel-Cluster geschrieben werden, verworfen werden.

- Kehren Sie die Replikationsrichtung um.

### Schritte

1. Führen Sie die [Reverse Resynchronisierung einer Failover-Replizierungsbeziehung](#) Schritte aus.
2. Führen Sie die [Richtung der Anwendungsreplikation umkehren](#) Schritte aus.

### Löschen einer Replikationsbeziehung

Sie können eine Replikationsbeziehung jederzeit löschen. Wenn Sie die Anwendungsreplikationsbeziehung löschen, führt dies zu zwei separaten Anwendungen ohne Beziehung zwischen ihnen.

### Schritte

1. Löschen Sie die AppMirrorRelationship-CR:

```
kubectl delete -f trident-protect-relationship.yaml -n my-app-namespace
```

## Migrieren Sie Applikationen mit Trident Protect

Sie können Ihre Applikationen zwischen Clustern oder Storage-Klassen migrieren, indem Sie Ihre Backup- oder Snapshot-Daten in einem anderen Cluster oder einer anderen Storage-Klasse wiederherstellen.



Wenn Sie eine Anwendung migrieren, werden alle für die Anwendung konfigurierten Ausführungshaken mit der App migriert. Wenn ein Hook für die Ausführung nach der Wiederherstellung vorhanden ist, wird er automatisch als Teil des Wiederherstellungsvorgangs ausgeführt.

### Backup- und Restore-Vorgänge

Um Backup- und Wiederherstellungsvorgänge für die folgenden Szenarien durchzuführen, können Sie bestimmte Backup- und Wiederherstellungsaufgaben automatisieren.

#### Klonen auf dasselbe Cluster

Um eine Anwendung auf demselben Cluster zu klonen, erstellen Sie einen Snapshot oder ein Backup, und stellen Sie die Daten im selben Cluster wieder her.

### Schritte

1. Führen Sie einen der folgenden Schritte aus:
  - a. ["Erstellen Sie einen Snapshot"](#).
  - b. ["Erstellen Sie ein Backup"](#).



2. Führen Sie auf demselben Cluster je nach Erstellung eines Snapshots oder eines Backups einen der folgenden Schritte aus:
  - a. "Stellen Sie Ihre Daten aus dem Snapshot wieder her".
  - b. "Stellen Sie Ihre Daten aus dem Backup wieder her".

### Klonen auf anderes Cluster

Um eine Anwendung auf einem anderen Cluster zu klonen (einen Cluster-übergreifenden Klon durchführen), erstellen Sie ein Backup auf dem Quellcluster, und stellen Sie das Backup dann auf einem anderen Cluster wieder her. Stellen Sie sicher, dass Trident Protect auf dem Ziel-Cluster installiert ist.



Sie können eine Anwendung zwischen verschiedenen Clustern mit replizieren "[SnapMirror Replizierung](#)".

### Schritte

1. "Erstellen Sie ein Backup".
2. Stellen Sie sicher, dass der AppVault CR für den Objektspeicher-Bucket, der das Backup enthält, auf dem Zielcluster konfiguriert wurde.
3. Auf dem Zielcluster, "Stellen Sie Ihre Daten aus dem Backup wieder her".

### Migrieren von Applikationen von einer Storage-Klasse zu einer anderen Storage-Klasse

Sie können Anwendungen von einer Storage-Klasse zu einer anderen Storage-Klasse migrieren, indem Sie einen Snapshot auf der anderen Ziel-Storage-Klasse wiederherstellen.

Beispiel: (Ohne die Geheimnisse aus dem Wiederherstellungs-CR):

```
apiVersion: protect.trident.netapp.io/v1
kind: SnapshotRestore
metadata:
  name: "${snapshotRestoreCRName}"
spec:
  appArchivePath: "${snapshotArchivePath}"
  appVaultRef: "${appVaultCRName}"
  namespaceMapping:
    destination: "${destinationNamespace}"
    source: "${sourceNamespace}"
  storageClassMapping:
    destination: "${destinationStorageClass}"
    source: "${sourceStorageClass}"
  resourceFilter:
    resourceMatchers:
      kind: Secret
      version: v1
    resourceSelectionCriteria: exclude
```

## Stellen Sie den Snapshot mithilfe eines CR-Systems wieder her

### Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie `trident-protect-snapshot-restore-cr.yaml`.
2. Konfigurieren Sie in der erstellten Datei die folgenden Attribute:
  - **metadata.name:** (*required*) der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und sinnvollen Namen für Ihre Umgebung.
  - **Spec.appArchivePath:** Der Pfad innerhalb von AppVault, wo der Snapshot-Inhalt gespeichert wird. Sie können den folgenden Befehl verwenden, um diesen Pfad zu finden:

```
kubectl get snapshots <my-snapshot-name> -n trident-protect -o jsonpath='{.status.appArchivePath}'
```

- **Spec.appVaultRef:** (*required*) der Name des AppVault, in dem der Snapshot-Inhalt gespeichert ist.
- **spec.namespaceMapping:** die Zuordnung des Quell-Namespace des Wiederherstellungsvorgangs zum Ziel-Namespace. Ersetzen `my-source-namespace` Sie und `my-destination-namespace` mit Informationen aus Ihrer Umgebung.

```
---
apiVersion: protect.trident.netapp.io/v1
kind: SnapshotRestore
metadata:
  name: my-cr-name
  namespace: trident-protect
spec:
  appArchivePath: my-snapshot-path
  appVaultRef: appvault-name
  namespaceMapping: [{"source": "my-source-namespace",
"destination": "my-destination-namespace"}]
```

3. Wenn Sie optional nur bestimmte Ressourcen der wiederherzustellenden Anwendung auswählen müssen, fügen Sie eine Filterung hinzu, die Ressourcen mit bestimmten Bezeichnungen enthält oder ausschließt:
  - **ResourceFilter.resourceSelectionCriteria:** (Erforderlich für die Filterung) Verwenden Sie `include` or `exclude`, um eine in `resourceMatchers` definierte Ressource ein- oder auszuschließen. Fügen Sie die folgenden `resourceMatchers`-Parameter hinzu, um die einzuschließenden oder auszuschließenden Ressourcen zu definieren:
    - **RefindeFilter.refindeMatchers:** Eine Reihe von `refindeMatcher`-Objekten. Wenn Sie mehrere Elemente in diesem Array definieren, stimmen sie als OP-Operation überein, und die Felder innerhalb jedes Elements (Gruppe, Typ, Version) stimmen mit einer UND-Operation überein.
      - **ResourceMatchers[].Group:** (*Optional*) Gruppe der zu filternden Ressource.
      - **ResourceMatchers[].Kind:** (*Optional*) Art der zu filternden Ressource.

- **ResourceMatchers[].Version:** (*Optional*) Version der zu filternden Ressource.
- **ResourceMatchers[].Namen:** (*Optional*) Namen im Kubernetes metadata.name-Feld der zu filternden Ressource.
- **ResourceMatchers[].Namespaces:** (*Optional*) Namespaces im Kubernetes metadata.name-Feld der zu filternden Ressource.
- **ResourceMatchers[].labelSelectors:** (*Optional*) Label selector string im Feld Kubernetes metadata.name der Ressource, wie im definiert "[Kubernetes-Dokumentation](#)". Zum Beispiel: "trident.netapp.io/os=linux".

Beispiel:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Nachdem Sie die Datei mit den richtigen Werten ausgefüllt trident-protect-snapshot-restore-cr.yaml haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-snapshot-restore-cr.yaml
```

## Stellen Sie den Snapshot mithilfe der CLI wieder her

### Schritte

1. Stellen Sie den Snapshot in einem anderen Namespace wieder her und ersetzen Sie Werte in Klammern durch Informationen aus Ihrer Umgebung.
  - Das snapshot Argument verwendet einen Namespace und Snapshot-Namen im Format <namespace>/<name>.
  - Das namespace-mapping Argument verwendet durch Doppelpunkte getrennte Namespaces, um Quellnamespaces im Format den richtigen Zielnamepaces zuzuordnen source1:dest1, source2:dest2.

Beispiel:

```
tridentctl-protect create snapshotrestore <my_restore_name>
--snapshot <namespace/snapshot_to_restore> --namespace-mapping
<source_to_destination_namespace_mapping>
```

## Managen von Trident Protect-Ausführungshaken

Ein Execution Hook ist eine benutzerdefinierte Aktion, die Sie so konfigurieren können, dass sie zusammen mit einem Datenschutzvorgang einer verwalteten App ausgeführt wird. Wenn Sie beispielsweise über eine Datenbank-App verfügen, können Sie mit einem Execution-Hook alle Datenbanktransaktionen vor einem Snapshot anhalten und die Transaktionen nach Abschluss des Snapshots wieder aufnehmen. Dies gewährleistet applikationskonsistente Snapshots.

### Arten von Ausführungshaken

Trident Protect unterstützt die folgenden Typen von Ausführungshaken, je nachdem, wann sie ausgeführt werden können:

- Vor dem Snapshot
- Nach dem Snapshot
- Vor dem Backup
- Nach dem Backup
- Nach dem Wiederherstellen
- Nach Failover

### Ausführungsreihenfolge

Wenn ein Datenschutzvorgang ausgeführt wird, finden Hakenereignisse in der folgenden Reihenfolge statt:

1. Alle entsprechenden benutzerdefinierten Testhaken für die Ausführung vor dem Betrieb werden auf den entsprechenden Containern ausgeführt. Sie können beliebig viele benutzerdefinierte Hooks für die Vorbedienung erstellen und ausführen, aber die Reihenfolge der Ausführung dieser Haken vor der Operation ist weder garantiert noch konfigurierbar.
2. Das Dateisystem hängt sich ggf. ein. ["Erfahren Sie mehr über das Konfigurieren des Dateisystemfrierens mit Trident Protect"](#).
3. Der Vorgang der Datensicherung wird durchgeführt.
4. Eingefrorene Dateisysteme werden gegebenenfalls nicht eingefroren.
5. Alle entsprechenden benutzerdefinierten Testhaken für die Ausführung nach der Operation werden auf den entsprechenden Containern ausgeführt. Sie können beliebig viele benutzerdefinierte Haken für die Nachbearbeitung erstellen und ausführen, aber die Reihenfolge der Ausführung dieser Haken nach der Operation ist weder garantiert noch konfigurierbar.

Wenn Sie mehrere Testausführungshaken desselben Typs erstellen (z. B. Pre-Snapshot), ist die Reihenfolge der Ausführung dieser Haken nicht garantiert. Die Reihenfolge der Ausführung von Haken unterschiedlicher Art ist jedoch garantiert. Im Folgenden wird beispielsweise die Reihenfolge der Ausführung einer Konfiguration

beschrieben, die alle verschiedenen Hooks umfasst:

1. Hooks vor dem Snapshot wurden ausgeführt
2. Hooks nach dem Snapshot wurden ausgeführt
3. Hooks vor dem Backup wurden ausgeführt
4. Hooks nach dem Backup ausgeführt



Das Beispiel der vorherigen Reihenfolge gilt nur, wenn Sie ein Backup ausführen, das keinen vorhandenen Snapshot verwendet.



Sie sollten Ihre Hook-Skripte immer testen, bevor Sie sie in einer Produktionsumgebung aktivieren. Mit dem Befehl 'kubectrl exec' können Sie die Skripte bequem testen. Nachdem Sie die Testausführungshaken in einer Produktionsumgebung aktiviert haben, testen Sie die erstellten Snapshots und Backups, um sicherzustellen, dass sie konsistent sind. Dazu klonen Sie die Applikation in einem temporären Namespace, stellen den Snapshot oder das Backup wieder her und testen anschließend die App.

## Wichtige Hinweise zu benutzerdefinierten Testausführungshaken

Bei der Planung von Testausführungshooks für Ihre Apps sollten Sie Folgendes berücksichtigen:

- Ein Testsuite muss ein Skript verwenden, um Aktionen durchzuführen. Viele Testsuitehooks können auf dasselbe Skript verweisen.
- Trident Protect erfordert, dass die Skripte, die Ausführungshaken verwenden, im Format ausführbarer Shell-Skripte geschrieben werden.
- Die Skriptgröße ist auf 96 KB begrenzt.
- Trident Protect verwendet Execution Hook-Einstellungen und alle übereinstimmenden Kriterien, um zu ermitteln, welche Hooks für einen Snapshot-, Backup- oder Wiederherstellungsvorgang gelten.



Da Testsuitehaken die Funktionalität der Anwendung, für die sie ausgeführt werden, oft reduzieren oder vollständig deaktivieren, sollten Sie immer versuchen, die Zeit zu minimieren, die Ihre benutzerdefinierten Testausführungshaken für die Ausführung benötigt. Wenn Sie eine Backup- oder Snapshot-Operation mit zugeordneten Testsuiten starten, diese aber dann abbrechen, können die Haken trotzdem ausgeführt werden, wenn der Backup- oder Snapshot-Vorgang bereits gestartet wurde. Das bedeutet, dass die in einem Testsuite nach dem Backup verwendete Logik nicht davon ausgehen kann, dass das Backup abgeschlossen wurde.

## Filter für Testausführungshaken

Wenn Sie einen Ausführungshaken für eine Anwendung hinzufügen oder bearbeiten, können Sie dem Ausführungshaken Filter hinzufügen, um zu verwalten, welche Container der Hook entsprechen soll. Filter sind für Applikationen nützlich, die in allen Containern dasselbe Container-Image nutzen. Jedes Image kann jedoch für einen anderen Zweck (wie Elasticsearch) verwendet werden. Mit Filtern können Sie Szenarien erstellen, in denen Ausführungshaken auf einigen, aber nicht unbedingt allen identischen Containern ausgeführt werden. Wenn Sie mehrere Filter für einen einzelnen Testausführungshaken erstellen, werden diese mit einem logischen UND einem Operator kombiniert. Pro Testsuite können Sie bis zu 10 aktive Filter haben.

Jeder Filter, den Sie einem Execution Hook hinzufügen, verwendet einen regulären Ausdruck, um Container in Ihrem Cluster zu entsprechen. Wenn ein Haken einem Container entspricht, führt der Haken sein zugehöriges Skript auf diesem Container aus. Reguläre Ausdrücke für Filter verwenden die Syntax des regulären

Ausdrucks 2 (RE2), die das Erstellen eines Filters nicht unterstützt, der Container aus der Liste der Übereinstimmungen ausschließt. Informationen zur Syntax, die Trident Protect für reguläre Ausdrücke in Ausführungshook-Filtern unterstützt, finden Sie unter "[Syntaxunterstützung für regulären Ausdruck 2 \(RE2\)](#)".



Wenn Sie einem Ausführungs-Hook einen Namespace-Filter hinzufügen, der nach einer Wiederherstellung oder einem Klonvorgang ausgeführt wird, und die Wiederherstellungs- oder Klonquelle und das Ziel in verschiedenen Namespaces liegen, wird der Namespace-Filter nur auf den Ziel-Namespace angewendet.

### **Beispiele für Testausführungshaken**

Besuchen Sie die "[NetApp Verda GitHub Projekt](#)", um echte Ausführungshaken für gängige Apps wie Apache Cassandra und Elasticsearch herunterzuladen. Sie können auch Beispiele sehen und Ideen für die Strukturierung Ihrer eigenen benutzerdefinierten Execution Hooks erhalten.

### **Erstellen Sie einen Ausführungshaken**

Mit Trident Protect können Sie einen benutzerdefinierten Ausführungshaken für eine App erstellen. Sie müssen über die Berechtigungen Eigentümer, Administrator oder Mitglied verfügen, um Testausführungshaken zu erstellen.

## CR verwenden

### Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR) und benennen Sie sie `trident-protect-hook.yaml`.
2. Konfigurieren Sie die folgenden Attribute entsprechend Ihrer Trident Protect-Umgebung und Cluster-Konfiguration:
  - **metadata.name:** (*required*) der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und sinnvollen Namen für Ihre Umgebung.
  - **Spec.applicationRef:** (*required*) der Kubernetes-Name der Anwendung, für die der Ausführungshaken ausgeführt werden soll.
  - **Spec.Stage:** (*required*) Eine Zeichenfolge, die angibt, welche Phase während der Aktion der Ausführungshaken ausgeführt werden soll. Mögliche Werte:
    - Vor
    - Post
  - **Spec.Action:** (*required*) Eine Zeichenfolge, die angibt, welche Aktion der Ausführungshaken ausführen wird, vorausgesetzt, dass alle angegebenen Ausführungshaken-Filter übereinstimmen. Mögliche Werte:
    - Snapshot
    - Backup
    - Wiederherstellen
    - Failover
  - **Spec.enabled:** (*Optional*) gibt an, ob dieser Ausführungshaken aktiviert oder deaktiviert ist. Wenn nicht angegeben, ist der Standardwert TRUE.
  - **Spec.hookSource:** (*required*) Ein String, der das base64-kodierte Hook-Skript enthält.
  - **Spec.timeout:** (*Optional*) Eine Zahl, die definiert, wie lange der Ausführungshaken in Minuten ausgeführt werden darf. Der Mindestwert beträgt 1 Minute, und der Standardwert ist 25 Minuten, wenn nicht angegeben.
  - **Spec.Arguments:** (*Optional*) Eine YAML-Liste von Argumenten, die Sie für den Ausführungshaken angeben können.
  - **Spec.matchingCriteria:** (*Optional*) eine optionale Liste von Kriterien-Schlüsselwertpaaren, jedes Paar, das einen Ausführungshook-Filter bildet. Sie können bis zu 10 Filter pro Ausführungshaken hinzufügen.
  - **Spec.matchingCriteria.type:** (*Optional*) Eine Zeichenfolge, die den Filtertyp für den Ausführungshaken identifiziert. Mögliche Werte:
    - ContainerImage
    - Containername
    - PodName
    - PodLabel
    - NamespaceName
  - **Spec.matchingCriteria.value:** (*Optional*) Ein String oder regulärer Ausdruck, der den Wert des Ausführungshook-Filters identifiziert.

## Beispiel YAML:

```
apiVersion: protect.trident.netapp.io/v1
kind: ExecHook
metadata:
  name: example-hook-cr
  namespace: my-app-namespace
  annotations:
    astra.netapp.io/astra-control-hook-source-id:
/account/test/hookSource/id
spec:
  applicationRef: my-app-name
  stage: Pre
  action: Snapshot
  enabled: true
  hookSource: IyEvYmluL2Jhc2gKZWNoYAiZXhhbXBsZSBzY3JpcHQiCg==
  timeout: 10
  arguments:
    - FirstExampleArg
    - SecondExampleArg
  matchingCriteria:
    - type: containerName
      value: mysql
    - type: containerImage
      value: bitnami/mysql
    - type: podName
      value: mysql
    - type: namespaceName
      value: mysql-a
    - type: podLabel
      value: app.kubernetes.io/component=primary
    - type: podLabel
      value: helm.sh/chart=mysql-10.1.0
    - type: podLabel
      value: deployment-type=production
```

3. Nachdem Sie die CR-Datei mit den richtigen Werten ausgefüllt haben, wenden Sie den CR an:

```
kubectl apply -f trident-protect-hook.yaml
```

## Verwenden Sie die CLI

### Schritte

1. Erstellen Sie den Ausführungshaken, und ersetzen Sie Werte in Klammern durch Informationen aus Ihrer Umgebung. Beispiel:



```
tridentctl-protect create exehook <my_exec_hook_name> --action  
<action_type> --app <app_to_use_hook> --stage <pre_or_post_stage>  
--source-file <script-file> -n <application_namespace>
```

## Deinstallieren Sie Trident Protect

Möglicherweise müssen Sie Trident Protect-Komponenten entfernen, wenn Sie ein Upgrade von einer Testversion auf eine Vollversion des Produkts durchführen.

So entfernen Sie Trident Protect:

### Schritte

1. Entfernen Sie die Trident Protect CR-Dateien:

```
helm uninstall -n trident-protect trident-protect-crds
```

2. Trident Protect entfernen:

```
helm uninstall -n trident-protect trident-protect
```

3. Entfernen Sie den Trident Protect Namespace:

```
kubectl delete ns trident-protect
```

# Wissen und Support

## Häufig gestellte Fragen

Hier finden Sie Antworten auf die häufig gestellten Fragen zur Installation, Konfiguration, Aktualisierung und Fehlerbehebung von Trident.

### Allgemeine Fragen

#### Wie oft wird Trident veröffentlicht?

Ab Version 24.02 wird Trident alle vier Monate veröffentlicht: Februar, Juni und Oktober.

#### Unterstützt Trident alle Funktionen, die in einer bestimmten Version von Kubernetes verfügbar sind?

Trident unterstützt in der Regel keine Alpha-Funktionen in Kubernetes. Trident unterstützt möglicherweise Beta-Funktionen in den beiden Trident Versionen, die nach der Kubernetes Beta-Version folgen.

#### Hat Trident irgendwelche Abhängigkeiten von anderen NetApp Produkten für seine Funktion?

Trident hat keine Abhängigkeiten von anderen NetApp Software-Produkten und funktioniert als Standalone-Applikation. Sie sollten jedoch ein NetApp Back-End Storage-Gerät haben.

#### Wie erhalte ich vollständige Trident Konfigurationsdetails?

Verwenden Sie den `tridentctl get` Befehl, um weitere Informationen zur Trident Konfiguration zu erhalten.

#### Kann ich Metriken abrufen, wie Storage von Trident bereitgestellt wird?

Ja. Prometheus Endpunkte, die zur Erfassung von Informationen über den Trident-Vorgang verwendet werden können, z. B. die Anzahl der gemanagten Back-Ends, die Anzahl der bereitgestellten Volumes, die verbrauchten Bytes usw. Sie können auch für Monitoring und Analysen verwenden "[Einblicke in die Cloud](#)".

#### Ändert sich die Benutzererfahrung bei der Verwendung von Trident als CSI-Provisionierung?

Nein. Es gibt keine Änderungen hinsichtlich der Benutzererfahrung und Funktionalitäten. Der verwendete bereitstellungsname ist `csi.trident.netapp.io`. Diese Methode zur Installation von Trident wird empfohlen, wenn Sie alle neuen Funktionen der aktuellen und zukünftigen Versionen verwenden möchten.

## Installation und Verwendung von Trident auf einem Kubernetes-Cluster

#### Unterstützt Trident eine Offline-Installation aus einer privaten Registrierung?

Ja, Trident kann offline installiert werden. Siehe "[Erfahren Sie mehr über die Trident Installation](#)".

#### Kann ich Trident Remote installieren?

Ja. Trident 18.10 und höher unterstützen Remote-Installationsfunktionen von jedem Computer aus, der Zugriff auf das Cluster hat `kubectl`. Nachdem `kubectl` der Zugriff überprüft wurde (z. B. Starten Sie einen `kubectl get nodes` Befehl vom Remote-Computer zur Überprüfung), befolgen Sie die Installationsanweisungen.

## **Kann ich Hochverfügbarkeit mit Trident konfigurieren?**

Trident wird als Kubernetes Deployment (ReplicaSet) mit einer Instanz installiert und verfügt daher über integrierte HA. Sie sollten die Anzahl der Replikate in der Bereitstellung nicht erhöhen. Wenn der Node, auf dem Trident installiert ist, verloren geht oder der Pod anderweitig nicht verfügbar ist, stellt Kubernetes den Pod automatisch wieder einem funktionstüchtigen Node im Cluster bereit. Trident arbeitet nur über die Kontrollebene. Derzeit gemountete Pods sind also nicht betroffen, wenn Trident erneut implementiert wird.

## **Braucht Trident Zugang zum kube-System-Namespace?**

Trident liest vom Kubernetes-API-Server, um zu bestimmen, wann Applikationen neue PVCs anfordern. Daher ist Zugriff auf das kube-System erforderlich.

## **Welche Rollen und Privileges verwendet Trident?**

Das Trident-Installationsprogramm erstellt ein Kubernetes ClusterRole, das spezifischen Zugriff auf die Ressourcen PersistentVolume, PersistentVolumeClaim, StorageClass und Secret des Kubernetes-Clusters hat. Siehe "[Die tridentctl-Installation anpassen](#)".

## **Kann ich die genauen Manifestdateien, die Trident für die Installation verwendet, lokal generieren?**

Sie können die genauen Manifestdateien, die Trident für die Installation verwendet, bei Bedarf lokal generieren und ändern. Siehe "[Die tridentctl-Installation anpassen](#)".

## **Kann ich dieselbe ONTAP-Backend-SVM für zwei separate Trident-Instanzen für zwei separate Kubernetes-Cluster nutzen?**

Es wird zwar nicht empfohlen, aber Sie können dieselbe Back-End-SVM für zwei Trident-Instanzen verwenden. Geben Sie während der Installation einen eindeutigen Volume-Namen für jede Instanz an und/oder geben Sie einen eindeutigen `StoragePrefix` Parameter in der `setup/backend.json` Datei an. Dadurch wird sichergestellt, dass nicht dieselbe FlexVol für beide Instanzen verwendet wird.

## **Ist es möglich, Trident unter ContainerLinux (früher CoreOS) zu installieren?**

Trident ist ganz einfach ein Kubernetes-Pod und kann überall dort installiert werden, wo Kubernetes ausgeführt wird.

## **Kann ich Trident mit NetApp Cloud Volumes ONTAP verwenden?**

Ja, Trident wird auf AWS, Google Cloud und Azure unterstützt.

## **Funktioniert Trident mit Cloud Volumes Services?**

Ja, Trident unterstützt den Azure NetApp Files-Service in Azure sowie den Cloud Volumes Service in GCP.

## **Fehlerbehebung und Support**

### **Unterstützt NetApp Trident?**

Obwohl Trident Open Source ist und kostenlos zur Verfügung gestellt wird, unterstützt NetApp es vollständig, vorausgesetzt, Ihr NetApp Backend wird unterstützt.

## Wie kann ich einen Support-Fall anheben?

Wenn Sie einen Support-Case anheben möchten, führen Sie einen der folgenden Schritte aus:

1. Kontaktieren Sie Ihren Support Account Manager und erhalten Sie Hilfe bei der Ticketausstellung.
2. Wenden Sie sich an "[NetApp Support](#)".

## Wie generiere ich ein Support Log-Paket?

Sie können ein Supportpaket erstellen, indem Sie ausführen `tridentctl logs -a`. Erfassen Sie zusätzlich zu den im Bundle erfassten Protokollen das kubelet-Protokoll, um die Mount-Probleme auf der Seite von Kubernetes zu diagnostizieren. Die Anweisungen zum Abrufen des kubelet-Protokolls variieren je nach der Installation von Kubernetes.

## Was muss ich tun, wenn ich einen Antrag auf eine neue Funktion stellen muss?

Erstellen Sie ein Problem "[Trident Github](#)" und erwähnen Sie **RFE** im Betreff und der Beschreibung des Problems.

## Wo kann ich einen Defekt aufwerfen?

Erstellen Sie ein Problem am "[Trident Github](#)". Achten Sie darauf, alle erforderlichen Informationen und Protokolle für das Problem einzubeziehen.

## Was passiert, wenn ich schnelle Frage zu Trident habe, bei der ich Klarstellung brauche? Gibt es eine Gemeinschaft oder ein Forum?

Sollten Sie Fragen oder Probleme haben oder Anfragen haben, wenden Sie sich bitte über unser Trident oder GitHub an uns "[Kanal abstecken](#)".

## Das Passwort meines Storage-Systems wurde geändert und Trident funktioniert nicht mehr. Wie kann ich das Recovery durchführen?

Aktualisieren Sie das Backend-Passwort mit `tridentctl update backend myBackend -f </path/to_new_backend.json> -n trident`. Ersetzen Sie `myBackend` im Beispiel durch Ihren Backend-Namen und `/path/to_new_backend.json` den Pfad zur richtigen `backend.json` Datei.

## Trident kann meinen Kubernetes-Node nicht finden. Wie kann ich das beheben?

Es gibt zwei wahrscheinliche Szenarien, warum Trident keinen Kubernetes-Node finden kann. Dies kann auf ein Netzwerkproblem innerhalb von Kubernetes oder auf ein DNS-Problem zurückzuführen sein. Das Trident Node-Demonset, das auf jedem Kubernetes Node ausgeführt wird, muss mit dem Trident Controller kommunizieren können, um den Node bei Trident zu registrieren. Wenn nach der Installation von Trident Netzwerkänderungen aufgetreten sind, tritt dieses Problem nur bei den neuen Kubernetes-Nodes auf, die dem Cluster hinzugefügt werden.

## Geht der Trident Pod verloren, gehen die Daten verloren?

Daten gehen nicht verloren, wenn der Trident Pod zerstört wird. Trident Metadaten werden in CRD-Objekten gespeichert. Alle PVS, die von Trident bereitgestellt wurden, funktionieren ordnungsgemäß.

## Upgrade von Trident

### Kann ich ein Upgrade von einer älteren Version direkt auf eine neuere Version durchführen (einige Versionen werden übersprungen)?

NetApp unterstützt das Upgrade von Trident von einer Hauptversion auf die nächste unmittelbare Hauptversion. Sie können ein Upgrade von Version 18.xx auf 19.xx, 19.xx auf 20.xx usw. durchführen. Sie sollten das Upgrade vor der Implementierung in einer Produktionsumgebung in einem Labor testen.

### Ist es möglich, Trident auf eine vorherige Version herunterzustufen?

Wenn Sie nach einem Upgrade, Abhängigkeitsproblemen oder einem nicht erfolgreichen oder unvollständigen Upgrade Fehler beheben müssen, sollten Sie ["Deinstallieren Sie Trident"](#) die frühere Version mithilfe der entsprechenden Anweisungen für diese Version neu installieren. Dies ist der einzige empfohlene Weg, um ein Downgrade auf eine frühere Version.

## Back-Ends und Volumes managen

### Muss ich Management- und Daten-LIFs in einer ONTAP-Back-End-Definitionsdatei definieren?

Die Management-LIF ist erforderlich. Logische Datenschnittstelle variiert:

- ONTAP SAN: Nicht für iSCSI angeben. Trident verwendet ["ONTAP selektive LUN-Zuordnung"](#), um die für die Einrichtung einer Multi-Path-Sitzung erforderlichen iSCSI LIFs zu ermitteln. Eine Warnung wird erzeugt, wenn `dataLIF` explizit definiert ist. Weitere Informationen finden Sie unter ["ONTAP-SAN-Konfigurationsoptionen und Beispiele"](#).
- ONTAP NAS: Wir empfehlen die Angabe `dataLIF`. Falls nicht bereitgestellt, ruft Trident die Daten-LIFs von der SVM ab. Sie können einen vollständig qualifizierten Domännennamen (FQDN) angeben, der für die NFS-Mount-Vorgänge verwendet werden soll. Damit können Sie ein Round-Robin-DNS zum Load-Balancing über mehrere Daten-LIFs erstellen. Weitere Informationen finden Sie unter ["ONTAP-NAS-Konfigurationsoptionen und Beispiele"](#).

### Kann Trident CHAP für ONTAP-Back-Ends konfigurieren?

Ja. Trident unterstützt bidirektionales CHAP für ONTAP Back-Ends. Dies erfordert die Einstellung `useCHAP=true` in Ihrer Backend-Konfiguration.

### Wie verwalte ich Exportrichtlinien mit Trident?

Trident kann Exportrichtlinien ab Version 20.04 dynamisch erstellen und verwalten. Dadurch kann der Storage-Administrator einen oder mehrere CIDR-Blöcke in seiner Back-End-Konfiguration bereitstellen und Trident Add-Node-IPs erstellen, die einer erstellten Exportrichtlinie innerhalb dieses Bereichs liegen. Auf diese Weise verwaltet Trident automatisch das Hinzufügen und Löschen von Regeln für Knoten mit IPs innerhalb der angegebenen CIDRs.

### Können IPv6-Adressen für das Management und die Daten-LIFs verwendet werden?

Trident unterstützt das Definieren von IPv6-Adressen für:

- `managementLIF` Und `dataLIF` für ONTAP NAS-Back-Ends.
- `managementLIF` Für ONTAP SAN Back-Ends. Sie können die Angabe auf einem ONTAP-SAN-Backend nicht `dataLIF` machen.

Trident muss mit dem Flag (für die `tridentctl` Installation), IPv6 (für den Trident-Operator) oder `tridentTPv6` (für die Helm-Installation) installiert `--use-ipv6` werden, damit es über IPv6 funktioniert.

### **Ist es möglich, die Management LIF auf dem Backend zu aktualisieren?**

Ja, es ist möglich, die Back-End-Management-LIF mit dem Befehl zu aktualisieren `tridentctl update backend`.

### **Ist es möglich, die Daten-LIF auf dem Backend zu aktualisieren?**

Sie können die Daten-LIF nur bei und `ontap-nas-economy` aktualisieren `ontap-nas`.

### **Kann ich mehrere Back-Ends in Trident für Kubernetes erstellen?**

Trident kann viele Backends gleichzeitig unterstützen, entweder mit dem gleichen Treiber oder mit verschiedenen Treibern.

### **Wie speichert Trident Back-End-Anmeldeinformationen?**

Trident speichert die Back-End-Zugangsdaten als Kubernetes Secrets.

### **Wie wählt Trident ein bestimmtes Backend aus?**

Wenn die Backend-Attribute nicht verwendet werden können, um automatisch die richtigen Pools für eine Klasse auszuwählen, werden die `storagePools` Parameter und `additionalStoragePools` verwendet, um einen bestimmten Pool-Satz auszuwählen.

### **Wie kann ich sicherstellen, dass die Trident nicht über ein bestimmtes Backend zur Verfügung stellt?**

Mit dem `excludeStoragePools` Parameter wird der Satz von Pools gefiltert, den Trident für die Bereitstellung verwendet, und alle passenden Pools werden entfernt.

### **Wenn es mehrere Back-Ends derselben Art gibt, wie wählt Trident das zu verwendende Back-End aus?**

Wenn mehrere konfigurierte Back-Ends des gleichen Typs vorhanden sind, wählt Trident das entsprechende Back-End basierend auf den in und `PersistentVolumeClaim` vorhandenen Parametern aus `StorageClass`. Wenn beispielsweise mehrere ONTAP-nas-Treiber-Backends vorhanden sind, versucht Trident, die Parameter im zu vergleichen `StorageClass` und `PersistentVolumeClaim` kombiniert und ein Backend zu verwenden, das die in und `PersistentVolumeClaim` aufgeführten Anforderungen erfüllen kann `StorageClass`. Wenn mehrere Back-Ends für die Anforderung vorhanden sind, wählt Trident zufällig einen aus.

### **Unterstützt Trident bidirektionales CHAP mit Element/SolidFire?**

Ja.

### **Wie implementiert Trident qtrees auf einem ONTAP Volume? Wie viele qtrees können auf einem einzelnen Volume implementiert werden?**

Der `ontap-nas-economy` Treiber erstellt bis zu 200 Qtrees in derselben FlexVol (konfigurierbar zwischen 50 und 300), 100,000 Qtrees pro Cluster-Node und 2,4 Millionen pro Cluster. Wenn Sie eine neue eingeben `PersistentVolumeClaim`, die vom Economy-Treiber gewartet wird, sucht der Fahrer, ob bereits eine FlexVol vorhanden ist, die den neuen Qtree bedienen kann. Wenn es keine FlexVol gibt, die für den Qtree

Services bereitstellen können, wird eine neue FlexVol erstellt.

### **Wie kann ich Unix Berechtigungen für Volumes festlegen, die auf ONTAP NAS bereitgestellt werden?**

Sie können Unix-Berechtigungen auf dem von Trident bereitgestellten Volume festlegen, indem Sie einen Parameter in der Back-End-Definitionsdatei festlegen.

### **Wie kann ich bei der Bereitstellung eines Volumes einen expliziten Satz von ONTAP-NFS-Mount-Optionen konfigurieren?**

Standardmäßig legt Trident für Kubernetes keine Mount-Optionen auf einen Wert fest. Folgen Sie dem Beispiel, um die Mount-Optionen in der Kubernetes Storage Class anzugeben "[Hier](#)".

### **Wie lege ich die bereitgestellten Volumes auf eine bestimmte Exportrichtlinie fest?**

Um den entsprechenden Hosts den Zugriff auf ein Volume zu ermöglichen, verwenden Sie den `exportPolicy` in der Back-End-Definitionsdatei konfigurierten Parameter.

### **Wie lege ich die Volume-Verschlüsselung über Trident mit ONTAP fest?**

Sie können die Verschlüsselung auf dem von Trident bereitgestellten Volume mit dem Verschlüsselungsparameter in der Back-End-Definitionsdatei festlegen. Weitere Informationen finden Sie unter: "[Funktionsweise von Trident mit NVE und NAE](#)"

### **Wie lässt sich QoS für ONTAP am besten über Trident implementieren?**

Verwenden Sie `StorageClasses`, um QoS für ONTAP zu implementieren.

### **Wie spezifiziere ich Thin oder Thick Provisioning über Trident?**

Die ONTAP-Treiber unterstützen entweder Thin Provisioning oder Thick Provisioning. Die ONTAP-Treiber verwenden Thin Provisioning standardmäßig. Wenn Thick Provisioning gewünscht ist, sollten Sie entweder die Backend-Definitionsdatei oder die konfigurieren `StorageClass`. Wenn beide konfiguriert sind, `StorageClass` hat Vorrang. Konfigurieren Sie Folgendes für ONTAP:

1. Ein `StorageClass`, setzt das `provisioningType` Attribut als dick.
2. Aktivieren Sie in der Definitionsdatei des Backends Thick Volumes, indem Sie als Volume festlegen `backend spaceReserve` parameter.

### **Wie kann ich sicherstellen, dass die verwendeten Volumes nicht gelöscht werden, auch wenn ich aus Versehen die PVC lösche?**

Der PVC-Schutz ist für Kubernetes ab Version 1.10 automatisch aktiviert.

### **Kann ich NFS-VES erweitern, die von Trident erstellt wurden?**

Ja. Sie können eine PVC erweitern, die von Trident erstellt wurde. Beachten Sie, dass Volume Autogrow eine ONTAP-Funktion ist, die nicht für Trident geeignet ist.

### **Kann ich ein Volume importieren, während es sich in SnapMirror Data Protection (DP) oder offline Modus befindet?**

Der Volumenimport schlägt fehl, wenn sich das externe Volume im DP-Modus befindet oder offline ist. Sie erhalten die folgende Fehlermeldung:

