Trident 25.06 Dokumentation

Trident

NetApp
November 18, 2025

This PDF was generated from https://docs.netapp.com/de-de/trident-2506/index.html on November 18, 2025. Always check docs.netapp.com for the latest.

Inhalt

Trident 25.06 Dokumentation . . . 1
Versionshinweise . . . 2
Was ist neu . . . 2
Was ist neu in 25.06.2 . . . 2
Änderungen in 25.06.1 . . . 2
Änderungen in 25.06 . . . 2
Änderungen in 25.02.1 . . . 5
Änderungen in 25.02 . . . 5
Änderungen in 24.10.1 . . . 7
Änderungen in 24.10 . . . 7
Änderungen in 24.06 . . . 9
Änderungen in 24.02 . . . 10
Änderungen in 23.10 . . . 10
Änderungen in 23.07.1 . . . 11
Änderungen in 23.07 . . . 11
Änderungen in 23.04 . . . 12
Änderungen in 23.01.1 . . . 13
Änderungen in 23.01 . . . 14
Änderungen in 22.10 . . . 14
Änderungen in 22.07 . . . 16
Änderungen in 22.04 . . . 17
Änderungen in 22.01.1 . . . 17
Änderungen in 22.01.0 . . . 18
Änderungen in 21.10.1 . . . 18
Änderungen in 21.10.0 . . . 19
Bekannte Probleme . . . 20
Weitere Informationen . . . 21
Frühere Versionen der Dokumentation . . . 21
Bekannte Probleme . . . 21
Das Wiederherstellen von Restic-Backups großer Dateien kann fehlschlagen . . . 21
Erste Schritte . . . 22
Erfahren Sie mehr über Trident . . . 22
Erfahren Sie mehr über Trident . . . 22
Trident -Architektur . . . 23
Konzepte . . . 25
Schnellstart für Trident . . . 29
Wie geht es weiter? . . . 30
Anforderungen . . . 30
Wichtige Informationen zu Trident . . . 30
Unterstützte Frontends (Orchestratoren) . . . 30
Unterstützte Backends (Speicher) . . . 31
Trident Unterstützung für KubeVirt- und OpenShift-Virtualisierung . . . 31
Funktionsanforderungen . . . 32

Getestete Host-Betriebssysteme . . . 32
Hostkonfiguration . . . 33
Speichersystemkonfiguration . . . 33
Trident Häfen . . . 33
Container-Images und zugehörige Kubernetes-Versionen . . . 33
Trident installieren . . . 35
Installation mit dem Trident Operator . . . 35
Installation mit tridentctl . . . 35
Installation mit einem OpenShift-zertifizierten Operator . . . 35
Benutze den Trident. . . . 36
Bereiten Sie den Worker-Knoten vor. . . . 36
Die richtigen Werkzeuge auswählen . . . 36
Knotendiensterkennung . . . 36
NFS -Volumes . . . 37
iSCSI-Volumes . . . 37
NVMe/TCP-Volumes . . . 41
SCSI über FC-Volumes . . . 42
Backends konfigurieren und verwalten . . . 45
Backends konfigurieren . . . 45
Azure NetApp Files . . . 45
Google Cloud NetApp Volumes . . . 65
Konfigurieren eines Cloud Volumes Service für das Google Cloud-Backend . . . 82
Konfigurieren Sie ein NetApp HCI oder SolidFire Backend . . . 94
ONTAP SAN-Treiber . . . 99
ONTAP NAS-Treiber . . . 130
Amazon FSx for NetApp ONTAP . . . 168
Backends mit kubectl erstellen . . . 204
Backends verwalten . . . 212
Speicherklassen erstellen und verwalten . . . 222
Erstellen einer Speicherklasse . . . 222
Speicherklassen verwalten . . . 225
Bereitstellung und Verwaltung von Volumina . . . 227
Bereitstellung eines Volumens . . . 227
Volumen erweitern . . . 231
Importmengen . . . 242
Datenträgernamen und -bezeichnungen anpassen . . . 250
Ein NFS-Volume über Namespaces hinweg freigeben . . . 253
Volumes über Namespaces hinweg klonen . . . 257
Volumes mit SnapMirror replizieren . . . 260
CSI-Topologie verwenden . . . 266
Mit Snapshots arbeiten . . . 274
Arbeiten mit Volumengruppen-Snapshots . . . 282
Trident verwalten und überwachen . . . 287
Upgrade Trident . . . 287
Upgrade Trident . . . 287

Upgrade beim Betreiber . . . 288
Upgrade mit tridentctl . . . 293
Trident mit tridentctl verwalten . . . 294
Befehle und globale Flaggen . . . 294
Befehlsoptionen und Flags . . . 296
Plugin-Unterstützung . . . 302
Monitor Trident . . . 302
Überblick . . . 302
Schritt 1: Definieren Sie ein Prometheus-Ziel . . . 302
Schritt 2: Erstellen Sie einen Prometheus ServiceMonitor . . . 303
Schritt 3: Trident -Metriken mit PromQL abfragen . . . 303
Erfahren Sie mehr über die Telemetrie von Trident AutoSupport. . . . 304
Trident -Metriken deaktivieren . . . 305
Trident deinstallieren . . . 306
Ermitteln Sie die ursprüngliche Installationsmethode . . . 306
Deinstallieren einer Trident -Operatorinstallation . . . 306
Deinstallieren Sie ein `tridentctl` Installation . . . 307
Trident für Docker . . . 308
Voraussetzungen für die Bereitstellung . . . 308
Überprüfen Sie die Anforderungen . . . 308
NVMe-Tools . . . 310
FC -Tools . . . 311
Trident einsetzen . . . 313
Docker-verwaltete Plugin-Methode (Version 1.13/17.03 und höher) . . . 313
Traditionelle Methode (Version 1.12 oder früher) . . . 315
Trident beim Systemstart starten . . . 316
Aktualisieren oder deinstallieren Sie Trident. . . . 317
Upgrade . . . 318
Deinstallieren . . . 319
Arbeiten mit Volumina . . . 319
Erstellen eines Volumes . . . 319
Ein Volume entfernen . . . 320
Klonen eines Volumes . . . 320
Zugriff auf extern erstellte Volumes . . . 322
Fahrerspezifische Lautstärkeoptionen . . . 322
Protokolle sammeln . . . 328
Sammeln Sie Protokolle zur Fehlerbehebung. . . . 328
Allgemeine Tipps zur Fehlerbehebung . . . 329
Mehrere Trident Instanzen verwalten . . . 329
Schritte für das Docker-verwaltete Plugin (Version 1.13/17.03 oder höher) . . . 329
Schritte für die traditionelle Version (Version 1.12 oder älter) . . . 330
Speicherkonfigurationsoptionen . . . 330
Globale Konfigurationsoptionen . . . 330
ONTAP-Konfiguration . . . 331
Element-Softwarekonfiguration . . . 339

Bekannte Probleme und Einschränkungen . . . 341
Beim Upgrade des Trident Docker Volume Plugins von älteren Versionen auf Version 20.10 oder höher tritt ein Upgrade-Fehler mit der Fehlermeldung „Datei oder Verzeichnis nicht gefunden“ auf. . . . 341
Die Bandnamen müssen mindestens 2 Zeichen lang sein. . . . 342
Docker Swarm weist bestimmte Verhaltensweisen auf, die Trident daran hindern, es mit jeder Speicher- und Treiberkombination zu unterstützen. . . . 342
Wird eine FlexGroup bereitgestellt, stellt ONTAP keine zweite FlexGroup bereit, wenn die zweite FlexGroup ein oder mehrere Aggregate mit der bereitzustellenden FlexGroup gemeinsam hat. . . . 342
Bewährte Verfahren und Empfehlungen . . . 343
Einsatz . . . 343
In einem dedizierten Namensraum bereitstellen . . . 343
Nutzen Sie Quoten und Bereichsgrenzen, um den Speicherverbrauch zu kontrollieren. . . . 343
Storage-Konfiguration . . . 343
Plattformübersicht . . . 343
ONTAP und Cloud Volumes ONTAP Best Practices . . . 344
SolidFire – bewährte Verfahren . . . 349
Wo finde ich weitere Informationen? . . . 350
Integriere Trident . . . 351
Fahrerauswahl und -einsatz . . . 351
Speicherklassendesign . . . 354
Virtuelles Pool-Design . . . 355
Volumenoperationen . . . 356
Metrikdienst . . . 360
Datenschutz und Notfallwiederherstellung . . . 361
Trident -Replikation und -Wiederherstellung . . . 361
SVM-Replikation und -Wiederherstellung . . . 362
Volumenreplikation und Wiederherstellung . . . 363
Snapshot-Datenschutz . . . 363
Sicherheit . . . 364
Sicherheit . . . 364
Linux Unified Key Setup (LUKS) . . . 365
Kerberos-Verschlüsselung während des Fluges . . . 371
Schützen Sie Ihre Anwendungen mit Trident Protect. . . . 379
Erfahren Sie mehr über Trident Protect. . . . 379
Wie geht es weiter? . . . 379
Installieren Sie Trident Protect. . . . 379
Trident Schutzanforderungen . . . 379
Trident Protect installieren und konfigurieren . . . 383
Installieren Sie das Trident Protect CLI-Plugin. . . . 386
Trident Schutzinstallation anpassen . . . 390
Trident Schutz verwalten . . . 395
Trident Protect-Autorisierung und Zugriffskontrolle verwalten . . . 395
Trident überwachen und Ressourcen schützen . . . 402
Generieren Sie ein Trident Protect Support-Paket . . . 407
Verbesserter Trident Schutz . . . 409

Anwendungen verwalten und schützen . . . 410
Verwenden Sie Trident Protect AppVault-Objekte, um Buckets zu verwalten. . . . 410
Definieren Sie eine Anwendung für die Verwaltung mit Trident Protect. . . . 425
Schützen Sie Ihre Anwendungen mit Trident Protect. . . . 429
Anwendungen wiederherstellen . . . 440
Anwendungen mit NetApp SnapMirror und Trident Protect replizieren . . . 458
Anwendungen mit Trident Protect migrieren . . . 475
Trident Protect-Ausführungshooks verwalten . . . 479
Trident Protect deinstallieren . . . 491
Trident und Trident schützen Blogs . . . 492
Trident -Blogs . . . 492
Trident schützt Blogs . . . 492
Wissen und Unterstützung . . . 494
Häufig gestellte Fragen . . . 494
Allgemeine Fragen . . . 494
Installieren und verwenden Sie Trident auf einem Kubernetes-Cluster . . . 494
Fehlerbehebung und Support . . . 495
Upgrade Trident . . . 497
Backends und Volumes verwalten . . . 497
Fehlerbehebung . . . 501
Allgemeine Fehlerbehebung . . . 501
Fehlgeschlagener Trident Einsatz mit dem Bediener . . . 503
Fehlgeschlagene Trident Bereitstellung mit `tridentctl` . . . 505
Trident und CRDs vollständig entfernen . . . 505
Fehler beim Entfernen von NVMe-Knoten aus dem Staging-Bereich bei Verwendung von RWX-Raw-Block-Namespaces in Kubernetes 1.26 . . . 506
NFSv4.2-Clients melden nach dem Upgrade von ONTAP „ungültiges Argument“, wenn erwartet wird, dass „v4.2-xattrs“ aktiviert ist . . . 507
Support . . . 507
Trident Unterstützung . . . 507
Selbstversorgung . . . 508
Unterstützung durch die Gemeinschaft . . . 508
Technischer Support von NetApp . . . 508
Weitere Informationen . . . 508
Referenz . . . 509
Trident Häfen . . . 509
Trident Häfen . . . 509
Trident REST-API . . . 509
Wann sollte die REST-API verwendet werden? . . . 509
Verwendung der REST-API . . . 509
Befehlszeilenoptionen . . . 510
Protokollierung . . . 510
Kubernetes . . . 510
Docker . . . 511
AUSRUHEN . . . 511