```
Error: could not import volume: volume import failed to get size of
volume: volume <name> was not found (400 Bad Request) command terminated
with exit code 1.
Make sure to remove the DP mode or put the volume online before importing
the volume.
```

### **Wie wird ein Ressourcenkontingent auf ein NetApp Cluster übersetzt?**

Die Kubernetes-Storage-Ressourcen-Quota sollte so lange funktionieren, wie NetApp Storage die Kapazität hat. Wenn der NetApp-Storage die Kubernetes-Kontingenteinstellungen aufgrund von Kapazitätsmangel nicht erfüllen kann, versucht Trident, die Bereitstellung zu übernehmen, es werden jedoch Fehler behoben.

### **Kann ich mit Trident Volume Snapshots erstellen?**

Ja. Das Erstellen von On-Demand-Volume-Snapshots und persistenten Volumes aus Snapshots wird von Trident unterstützt. Um PVS aus Snapshots zu erstellen, stellen Sie sicher, dass das `VolumeSnapshotDataSource` Feature Gate aktiviert wurde.

### **Welche Treiber unterstützen Trident-Volume-Snapshots?**

Ab sofort ist für unsere `ontap-nas,,,,ontap-nas-flexgroup,,,ontap-san ontap-san-economy solidfire-san gcp-cvs`` und ``azure-netapp-files` Backend-Treiber.

### **Wie mache ich ein Snapshot-Backup eines Volumes, das von Trident mit ONTAP bereitgestellt wird?**

Dies ist auf, `ontap-san` und `ontap-nas-flexgroup` Treiber verfügbar `ontap-nas`. Sie können auch einen für den `ontap-san-economy` Treiber auf FlexVol-Ebene angeben `snapshotPolicy`.

Dies ist auch auf den Treibern verfügbar `ontap-nas-economy`, aber auf der Granularität auf FlexVol-Ebene und nicht auf qtree-Ebene. Um die Fähigkeit zu aktivieren, von Trident bereitgestellte Snapshots von Volumes zu erstellen, setzen Sie die Option für den Backend-Parameter `snapshotPolicy` auf die gewünschte Snapshot-Richtlinie, wie auf dem ONTAP-Backend definiert. Alle vom Storage Controller erstellten Snapshots sind von Trident nicht bekannt.

### **Kann ich einen Snapshot-Reserve-Prozentsatz für ein über Trident bereitgestelltes Volume einstellen?**

Ja, Sie können einen bestimmten Prozentsatz an Festplattenspeicher für das Speichern der Snapshot-Kopien über Trident reservieren, indem Sie das Attribut in der Back-End-Definitionsdatei festlegen `snapshotReserve`. Wenn Sie konfiguriert haben `snapshotPolicy` und `snapshotReserve` in der Back-End-Definitionsdatei, wird der Prozentsatz der Snapshot-Reserve entsprechend dem Prozentsatz festgelegt `snapshotReserve`, der in der Backend-Datei angegeben ist. Wenn die `snapshotReserve` Prozentzahl nicht erwähnt wird, nimmt ONTAP den Prozentwert der Snapshot-Reserve standardmäßig auf 5. Wenn die `snapshotPolicy` Option auf keine gesetzt ist, wird der Prozentsatz der Snapshot-Reserve auf 0 gesetzt.

### **Kann ich direkt auf das Snapshot-Verzeichnis des Volumes zugreifen und Dateien kopieren?**

Ja, Sie können auf das Snapshot-Verzeichnis auf dem von Trident bereitgestellten Volume zugreifen, indem Sie den Parameter in der Back-End-Definitionsdatei festlegen `snapshotDir`.



## Kann ich SnapMirror für Volumes über Trident einrichten?

Derzeit muss SnapMirror extern über ONTAP CLI oder OnCommand System Manager festgelegt werden.

## Wie kann ich persistente Volumes auf einen bestimmten ONTAP Snapshot wiederherstellen?

So stellen Sie ein Volume auf einem ONTAP-Snapshot wieder her:

1. Legen Sie den Applikations-POD still, der das persistente Volume nutzt.
2. Zurücksetzen des erforderlichen Snapshots mithilfe von ONTAP CLI oder OnCommand System Manager
3. Starten Sie den Anwendungs-POD neu.

## Kann Trident Volumes auf SVMs bereitstellen, die ein Load Sharing Mirror konfiguriert haben?

Load-Sharing-Spiegelungen können für Root-Volumes von SVMs erstellt werden, die Daten über NFS bereitstellen. ONTAP aktualisiert automatisch die Spiegelungen zur Lastverteilung für Volumes, die von Trident erstellt wurden. Dies kann zu Verzögerungen bei der Montage der Volumes führen. Wenn mehrere Volumes mit Trident erstellt werden, hängt die Bereitstellung eines Volumes davon ab, ob ONTAP die Load-Sharing-Spiegelung aktualisiert.

## Wie lässt sich die Storage-Klassennutzung für jeden Kunden/Mandanten trennen?

Kubernetes erlaubt Storage-Klassen nicht in Namespaces. Kubernetes lässt sich jedoch mithilfe von Storage-Ressourcenkontingenten, die pro Namespace gelten, die Nutzung einer bestimmten Storage-Klasse pro Namespace begrenzen. Um einem bestimmten Namespace-Zugriff auf einen bestimmten Speicher zu verweigern, setzen Sie das Ressourcenkontingent für diese Speicherklasse auf 0.

# Fehlerbehebung

Verwenden Sie die hier angegebenen Zeiger zur Fehlerbehebung bei Problemen, die bei der Installation und Verwendung von Trident auftreten können.

## Allgemeine Fehlerbehebung

- Wenn der Trident Pod nicht ordnungsgemäß ausgeführt werden kann (wenn z. B. der Trident Pod in der Phase mit weniger als zwei einsatzbereiten Containern stecken bleibt `ContainerCreating`), wird ausgeführt `kubectl -n trident describe deployment trident` und kann zusätzliche Einblicke bieten. `kubectl -n trident describe pod trident--**` Auch das Abrufen von Kubelet-Protokollen (zum Beispiel via `journalctl -xeu kubelet`) kann hilfreich sein.
- Wenn in den Trident-Protokollen nicht genügend Informationen vorhanden sind, können Sie versuchen, den Debug-Modus für Trident zu aktivieren, indem Sie das Flag an den Installationsparameter basierend auf Ihrer Installationsoption übergeben `-d`.

Bestätigen Sie dann, dass Debug mit festgelegt ist `./tridentctl logs -n trident`, und suchen Sie im Protokoll nach `level=debug msg`.

### Mit Operator installiert

```
kubectl patch torc trident -n <namespace> --type=merge -p
'{"spec":{"debug":true}}'
```

Dadurch werden alle Trident Pods neu gestartet, was mehrere Sekunden dauern kann. Sie können dies überprüfen, indem Sie die Spalte 'ALTER' in der Ausgabe von `kubectl get pod -n trident`.

Für Trident 20.07 und 20.10 Verwendung `tprov` anstelle von `torc`.

### Installiert mit Helm

```
helm upgrade <name> trident-operator-21.07.1-custom.tgz --set
tridentDebug=true`
```

### Mit tridentctl installiert

```
./tridentctl uninstall -n trident
./tridentctl install -d -n trident
```

- Sie können auch Debug-Protokolle für jedes Backend erhalten, indem Sie in Ihre Backend-Definition `ingl`. `debugTraceFlags` Beispiel: Einbeziehen `debugTraceFlags: {"api":true, "method":true,}`, um API-Aufrufe und Methodenüberschriften in den Trident-Protokollen zu erhalten. Vorhandene Back-Ends können `debugTraceFlags` mit einem konfiguriert `tridentctl backend update` werden.
- Wenn Sie RedHat CoreOS verwenden, stellen Sie sicher, dass dies `iscsid` auf den Workerknoten aktiviert und standardmäßig gestartet ist. Dies kann mit OpenShift MachineConfigs oder durch Ändern der Zündvorlagen erfolgen.
- Ein häufiges Problem, das bei der Verwendung von Trident mit auftreten kann "[Azure NetApp Dateien](#)", ist, wenn die Mandanten- und Client-Geheimnisse aus einer App-Registrierung mit unzureichenden Berechtigungen stammen. Eine vollständige Liste der Trident-Anforderungen finden Sie unter "[Azure NetApp Dateien](#)" Konfiguration.
- Wenn Probleme bei der Montage eines PV an einem Container auftreten, stellen Sie sicher, dass `rpcbind` installiert und ausgeführt wird. Verwenden Sie den erforderlichen Paketmanager für das Host-Betriebssystem, und überprüfen Sie, ob `rpcbind` ausgeführt wird. Sie können den Status des Dienstes überprüfen `rpcbind`, indem Sie einen oder dessen Äquivalent ausführen `systemctl status rpcbind`.
- Wenn ein Trident-Backend meldet, dass es sich in einem Zustand befindet, obwohl es `failed` zuvor gearbeitet hat, ist dies wahrscheinlich durch eine Änderung der SVM/Admin-Anmeldeinformationen des Back-End verursacht. Durch das Aktualisieren der Backend-Informationen mithilfe `tridentctl update backend` des Trident POD oder das Bouncing wird dieses Problem behoben.
- Wenn bei der Installation von Trident mit Docker als Container-Laufzeit Berechtigungsprobleme auftreten, versuchen Sie, Trident mit dem Flag zu installieren `--in cluster=false`. Dadurch wird kein Installationspod verwendet und es werden keine Berechtigungssorgen aufgrund des Benutzers angezeigt `trident-installer`.
- Verwenden Sie den `uninstall` parameter `<Uninstalling Trident>` zum Bereinigen nach einem fehlgeschlagenen Durchlauf. Standardmäßig werden die von Trident erstellten CRDs nicht vom Skript entfernt, sodass es sicher ist, auch in einer laufenden Implementierung zu deinstallieren und wieder zu installieren.
- Wenn Sie ein Downgrade auf eine frühere Version von Trident durchführen möchten, führen Sie zuerst den Befehl aus `tridentctl uninstall`, um Trident zu entfernen. Laden Sie das gewünschte herunter "[Trident Version](#)", und installieren Sie es mit dem `tridentctl install` Befehl.

- Wenn nach einer erfolgreichen Installation eine PVC in der Phase feststeckt `Pending`, kann die Ausführung `kubectl describe pvc` weitere Informationen darüber liefern, warum Trident kein PV für diese PVC bereitgestellt hat.

## Die Bereitstellung von Trident mit dem Operator ist fehlgeschlagen

Wenn Sie Trident mit dem Operator bereitstellen, ändert sich `Installing` der Status von `TridentOrchestrator` in `Installed`. Wenn Sie den Status beobachten `Failed` und der Operator sich nicht selbst wiederherstellen kann, sollten Sie die Protokolle des Operators überprüfen, indem Sie den folgenden Befehl ausführen:

```
tridentctl logs -l trident-operator
```

Das Nachführen der Protokolle des Dreizack-Operators kann auf den Punkt verweisen, an dem das Problem liegt. Ein solches Problem könnte beispielsweise darin liegen, dass die erforderlichen Container-Images nicht von vorgelagerten Registern in einer Airgoed-Umgebung übertragen werden können.

Um zu verstehen, warum die Installation von Trident nicht erfolgreich war, sollten Sie einen Blick auf den `TridentOrchestrator` Status werfen.

```

kubect1 describe torc trident-2
Name:          trident-2
Namespace:
Labels:        <none>
Annotations:   <none>
API Version:   trident.netapp.io/v1
Kind:          TridentOrchestrator
...
Status:
  Current Installation Params:
    IPv6:
    Autosupport Hostname:
    Autosupport Image:
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:
    Image Pull Secrets:          <nil>
    Image Registry:
    k8sTimeout:
    Kubelet Dir:
    Log Format:
    Silence Autosupport:
    Trident Image:
  Message:          Trident is bound to another CR 'trident'
  Namespace:        trident-2
  Status:           Error
  Version:
Events:
  Type          Reason  Age                From                Message
  ----          -
  Warning       Error   16s (x2 over 16s)  trident-operator.netapp.io  Trident
is bound to another CR 'trident'

```

Dieser Fehler zeigt an, dass bereits ein vorhanden ist `TridentOrchestrator`, der zur Installation von Trident verwendet wurde. Da jeder Kubernetes-Cluster nur eine Instanz von Trident haben kann, sorgt der Operator dafür, dass zu einem beliebigen Zeitpunkt nur eine aktive vorhanden ist `TridentOrchestrator`, die er erstellen kann.

Zusätzlich können Sie durch die Beobachtung des Status der Trident Pods oft angeben, ob etwas nicht richtig ist.

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

NAME	READY	STATUS	RESTARTS
trident-csi-4p5kq 5m18s	1/2	ImagePullBackOff	0
trident-csi-6f45bfd8b6-vfrkw 5m19s	4/5	ImagePullBackOff	0
trident-csi-9q5xc 5m18s	1/2	ImagePullBackOff	0
trident-csi-9v95z 5m18s	1/2	ImagePullBackOff	0
trident-operator-766f7b8658-ldzsv 8m17s	1/1	Running	0

Sie können klar sehen, dass die Pods nicht vollständig initialisiert werden können, da ein oder mehrere Container-Images nicht abgerufen wurden.

Um das Problem zu beheben, sollten Sie den CR bearbeiten `TridentOrchestrator`. Alternativ können Sie , löschen `TridentOrchestrator` und eine neue mit der geänderten und genauen Definition erstellen.

## Erfolgreiche Trident-Bereitstellung mit `tridentctl`

Um herauszufinden, was schief gelaufen ist, können Sie das Installationsprogramm mit dem Argument erneut ausführen, das den `-d` Debug-Modus aktiviert und Ihnen hilft, das Problem zu verstehen:

```
./tridentctl install -n trident -d
```

Nachdem Sie das Problem behoben haben, können Sie die Installation wie folgt bereinigen und dann den Befehl erneut ausführen `tridentctl install`:

```
./tridentctl uninstall -n trident  
INFO Deleted Trident deployment.  
INFO Deleted cluster role binding.  
INFO Deleted cluster role.  
INFO Deleted service account.  
INFO Removed Trident user from security context constraint.  
INFO Trident uninstallation succeeded.
```

## Entfernen Sie Trident und CRDs vollständig

Sie können Trident und alle erstellten CRDs und zugehörigen benutzerdefinierten Ressourcen vollständig entfernen.



Dieser Vorgang kann nicht rückgängig gemacht werden. Tun Sie dies nicht, es sei denn, Sie möchten eine völlig neue Installation von Trident. Informationen zum Deinstallieren von Trident ohne Entfernen von CRDs finden Sie unter ["Deinstallieren Sie Trident"](#).

### Betreiber von Trident

So deinstallieren Sie Trident und entfernen CRDs vollständig mit dem Trident-Operator:

```
kubectl patch torc <trident-orchestrator-name> --type=merge -p
'{"spec":{"wipeout":["crds"],"uninstall":true}}'
```

### Helm

So deinstallieren Sie Trident und entfernen CRDs vollständig mit Helm:

```
kubectl patch torc trident --type=merge -p
'{"spec":{"wipeout":["crds"],"uninstall":true}}'
```

### `<code>-Datei findet </code>`

So entfernen Sie CRDs nach der Deinstallation von Trident mit vollständig `tridentctl`

```
tridentctl obliviate crd
```

## Fehler beim Entstopfen des NVMe-Node bei den RWX-RAW-Block-Namespaces o Kubernetes 1.26

Wenn Sie Kubernetes 1.26 ausführen, schlägt das Entstauen der Nodes möglicherweise fehl, wenn NVMe/TCP mit RWX-unformatierten Block-Namespaces verwendet wird. Die folgenden Szenarien bieten eine Behelfslösung für den Fehler. Alternativ können Sie ein Upgrade von Kubernetes auf 1.27 durchführen.

### Namespace und Pod wurden gelöscht

Stellen Sie sich ein Szenario vor, in dem ein über Trident verwalteter Namespace (persistentes Volume NVMe) mit einem Pod verbunden ist. Wenn Sie den Namespace direkt aus dem ONTAP-Backend löschen, bleibt der Entstempungsprozess hängen, nachdem Sie versucht haben, den Pod zu löschen. Dieses Szenario beeinträchtigt nicht das Kubernetes-Cluster oder andere Funktionen.

### Behelfslösung

Heben Sie das persistente Volume (entsprechend dem Namespace) vom entsprechenden Node auf und löschen Sie es.

### Blockierte Daten-LIFs

If you block (or bring down) all the dataLIFs of the NVMe Trident backend, the unstaging process gets stuck when you attempt to delete the pod. In this scenario, you cannot run any NVMe CLI commands on the Kubernetes node.

.Behelfslösung

Das DataLIFS wird zur Wiederherstellung der vollen Funktionalität angezeigt.

### Namespace-Zuordnung wurde gelöscht

If you remove the `hostNQN` of the worker node from the corresponding subsystem, the unstaging process gets stuck when you attempt to delete the pod. In this scenario, you cannot run any NVMe CLI commands on the Kubernetes node.

.Behelfslösung

Fügen Sie die Rückseite dem Subsystem hinzu `hostNQN`.

## Support

NetApp unterstützt Trident auf unterschiedliche Weise. Umfangreiche kostenlose Self-Support-Optionen stehen rund um die Uhr zur Verfügung, wie z. B. Knowledge Base-Artikel (KB) und ein Einseilkanal.

### Lebenszyklus des Trident Supports

Trident bietet je nach Ihrer Version drei Support-Level. Siehe "[Unterstützung der NetApp Softwareversion für Definitionen](#)".

#### Volle Unterstützung

Trident bietet ab dem Veröffentlichungsdatum vollen Support für zwölf Monate.

#### Eingeschränkter Support

Trident bietet eingeschränkten Support für die Monate 13 bis 24 nach dem Veröffentlichungsdatum.

#### Self-Support

Die Trident-Dokumentation ist ab Veröffentlichungsdatum für die Monate 25 - 36 verfügbar.

Version	Volle Unterstützung	Eingeschränkter Support	Self-Support
"24,10"	Oktober 2025	Oktober 2026	Oktober 2027
"24,06"	Juni 2025	Juni 2026	Juni 2027
"24,02"	—	Februar 2026	Februar 2027

Version	Volle Unterstützung	Eingeschränkter Support	Self-Support
"23,10"	—	Oktober 2025	Oktober 2026
"23,07"	—	Juli 2025	Juli 2026
"23.04"	—	April 2025	April 2026
"23.01"	—	—	Januar 2026
"22.10"	—	—	Oktober 2025
"22.07"	—	—	Juli 2025
"22.04"	—	—	April 2025

## Self-Support

Eine umfassende Liste von Artikeln zur Fehlerbehebung finden Sie unter ["NetApp Knowledge Base \(Anmeldung erforderlich\)"](#).

## Community-Support

Es gibt eine lebendige öffentliche Gemeinschaft von Container-Nutzer (einschließlich Trident-Entwickler) auf unserem ["Kanal abstecken"](#). Hier können Sie allgemeine Fragen zum Projekt stellen und verwandte Themen mit Gleichgesinnten diskutieren.

## Technischer Support von NetApp

Um Hilfe mit Trident zu erhalten, erstellen Sie ein Support Bundle mit `tridentctl logs -a -n trident` und senden Sie es an `NetApp Support <Getting Help>`.

## Finden Sie weitere Informationen

- ["Trident-Ressourcen"](#)
- ["Kubernetes Hub"](#)



# Referenz

## Trident-Ports

Erfahren Sie mehr über die Ports, die Trident für die Kommunikation verwendet.

### Trident-Ports

Trident kommuniziert über folgende Ports:

Port	Zweck
8443	Backchannel HTTPS
8001	Endpunkt der Prometheus Kennzahlen
8000	Trident REST-Server
17546	Anschluss für Liveness/Readiness-Sonde, der von Trident Demonset-Pods verwendet wird



Der Anschluss für die Liveness/Readiness-Sonde kann während der Installation mit dem Flag geändert `--probe-port` werden. Es ist wichtig, sicherzustellen, dass dieser Port nicht von einem anderen Prozess auf den Worker-Knoten verwendet wird.

## Trident REST-API

Sie "[Tridentctl-Befehle und -Optionen](#)" sind die einfachste Möglichkeit zur Interaktion mit der Trident REST-API, können Sie, falls gewünscht, den REST-Endpunkt direkt verwenden.

### Wann die REST-API verwendet werden soll

DIE REST-API ist nützlich für erweiterte Installationen, die Trident als Standalone-Binärdatei in Implementierungen ohne Kubernetes verwenden.

Zur Verbesserung der Sicherheit ist das Trident REST API standardmäßig auf localhost beschränkt, wenn es innerhalb eines Pods ausgeführt wird. Um dieses Verhalten zu ändern, müssen Sie das Argument von Trident in der POD-Konfiguration festlegen `-address`.

### REST-API wird verwendet

Für Beispiele, wie diese APIs aufgerufen werden, übergeben Sie das (`-d` Flag debug ). Weitere Informationen finden Sie unter "[Managen Sie Trident mit tridentctl](#)".

Die API funktioniert wie folgt:

#### GET

**GET** <trident-address>/trident/v1/<object-type>

Listet alle Objekte dieses Typs auf.

**GET** <trident-address>/trident/v1/<object-type>/<object-name>

Ruft die Details des benannten Objekts ab.

## POST

**POST** <trident-address>/trident/v1/<object-type>

Erstellt ein Objekt des angegebenen Typs.

- Eine JSON-Konfiguration für das zu erstellende Objekt erforderlich. Informationen zur Spezifikation der einzelnen Objekttypen finden Sie unter "[Managen Sie Trident mit tridentctl](#)".
- Falls das Objekt bereits vorhanden ist, variiert das Verhalten: Back-Ends aktualisiert das vorhandene Objekt, während alle anderen Objekttypen den Vorgang nicht ausführen.

## Löschen

**DELETE** <trident-address>/trident/v1/<object-type>/<object-name>

Löscht die benannte Ressource.



Es existieren weiterhin Volumes, die mit Back-Ends oder Storage-Klassen verbunden sind. Diese müssen separat gelöscht werden. Weitere Informationen finden Sie unter "[Managen Sie Trident mit tridentctl](#)".

## Befehlszeilenoptionen

Trident bietet mehrere Befehlszeilenoptionen für den Trident Orchestrator. Sie können diese Optionen verwenden, um Ihre Bereitstellung zu ändern.

### Protokollierung

**-debug**

Aktiviert die Debugging-Ausgabe.

**-loglevel <level>**

Legt die Protokollierungsebene fest (Debug, Info, Warn, ERROR, Fatal). Standardmäßig Info.

### Kubernetes

**-k8s\_pod**

Verwenden Sie diese Option oder `-k8s_api_server`, um den Kubernetes-Support zu aktivieren. Durch diese Einstellung verwendet Trident die Zugangsdaten für das Kubernetes-Servicekonto eines Pods, um den API-Server zu kontaktieren. Dies funktioniert nur, wenn Trident als Pod in einem Kubernetes-Cluster mit aktivierten Service-Konten ausgeführt wird.

**-k8s\_api\_server <insecure-address:insecure-port>**

Verwenden Sie diese Option oder `-k8s_pod`, um den Kubernetes-Support zu aktivieren. Bei Angabe von `insecure-address` stellt Trident über die angegebene unsichere Adresse und den angegebenen Port eine Verbindung zum

Kubernetes-API-Server her. Dadurch kann Trident außerhalb eines Pods bereitgestellt werden. Es werden jedoch nur unsichere Verbindungen zum API-Server unterstützt. Um eine sichere Verbindung herzustellen, implementieren Sie Trident in einem Pod mit der `-k8s_pod` Option.

## Docker

**-volume\_driver <name>**

Treibername, der bei der Registrierung des Docker-Plug-ins verwendet wird. Standardmäßig ist `netapp`.

**-driver\_port <port-number>**

Hören Sie auf diesen Port statt auf einen UNIX-Domain-Socket.

**-config <file>**

Erforderlich; Sie müssen diesen Pfad zu einer Back-End-Konfigurationsdatei angeben.

## RUHE

**-address <ip-or-host>**

Gibt die Adresse an, auf der der REST-Server von Trident hören soll. Standardmäßig `localhost`. Wenn auf dem `localhost` zuhören und in einem Kubernetes Pod ausgeführt werden, ist der ZUGRIFF auf DIE REST-Schnittstelle nicht direkt von außerhalb des Pods möglich. Verwenden Sie `-address ""`, um den Zugriff auf die REST-Schnittstelle über die POD-IP-Adresse zu ermöglichen.



Die Trident REST-Schnittstelle kann nur für die Wiedergabe unter `127.0.0.1` (für IPv4) oder `[:1]` (für IPv6) konfiguriert werden.

**-port <port-number>**

Gibt den Port an, auf dem der REST-Server von Trident lauschen soll. Die Standardeinstellung ist `8000`.

**-rest**

Aktiviert die REST-Schnittstelle. Standardmäßig auf „true“ gesetzt.

## Kubernetes und Trident Objekte

Kubernetes und Trident lassen sich über REST-APIs miteinander interagieren, indem Objekte gelesen und geschrieben werden. Es gibt verschiedene Ressourcenobjekte, die die Beziehung zwischen Kubernetes und Trident, Trident und Storage sowie Kubernetes und Storage vorschreiben. Einige dieser Objekte werden über Kubernetes verwaltet, andere wiederum über Trident.

### Wie interagieren die Objekte miteinander?

Am einfachsten ist es, die Objekte, deren Bedeutung und ihre Interaktion zu verstehen, wenn ein Kubernetes-Benutzer eine einzelne Storage-Anfrage bearbeitet:

1. Ein Benutzer erstellt ein, das `PersistentVolumeClaim` eine neue Anforderung einer bestimmten Größe von einem Kubernetes `StorageClass` anfordert `PersistentVolume`, das zuvor vom Administrator konfiguriert wurde.

2. Kubernetes `StorageClass` identifiziert Trident als bereitstellung und enthält Parameter, die Trident sagen, wie ein Volume für die angeforderte Klasse bereitgestellt werden kann.
3. Trident betrachtet seinen eigenen `StorageClass` Namen mit dem gleichen Namen, der die Abgleichung identifiziert `Backends` und `StoragePools` den es zur Bereitstellung von Volumes für die Klasse verwenden kann.
4. Trident stellt Storage auf einem passenden Back-End bereit und erstellt zwei Objekte: Ein `PersistentVolume` in Kubernetes, das Kubernetes über die Suche, das Mounten und die Behandlung des Volume informiert, und ein Volume in Trident, das die Beziehung zwischen dem und dem tatsächlichen Storage beibehält. `PersistentVolume`
5. Kubernetes bindet das `PersistentVolumeClaim` an das neue `PersistentVolume`. Pods, die das Mount von `PersistentVolume` auf jedem Host enthalten `PersistentVolumeClaim`, auf dem es ausgeführt wird.
6. Ein Benutzer erstellt `VolumeSnapshot` mithilfe eines s, das auf Trident verweist, eine einer vorhandenen PVC `VolumeSnapshotClass`.
7. Trident identifiziert das dem PVC zugeordnete Volume und erstellt einen Snapshot des Volumes auf dem Back-End. Es erstellt auch ein `VolumeSnapshotContent`, das Kubernetes über die Identifizierung des Snapshots anweist.
8. Ein Benutzer kann ein Verwenden `VolumeSnapshot` als Quelle erstellen `PersistentVolumeClaim`.
9. Trident identifiziert den erforderlichen Snapshot und führt die gleichen Schritte aus, die beim Erstellen von A und A `Volume` erforderlich sind `PersistentVolume`.



Für weitere Informationen zu Kubernetes-Objekten empfehlen wir, den Abschnitt der Kubernetes-Dokumentation zu lesen "[Persistente Volumes](#)".

## `PersistentVolumeClaim` Kubernetes Objekte

Ein Kubernetes- `PersistentVolumeClaim` Objekt ist eine Anforderung von Storage, der von einem Kubernetes-Cluster-Benutzer erstellt wird.

Zusätzlich zur Standardspezifikation können Benutzer mit Trident die folgenden Volume-spezifischen Anmerkungen angeben, wenn sie die in der Back-End-Konfiguration festgelegten Standardeinstellungen überschreiben möchten:

Anmerkung	Volume-Option	Unterstützte Treiber
<code>trident.netapp.io/fileSystem</code>	Dateisystem	ontap-san, solidfire-san, ontap-san-Economy
<code>trident.netapp.io/cloneFromPVC</code>	KlonSourceVolume	ontap-nas, ontap-san, solidfire-san, Azure-netapp-Dateien, gcp-cvs, ontap-san-Ökonomie
<code>trident.netapp.io/splitOnClone</code>	SPLITOnClone	ontap-nas, ontap-san
<code>trident.netapp.io/protocol</code>	Protokoll	Alle
<code>trident.netapp.io/exportPolicy</code>	Exportpolitik	ontap-nas, ontap-nas-Economy, ontap-nas-Flexgroup
<code>trident.netapp.io/snapshotPolicy</code>	SnapshotPolicy	ontap-nas, ontap-nas-Economy, ontap-nas-Flexgroup, ontap-san

Anmerkung	Volume-Option	Unterstützte Treiber
trident.netapp.io/snapshotReserve	SnapshotReserve	ontap-nas, ontap-nas-Flexgroup, ontap-san, gcp-cvs
trident.netapp.io/snapshotDirectory	SnapshotDirectory	ontap-nas, ontap-nas-Economy, ontap-nas-Flexgroup
trident.netapp.io/unixPermissions	UnxPermissions	ontap-nas, ontap-nas-Economy, ontap-nas-Flexgroup
trident.netapp.io/blockSize	Blocksize	solidfire-san

Wenn das erstellte PV über die Zurückgewinnungsrichtlinie verfügt `Delete`, löscht Trident sowohl das PV als auch das Back-Volume, wenn das PV freigegeben wird (d. h. wenn der Benutzer die PVC löscht). Sollte die Löschkaktion fehlschlagen, markiert Trident den PV als solche und wiederholt den Vorgang periodisch, bis er erfolgreich ist oder der PV manuell gelöscht wird. Wenn das PV die Richtlinie verwendet `Retain`, ignoriert Trident sie und geht davon aus, dass der Administrator sie von Kubernetes und dem Back-End bereinigt, sodass das Volume vor dem Entfernen gesichert oder inspiziert werden kann. Beachten Sie, dass das Löschen des PV nicht dazu führt, dass Trident das Backing-Volume löscht. Sie sollten es mit der REST API entfernen (`tridentctl`).

Trident unterstützt die Erstellung von Volume Snapshots anhand der CSI-Spezifikation: Sie können einen Volume Snapshot erstellen und ihn als Datenquelle zum Klonen vorhandener PVCs verwenden. So können zeitpunktgenaue Kopien von PVS in Form von Snapshots Kubernetes zugänglich gemacht werden. Die Snapshots können dann verwendet werden, um neue PVS zu erstellen. Schauen Sie sich an `On-Demand Volume Snapshots`, um zu sehen, wie das funktionieren würde.

Trident liefert außerdem die `cloneFromPVC` Annotationen und `splitOnClone` zum Erstellen von Klonen. Mit diesen Anmerkungen können Sie eine PVC klonen, ohne die CSI-Implementierung verwenden zu müssen.

Hier ist ein Beispiel: Wenn ein Benutzer bereits eine PVC aufgerufen hat `mysql`, kann der Benutzer eine neue PVC erstellen, die über die Anmerkung aufgerufen wird `mysqlclone`, wie `trident.netapp.io/cloneFromPVC: mysql`z.B. .` Mit diesem Anmerkungsset klonst Trident das Volume, das dem `mysql` PVC entspricht, anstatt ein Volume von Grund auf neu bereitzustellen.

Berücksichtigen Sie folgende Punkte:

- Wir empfehlen das Klonen eines inaktiven Volumes.
- Ein PVC und sein Klon sollten sich im gleichen Kubernetes Namespace befinden und dieselbe Storage-Klasse haben.
- Bei den `ontap-nas` und `ontap-san`-Treibern könnte es wünschenswert sein, die PVC-Beschriftung in Verbindung mit `trident.netapp.io/cloneFromPVC` einzustellen `trident.netapp.io/splitOnClone`. Mit `trident.netapp.io/splitOnClone` Set-auf teilt Trident das geklonte Volume vom übergeordneten Volume auf `true` und entkoppelt damit den Lebenszyklus des geklonten Volume vollständig von seinem übergeordneten Volume, was den Verlust einiger Storage-Effizienz bedeutet. Wenn Sie diese Einstellung nicht festlegen oder auf `false` diese Einstellung setzen `trident.netapp.io/splitOnClone`, verringert sich der Speicherplatzverbrauch im Backend auf Kosten des Erstellens von Abhängigkeiten zwischen den übergeordneten und den Klon-Volumes, sodass das übergeordnete Volume nicht gelöscht werden kann, es sei denn, der Klon wird zuerst gelöscht. Ein Szenario, in dem das Aufteilen des Klons sinnvoll ist, ist das Klonen eines leeren Datenbank-Volumes, in dem erwartet wird, dass das Volume und der zugehörige Klon eine große Divergenz sind. Es profitieren nicht von der Storage-Effizienz des ONTAP.

Das `sample-input` Verzeichnis enthält Beispiele für PVC-Definitionen für die Verwendung mit Trident. Eine vollständige Beschreibung der Parameter und Einstellungen zu Trident Volumes finden Sie unter.

## `PersistentVolume`Kubernetes Objekte

Ein Kubernetes- `PersistentVolume` Objekt ist ein Storage-Element, der dem Kubernetes Cluster zur Verfügung gestellt wird. Es weist einen Lebenszyklus auf, der unabhängig vom POD ist, der ihn nutzt.



Trident erstellt `PersistentVolume` auf Basis der bereitstehenden Volumes automatisch Objekte und registriert sie beim Kubernetes-Cluster. Sie sollten diese nicht selbst verwalten.

Wenn Sie eine PVC erstellen, die sich auf ein Trident-basiertes bezieht `StorageClass`, stellt Trident ein neues Volume mit der entsprechenden Speicherklasse bereit und registriert ein neues PV für dieses Volume. Bei der Konfiguration des bereitgestellten Volume und des entsprechenden PV befolgt Trident folgende Regeln:

- Trident generiert einen PV-Namen für Kubernetes mit einem internen Namen, der zur Bereitstellung des Storage verwendet wird. In beiden Fällen wird sichergestellt, dass die Namen in ihrem Geltungsbereich eindeutig sind.
- Die Größe des Volumens entspricht der gewünschten Größe in der PVC so genau wie möglich, obwohl es möglicherweise auf die nächste zuteilbare Menge aufgerundet werden, je nach Plattform.

## `StorageClass`Kubernetes Objekte

Kubernetes- `StorageClass` Objekte werden mithilfe des Namens in angegeben `PersistentVolumeClaims`, um Storage mit einem Satz von Eigenschaften bereitzustellen. Die Storage-Klasse selbst gibt die zu verwendenden bereitstellungsunternehmen an und definiert die Eigenschaftengruppe in Bezug auf die provisionierung von.

Es handelt sich um eines von zwei grundlegenden Objekten, die vom Administrator erstellt und verwaltet werden müssen. Das andere ist das Trident Back-End-Objekt.

Ein Kubernetes- `StorageClass` Objekt, das Trident verwendet, sieht folgendermaßen aus:

```
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
```

Diese Parameter sind Trident-spezifisch und Trident erläutert die Bereitstellung von Volumes für die Klasse.

Parameter der Storage-Klasse sind:

Attribut	Typ	Erforderlich	Beschreibung
Merkmale	Zuordnen einer Zeichenfolge[string]	Nein	Weitere Informationen finden Sie im Abschnitt Attribute unten
Storage Pools	Zuordnen[String]StringList	Nein	Zuordnung von Back-End-Namen zu Listen von Storage-Pools innerhalb
Zusätzlich StoragePools	Zuordnen[String]StringList	Nein	Zuordnung von Back-End-Namen zu Listen von Storage-Pools innerhalb
Unter Ausnahme von StoragePools	Zuordnen[String]StringList	Nein	Zuordnung von Back-End-Namen zu Listen von Storage-Pools innerhalb

Storage-Attribute und ihre möglichen Werte können in Auswahlebene und Kubernetes-Attribute des Storage-Pools klassifiziert werden.

### Auswahlebene für Storage-Pools

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 <sup>1</sup>	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

Attribut	Typ	Werte	Angebot	Anfrage	Unterstützt von
Klone	bool	Richtig, falsch	Pool unterstützt das Klonen von Volumes	Volume mit aktivierten Klonen	ontap-nas, ontap-san, solidfire-san, gcp-cvs
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

<sup>1</sup>: Nicht unterstützt von ONTAP Select-Systemen

In den meisten Fällen beeinflussen die angeforderten Werte direkt die Bereitstellung. Wenn Sie beispielsweise Thick Provisioning anfordern, entsteht ein Volume mit Thick Provisioning. Ein Element Storage-Pool nutzt jedoch den angebotenen IOPS-Minimum und das Maximum, um QoS-Werte anstelle des angeforderten Werts festzulegen. In diesem Fall wird der angeforderte Wert nur verwendet, um den Speicherpool auszuwählen.

Idealerweise können Sie `attributes` allein die Qualitäten des Storage modellieren, den Sie zur Erfüllung der Anforderungen einer bestimmten Klasse benötigen. Trident erkennt und wählt automatisch Speicherpools aus, die mit den von Ihnen angegebenen *allen* übereinstimmen `attributes`.

Wenn Sie nicht in der Lage sind, `attributes` automatisch die richtigen Pools für eine Klasse auszuwählen, können Sie die Parameter und `additionalStoragePools` verwenden `storagePools`, um die Pools weiter zu verfeinern oder sogar eine bestimmte Gruppe von Pools auszuwählen.

Mit dem Parameter können Sie `storagePools` die Anzahl der Pools, die mit den angegebenen übereinstimmen, weiter einschränken `attributes`. Mit anderen Worten: Trident verwendet die Kreuzung von Pools, die durch die Parameter und `storagePools` für das Provisioning identifiziert `attributes` werden. Sie können entweder allein oder beides zusammen verwenden.

Sie können den Parameter verwenden `additionalStoragePools`, um den Pool-Satz zu erweitern, den Trident für das Provisioning verwendet, unabhängig von den durch die Parameter und `storagePools` ausgewählten Pools `attributes`.

Sie können den Parameter verwenden `excludeStoragePools`, um den Satz von Pools zu filtern, den Trident für das Provisioning verwendet. Mit diesem Parameter werden alle Pools entfernt, die übereinstimmen.

In den `storagePools` Parametern und `additionalStoragePools` hat jeder Eintrag das Formular `<backend>:<storagePoolList>`, wobei `<storagePoolList>` eine kommagetrennte Liste von Speicherpools für das angegebene Backend ist. Beispielsweise könnte ein Wert für `additionalStoragePools` wie aussehen