Kubernetes- und Trident Objekte . . . 511
Wie interagieren die Objekte miteinander? . . . 511
Kubernetes `PersistentVolumeClaim` Objekte . . . 512
Kubernetes `PersistentVolume` Objekte . . . 514
Kubernetes `StorageClass` Objekte . . . 514
Kubernetes `VolumeSnapshotClass` Objekte . . . 518
Kubernetes `VolumeSnapshot` Objekte . . . 518
Kubernetes `VolumeSnapshotContent` Objekte . . . 519
Kubernetes `VolumeGroupSnapshotClass` Objekte . . . 519
Kubernetes `VolumeGroupSnapshot` Objekte . . . 520
Kubernetes `VolumeGroupSnapshotContent` Objekte . . . 520
Kubernetes `CustomResourceDefinition` Objekte . . . 521
Trident `StorageClass` Objekte . . . 521
Trident Backend-Objekte . . . 521
Trident `StoragePool` Objekte . . . 522
Trident `Volume` Objekte . . . 522
Trident `Snapshot` Objekte . . . 523
Trident `ResourceQuota` Objekt . . . 524
Pod-Sicherheitsstandards (PSS) und Sicherheitskontextbeschränkungen (SCC) . . . 525
Erforderlicher Kubernetes-Sicherheitskontext und zugehörige Felder . . . 526
Pod-Sicherheitsstandards (PSS) . . . 526
Pod-Sicherheitsrichtlinien (PSP) . . . 527
Sicherheitskontextbeschränkungen (SCC) . . . 528
Rechtliche Hinweise . . . 531
Copyright . . . 531
Marken . . . 531
Patente . . . 531
Datenschutzrichtlinie . . . 531
Open Source . . . 531

Trident 25.06 Dokumentation

Versionshinweise

Was ist neu

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

> Der `tridentctl` Die im Installations-ZIP-Archiv enthaltene Binärdatei für Linux ist die getestete und unterstützte Version. Beachten Sie, dass `macos` Die im Lieferumfang enthaltene Binärdatei `/extras` Ein Teil der ZIP-Datei wird nicht getestet oder unterstützt.

Was ist neu in 25.06.2

Die Zusammenfassung „Neuerungen“ enthält Details zu Verbesserungen, Fehlerbehebungen und veralteten Funktionen in den Versionen Trident und Trident Protect.

Trident

Fehlerbehebungen

- **Kubernetes**: Kritisches Problem behoben, bei dem beim Trennen von Volumes von Kubernetes-Knoten falsche iSCSI-Geräte erkannt wurden.

Änderungen in 25.06.1

Trident

> Kunden, die SolidFire verwenden, sollten aufgrund eines bekannten Problems beim Aufheben der Veröffentlichung von Volumes kein Upgrade auf 25.06.1 durchführen. Um dieses Problem zu beheben, wird in Kürze 25.06.2 veröffentlicht.

Fehlerbehebungen

- **Kubernetes**:
  - Ein Problem wurde behoben, bei dem NQNs nicht überprüft wurden, bevor sie von Subsystemen getrennt wurden.
  - Ein Problem wurde behoben, bei dem mehrere Versuche, ein LUKS-Gerät zu schließen, zu Fehlern beim Trennen von Volumes führten.
  - Behoben: Unstance des iSCSI-Volumes, wenn sich der Gerätepfad seit seiner Erstellung geändert hat.
  - Blockieren Sie das Klonen von Volumes über Speicherklassen hinweg.
- **OpenShift**: Ein Problem behoben, bei dem die iSCSI-Knotenvorbereitung mit OCP 4.19 fehlschlug.
- Das Timeout beim Klonen eines Volumes mit SolidFire -Backends wurde erhöht ("Ausgabe Nr. 1008").

Änderungen in 25.06

Trident

Verbesserungen

- **Kubernetes**:
  - Unterstützung für CSI-Volume-Group-Snapshots hinzugefügt `v1beta1` Volume Group Snapshot Kubernetes APIs für ONTAP-SAN iSCSI-Treiber. Sehen"Arbeiten mit Volumengruppen-Snapshots" .

    VolumeGroupSnapshot ist eine Beta-Funktion in Kubernetes mit Beta-APIs. Für VolumeGroupSnapshot ist mindestens die Version Kubernetes 1.32 erforderlich.

  - Zusätzlich zu iSCSI wurde die Unterstützung für ONTAP ASA r2 für NVMe/TCP hinzugefügt. Sehenlink:"ONTAP SAN-Konfigurationsoptionen und Beispiele" .
  - Sichere SMB-Unterstützung für ONTAP-NAS- und ONTAP-NAS-Economy-Volumes hinzugefügt. Active Directory-Benutzer und -Gruppen können nun für eine verbesserte Sicherheit mit SMB-Volumes verwendet werden. Sehen"Sichere SMB-Verbindungen aktivieren" .
  - Verbesserte Trident Knotenkonkurrenz für höhere Skalierbarkeit bei Knotenoperationen für iSCSI-Volumes.
  - Hinzugefügt `--allow-discards` beim Öffnen von LUKS-Volumes, um Discard-/TRIM-Befehle zur Speicherplatzrückgewinnung zu ermöglichen.
  - Verbesserte Leistung beim Formatieren von LUKS-verschlüsselten Datenträgern.
  - Erweiterte LUKS-Bereinigung für fehlgeschlagene, aber teilweise formatierte LUKS-Geräte.
  - Verbesserte Trident -Knoten-Idempotenz für das Anhängen und Trennen von NVMe-Volumes.
  - Hinzugefügt `internalID` Feld zur Trident -Volume-Konfiguration für den ONTAP-SAN-Economy-Treiber.
  - Unterstützung für die Volume-Replikation mit SnapMirror für NVMe-Backends hinzugefügt. Sehen"Volumes mit SnapMirror replizieren" .

Experimentelle Verbesserungen

Nicht für den Einsatz in Produktionsumgebungen geeignet.

- [Technische Vorschau] Gleichzeitige Trident -Controller-Operationen über die `--enable-concurrency` Feature-Flag. Dadurch können Controller-Operationen parallel ausgeführt werden, was die Leistung in stark ausgelasteten oder großen Umgebungen verbessert.

  Diese Funktion ist experimentell und unterstützt derzeit nur eingeschränkte parallele Arbeitsabläufe mit dem ONTAP-SAN-Treiber (iSCSI- und FCP-Protokolle).

- [Tech Preview] Manuelle QoS-Unterstützung mit dem ANF-Treiber hinzugefügt.

Fehlerbehebungen

- **Kubernetes**:
  - Es wurde ein Problem mit CSI NodeExpandVolume behoben, bei dem Multipath-Geräte mit inkongruenten Größen zurückbleiben konnten, wenn die zugrunde liegenden SCSI-Festplatten nicht verfügbar waren.
  - Es wurde ein Fehler behoben, der dazu führte, dass doppelte Exportrichtlinien für ONTAP-NAS- und

ONTAP-NAS-Economy-Treiber nicht mehr bereinigt wurden.

  - Behoben: GCNV-Volumes verwenden standardmäßig NFSv3, wenn `nfsMountOptions` ist nicht festgelegt; jetzt werden sowohl NFSv3- als auch NFSv4-Protokolle unterstützt. Wenn `nfsMountOptions` Wird keine NFS-Version angegeben, wird die standardmäßige NFS-Version des Hosts (NFSv3 oder NFSv4) verwendet.
  - Behobenes Bereitstellungsproblem bei der Installation von Trident mit Kustomize ("Ausgabe Nr. 831").
  - Fehlende Exportrichtlinien für aus Snapshots erstellte PVCs behoben ("Ausgabe Nr. 1016").
  - Problem behoben, bei dem die ANF-Volume-Größen nicht automatisch in 1-GiB-Schritten ausgerichtet wurden.
  - Problem bei der Verwendung von NFSv3 mit Bottlerocket behoben.
- Timeout beim Klonen eines Volumes mit SolidFire -Backends behoben ("Ausgabe Nr. 1008").
- Problem behoben, bei dem ONTAP-NAS-Economy-Volumes trotz fehlgeschlagener Größenänderung auf bis zu 300 TB erweitert wurden.
- Problem behoben, bei dem Klon- und Aufteilungsvorgänge bei Verwendung der ONTAP REST API synchron ausgeführt wurden.

Abschaffung:

- **Kubernetes**: Die minimal unterstützte Kubernetes-Version wurde auf v1.27 aktualisiert.

Trident -Schutz

NetApp Trident Protect bietet erweiterte Funktionen zur Anwendungsdatenverwaltung, die die Funktionalität und Verfügbarkeit von Stateful-Kubernetes-Anwendungen verbessern, die von NetApp ONTAP Speichersystemen und dem NetApp Trident CSI-Speicherbereitsteller unterstützt werden.

Verbesserungen

- Verbesserte Wiederherstellungszeiten, wodurch die Möglichkeit besteht, häufigere vollständige Datensicherungen durchzuführen.
- Verbesserte Granularität der Anwendungsdefinition und selektive Wiederherstellung mit Group-Version-Kind (GVK)-Filterung.
- Effiziente Resynchronisierung und umgekehrte Replikation bei Verwendung von AppMirrorRelationship (AMR) mit NetApp SnapMirror, um eine vollständige PVC-Replikation zu vermeiden.
- Die Möglichkeit, AppVault-Buckets mithilfe von EKS Pod Identity zu erstellen, wurde hinzugefügt. Dadurch entfällt die Notwendigkeit, ein Geheimnis mit den Bucket-Anmeldeinformationen für EKS-Cluster anzugeben.
- Es wurde die Möglichkeit hinzugefügt, das Wiederherstellen von Labels und Annotationen im Wiederherstellungs-Namespace bei Bedarf zu überspringen.
- AppMirrorRelationship (AMR) prüft nun, ob eine Quell-PVC-Erweiterung erforderlich ist, und führt bei Bedarf die entsprechende Erweiterung auf der Ziel-PVC durch.

Fehlerbehebungen

- Es wurde ein Fehler behoben, bei dem Snapshot-Anmerkungswerte aus früheren Snapshots auf neuere Snapshots angewendet wurden. Alle Snapshot-Anmerkungen werden nun korrekt angewendet.
- Standardmäßig wird ein Geheimnis für die Datenübertragungsverschlüsselung (Kopia / Restic) definiert, falls nicht anders angegeben.

- Verbesserte Validierung und Fehlermeldungen für die Erstellung von S3 Appvaults hinzugefügt.
- AppMirrorRelationship (AMR) repliziert PVs jetzt nur noch im Bound-Status, um Fehlversuche zu vermeiden.
- Problem behoben, bei dem Fehler angezeigt wurden, wenn AppVaultContent in einem AppVault mit einer großen Anzahl von Backups abgerufen wurde.
- KubeVirt VMSnapshots sind von Wiederherstellungs- und Failover-Operationen ausgeschlossen, um Fehler zu vermeiden.
- Es wurde ein Problem mit Kopia behoben, bei dem Snapshots vorzeitig gelöscht wurden, da der standardmäßige Aufbewahrungsplan von Kopia die vom Benutzer im Zeitplan festgelegten Einstellungen überschrieb.

Änderungen in 25.02.1

Trident

Fehlerbehebungen

- **Kubernetes**:
  - Es wurde ein Problem im Trident-Operator behoben, bei dem die Namen und Versionen der Sidecar-Images bei Verwendung einer nicht standardmäßigen Image-Registry falsch angezeigt wurden ("Ausgabe Nr. 983").
  - Das Problem wurde behoben, bei dem Multipath-Sitzungen während eines ONTAP Failover-Givebacks nicht wiederhergestellt werden konnten ("Ausgabe Nr. 961").

Änderungen in 25.02

Ab Trident 25.02 enthält die Zusammenfassung „Neuerungen“ Details zu Verbesserungen, Fehlerbehebungen und veralteten Funktionen sowohl für Trident als auch für Trident Protect-Versionen.