```
ontapnas_192.168.1.100:aggr1,aggr2;solidfire_192.168.1.101:bronze
```

Diese Listen akzeptieren Regex-Werte sowohl für das Backend als auch für Listenwerte. Sie können verwenden `tridentctl get backend`, um die Liste der Back-Ends und deren Pools zu erhalten.



## Attribute für Kubernetes

Diese Attribute haben keine Auswirkung auf die Auswahl von Storage-Pools/Back-Ends, die von Trident während der dynamischen Provisionierung durchgeführt werden. Stattdessen liefern diese Attribute einfach Parameter, die von Kubernetes Persistent Volumes unterstützt werden. Worker-Knoten sind für die Erstellung von Dateisystem-Operationen verantwortlich und benötigen möglicherweise Dateisystem-Dienstprogramme, wie z. B. xfsprogs.

Attribut	Typ	Werte	Beschreibung	Wichtige Faktoren	Kubernetes-Version
Fstype	Zeichenfolge	Ext4, ext3, xfs	Der Filesystem-Typ für Block-Volumes	solidfire-san, ontap-nas, ontap-nas-Economy, ontap-nas-Flexgroup, ontap-san, ontap-san-Ökonomie	Alle
VolumeErweiterung	boolesch	Richtig, falsch	Aktivieren oder deaktivieren Sie die Unterstützung für das Vergrößern der PVC-Größe	ontap-nas, ontap-nas-Ökonomie, ontap-nas-Flexgroup, ontap-san, ontap-san-Ökonomie, solidfire-san, gcp-cvs, Azure-netapp-Files	1.11+
VolumeBindingmodus	Zeichenfolge	Sofort, WaitForFirstConsumer	Legen Sie fest, wann Volume Binding und dynamische Bereitstellung stattfindet	Alle	1.19 - 1.26

- Mit dem `fsType` Parameter wird der gewünschte Dateisystemtyp für SAN-LUNs gesteuert. Außerdem verwendet Kubernetes die Anwesenheit von `fsType` in einer Storage-Klasse, um anzugeben, dass ein Dateisystem vorhanden ist. Die Volume-Eigentumsrechte können nur mit dem Sicherheitskontext eines Pods gesteuert werden `fsGroup`, wenn `fsType` festgelegt ist. Eine Übersicht über die Einstellung der Volume-Eigentumsrechte mithilfe des `fsGroup` Kontexts finden Sie unter "[Kubernetes: Einen Sicherheitskontext für einen Pod oder Container konfigurieren](#)". Kubernetes setzt diesen `fsGroup` Wert nur ein, wenn:



- `fsType` Wird in der Storage-Klasse festgelegt.
- Der PVC-Zugriffsmodus ist `RWO`.

Für NFS-Speichertreiber ist bereits ein Dateisystem als Teil des NFS-Exports vorhanden. Um die Storage-Klasse zu verwenden, `fsGroup` muss noch ein angegeben werden `fsType`. Sie können es auf oder einen Wert ungleich Null setzen `nfs`.

- Weitere Details zur Volume-Erweiterung finden Sie unter "[Erweitern Sie Volumes](#)".
- Das Trident Installer-Paket enthält mehrere Beispiele für Speicherklassen-Definitionen für die Verwendung mit Trident in `sample-input/storage-class-*.yaml`. Durch das Löschen einer Kubernetes-Storage-Klasse wird auch die entsprechende Trident-Storage-Klasse gelöscht.

## VolumeSnapshotClass`Kubernetes Objekte

Kubernetes- `VolumeSnapshotClass`` Objekte sind analog zu `StorageClasses`. Sie helfen, mehrere Speicherklassen zu definieren und werden von Volume-Snapshots referenziert, um den Snapshot der erforderlichen Snapshot-Klasse zuzuordnen. Jeder Volume Snapshot ist einer einzelnen Volume-Snapshot-Klasse zugeordnet.

Ein `VolumeSnapshotClass` sollte von einem Administrator definiert werden, um Snapshots zu erstellen. Eine Volume-Snapshot-Klasse wird mit folgender Definition erstellt:

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: csi-snapclass
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

Der `driver` gibt an Kubernetes an, dass Anforderungen von Volume-Snapshots der `csi-snapclass` Klasse von Trident verarbeitet werden. Der `deletionPolicy` gibt die Aktion an, die ausgeführt werden soll, wenn ein Snapshot gelöscht werden muss. Wenn `deletionPolicy` auf festgelegt ist `Delete`, werden die Volume-Snapshot-Objekte sowie der zugrunde liegende Snapshot auf dem Speicher-Cluster entfernt, wenn ein Snapshot gelöscht wird. Wenn Sie diese Einstellung auf setzen `Retain`, bedeutet dies, dass `VolumeSnapshotContent` der physische Snapshot beibehalten wird.

## VolumeSnapshot`Kubernetes Objekte

Ein Kubernetes- `VolumeSnapshot`` Objekt ist eine Anforderung zur Erstellung eines Snapshots eines Volumes. So wie eine PVC eine von einem Benutzer erstellte Anfrage für ein Volume darstellt, besteht bei einem

Volume-Snapshot die Anforderung eines Benutzers, einen Snapshot eines vorhandenen PVC zu erstellen.

Wenn eine Volume-Snapshot-Anfrage eingeht, managt Trident automatisch die Erstellung des Snapshots für das Volume auf dem Backend und legt den Snapshot durch Erstellen eines eindeutigen Objekts dar.

`VolumeSnapshotContent` Sie können Snapshots aus vorhandenen VES erstellen und die Snapshots als Datenquelle beim Erstellen neuer VES verwenden.



Der Lebenszyklus eines VolumeSnapshots ist unabhängig von der Quelle PVC: Ein Snapshot bleibt auch nach dem Löschen der Quelle PVC erhalten. Beim Löschen eines PVC mit zugehörigen Snapshots markiert Trident das Backing-Volume für dieses PVC in einem **Deleting**-Zustand, entfernt es aber nicht vollständig. Das Volume wird entfernt, wenn alle zugehörigen Snapshots gelöscht werden.

## `VolumeSnapshotContent` Kubernetes Objekte

Ein Kubernetes- `VolumeSnapshotContent` Objekt ist ein Snapshot, der von einem bereits bereitgestellten Volume erstellt wurde. Er ist analog zu einem `PersistentVolume` und bedeutet einen bereitgestellten Snapshot auf dem Storage-Cluster. Wenn ein Snapshot erstellt wird, behält das Objekt, ähnlich wie `PersistentVolumeClaim` Objekte `VolumeSnapshotContent` von und `PersistentVolume`, eine Eins-zu-eins-Zuordnung zu dem `VolumeSnapshot` Objekt bei, das die Snapshot-Erstellung angefordert hatte.

Das `VolumeSnapshotContent` Objekt enthält Details, die den Snapshot eindeutig identifizieren, z. B. `snapshotHandle`. Dies `snapshotHandle` ist eine eindeutige Kombination aus dem Namen des PV und dem Namen des `VolumeSnapshotContent` Objekts.

Wenn eine Snapshot-Anfrage eingeht, erstellt Trident den Snapshot auf dem Back-End. Nachdem der Snapshot erstellt wurde, konfiguriert Trident ein `VolumeSnapshotContent` Objekt und legt den Snapshot der Kubernetes-API vor.



In der Regel müssen Sie das Objekt nicht verwalten `VolumeSnapshotContent`. Eine Ausnahme ist, wenn Sie außerhalb von Trident erstellen möchten "[Importieren Sie einen Volume-Snapshot](#)".

## `CustomResourceDefinition` Kubernetes Objekte

Kubernetes Custom Ressourcen sind Endpunkte in der Kubernetes API, die vom Administrator definiert werden und zum Gruppieren ähnlicher Objekte verwendet werden. Kubernetes unterstützt das Erstellen individueller Ressourcen zum Speichern einer Sammlung von Objekten. Sie können diese Ressourcendefinitionen erhalten, indem Sie ausführen `kubectl get crds`.

CRDs (Custom Resource Definitions) und die zugehörigen Objektmetadaten werden durch Kubernetes im Metadatenpeicher gespeichert. Dadurch ist kein separater Speicher für Trident erforderlich.

Trident verwendet `CustomResourceDefinition` Objekte, um die Identität von Trident Objekten wie Trident Back-Ends, Trident Storage-Klassen und Trident Volumes zu erhalten. Diese Objekte werden von Trident gemanagt. Darüber hinaus werden im CSI-Volume-Snapshot-Framework einige CRS-IDs verwendet, die zum Definieren von Volume-Snapshots erforderlich sind.

CRDs stellen ein Kubernetes-Konstrukt dar. Objekte der oben definierten Ressourcen werden von Trident erstellt. Ein einfaches Beispiel: Wenn ein Backend mit `tridentctl` erstellt wird, wird ein entsprechendes `tridentbackends` CRD-Objekt für den Verbrauch durch Kubernetes erstellt.

Beachten Sie die folgenden CRDs von Trident:

- Wenn Trident installiert ist, werden eine Reihe von CRDs erstellt und können wie alle anderen Ressourcentypen verwendet werden.
- Wenn Sie Trident mit dem Befehl `tridentctl uninstall`, werden Trident-Pods gelöscht, die erstellten CRDs werden jedoch nicht bereinigt. Informationen dazu, wie Trident vollständig entfernt und neu konfiguriert werden kann, finden Sie unter "[Deinstallieren Sie Trident](#)".

## Trident-Objekte `StorageClass`

Trident erstellt passende Storage-Klassen für Kubernetes- `StorageClass`-Objekte, die in ihrem Feld „bereitstellung“ angegeben werden `csi.trident.netapp.io`. Der Name der Storage-Klasse stimmt mit dem Kubernetes-Objekt überein `StorageClass`, das sie darstellt.



Mit Kubernetes werden diese Objekte automatisch erstellt, wenn ein Kubernetes `StorageClass`, das Trident als bereitstellungsunternehmen verwendet, registriert wird.

Storage-Klassen umfassen eine Reihe von Anforderungen für Volumes. Trident stimmt diese Anforderungen mit den in jedem Storage-Pool vorhandenen Attributen überein. Ist dieser Storage-Pool ein gültiges Ziel für die Bereitstellung von Volumes anhand dieser Storage-Klasse.

Sie können Storage-Klassen-Konfigurationen erstellen, um Storage-Klassen direkt über DIE REST API zu definieren. Bei Kubernetes-Implementierungen erwarten wir jedoch, dass sie bei der Registrierung neuer Kubernetes-Objekte erstellt werden `StorageClass`.

## Trident Back-End-Objekte

Back-Ends stellen die Storage-Anbieter dar, über die Trident Volumes bereitstellt. Eine einzelne Trident Instanz kann eine beliebige Anzahl von Back-Ends managen.



Dies ist einer der beiden Objekttypen, die Sie selbst erstellen und verwalten. Die andere ist das Kubernetes- `StorageClass`-Objekt.

Weitere Informationen zum Erstellen dieser Objekte finden Sie unter "[Back-Ends werden konfiguriert](#)".

## Trident-Objekte `StoragePool`

Storage-Pools stellen die verschiedenen Standorte dar, die für die Provisionierung an jedem Back-End verfügbar sind. Für ONTAP entsprechen diese Aggregaten in SVMs. Bei NetApp HCI/SolidFire entsprechen diese den vom Administrator festgelegten QoS-Bands. Für Cloud Volumes Service entsprechen diese Regionen Cloud-Provider. Jeder Storage-Pool verfügt über eine Reihe individueller Storage-Attribute, die seine Performance-Merkmale und Datensicherungsmerkmale definieren.

Im Gegensatz zu den anderen Objekten hier werden Storage-Pool-Kandidaten immer automatisch erkannt und gemanagt.

## Trident-Objekte `Volume`

Volumes sind die grundlegende Bereitstellungseinheit, die Back-End-Endpunkte umfasst, wie NFS-Freigaben und iSCSI-LUNs. In Kubernetes entsprechen diese direkt `PersistentVolumes`. Wenn Sie ein Volume erstellen, stellen Sie sicher, dass es über eine Storage-Klasse verfügt, die bestimmt, wo das Volume

zusammen mit einer Größe bereitgestellt werden kann.



- In Kubernetes werden diese Objekte automatisch gemanagt. Sie können sich anzeigen lassen, welche Bereitstellung von Trident bereitgestellt wurde.
- Wenn Sie ein PV mit den zugehörigen Snapshots löschen, wird das entsprechende Trident-Volume auf den Status **Löschen** aktualisiert. Damit das Trident Volume gelöscht werden kann, sollten Sie die Snapshots des Volume entfernen.

Eine Volume-Konfiguration definiert die Eigenschaften, über die ein bereitgestelltes Volume verfügen sollte.

Attribut	Typ	Erforderlich	Beschreibung
Version	Zeichenfolge	Nein	Version der Trident API („1“)
Name	Zeichenfolge	ja	Name des zu erstellenden Volumes
Storage Class	Zeichenfolge	ja	Storage-Klasse, die bei der Bereitstellung des Volumes verwendet werden muss
Größe	Zeichenfolge	ja	Größe des Volumes, das in Byte bereitgestellt werden soll
Protokoll	Zeichenfolge	Nein	Zu verwendenden Protokolltyp; „Datei“ oder „Block“
InternalName	Zeichenfolge	Nein	Name des Objekts auf dem Storage-System, das von Trident generiert wird
KlonSourceVolume	Zeichenfolge	Nein	ONTAP (nas, san) & SolidFire-*: Name des Volumes aus dem geklont werden soll
SPLITonClone	Zeichenfolge	Nein	ONTAP (nas, san): Den Klon von seinem übergeordneten Objekt trennen
SnapshotPolicy	Zeichenfolge	Nein	ONTAP-*: Die Snapshot-Richtlinie zu verwenden
SnapshotReserve	Zeichenfolge	Nein	ONTAP-*: Prozentsatz des für Schnappschüsse reservierten Volumens
Exportpolitik	Zeichenfolge	Nein	ontap-nas*: Richtlinie für den Export zu verwenden
SnapshotDirectory	bool	Nein	ontap-nas*: Ob das Snapshot-Verzeichnis sichtbar ist

Attribut	Typ	Erforderlich	Beschreibung
UnxPermissions	Zeichenfolge	Nein	ontap-nas*: Anfängliche UNIX-Berechtigungen
Blocksize	Zeichenfolge	Nein	SolidFire-*: Block-/Sektorgröße
Dateisystem	Zeichenfolge	Nein	Typ des Filesystems

Trident wird beim Erstellen des Volume generiert `internalName`. Dies besteht aus zwei Schritten. Zuerst wird das Speicherpräfix (entweder der Standard oder das Präfix in der Backend-Konfiguration) dem Volume-Namen vorangestellt `trident`, was zu einem Namen des Formulars führt `<prefix>-<volume-name>`. Anschließend wird der Name desinifiziert und die im Backend nicht zulässigen Zeichen ersetzt. Für ONTAP-Back-Ends ersetzt er Bindestriche durch Unterstriche (der interne Name lautet also `<prefix>_<volume-name>`). Bei Element-Back-Ends werden Unterstriche durch Bindestriche ersetzt.

Sie können Volume-Konfigurationen verwenden, um Volumes direkt mit der REST-API bereitzustellen, doch in Kubernetes-Implementierungen erwarten wir, dass die meisten Benutzer die standardmäßige Kubernetes-Methode verwenden `PersistentVolumeClaim`. Trident erstellt dieses Volume-Objekt automatisch im Rahmen des Bereitstellungsprozesses.

## Trident-Objekte Snapshot

Snapshots sind eine zeitpunktgenaue Kopie von Volumes, die zur Bereitstellung neuer Volumes oder für Restores verwendet werden kann. In Kubernetes entsprechen diese direkt `VolumeSnapshotContent` Objekten. Jeder Snapshot ist einem Volume zugeordnet, das die Quelle der Daten für den Snapshot ist.

Jedes `Snapshot` Objekt enthält die nachfolgend aufgeführten Eigenschaften:

Attribut	Typ	Erforderlich	Beschreibung
Version	Zeichenfolge	Ja.	Version der Trident API („1“)
Name	Zeichenfolge	Ja.	Name des Trident Snapshot-Objekts
InternalName	Zeichenfolge	Ja.	Name des Trident Snapshot-Objekts auf dem Storage-System
VolumeName	Zeichenfolge	Ja.	Name des Persistent Volume, für das der Snapshot erstellt wird
VolumeInternalName	Zeichenfolge	Ja.	Name des zugehörigen Trident-Volume-Objekts auf dem Storage-System



In Kubernetes werden diese Objekte automatisch gemanagt. Sie können sich anzeigen lassen, welche Bereitstellung von Trident bereitgestellt wurde.

Bei der Erstellung einer Kubernetes- `VolumeSnapshot``Objektanforderung erstellt Trident ein Snapshot-Objekt auf dem zugrunde liegende Storage-System. Die ``internalName` des

Snapshot-Objekts wird durch die Kombination des Präfixes mit dem UID des VolumeSnapshot Objekts generiert snapshot- (z. B. snapshot-e8d8a0ca-9826-11e9-9807-525400f3f660 ). volumeName Und volumeInternalName werden mit den Details des Backing-Volumes gefüllt.

## Trident ResourceQuota-Objekt

Die Trident-Deamonset-Technologie nutzt eine `system-node-critical` Prioritätsklasse – die höchste in Kubernetes verfügbare Klasse –, um sicherzustellen, dass Trident Volumes während des ordnungsgemäßen Shutdowns identifizieren und bereinigen kann. Trident-Dämonset-Pods vermeiden Workloads mit einer niedrigeren Priorität in Clustern, bei denen der Ressourcendruck hoch ist.

Um dies zu erreichen, verwendet Trident ein `ResourceQuota` Objekt, um sicherzustellen, dass eine „systemNode-kritische“ Prioritätsklasse auf dem Trident-Dämonenset erfüllt ist. Vor der Bereitstellung und der Erstellung von Dämonensets sucht Trident nach dem `ResourceQuota` Objekt und wendet es an, falls es nicht erkannt wird.

Wenn Sie mehr Kontrolle über die Standardkontingente und Prioritätsklasse benötigen, können Sie ein Objekt mithilfe des Helm-Diagramms erzeugen `custom.yaml` oder konfigurieren `ResourceQuota`.

Im Folgenden finden Sie ein Beispiel für ein `ResourceQuota` Objekt mit Priorität des Trident-Dämonenset.