Trident

Verbesserungen

- **Kubernetes**:
  - Unterstützung für ONTAP ASA r2 für iSCSI hinzugefügt.
  - Unterstützung für das erzwungene Trennen von ONTAP-NAS-Volumes bei nicht ordnungsgemäßem Herunterfahren von Knoten hinzugefügt. Neue ONTAP-NAS-Volumes nutzen nun volumenspezifische Exportrichtlinien, die von Trident verwaltet werden. Es wurde ein Upgrade-Pfad für bestehende Volumes bereitgestellt, um beim Aufheben der Veröffentlichung auf das neue Exportrichtlinienmodell umzustellen, ohne die aktiven Workloads zu beeinträchtigen.
  - Die Annotation cloneFromSnapshot wurde hinzugefügt.
  - Unterstützung für das Klonen von Volumes über Namespaces hinweg hinzugefügt.
  - Erweiterte iSCSI-Selbstheilungsscan-Behebungsmechanismen zur Initiierung von erneuten Scans anhand des exakten Hosts, Kanals, Ziels und der LUN-ID.
  - Unterstützung für Kubernetes 1.32 hinzugefügt.
- **OpenShift**:
  - Unterstützung für die automatische iSCSI-Knotenvorbereitung für RHCOS auf ROSA-Clustern

hinzugefügt.

  - Unterstützung für OpenShift Virtualization für ONTAP -Treiber hinzugefügt.
- Fibre Channel-Unterstützung für ONTAP-SAN-Treiber hinzugefügt.
- NVMe LUKS-Unterstützung hinzugefügt.
- Für alle Basisimages wurde auf ein Scratch-Image umgestellt.
- Die Erkennung und Protokollierung des iSCSI-Verbindungsstatus wurde hinzugefügt, wenn iSCSI-Sitzungen protokolliert werden sollten, dies aber nicht der Fall ist ("Ausgabe Nr. 961").
- Unterstützung für SMB-Volumes mit dem google-cloud-netapp-volumes-Treiber hinzugefügt.
- Es wurde Unterstützung hinzugefügt, um ONTAP -Volumes zu ermöglichen, die Wiederherstellungswarteschlange beim Löschen zu überspringen.
- Es wurde die Möglichkeit hinzugefügt, Standardbilder mithilfe von SHAs anstelle von Tags zu überschreiben.
- Dem tridentctl-Installer wurde das Flag image-pull-secrets hinzugefügt.

Fehlerbehebungen

- **Kubernetes**:
  - Fehlende Knoten-IP-Adressen aus automatischen Exportrichtlinien behoben ("Ausgabe Nr. 965").
  - Behoben: Automatische Exportrichtlinien, die für ONTAP-NAS-Economy vorzeitig auf eine volumenbasierte Richtlinie umgestellt wurden.
  - Die Backend-Konfigurationszugangsdaten wurden korrigiert, um alle verfügbaren AWS-ARN-Partitionen zu unterstützen ("Ausgabe Nr. 913").
  - Option zum Deaktivieren des automatischen Konfiguratorabgleichs im Trident -Operator hinzugefügt ("Ausgabe Nr. 924").
  - Sicherheitskontext für den csi-resizer-Container hinzugefügt ("Ausgabe Nr. 976").

Trident -Schutz

NetApp Trident Protect bietet erweiterte Funktionen zur Anwendungsdatenverwaltung, die die Funktionalität und Verfügbarkeit von Stateful-Kubernetes-Anwendungen verbessern, die von NetApp ONTAP Speichersystemen und dem NetApp Trident CSI-Speicherbereitsteller unterstützt werden.

Verbesserungen

- Unterstützung für Backup und Wiederherstellung von KubeVirt / OpenShift Virtualisierungs-VMs für volumeMode: File und volumeMode: Block (raw device) Speicher hinzugefügt. Diese Unterstützung ist mit allen Trident -Treibern kompatibel und erweitert die bestehenden Schutzfunktionen bei der Replikation von Speicher mit NetApp SnapMirror und Trident Protect.
- Die Möglichkeit, das Einfrieren von Anwendungen in Kubevirt-Umgebungen zu steuern, wurde hinzugefügt.
- Unterstützung für die Konfiguration von AutoSupport Proxy-Verbindungen hinzugefügt.
- Es wurde die Möglichkeit hinzugefügt, ein Geheimnis für die Datenübertragungsverschlüsselung (Kopia / Restic) zu definieren.
- Die Möglichkeit, einen Ausführungs-Hook manuell auszuführen, wurde hinzugefügt.
- Die Möglichkeit, Sicherheitskontextbeschränkungen (SCCs) während der Installation von Trident Protect zu konfigurieren, wurde hinzugefügt.

- Unterstützung für die Konfiguration von nodeSelector während der Installation von Trident Protect hinzugefügt.
- Unterstützung für HTTP/HTTPS-Ausgangsproxy für AppVault-Objekte hinzugefügt.
- Erweiterter ResourceFilter, um den Ausschluss von Ressourcen mit Cluster-Bezug zu ermöglichen.
- Unterstützung für das AWS-Sitzungstoken in den S3 AppVault-Anmeldeinformationen hinzugefügt.
- Unterstützung für die Ressourcenerfassung nach Pre-Snapshot-Ausführungs-Hooks hinzugefügt.

Fehlerbehebungen

- Die Verwaltung temporärer Datenträger wurde verbessert, um die ONTAP Datenträgerwiederherstellungswarteschlange zu umgehen.
- Die SCC-Anmerkungen wurden nun auf ihre ursprünglichen Werte zurückgesetzt.
- Verbesserte Wiederherstellungseffizienz durch Unterstützung paralleler Operationen.
- Verbesserte Unterstützung für Timeouts von Ausführungs-Hooks für größere Anwendungen.

Änderungen in 24.10.1

Verbesserungen

- **Kubernetes**: Unterstützung für Kubernetes 1.32 hinzugefügt.
- Die Erkennung und Protokollierung des iSCSI-Verbindungsstatus wurde hinzugefügt, wenn iSCSI-Sitzungen protokolliert werden sollten, dies aber nicht der Fall ist ("Ausgabe Nr. 961").

Fehlerbehebungen

- Fehlende Knoten-IP-Adressen aus automatischen Exportrichtlinien behoben ("Ausgabe Nr. 965").
- Behoben: Automatische Exportrichtlinien, die für ONTAP-NAS-Economy vorzeitig auf eine volumenbasierte Richtlinie umgestellt wurden.
- Die Abhängigkeiten von Trident und Trident-ASUP wurden aktualisiert, um CVE-2024-45337 und CVE-2024-45310 zu beheben.
- Abmeldungen für zeitweise fehlerhafte Nicht-CHAP-Portale während der iSCSI-Selbstheilung wurden entfernt ("Ausgabe Nr. 961").

Änderungen in 24.10

Verbesserungen

- Der Google Cloud NetApp Volumes -Treiber ist jetzt allgemein für NFS-Volumes verfügbar und unterstützt zonenbasierte Bereitstellung.
- Die GCP-Workload-Identität wird als Cloud-Identität für Google Cloud NetApp Volumes mit GKE verwendet.
- Hinzugefügt `formatOptions` Konfigurationsparameter für die ONTAP-SAN- und ONTAP-SAN-Economy-Treiber, um Benutzern die Angabe von LUN-Formatoptionen zu ermöglichen.
- Die minimale Volume-Größe für Azure NetApp Files wurde auf 50 GiB reduziert. Die neue Mindestgröße für Azure wird voraussichtlich im November allgemein verfügbar sein.
- Hinzugefügt `denyNewVolumePools` Konfigurationsparameter zur Beschränkung der ONTAP-NAS-Economy- und ONTAP-SAN-Economy-Treiber auf bereits vorhandene Flexvol-Pools.

- Hinzugefügte Erkennung für das Hinzufügen, Entfernen oder Umbenennen von Aggregaten aus dem SVM über alle ONTAP Treiber hinweg.
- 18 MiB Overhead wurden zu LUKS-LUNs hinzugefügt, um sicherzustellen, dass die gemeldete PVC-Größe nutzbar ist.
- Verbesserte Fehlerbehandlung für das Stagen und Unstage von ONTAP-SAN- und ONTAP-SAN-Economy-Knoten, um das Unstage-Verfahren zu ermöglichen, das die Entfernung von Geräten nach einem fehlgeschlagenen Stage erlaubt.
- Es wurde ein benutzerdefinierter Rollengenerator hinzugefügt, mit dem Kunden eine minimalistische Rolle für Trident in ONTAP erstellen können.
- Zusätzliche Protokollierung zur Fehlerbehebung hinzugefügt `lsscsi` ("Ausgabe Nr. 792").

Kubernetes

- Neue Trident Funktionen für Kubernetes-native Workflows hinzugefügt:
  - Datenschutz
  - Datenmigration
  - Notfallwiederherstellung
  - Anwendungsmobilität

  "Erfahren Sie mehr über Trident Protect.".
- Eine neue Flagge wurde hinzugefügt `--k8s-api-qps` Installationsprogramme sollen den QPS-Wert festlegen, der von Trident für die Kommunikation mit dem Kubernetes-API-Server verwendet wird.
- Hinzugefügt `--node-prep` Flag für Installationsprogramme zur automatischen Verwaltung von Speicherprotokollabhängigkeiten auf Kubernetes-Clusterknoten. Die Kompatibilität mit dem Amazon Linux 2023 iSCSI-Speicherprotokoll wurde getestet und verifiziert.
- Unterstützung für das erzwungene Trennen von ONTAP-NAS-Economy-Volumes bei nicht ordnungsgemäßem Herunterfahren von Knoten hinzugefügt.
- Neue ONTAP-NAS-Economy NFS-Volumes verwenden pro-qtree-Exportrichtlinien bei der Verwendung `autoExportPolicy` Backend-Option. Qtrees werden nur zum Zeitpunkt der Veröffentlichung knotenbeschränkten Exportrichtlinien zugeordnet, um die Zugriffskontrolle und Sicherheit zu verbessern. Bestehende Qtrees werden auf das neue Exportrichtlinienmodell umgestellt, wenn Trident das Volume von allen Knoten entfernt, um dies zu erreichen, ohne die aktiven Workloads zu beeinträchtigen.
- Unterstützung für Kubernetes 1.31 hinzugefügt.

Experimentelle Verbesserungen

- Technische Vorschau für Fibre Channel-Unterstützung im ONTAP-SAN-Treiber hinzugefügt.

Fehlerbehebungen

- **Kubernetes**:
  - Behobener Rancher-Zugangswebhook verhindert Trident Helm-Installationen ("Ausgabe Nr. 839").
  - Fester Affinitätsschlüssel in Helm-Chart-Werten ("Ausgabe Nr. 898").
  - Der Fehler „tridentControllerPluginNodeSelector/tridentNodePluginNodeSelector funktioniert nicht mit dem Wert "true"“ ("Ausgabe Nr. 899").
  - Gelöschte temporäre Snapshots, die während des Klonvorgangs erstellt wurden ("Ausgabe Nr. 901").

- Unterstützung für Windows Server 2019 hinzugefügt.
- `go mod tidy` im Trident Repository behoben ("Ausgabe Nr. 767").

Veraltete Funktionen

- **Kubernetes:**
  - Die minimal unterstützte Kubernetes-Version wurde auf 1.25 aktualisiert.
  - Die Unterstützung für die POD-Sicherheitsrichtlinie wurde entfernt.

Produkt-Rebranding

Mit der Version 24.10 wird Astra Trident in Trident (Netapp Trident) umbenannt. Diese Umbenennung hat keinerlei Auswirkungen auf Funktionen, unterstützte Plattformen oder die Interoperabilität von Trident.

Änderungen in 24.06

Verbesserungen

- **WICHTIG**: Die `limitVolumeSize` Der Parameter begrenzt nun die Qtree-/LUN-Größen in den ONTAP Economy-Treibern. Nutzen Sie das Neue `limitVolumePoolSize` Parameter zur Steuerung der Flexvol-Größen in diesen Treibern. ("Ausgabe Nr. 341").
- Die Möglichkeit zur iSCSI-Selbstheilung wurde hinzugefügt, um SCSI-Scans anhand der exakten LUN-ID zu initiieren, falls veraltete iGroups verwendet werden ("Ausgabe Nr. 883").
- Es wurde die Unterstützung für Volume-Klon- und Größenänderungsvorgänge hinzugefügt, sodass diese auch dann möglich sind, wenn sich das Backend im Ruhezustand befindet.
- Es wurde die Möglichkeit hinzugefügt, benutzerdefinierte Protokollierungseinstellungen für den Trident -Controller an die Trident Knoten-Pods weiterzugeben.
- In Trident wurde die Unterstützung hinzugefügt, um standardmäßig REST anstelle von ONTAPI (ZAPI) für ONTAP -Versionen 9.15.1 und höher zu verwenden.
- Unterstützung für benutzerdefinierte Volumennamen und Metadaten auf den ONTAP -Speicher-Backends für neue persistente Volumen hinzugefügt.
- Verbesserte `azure-netapp-files` (ANF)-Treiber, um das Snapshot-Verzeichnis standardmäßig automatisch zu aktivieren, wenn die NFS-Mount-Optionen auf die Verwendung von NFS Version 4.x eingestellt sind.
- Bottlerocket-Unterstützung für NFS-Volumes hinzugefügt.
- Technische Vorschauunterstützung für Google Cloud NetApp Volumes hinzugefügt.

Kubernetes

- Unterstützung für Kubernetes 1.30 hinzugefügt.
- Die Möglichkeit, Zombie-Mounts und verbleibende Tracking-Dateien beim Start zu Trident , wurde hinzugefügt ("Ausgabe Nr. 883").
- PVC-Anmerkung hinzugefügt `trident.netapp.io/luksEncryption` zum dynamischen Importieren von LUKS-Volumes ("Ausgabe Nr. 849").
- Topologieerkennung zum ANF-Treiber hinzugefügt.
- Unterstützung für Windows Server 2022-Knoten hinzugefügt.

Fehlerbehebungen

- Installationsfehler von Trident aufgrund veralteter Transaktionen wurden behoben.
- tridentctl wurde so korrigiert, dass Warnmeldungen von Kubernetes ignoriert werden ("Ausgabe Nr. 892").
- Trident Controller ausgetauscht `SecurityContextConstraint` Priorität `0` ("Ausgabe Nr. 887").
- ONTAP -Treiber akzeptieren jetzt Datenträgergrößen unter 20 MiB ("Ausgabe [#885"]).
- Trident wurde so korrigiert, dass das Verkleinern von FlexVol -Volumes während der Größenänderung für den ONTAP-SAN-Treiber verhindert wird.
- Ein Fehler beim Importieren von ANF-Volumes mit NFS v4.1 wurde behoben.

Änderungen in 24.02

Verbesserungen

- Unterstützung für Cloud Identity hinzugefügt.
  - AKS mit ANF – Azure Workload Identity wird als Cloud-Identität verwendet.
  - EKS mit FSxN - Die AWS IAM-Rolle wird als Cloud-Identität verwendet.
- Es wurde die Möglichkeit hinzugefügt, Trident als Add-on auf einem EKS-Cluster über die EKS-Konsole zu installieren.
- Die Möglichkeit, die iSCSI-Selbstheilung zu konfigurieren und zu deaktivieren, wurde hinzugefügt ("Ausgabe Nr. 864").
- Die Amazon FSx Persönlichkeit wurde den ONTAP -Treibern hinzugefügt, um die Integration mit AWS IAM und SecretsManager zu ermöglichen und Trident das Löschen von FSx-Volumes mit Backups zu ermöglichen ("Ausgabe Nr. 453").

Kubernetes

- Unterstützung für Kubernetes 1.29 hinzugefügt.

Fehlerbehebungen

- Behobene ACP-Warnmeldungen, wenn ACP nicht aktiviert ist ("Ausgabe Nr. 866").
- Bei ONTAP -Treibern wurde eine Verzögerung von 10 Sekunden vor der Klonaufteilung während des Snapshot-Löschens hinzugefügt, wenn ein Klon mit dem Snapshot verknüpft ist.

Veraltete Funktionen

- Das In-Toto-Attestierungsframework wurde aus den plattformübergreifenden Image-Manifesten entfernt.

Änderungen in 23.10

Fehlerbehebungen

- Feste Volumenerweiterung, wenn eine neu angeforderte Größe kleiner ist als die Gesamtvolumengröße für ontap-nas- und ontap-nas-flexgroup-Speichertreiber ("Ausgabe Nr. 834").
- Feste Volume-Größe, um beim Import für die Speichertreiber ontap-nas und ontap-nas-flexgroup nur die nutzbare Größe des Volumes anzuzeigen ("Ausgabe Nr. 722").
- Die FlexVol Namensumwandlung für ONTAP-NAS-Economy wurde korrigiert.

- Problem mit der Trident -Initialisierung auf einem Windows-Knoten beim Neustart des Knotens behoben.

Verbesserungen

Kubernetes

Unterstützung für Kubernetes 1.28 hinzugefügt.

Trident

- Unterstützung für die Verwendung von Azure Managed Identities (AMI) mit dem Speichertreiber azure-netapp-files hinzugefügt.
- Unterstützung für NVMe über TCP für den ONTAP-SAN-Treiber hinzugefügt.
- Die Möglichkeit, die Bereitstellung eines Volumes anzuhalten, wenn das Backend vom Benutzer in den angehaltenen Zustand versetzt wird, wurde hinzugefügt ("Ausgabe Nr. 558").

Änderungen in 23.07.1

Kubernetes: Problem mit der Daemonset-Löschung behoben, um Upgrades ohne Ausfallzeiten zu unterstützen ("Ausgabe Nr. 740").

Änderungen in 23.07

Fehlerbehebungen

Kubernetes

- Das Trident -Upgrade wurde so korrigiert, dass alte Pods, die im Status „Beendigung“ feststecken, nicht mehr berücksichtigt werden ("Ausgabe Nr. 740").
- Toleranz zur Definition "transient-trident-version-pod" hinzugefügt ("Ausgabe Nr. 795").

Trident

- Die ONTAPI (ZAPI)-Anfragen wurden korrigiert, um sicherzustellen, dass beim Abrufen von LUN-Attributen LUN-Seriennummern abgefragt werden, um während Node-Staging-Operationen Geister-iSCSI-Geräte zu identifizieren und zu beheben.
- Fehlerbehandlung im Speichertreibercode behoben ("Ausgabe Nr. 816").
- Behobene Quotenanpassung bei Verwendung von ONTAP -Treibern mit use-rest=true.
- Die Erstellung von LUN-Klonen in ontap-san-economy wurde korrigiert.
- Veröffentlichungsinformationsfeld zurücksetzen von `rawDevicePath` Zu `devicePath` ; Logik zum Befüllen und Wiederherstellen (in einigen Fällen) hinzugefügt `devicePath` Feld.

Verbesserungen

Kubernetes

- Unterstützung für den Import von vorab bereitgestellten Snapshots hinzugefügt.
- Minimierte Bereitstellung und Linux-Berechtigungen für DaemonSets ("Ausgabe Nr. 817").

Trident

- Das Statusfeld für „Online“-Volumes und Snapshots wird nicht mehr gemeldet.
- Aktualisiert den Backend-Status, wenn das ONTAP -Backend offline ist ("Ausgabe Nr. 801" ,"#543").
- Die LUN-Seriennummer wird immer während des ControllerVolumePublish-Workflows abgerufen und veröffentlicht.
- Es wurde zusätzliche Logik hinzugefügt, um die Seriennummer und Größe des iSCSI-Multipath-Geräts zu überprüfen.
- Zusätzliche Überprüfung der iSCSI-Volumes, um sicherzustellen, dass das richtige Multipath-Gerät nicht bereitgestellt ist.

Experimentelle Verbesserung

Tech Preview-Unterstützung für NVMe over TCP für den ONTAP-SAN-Treiber hinzugefügt.

Dokumentation

Es wurden zahlreiche organisatorische und formattechnische Verbesserungen vorgenommen.

Veraltete Funktionen

Kubernetes

- Die Unterstützung für v1beta1-Snapshots wurde entfernt.
- Die Unterstützung für Pre-CSI-Volumes und Speicherklassen wurde entfernt.
- Die minimal unterstützte Kubernetes-Version wurde auf 1.22 aktualisiert.

Änderungen in 23.04

> Das erzwungene Trennen von ONTAP-SAN-* Volumes wird nur bei Kubernetes-Versionen unterstützt, bei denen das Feature Gate „Non-Graceful Node Shutdown“ aktiviert ist. Die erzwungene Trennung muss bei der Installation über die folgende Funktion aktiviert werden: `--enable-force-detach` Trident Installationsflag.

Fehlerbehebungen

- Der Trident -Operator wurde so korrigiert, dass er bei Bedarf, wenn in der Spezifikation angegeben, IPv6 localhost für die Installation verwendet.
- Die Berechtigungen der Trident -Operator-Clusterrolle wurden so angepasst, dass sie mit den Bundle-Berechtigungen übereinstimmen ("Ausgabe Nr. 799").
- Problem beim Anhängen von Raw-Block-Volumes an mehrere Knoten im RWX-Modus behoben.
- Die Unterstützung für FlexGroup -Klonen und den Volumenimport für SMB-Volumes wurde behoben.
- Problem behoben, bei dem der Trident -Controller nicht sofort heruntergefahren werden konnte ("Ausgabe Nr. 811").
- Es wurde ein Fix hinzugefügt, um alle igroup-Namen aufzulisten, die einer bestimmten LUN zugeordnet sind, die mit ontap-san-* Treibern bereitgestellt wurde.
- Es wurde ein Fehler behoben, der es externen Prozessen ermöglicht, bis zum Ende ausgeführt zu werden.
- Kompilierungsfehler für die S390-Architektur behoben ("Ausgabe Nr. 537").

- Fehlerhafte Protokollierungsstufe bei Volume-Mount-Operationen behoben ("Ausgabe Nr. 781").
- Möglicher Typzusicherungsfehler behoben ("Ausgabe Nr. 802").

Verbesserungen

- Kubernetes:
  - Unterstützung für Kubernetes 1.27 hinzugefügt.
  - Unterstützung für den Import von LUKS-Volumes hinzugefügt.
  - Unterstützung für den ReadWriteOncePod PVC-Zugriffsmodus hinzugefügt.
  - Unterstützung für das erzwungene Trennen von ONTAP-SAN-* Volumes bei nicht ordnungsgemäßem Herunterfahren von Knoten hinzugefügt.
  - Alle ONTAP-SAN-* Volumes verwenden nun pro-node igroups. Um unsere Sicherheitslage zu verbessern, werden LUNs nur dann igroups zugeordnet, wenn sie aktiv auf diesen Knoten veröffentlicht sind. Bestehende Volumes werden opportunistisch auf das neue igroup-Schema umgestellt, sobald Trident feststellt, dass dies ohne Beeinträchtigung aktiver Workloads sicher möglich ist."Ausgabe Nr. 758").
  - Verbesserte Trident Sicherheit durch Bereinigung nicht verwendeter, von Trident verwalteter igroups aus ONTAP-SAN-* Backends.
- Die Unterstützung für SMB-Volumes mit Amazon FSx wurde zu den Speichertreibern ontap-nas-economy und ontap-nas-flexgroup hinzugefügt.
- Unterstützung für SMB-Freigaben mit den Speichertreibern ontap-nas, ontap-nas-economy und ontap-nas-flexgroup hinzugefügt.
- Unterstützung für arm64-Knoten hinzugefügt ("Ausgabe Nr. 732").
- Verbesserte Trident Abschaltprozedur durch vorherige Deaktivierung der API-Server ("Ausgabe Nr. 811").
- Die Makefile wurde um plattformübergreifende Build-Unterstützung für Windows- und arm64-Hosts erweitert; siehe BUILD.md.

Veraltete Funktionen

Kubernetes: Backend-bezogene igroups werden beim Konfigurieren der Treiber ontap-san und ontap-san-economy nicht mehr erstellt ("Ausgabe Nr. 758").