```
apiVersion: <version>
kind: ResourceQuota
metadata:
  name: trident-csi
  labels:
    app: node.csi.trident.netapp.io
spec:
  scopeSelector:
    matchExpressions:
      - operator : In
        scopeName: PriorityClass
        values: ["system-node-critical"]
```

Weitere Informationen zu Ressourcenquoten finden Sie unter "[Kubernetes: Ressourcenkontingente](#)".

### Bereinigen Sie sich `ResourceQuota`, wenn die Installation fehlschlägt

In dem seltenen Fall, in dem die Installation nach der Erstellung des Objekts fehlschlägt `ResourceQuota`, versuchen Sie zuerst "[Deinstallation](#)", und installieren Sie dann erneut.

Wenn das nicht funktioniert, entfernen Sie das Objekt manuell `ResourceQuota`.

### Entfernen `ResourceQuota`

Wenn Sie die Kontrolle über Ihre eigene Ressourcenzuweisung bevorzugen, können Sie das Trident-Objekt mit dem folgenden Befehl entfernen `ResourceQuota`:

```
kubectl delete quota trident-csi -n trident
```

## Pod Security Standards (PSS) und Security Context Constraints (SCC)

Kubernetes Pod Security Standards (PSS) und Pod Security Policies (PSP) definieren Berechtigungsebenen und schränken das Verhalten von Pods ein. OpenShift Security Context Constraints (SCC) definieren ebenfalls die Pod-Einschränkung speziell für die OpenShift Kubernetes Engine. Um diese Anpassung zu ermöglichen, aktiviert Trident bestimmte Berechtigungen während der Installation. In den folgenden Abschnitten werden die von Trident festgelegten Berechtigungen beschrieben.



PSS ersetzt Pod Security Policies (PSP). PSP war in Kubernetes v1.21 veraltet und wird in v1.25 entfernt. Weitere Informationen finden Sie unter "[Kubernetes: Sicherheit](#)".

### Erforderlicher Kubernetes-Sicherheitskontext und zugehörige Felder

Berechtigung	Beschreibung
Privileged	Bei CSI müssen Mount-Punkte bidirektional sein. Das Trident Node-POD muss einen privilegierten Container ausführen. Weitere Informationen finden Sie unter " <a href="#">Kubernetes: Mount-Ausbreitung</a> ".
Host-Netzwerk	Erforderlich für den iSCSI-Daemon. <code>iscsiadm</code> verwaltet iSCSI-Mounts und verwendet Host-Netzwerke zur Kommunikation mit dem iSCSI-Daemon.
Host-IPC	NFS nutzt Prozesskommunikation (IPC) mit dem NFSD.
Host-PID	Erforderlich für den Start <code>rpc-statd</code> von NFS. Trident fragt Hostprozesse ab, um festzustellen, ob <code>rpc-statd</code> vor dem Mounten von NFS-Volumen ausgeführt wird.
Sorgen	Diese <code>SYS_ADMIN</code> Funktion ist Bestandteil der Standardfunktionen für privilegierte Container. Docker setzt beispielsweise diese Funktionen für privilegierte Container: <code>CapPrm: 0000003fffffffff</code> <code>CapEff: 0000003fffffffff</code>
Abt	Seccomp-Profil ist in privilegierten Containern immer „unbeschränkt“; daher kann es in Trident nicht aktiviert werden.



Berechtigung	Beschreibung
SELinux	Auf OpenShift werden privilegierte Container in der Domäne („Super Privileged Container“) ausgeführt <code>spc_t</code> , und nicht privilegierte Container werden in der Domäne ausgeführt <code>container_t</code> . Auf <code>containerd</code> , bei <code>container-selinux</code> installed werden alle Container in der Domain ausgeführt <code>spc_t</code> , was SELinux effektiv deaktiviert. Aus diesem Grund wird Trident den Containern nicht hinzugefügt <code>seLinuxOptions</code> .
DAC	Privilegierte Container müssen als Root ausgeführt werden. Nicht privilegierte Container werden als Root ausgeführt, um auf unix-Sockets zuzugreifen, die von CSI benötigt werden.

## Pod-Sicherheitsstandards (PSS)

Etikett	Beschreibung	Standard
<code>pod-security.kubernetes.io/enforce</code> <code>pod-security.kubernetes.io/enforce-version</code>	Ermöglicht die Aufnahme der Trident Controller und Knoten im Namespace für die Installation. Ändern Sie nicht die Namespace-Bezeichnung.	<code>enforce: privileged</code> <code>enforce-version: &lt;version of the current cluster or highest version of PSS tested.&gt;</code>



Das Ändern der Namespace-Labels kann dazu führen, dass Pods nicht geplant werden, ein „Error Creating: ...“ oder „Warnung: trident-csi-...“. Überprüfen Sie in diesem Fall, ob die Namespace-Bezeichnung für `privileged` geändert wurde. Falls ja, installieren Sie Trident neu.

## Pod-Sicherheitsrichtlinien (PSP)

Feld	Beschreibung	Standard
<code>allowPrivilegeEscalation</code>	Privilegierte Container müssen die Eskalation von Berechtigungen ermöglichen.	<code>true</code>
<code>allowedCSIDrivers</code>	Trident verwendet keine kurzlebigen CSI-Inline-Volumes.	Leer
<code>allowedCapabilities</code>	Für Trident Container ohne Privilegien sind nicht mehr Funktionen erforderlich als für die Standardwerte. Privilegierte Container erhalten alle möglichen Funktionen.	Leer
<code>allowedFlexVolumes</code>	Trident verwendet keinen "FlexVolume-Treiber", daher sind sie nicht in der Liste der erlaubten Volumes enthalten.	Leer

Feld	Beschreibung	Standard
allowedHostPaths	Der Trident-Node-Pod hängt das Root-Dateisystem des Node zusammen, daher bietet es keinen Vorteil, diese Liste zu setzen.	Leer
allowedProcMountTypes	Trident verwendet keine ProcMountTypes.	Leer
allowedUnsafeSysctls	Trident erfordert keine unsicheren sysctls.	Leer
defaultAddCapabilities	Zu privilegierten Containern müssen keine Funktionen hinzugefügt werden.	Leer
defaultAllowPrivilegeEscalation	In jedem Trident Pod werden Berechtigungen erteilt.	false
forbiddenSysctls	Nein sysctls sind zulässig.	Leer
fsGroup	Trident Container werden als Root ausgeführt.	RunAsAny
hostIPC	Für das Mounten von NFS-Volumes ist Host-IPC zur Kommunikation mit erforderlich <code>nfsd</code>	true
hostNetwork	Isctsiadm erfordert, dass das Hostnetzwerk mit dem iSCSI-Daemon kommunizieren kann.	true
hostPID	Host-PID ist erforderlich, um zu überprüfen, ob <code>rpc-statd</code> auf dem Knoten ausgeführt wird.	true
hostPorts	Trident verwendet keine Host Ports.	Leer
privileged	Trident Node-Pods müssen einen privilegierten Container ausführen, um Volumes mounten zu können.	true
readOnlyRootFilesystem	Trident Node-Pods müssen in das Node-Dateisystem schreiben.	false
requiredDropCapabilities	Trident Node-Pods führen einen privilegierten Container aus und können Funktionen nicht ablegen.	none
runAsGroup	Trident Container werden als Root ausgeführt.	RunAsAny
runAsUser	Trident Container werden als Root ausgeführt.	runAsAny
runtimeClass	Trident verwendet nicht <code>RuntimeClasses</code> .	Leer

Feld	Beschreibung	Standard
seLinux	Trident ist nicht festgelegt seLinuxOptions, da es derzeit Unterschiede gibt, wie Container-Laufzeiten und Kubernetes-Distributionen SELinux handhaben.	Leer
supplementalGroups	Trident Container werden als Root ausgeführt.	RunAsAny
volumes	Trident Pods erfordern diese Volume-Plug-ins.	hostPath, projected, emptyDir

## Sicherheitskontexteinschränkungen (SCC)

Etiketten	Beschreibung	Standard
allowHostDirVolumePlugin	Trident-Node-Pods mounten das Root-Dateisystem des Node.	true
allowHostIPC	Für das Mounten von NFS-Volumes muss Host IPC mit kommunizieren nfsd.	true
allowHostNetwork	Isctadm erfordert, dass das Hostnetzwerk mit dem iSCSI-Daemon kommunizieren kann.	true
allowHostPID	Host-PID ist erforderlich, um zu überprüfen, ob rpc-statd auf dem Knoten ausgeführt wird.	true
allowHostPorts	Trident verwendet keine Host Ports.	false
allowPrivilegeEscalation	Privilegierte Container müssen die Eskalation von Berechtigungen ermöglichen.	true
allowPrivilegedContainer	Trident Node-Pods müssen einen privilegierten Container ausführen, um Volumes mounten zu können.	true
allowedUnsafeSysctls	Trident erfordert keine unsicheren sysctls.	none
allowedCapabilities	Für Trident Container ohne Privilegien sind nicht mehr Funktionen erforderlich als für die Standardwerte. Privilegierte Container erhalten alle möglichen Funktionen.	Leer
defaultAddCapabilities	Zu privilegierten Containern müssen keine Funktionen hinzugefügt werden.	Leer
fsGroup	Trident Container werden als Root ausgeführt.	RunAsAny

<b>Etiketten</b>	<b>Beschreibung</b>	<b>Standard</b>
groups	Dieses SCC ist speziell für Trident bestimmt und an den Anwender gebunden.	Leer
readOnlyRootFilesystem	Trident Node-Pods müssen in das Node-Dateisystem schreiben.	false
requiredDropCapabilities	Trident Node-Pods führen einen privilegierten Container aus und können Funktionen nicht ablegen.	none
runAsUser	Trident Container werden als Root ausgeführt.	RunAsAny
seLinuxContext	Trident ist nicht festgelegt <code>seLinuxOptions</code> , da es derzeit Unterschiede gibt, wie Container-Laufzeiten und Kubernetes-Distributionen SELinux handhaben.	Leer
seccompProfiles	Privilegierte Container laufen immer „unbegrenzt“.	Leer
supplementalGroups	Trident Container werden als Root ausgeführt.	RunAsAny
users	Es ist ein Eintrag verfügbar, um diesen SCC an den Trident-Benutzer im Trident Namespace zu binden.	k. A.
volumes	Trident Pods erfordern diese Volume-Plug-ins.	hostPath, downwardAPI, projected, emptyDir

# Rechtliche Hinweise

Rechtliche Hinweise ermöglichen den Zugriff auf Copyright-Erklärungen, Marken, Patente und mehr.

## Urheberrecht

["https://www.netapp.com/company/legal/copyright/"](https://www.netapp.com/company/legal/copyright/)

## Marken

NetApp, das NETAPP Logo und die auf der NetApp Markenseite aufgeführten Marken sind Marken von NetApp Inc. Andere Firmen- und Produktnamen können Marken der jeweiligen Eigentümer sein.

["https://www.netapp.com/company/legal/trademarks/"](https://www.netapp.com/company/legal/trademarks/)

## Patente

Eine aktuelle Liste der NetApp Patente finden Sie unter:

<https://www.netapp.com/pdf.html?item=/media/11887-patentspage.pdf>

## Datenschutzrichtlinie

["https://www.netapp.com/company/legal/privacy-policy/"](https://www.netapp.com/company/legal/privacy-policy/)

## Open Source

Sie können das Urheberrecht und die in der NetApp-Software für Trident verwendeten Lizenzen von Drittanbietern in der Notices-Datei für jedes Release unter nachlesen. <https://github.com/NetApp/trident/>

## Copyright-Informationen

Copyright © 2025 NetApp. Alle Rechte vorbehalten. Gedruckt in den USA. Dieses urheberrechtlich geschützte Dokument darf ohne die vorherige schriftliche Genehmigung des Urheberrechtinhabers in keiner Form und durch keine Mittel – weder grafische noch elektronische oder mechanische, einschließlich Fotokopieren, Aufnehmen oder Speichern in einem elektronischen Abrufsystem – auch nicht in Teilen, vervielfältigt werden.

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

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

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

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

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

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

## Markeninformationen

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