Änderungen in 23.01.1

Fehlerbehebungen

- Der Trident -Operator wurde so korrigiert, dass er bei Bedarf, wenn in der Spezifikation angegeben, IPv6 localhost für die Installation verwendet.
- Die Berechtigungen der Trident -Operator-Clusterrolle wurden so angepasst, dass sie mit den Bundle-Berechtigungen synchronisiert sind."Ausgabe Nr. 799" .
- Es wurde ein Fehler behoben, der es externen Prozessen ermöglicht, bis zum Ende ausgeführt zu werden.
- Problem beim Anhängen von Raw-Block-Volumes an mehrere Knoten im RWX-Modus behoben.
- Die Unterstützung für FlexGroup -Klonen und den Volumenimport für SMB-Volumes wurde behoben.

Änderungen in 23.01

Kubernetes 1.27 wird jetzt in Trident unterstützt. Bitte aktualisieren Sie Trident, bevor Sie Kubernetes aktualisieren.

Fehlerbehebungen

- Kubernetes: Optionen zum Ausschließen der Pod-Sicherheitsrichtlinienerstellung hinzugefügt, um Trident -Installationen über Helm zu beheben ("Probleme #783, #794").

Verbesserungen

Kubernetes

- Unterstützung für Kubernetes 1.26 hinzugefügt.
- Verbesserte Gesamtressourcennutzung von Trident RBAC ("Ausgabe Nr. 757").
- Es wurde eine Automatisierung hinzugefügt, um fehlerhafte oder veraltete iSCSI-Sitzungen auf Hostknoten zu erkennen und zu beheben.
- Unterstützung für die Erweiterung von LUKS-verschlüsselten Volumes hinzugefügt.
- Kubernetes: Unterstützung für die Rotation von Anmeldeinformationen für LUKS-verschlüsselte Volumes hinzugefügt.

Trident

- Dem ontap-nas-Speichertreiber wurde die Unterstützung für SMB-Volumes mit Amazon FSx for NetApp ONTAP hinzugefügt.
- Unterstützung für NTFS-Berechtigungen bei Verwendung von SMB-Volumes hinzugefügt.
- Unterstützung für Speicherpools für GCP-Volumes mit CVS-Dienstlevel hinzugefügt.
- Unterstützung für die optionale Verwendung von flexgroupAggregateList beim Erstellen von FlexGroups mit dem ontap-nas-flexgroup-Speichertreiber hinzugefügt.
- Verbesserte Leistung des ontap-nas-economy-Speichertreibers bei der Verwaltung mehrerer FlexVol -Volumes
- Aktivierte dataLIF-Updates für alle ONTAP NAS-Speichertreiber.
- Die Namenskonvention für Trident Deployment und DaemonSet wurde aktualisiert, um das Betriebssystem des Host-Knotens widerzuspiegeln.

Veraltete Funktionen

- Kubernetes: Die minimal unterstützte Kubernetes-Version wurde auf 1.21 aktualisiert.
- DataLIFs sollten bei der Konfiguration nicht mehr angegeben werden. `ontap-san` oder `ontap-san-economy` Fahrer.

Änderungen in 22.10

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

Wichtige Informationen zu Trident 22.10

- Kubernetes 1.25 wird jetzt in Trident unterstützt. Sie müssen Trident auf Version 22.10 aktualisieren, bevor Sie auf Kubernetes 1.25 aktualisieren können.
- Trident setzt nun die Verwendung von Multipathing-Konfigurationen in SAN-Umgebungen strikt durch, mit einem empfohlenen Wert von `find_multipaths: no` in der Datei multipath.conf.

  Verwendung einer Nicht-Multipathing-Konfiguration oder Verwendung von `find_multipaths: yes` oder `find_multipaths: smart` Ein falscher Wert in der Datei multipath.conf führt zu Mount-Fehlern. Trident hat die Verwendung von `find_multipaths: no` seit der Version vom 21.07.

Fehlerbehebungen

- Problem behoben, das spezifisch für das ONTAP Backend war, das mit `credentials` Feld konnte während des Upgrades auf Version 22.07.0 nicht online geschaltet werden ("Ausgabe Nr. 759").
- **Docker:** Ein Problem wurde behoben, das dazu führte, dass das Docker-Volume-Plugin in einigen Umgebungen nicht gestartet werden konnte ("Ausgabe Nr. 548" Und"Ausgabe Nr. 760").
- Ein SLM-Problem speziell für ONTAP SAN-Backends wurde behoben, um sicherzustellen, dass nur eine Teilmenge der dataLIFs, die zu den meldenden Knoten gehören, veröffentlicht wird.
- Es wurde ein Leistungsproblem behoben, bei dem beim Anhängen eines Volumes unnötige Scans nach iSCSI-LUNs auftraten.
- Die detaillierten Wiederholungsversuche innerhalb des Trident iSCSI-Workflows wurden entfernt, um ein schnelles Scheitern zu ermöglichen und die externen Wiederholungsintervalle zu reduzieren.
- Es wurde ein Problem behoben, bei dem ein Fehler zurückgegeben wurde, wenn ein iSCSI-Gerät gespült wurde, obwohl das entsprechende Multipath-Gerät bereits gespült worden war.

Verbesserungen

- Kubernetes:
  - Unterstützung für Kubernetes 1.25 hinzugefügt. Sie müssen Trident auf Version 22.10 aktualisieren, bevor Sie auf Kubernetes 1.25 aktualisieren können.
  - Um zukünftige Berechtigungserweiterungen zu ermöglichen, wurden ein separates ServiceAccount, eine separate ClusterRole und eine separate ClusterRoleBinding für die Trident Bereitstellung und das DaemonSet hinzugefügt.
  - Unterstützung hinzugefügt für"Namensraumübergreifende Volumenfreigabe" .
- Alle Trident `ontap-*` Speichertreiber arbeiten jetzt mit der ONTAP REST API.
- Neuer Operator (YAML) hinzugefügt(`bundle_post_1_25.yaml`) ohne ein `PodSecurityPolicy` zur Unterstützung von Kubernetes 1.25.
- Hinzugefügt"Unterstützung für LUKS-verschlüsselte Datenträger" für `ontap-san` Und `ontap-san-economy` Speichertreiber.
- Unterstützung für Windows Server 2019-Knoten hinzugefügt.
- Hinzugefügt"Unterstützung für SMB-Volumes auf Windows-Knoten" durch die `azure-netapp-files` Speichertreiber.
- Die automatische MetroCluster Umschalterkennung für ONTAP -Treiber ist jetzt allgemein verfügbar.

Veraltete Funktionen

- **Kubernetes:** Die minimal unterstützte Kubernetes-Version wurde auf 1.20 aktualisiert.
- Der Astra Data Store (ADS)-Treiber wurde entfernt.
- Unterstützung für entfernt `yes` Und `smart` Optionen für `find_multipaths` bei der Konfiguration von Worker-Node-Multipathing für iSCSI.

Änderungen in 22.07

Fehlerbehebungen

Kubernetes

- Problem bei der Verarbeitung von booleschen und numerischen Werten für den Knotenselektor bei der Konfiguration von Trident mit Helm oder dem Trident Operator behoben. ("GitHub-Problem #700")
- Ein Problem bei der Fehlerbehandlung von Nicht-CHAP-Pfaden wurde behoben, sodass kubelet im Fehlerfall einen erneuten Versuch unternimmt. "GitHub-Problem #736")

Verbesserungen

- Umstellung von k8s.gcr.io auf registry.k8s.io als Standard-Registry für CSI-Images
- ONTAP-SAN-Volumes verwenden nun pro-node igroups und ordnen LUNs nur dann igroups zu, wenn sie aktiv auf diesen Knoten veröffentlicht werden, um unsere Sicherheitslage zu verbessern. Bestehende Volumes werden opportunistisch auf das neue igroup-Schema umgestellt, sobald Trident feststellt, dass dies ohne Beeinträchtigung aktiver Workloads gefahrlos möglich ist.
- Bei Trident -Installationen wurde ein ResourceQuota hinzugefügt, um sicherzustellen, dass das Trident DaemonSet dann eingeplant wird, wenn der Verbrauch der PriorityClass standardmäßig begrenzt ist.
- Dem Azure NetApp Files -Treiber wurde Unterstützung für Netzwerkfunktionen hinzugefügt. ("GitHub-Problem #717")
- Die automatische MetroCluster -Umschalterkennung wurde in ONTAP -Treibern in der technischen Vorschau hinzugefügt. ("GitHub-Problem #228")

Veraltete Funktionen

- **Kubernetes:** Die minimal unterstützte Kubernetes-Version wurde auf 1.19 aktualisiert.
- Die Backend-Konfiguration erlaubt nicht mehr mehrere Authentifizierungstypen in einer einzigen Konfiguration.

Umzüge

- Der AWS CVS-Treiber (veraltet seit Version 22.04) wurde entfernt.
- Kubernetes
  - Die unnötige SYS_ADMIN-Berechtigung wurde aus den Node-Pods entfernt.
  - Reduziert nodeprep auf einfache Hostinformationen und aktive Diensterkennung, um bestmöglich zu bestätigen, dass NFS/iSCSI-Dienste auf den Worker-Knoten verfügbar sind.

Dokumentation

Ein neues"Pod-Sicherheitsstandards" Der Abschnitt (PSS) wurde hinzugefügt, der die von Trident bei der

Installation aktivierten Berechtigungen detailliert beschreibt.

Änderungen in 22.04

NetApp verbessert und erweitert seine Produkte und Services kontinuierlich. Hier sind einige der neuesten Funktionen in Trident. Informationen zu früheren Versionen finden Sie unter "Frühere Versionen der Dokumentation" .

> Wenn Sie von einer früheren Trident Version aktualisieren und Azure NetApp Files verwenden,`location` Der Konfigurationsparameter ist nun ein obligatorisches Einzelfeld.

Fehlerbehebungen

- Verbesserte Analyse von iSCSI-Initiatornamen. ("GitHub-Problem #681")
- Problem behoben, bei dem CSI-Speicherklassenparameter nicht zulässig waren. ("GitHub-Problem #598")
- Doppelte Schlüsseldeklaration in Trident CRD behoben. ("GitHub-Problem #671")
- Fehlerhafte CSI-Snapshot-Protokolle korrigiert. ("GitHub-Problem #629"))
- Problem beim Aufheben der Veröffentlichung von Volumes auf gelöschten Knoten behoben. ("GitHub-Problem #691")
- Hinzugefügte Behandlung von Dateisysteminkonsistenzen auf Blockgeräten. ("GitHub-Problem #656")
- Problem beim Abrufen von Auto-Support-Bildern beim Festlegen der `imageRegistry` Flagge während der Installation. ("GitHub-Problem #715")
- Problem behoben, bei dem der Azure NetApp Files Treiber ein Volume mit mehreren Exportregeln nicht klonen konnte.

Verbesserungen

- Eingehende Verbindungen zu den sicheren Endpunkten von Trident erfordern nun mindestens TLS 1.3. ("GitHub-Problem #698")
- Trident fügt nun HSTS-Header zu den Antworten seiner sicheren Endpunkte hinzu.
- Trident versucht nun, die Unix-Berechtigungsfunktion von Azure NetApp Files automatisch zu aktivieren.
- **Kubernetes**: Das Trident Daemonset läuft jetzt mit der Prioritätsklasse system-node-critical. ("GitHub-Problem #694")

Umzüge

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

Änderungen in 22.01.1

Fehlerbehebungen

- Problem beim Aufheben der Veröffentlichung von Volumes auf gelöschten Knoten behoben. ("GitHub-Problem #691")
- Behoben wurde ein Fehler, der beim Zugriff auf Nullwerte in Feldern für aggregierten Speicherplatz in ONTAP -API-Antworten auftrat.

Änderungen in 22.01.0

Fehlerbehebungen

- **Kubernetes:** Erhöhung der Wiederholungszeit für die Knotenregistrierung bei großen Clustern.
- Es wurde ein Problem behoben, bei dem der azure-netapp-files-Treiber durch mehrere Ressourcen mit demselben Namen verwirrt werden konnte.
- ONTAP SAN IPv6 DataLIFs funktionieren jetzt, wenn sie mit Klammern angegeben werden.
- Problem behoben, bei dem beim Versuch, ein bereits importiertes Volume zu importieren, EOF zurückgegeben wurde und der PVC im Status „Ausstehend“ verblieb. ("GitHub-Problem #489")
- Problem behoben, bei dem die Trident -Performance nachließ, wenn mehr als 32 Snapshots auf einem SolidFire -Volume erstellt wurden.
- Bei der Erstellung von SSL-Zertifikaten wurde SHA-1 durch SHA-256 ersetzt.
- Der Azure NetApp Files -Treiber wurde so angepasst, dass doppelte Ressourcennamen nicht mehr zulässig sind und die Vorgänge auf einen einzigen Speicherort beschränkt werden.
- Der Azure NetApp Files -Treiber wurde so angepasst, dass doppelte Ressourcennamen nicht mehr zulässig sind und die Vorgänge auf einen einzigen Speicherort beschränkt werden.

Verbesserungen

- Kubernetes-Erweiterungen:
  - Unterstützung für Kubernetes 1.23 hinzugefügt.
  - Fügen Sie Planungsoptionen für Trident -Pods hinzu, wenn diese über Trident Operator oder Helm installiert werden. ("GitHub-Problem #651")
- Regionsübergreifende Volumes im GCP-Treiber zulassen. ("GitHub-Problem #633")
- Unterstützung für die Option „unixPermissions“ wurde für Azure NetApp Files -Volumes hinzugefügt. ("GitHub-Problem #666")

Veraltete Funktionen

Die Trident REST-Schnittstelle kann nur unter den Adressen 127.0.0.1 oder [::1] lauschen und Anfragen bearbeiten.

Änderungen in 21.10.1

Die Version v21.10.0 weist ein Problem auf, das dazu führen kann, dass der Trident Controller in den Zustand CrashLoopBackOff gerät, wenn ein Knoten aus dem Kubernetes-Cluster entfernt und anschließend wieder hinzugefügt wird. Dieses Problem wurde in Version 21.10.1 behoben (GitHub-Problem 669).

Fehlerbehebungen

- Ein potenzieller Race Condition beim Importieren eines Volumes auf einem GCP CVS-Backend, der zu einem Importfehler führte, wurde behoben.
- Es wurde ein Problem behoben, das dazu führen konnte, dass der Trident -Controller in den Zustand CrashLoopBackOff geriet, wenn ein Knoten aus dem Kubernetes-Cluster entfernt und anschließend wieder hinzugefügt wurde (GitHub-Problem 669).

- Problem behoben, bei dem SVMs nicht mehr gefunden wurden, wenn kein SVM-Name angegeben wurde (GitHub-Problem 612).

Änderungen in 21.10.0

Fehlerbehebungen

- Problem behoben, bei dem Klone von XFS-Volumes nicht auf demselben Knoten wie das Quell-Volume eingebunden werden konnten (GitHub-Problem 514).
- Problem behoben, bei dem Trident beim Herunterfahren einen schwerwiegenden Fehler protokollierte (GitHub-Problem 597).
- Kubernetes-bezogene Korrekturen:
  - Beim Erstellen von Snapshots wird der belegte Speicherplatz eines Volumes als minimale restoreSize zurückgegeben. `ontap-nas` Und `ontap-nas-flexgroup` Treiber (GitHub-Problem 645).
  - Problem behoben, bei dem `Failed to expand filesystem` Nach der Größenänderung des Volumes wurde ein Fehler protokolliert (GitHub-Problem 560).
  - Problem behoben, bei dem eine Kapsel stecken bleiben konnte `Terminating` Status (GitHub-Problem 572).
  - Der Fall, in dem ein `ontap-san-economy` FlexVol könnte mit Snapshot-LUNs überladen sein (GitHub-Problem 533).
  - Problem mit dem benutzerdefinierten YAML-Installer bei Verwendung eines anderen Images behoben (GitHub-Problem 613).
  - Korrektur der Berechnung der Snapshot-Größe (GitHub-Problem 611).
  - Problem behoben, bei dem alle Trident Installationsprogramme normales Kubernetes als OpenShift identifizieren konnten (GitHub-Problem 639).
  - Der Trident -Operator wurde so korrigiert, dass die Abgleichung gestoppt wird, wenn der Kubernetes-API-Server nicht erreichbar ist (GitHub-Problem 599).

Verbesserungen

- Unterstützung hinzugefügt für `unixPermissions` Option zu GCP-CVS Performance-Volumes.
- Unterstützung für skalierungsoptimierte CVS-Volumina in GCP im Bereich von 600 GiB bis 1 TiB hinzugefügt.
- Kubernetes-bezogene Verbesserungen:
  - Unterstützung für Kubernetes 1.22 hinzugefügt.
  - Der Trident -Operator und das Helm-Chart wurden für Kubernetes 1.22 aktiviert (GitHub-Problem 628).
  - Bedienerbild hinzugefügt `tridentctl` images-Befehl (GitHub-Problem 570).

Experimentelle Erweiterungen

- Unterstützung für die Volumenreplikation hinzugefügt in `ontap-san` Treiber.
- **Tech Preview** REST-Unterstützung für die `ontap-nas-flexgroup` , `ontap-san` , Und `ontap-nas-economy` Fahrer.

Bekannte Probleme

Bekannte Probleme kennzeichnen Schwierigkeiten, die eine erfolgreiche Nutzung des Produkts verhindern könnten.

- Beim Upgrade eines Kubernetes-Clusters von Version 1.24 auf 1.25 oder höher, auf dem Trident installiert ist, muss die Datei values.yaml aktualisiert werden, um Folgendes festzulegen: `excludePodSecurityPolicy` Zu `true` oder hinzufügen `--set excludePodSecurityPolicy=true` zum `helm upgrade` Führen Sie diesen Befehl aus, bevor Sie den Cluster aktualisieren können.
- Trident erzwingt nun eine leere `fsType` (`fsType=""`) für Bände, die nicht die `fsType` in ihrer Speicherklasse angegeben. Bei der Arbeit mit Kubernetes 1.17 oder höher unterstützt Trident die Bereitstellung eines leeren `fsType` für NFS-Volumes. Für iSCSI-Volumes müssen Sie die folgenden Einstellungen vornehmen: `fsType` bei Ihrer StorageClass, wenn Sie eine erzwingen `fsGroup` unter Verwendung eines Sicherheitskontexts.
- Bei der Verwendung eines Backends über mehrere Trident -Instanzen hinweg sollte jede Backend-Konfigurationsdatei eine andere Konfigurationsdatei enthalten. `storagePrefix` Wert für ONTAP Backends oder verwenden Sie einen anderen `TenantName` für SolidFire -Backends. Trident kann keine Volumes erkennen, die von anderen Trident -Instanzen erstellt wurden. Der Versuch, ein bestehendes Volume auf einem ONTAP oder SolidFire -Backend zu erstellen, gelingt, da Trident die Volume-Erstellung als idempotente Operation behandelt. Wenn `storagePrefix` oder `TenantName` Wenn sie sich nicht unterscheiden, kann es zu Namenskonflikten bei Volumes kommen, die auf demselben Backend erstellt wurden.
- Bei der Installation von Trident (mit `tridentctl` oder dem Trident -Operator) und unter Verwendung `tridentctl` Um Trident zu verwalten, sollten Sie Folgendes sicherstellen: `KUBECONFIG` Die Umgebungsvariable ist gesetzt. Dies ist notwendig, um den Kubernetes-Cluster anzugeben, der `tridentctl` sollte entgegenwirken. Bei der Arbeit mit mehreren Kubernetes-Umgebungen sollten Sie sicherstellen, dass die `KUBECONFIG` Die Datei wurde korrekt bezogen.
- Um die Online-Speicherplatzfreigabe für iSCSI-PVs durchzuführen, benötigt das zugrunde liegende Betriebssystem auf dem Worker-Knoten möglicherweise Mount-Optionen, die an das Volume übergeben werden. Dies gilt für RHEL/Red Hat Enterprise Linux CoreOS (RHCOS)-Instanzen, die Folgendes erfordern: `discard` "Montageoption" Stellen Sie sicher, dass die Option zum Verwerfen der Mount-Option in Ihrer Datei enthalten ist.[`StorageClass` ^] zur Unterstützung des Online-Blockverwerfens.
- Wenn Sie mehr als eine Trident -Instanz pro Kubernetes-Cluster haben, kann Trident nicht mit anderen Instanzen kommunizieren und kann andere von ihnen erstellte Volumes nicht erkennen, was zu unerwartetem und fehlerhaftem Verhalten führt, wenn mehr als eine Instanz innerhalb eines Clusters ausgeführt wird. Pro Kubernetes-Cluster sollte nur eine Trident -Instanz vorhanden sein.
- Wenn Trident-basiert `StorageClass` Objekte werden aus Kubernetes gelöscht, während Trident offline ist. Trident entfernt jedoch nicht die entsprechenden Speicherklassen aus seiner Datenbank, wenn es wieder online geht. Sie sollten diese Speicherklassen löschen mit `tridentctl` oder die REST-API.
- Wenn ein Benutzer ein von Trident bereitgestelltes PV löscht, bevor er das zugehörige PVC löscht, löscht Trident das zugrunde liegende Volume nicht automatisch. Sie sollten das Volumen entfernen über `tridentctl` oder die REST-API.
- ONTAP kann nicht gleichzeitig mehr als eine FlexGroup bereitstellen, es sei denn, die Menge der Aggregate ist für jede Bereitstellungsanforderung eindeutig.
- Bei der Verwendung von Trident über IPv6 sollten Sie Folgendes angeben: `managementLIF` Und `dataLIF` in der Backend-Definition innerhalb eckiger Klammern. Zum Beispiel,`[fd20:8b1e:b258:2000:f816:3eff:feec:0]` .

Sie können nicht angeben `dataLIF` auf einem ONTAP SAN-Backend. Trident erkennt alle verfügbaren iSCSI LIFs und nutzt diese, um die Multipath-Sitzung herzustellen.

- Bei Verwendung der `solidfire-san` Treiber mit OpenShift 4.5, stellen Sie sicher, dass die zugrunde liegenden Worker-Knoten MD5 als CHAP-Authentifizierungsalgorithmus verwenden. Mit Element 12.7 stehen sichere, FIPS-konforme CHAP-Algorithmen wie SHA1, SHA-256 und SHA3-256 zur Verfügung.

Weitere Informationen

- "Trident GitHub"
- "Trident -Blogs"

Frühere Versionen der Dokumentation

Falls Sie nicht Trident 25.06 verwenden, ist die Dokumentation für frühere Versionen verfügbar."Trident Support-Lebenszyklus" .

- "Trident 25.02"
- "Trident 24.10"
- "Trident 24.06"
- "Trident 24.02"
- "Trident 23.10"
- "Trident 23.07"
- "Trident 23.04"
- "Trident 23.01"
- "Trident 22.10"

Bekannte Probleme

Bekannte Probleme identifizieren Probleme, die Sie möglicherweise daran hindern, diese Version des Produkts erfolgreich zu verwenden.

Folgende bekannte Probleme betreffen die aktuelle Version:

Das Wiederherstellen von Restic-Backups großer Dateien kann fehlschlagen

Beim Wiederherstellen von 30 GB oder größeren Dateien aus einem Amazon S3-Backup, das mit Restic erstellt wurde, kann der Wiederherstellungsvorgang fehlschlagen. Als Ausweichlösung können Sie die Daten mit Kopia als Datenmigrationsprogramm sichern (Kopia ist das Standardprogramm für Datenmigration bei Backups). Siehe "Schützen Sie Ihre Anwendungen mit Trident Protect." für Anweisungen.

Erste Schritte

Erfahren Sie mehr über Trident

Erfahren Sie mehr über Trident

Trident ist ein vollständig unterstütztes Open-Source-Projekt, das von NetApp betreut wird. Es wurde entwickelt, um Ihnen dabei zu helfen, die Persistenzanforderungen Ihrer containerisierten Anwendung mithilfe von branchenüblichen Schnittstellen wie dem Container Storage Interface (CSI) zu erfüllen.

Was ist Trident?

NetApp Trident ermöglicht die Nutzung und Verwaltung von Speicherressourcen über alle gängigen NetApp Speicherplattformen hinweg, sowohl in der Public Cloud als auch On-Premises, einschließlich On-Premises ONTAP Cluster (AFF, FAS und ASA), ONTAP Select, Cloud Volumes ONTAP, Element-Software (NetApp HCI, SolidFire), Azure NetApp Files, Amazon FSx for NetApp ONTAP und Cloud Volumes Service auf Google Cloud.

Trident ist ein Container Storage Interface (CSI)-konformer dynamischer Speicherorchestrator, der sich nativ integrieren lässt mit"Kubernetes" . Trident läuft als einzelner Controller-Pod plus ein Node-Pod auf jedem Worker-Knoten im Cluster. Siehe "Trident -Architektur" für Details.

Trident bietet außerdem eine direkte Integration mit dem Docker-Ökosystem für NetApp Speicherplattformen. Das NetApp Docker Volume Plugin (nDVP) unterstützt die Bereitstellung und Verwaltung von Speicherressourcen von der Speicherplattform zu Docker-Hosts. Siehe "Trident für Docker bereitstellen" für Details.

> Wenn Sie Kubernetes zum ersten Mal verwenden, sollten Sie sich mit den folgenden Grundlagen vertraut machen:"Kubernetes-Konzepte und -Werkzeuge" .

Kubernetes-Integration mit NetApp -Produkten

Das NetApp Portfolio an Speicherprodukten integriert sich in viele Aspekte eines Kubernetes-Clusters und bietet fortschrittliche Datenverwaltungsfunktionen, die die Funktionalität, Leistungsfähigkeit, Performance und Verfügbarkeit der Kubernetes-Bereitstellung verbessern.

Amazon FSx for NetApp ONTAP

"Amazon FSx for NetApp ONTAP"ist ein vollständig verwalteter AWS-Service, mit dem Sie Dateisysteme starten und ausführen können, die auf dem Speicherbetriebssystem NetApp ONTAP basieren.

Azure NetApp Files

"Azure NetApp Files"ist ein Azure-Dateifreigabedienst der Enterprise-Klasse, der von NetApp bereitgestellt wird. Sie können Ihre anspruchsvollsten dateibasierten Workloads nativ in Azure ausführen und dabei die Leistung und das umfassende Datenmanagement nutzen, die Sie von NetApp erwarten.

Cloud Volumes ONTAP

"Cloud Volumes ONTAP"ist ein rein softwarebasiertes Speichersystem, auf dem die Datenverwaltungssoftware ONTAP in der Cloud ausgeführt wird.

Google Cloud NetApp Volumes

"Google Cloud NetApp Volumes"ist ein vollständig verwalteter Dateispeicherdienst in Google Cloud, der hochleistungsfähigen Dateispeicher auf Unternehmensebene bietet.

Element Software

"Element"ermöglicht es dem Speicheradministrator, Arbeitslasten zu konsolidieren, indem die Leistung gewährleistet und ein vereinfachter und optimierter Speicherbedarf ermöglicht wird.

NetApp HCI

"NetApp HCI"vereinfacht die Verwaltung und Skalierung des Rechenzentrums durch die Automatisierung von Routineaufgaben und ermöglicht es Infrastrukturadministratoren, sich auf wichtigere Funktionen zu konzentrieren.

Trident kann Speichergeräte für containerisierte Anwendungen direkt auf der zugrunde liegenden NetApp HCI -Speicherplattform bereitstellen und verwalten.

NetApp ONTAP

"NetApp ONTAP"ist das Multiprotokoll-Betriebssystem von NetApp für einheitliche Speichersysteme, das fortschrittliche Datenverwaltungsfunktionen für jede Anwendung bietet.

ONTAP Systeme verfügen über All-Flash-, Hybrid- oder All-HDD-Konfigurationen und bieten viele verschiedene Bereitstellungsmodelle: lokale FAS, AFA- und ASA Cluster, ONTAP Select und Cloud Volumes ONTAP. Trident unterstützt diese ONTAP Bereitstellungsmodelle.

Trident -Architektur

Trident läuft als einzelner Controller-Pod plus ein Node-Pod auf jedem Worker-Knoten im Cluster. Der Node-Pod muss auf jedem Host ausgeführt werden, auf dem Sie möglicherweise ein Trident -Volume einbinden möchten.

Controller-Pods und Node-Pods verstehen

Trident wird als einzelnes System eingesetztTrident Controller Pod und ein oder mehrereTrident Knotenkapseln auf dem Kubernetes-Cluster und verwendet standardmäßige Kubernetes *CSI Sidecar Container*, um die Bereitstellung von CSI-Plugins zu vereinfachen. "Kubernetes CSI Sidecar-Container" werden von der Kubernetes Storage-Community gepflegt.

Kubernetes"Knotenselektoren" Und"Toleranzen und Verunreinigungen" werden verwendet, um einen Pod so einzuschränken, dass er auf einem bestimmten oder bevorzugten Knoten ausgeführt wird. Sie können

während der Trident Installation Knotenselektoren und Toleranzen für Controller- und Knotenpods konfigurieren.

- Das Controller-Plugin übernimmt die Bereitstellung und Verwaltung von Volumes, wie z. B. Snapshots und Größenänderungen.
- Das Node-Plugin kümmert sich um die Anbindung des Speichers an den Knoten.

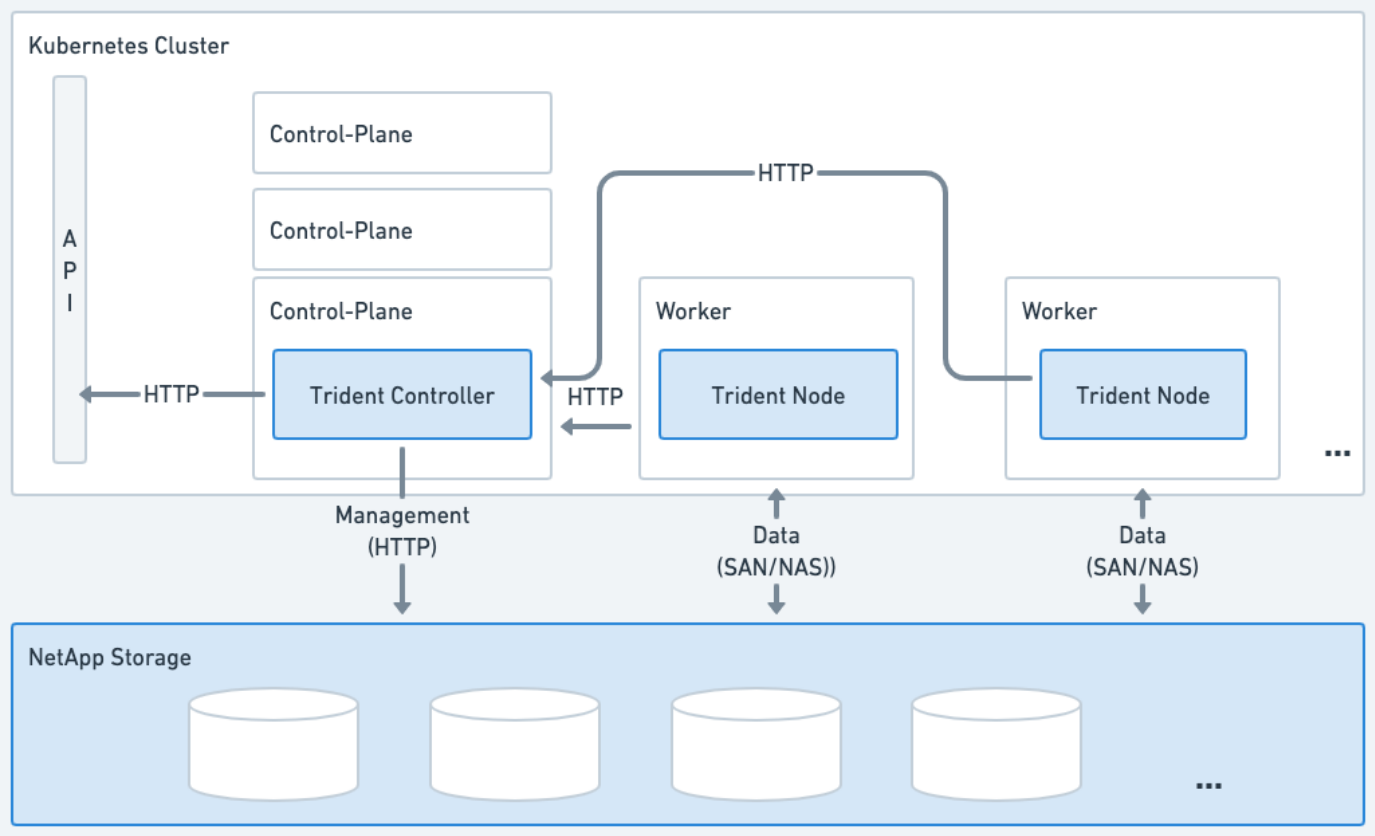


Abbildung 1. Trident wurde auf dem Kubernetes-Cluster bereitgestellt.

Trident Controller Pod

Der Trident Controller Pod ist ein einzelner Pod, auf dem das CSI Controller-Plugin ausgeführt wird.

- Verantwortlich für die Bereitstellung und Verwaltung von Volumes im NetApp -Speicher
- Verwaltet durch eine Kubernetes-Bereitstellung
- Kann je nach Installationsparametern auf der Steuerungsebene oder auf den Arbeitsknoten ausgeführt werden.

Abbildung 2. Diagramm der Trident -Controller-Kapsel

Trident Knotenkapseln

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

- Verantwortlich für das Ein- und Aushängen von Speichermedien für Pods, die auf dem Host ausgeführt werden.
- Verwaltet von einem Kubernetes-DaemonSet
- Muss auf jedem Knoten ausgeführt werden, der NetApp -Speicher einbinden kann.

Abbildung 3. Diagramm des Trident

Unterstützte Kubernetes-Clusterarchitekturen

Trident wird von den folgenden Kubernetes-Architekturen unterstützt:

Kubernetes-Clusterarchitekturen	Unterstützt	Standardinstallation
Einzelner Master, Berechnung	Ja	Ja
Mehrere Master, Berechnung	Ja	Ja
Master, `etcd` , berechnen	Ja	Ja
Master, Infrastruktur, Rechner	Ja	Ja

Konzepte

Bereitstellung

Die Bereitstellung in Trident umfasst zwei Hauptphasen. In der ersten Phase wird eine Speicherklasse mit den passenden Backend-Speicherpools verknüpft. Dies erfolgt als notwendige Vorbereitung vor der Bereitstellung. Die zweite Phase umfasst die eigentliche Volume-Erstellung und erfordert die Auswahl eines Speicherpools aus denjenigen, die der Speicherklasse des ausstehenden Volumes zugeordnet sind.

Speicherklassenzuordnung

Die Zuordnung von Backend-Speicherpools zu einer Speicherklasse basiert sowohl auf den angeforderten Attributen der Speicherklasse als auch auf deren `storagePools` , `additionalStoragePools` , Und `excludeStoragePools` Listen. Wenn Sie eine Speicherklasse erstellen, vergleicht Trident die von jedem seiner Backends angebotenen Attribute und Pools mit denjenigen, die von der Speicherklasse angefordert werden. Wenn die Attribute und der Name eines Speicherpools mit allen angeforderten Attributen und Poolnamen übereinstimmen, fügt Trident diesen Speicherpool der Menge der geeigneten Speicherpools für diese Speicherklasse hinzu. Darüber hinaus fügt Trident alle im Verzeichnis aufgeführten Speicherpools hinzu. `additionalStoragePools` Fügen Sie die Elemente dieser Liste hinzu, auch wenn deren Attribute nicht alle oder keine der angeforderten Attribute der Speicherklasse erfüllen. Sie sollten die `excludeStoragePools` Liste zum Überschreiben und Entfernen von Speicherpools für eine Speicherklasse. Trident führt bei jedem Hinzufügen eines neuen Backends einen ähnlichen Prozess durch, indem es prüft, ob dessen Speicherpools denen der vorhandenen Speicherklassen entsprechen, und alle als ausgeschlossen markierten entfernt.

Volumenerzeugung

Trident verwendet dann die Verknüpfungen zwischen Speicherklassen und Speicherpools, um zu bestimmen, wo Datenträger bereitgestellt werden sollen. Wenn Sie ein Volume erstellen, ermittelt Trident zunächst die Speicherpools für die Speicherklasse dieses Volumes. Wenn Sie ein Protokoll für das Volume angeben, entfernt Trident anschließend diejenigen Speicherpools, die das angeforderte Protokoll nicht bereitstellen können (beispielsweise kann ein NetApp HCI/ SolidFire Backend kein dateibasiertes Volume bereitstellen, während ein ONTAP NAS-Backend kein blockbasiertes Volume bereitstellen kann). Trident randomisiert die Reihenfolge dieser resultierenden Menge, um eine gleichmäßige Verteilung der Volumes zu ermöglichen, und durchläuft sie dann, um das Volume nacheinander auf jedem Speicherpool bereitzustellen. Wenn ein Test

erfolgreich ist, gibt die Funktion eine positive Rückmeldung und protokolliert dabei alle aufgetretenen Fehler. Trident gibt nur dann einen Fehler zurück, wenn die Bereitstellung auf **allen** verfügbaren Speicherpools für die angeforderte Speicherklasse und das Protokoll fehlschlägt.

Volumen-Snapshots

Erfahren Sie mehr darüber, wie Trident die Erstellung von Volume-Snapshots für seine Treiber handhabt.

Erfahren Sie mehr über die Erstellung von Volume-Snapshots.

- Für die `ontap-nas` , `ontap-san` , `gcp-cvs` , Und `azure-netapp-files` Treiber, jedes Persistent Volume (PV) wird einem FlexVol volume zugeordnet. Dadurch werden Volume-Snapshots als NetApp Snapshots erstellt. Die Snapshot-Technologie von NetApp bietet mehr Stabilität, Skalierbarkeit, Wiederherstellbarkeit und Leistung als konkurrierende Snapshot-Technologien. Diese Snapshot-Kopien sind sowohl hinsichtlich der Erstellungszeit als auch des Speicherplatzes äußerst effizient.
- Für die `ontap-nas-flexgroup` Jeder Persistent Volume (PV) wird vom Treiber einer FlexGroup zugeordnet. Daher werden Volume-Snapshots als NetApp FlexGroup Snapshots erstellt. Die Snapshot-Technologie von NetApp bietet mehr Stabilität, Skalierbarkeit, Wiederherstellbarkeit und Leistung als konkurrierende Snapshot-Technologien. Diese Snapshot-Kopien sind sowohl hinsichtlich der Erstellungszeit als auch des Speicherplatzes äußerst effizient.
- Für die `ontap-san-economy` Treiber, PVs werden LUNs zugeordnet, die auf gemeinsam genutzten FlexVol -Volumes erstellt wurden. VolumeSnapshots von PVs werden durch die Durchführung von FlexClones der zugehörigen LUN erreicht. Die ONTAP FlexClone -Technologie ermöglicht es, Kopien selbst größter Datensätze nahezu augenblicklich zu erstellen. Kopien teilen sich Datenblöcke mit ihren Elternkopien und verbrauchen dabei keinen Speicherplatz außer dem für Metadaten erforderlichen.
- Für die `solidfire-san` Der Treiber ordnet jedem PV eine LUN zu, die auf der NetApp Element Software/ dem NetApp HCI Cluster erstellt wurde. VolumeSnapshots werden durch Element-Snapshots der zugrunde liegenden LUN dargestellt. Diese Momentaufnahmen sind Momentaufnahmen und beanspruchen nur wenig Systemressourcen und Speicherplatz.
- Bei der Arbeit mit dem `ontap-nas` Und `ontap-san` Treiber, ONTAP Snapshots sind Momentaufnahmen des FlexVol und belegen Speicherplatz auf dem FlexVol selbst. Dies kann dazu führen, dass sich der beschreibbare Speicherplatz im Volume mit der Zeit verringert, da Snapshots erstellt/geplant werden. Eine einfache Möglichkeit, dieses Problem zu lösen, besteht darin, das Volumen durch Größenänderung über Kubernetes zu vergrößern. Eine weitere Möglichkeit besteht darin, nicht mehr benötigte Snapshots zu löschen. Wenn ein über Kubernetes erstellter VolumeSnapshot gelöscht wird, löscht Trident den zugehörigen ONTAP Snapshot. Auch ONTAP Snapshots, die nicht über Kubernetes erstellt wurden, können gelöscht werden.

Mit Trident können Sie VolumeSnapshots verwenden, um daraus neue PVs zu erstellen. Die Erstellung von PVs aus diesen Snapshots erfolgt mithilfe der FlexClone -Technologie für unterstützte ONTAP und CVS-Backends. Beim Erstellen eines PV aus einem Snapshot ist das zugrundeliegende Volume ein FlexClone des übergeordneten Volumes des Snapshots. Der `solidfire-san` Der Treiber verwendet Element Software-Volume-Klone, um PVs aus Snapshots zu erstellen. Hier wird ein Klon aus dem Element-Snapshot erstellt.

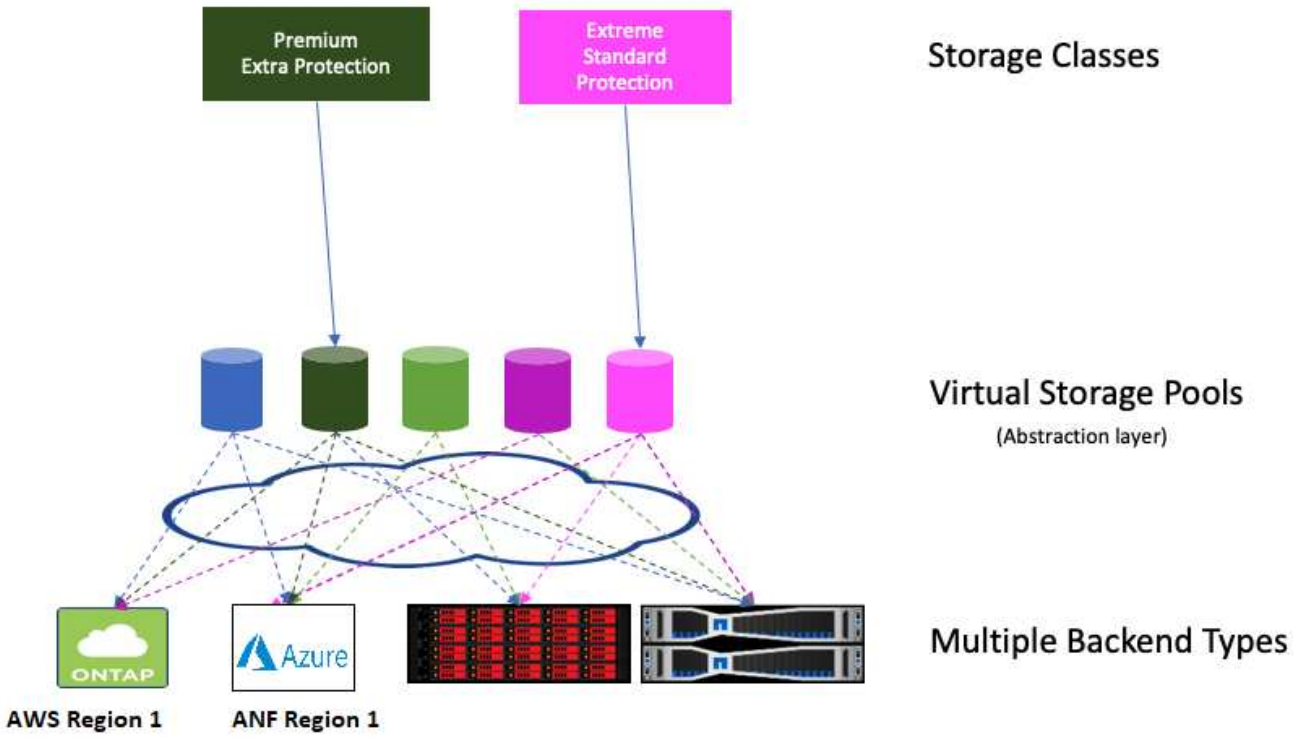
Virtuelle Pools

Virtuelle Pools bieten eine Abstraktionsschicht zwischen Trident -Speicher-Backends und Kubernetes. `StorageClasses` . Sie ermöglichen es einem Administrator, Aspekte wie Standort, Leistung und Schutz für jedes Backend auf eine gemeinsame, backendunabhängige Weise zu definieren, ohne eine `StorageClass` Geben Sie an,

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

Erfahren Sie mehr über virtuelle Pools

Der Speicheradministrator kann virtuelle Pools auf jedem der Trident -Backends in einer JSON- oder YAML-Definitionsdatei definieren.



Alle Aspekte, die außerhalb der Liste der virtuellen Pools angegeben werden, sind global für das Backend und gelten für alle virtuellen Pools, während jeder virtuelle Pool einen oder mehrere Aspekte individuell angeben kann (wodurch alle globalen Aspekte des Backends überschrieben werden).

- Beim Definieren virtueller Pools sollte die Reihenfolge vorhandener virtueller Pools in einer Backend-Definition nicht verändert werden.
- Wir raten davon ab, Attribute eines bestehenden virtuellen Pools zu ändern. Sie sollten einen neuen virtuellen Pool definieren, um Änderungen vorzunehmen.

Die meisten Aspekte werden in Backend-spezifischen Begriffen spezifiziert. Entscheidend ist, dass die Aspektwerte außerhalb des Backend-Treibers nicht zugänglich sind und nicht für den Abgleich zur Verfügung stehen. `StorageClasses` Stattdessen definiert der Administrator für jeden virtuellen Pool eine oder mehrere Bezeichnungen. Jedes Label ist ein Schlüssel-Wert-Paar, und Labels können über verschiedene Backends hinweg übereinstimmen. Ähnlich wie Aspekte können Labels pro Pool oder global für das Backend festgelegt werden. Im Gegensatz zu Aspekten, die über vordefinierte Namen und Werte verfügen, hat der Administrator bei der Definition von Bezeichnungsschlüsseln und -werten nach Bedarf die volle Entscheidungsfreiheit. Zur Vereinfachung können Speicheradministratoren Bezeichnungen pro virtuellem Pool definieren und Volumes nach Bezeichnung gruppieren.

Die Bezeichnungen des virtuellen Pools können mit diesen Zeichen definiert werden:

- Großbuchstaben `A-Z`
- Kleinbuchstaben `a-z`
- Zahlen `0-9`
- Unterstriche _
- Bindestriche -

A `StorageClass` Identifiziert den zu verwendenden virtuellen Pool durch Bezugnahme auf die Bezeichnungen innerhalb eines Selektorparameters. Virtuelle Poolselektoren unterstützen die folgenden Operatoren:

Operator	Beispiel	Der Labelwert eines Pools muss Folgendes aufweisen:
`=`	Leistung = Premium	Übereinstimmen
`!=`	Leistung!=extrem	Nicht übereinstimmend
`in`	Lage in (Osten, Westen)	Sei Teil der Wertemenge
`notin`	Leistung Notin (Silber, Bronze)	nicht in der Wertemenge enthalten sein
`<key>`	Schutz	Existieren Sie mit jedem beliebigen Wert
`!<key>`	!Schutz	Existiert nicht

Volumenzugriffsgruppen

Erfahren Sie mehr darüber, wie Trident verwendet "Volumenzugriffsgruppen" .

> 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 erweiterter CSI-Provisioner installiert wird.

Erfahren Sie mehr über Volumenzugriffsgruppen

Trident kann Volume-Zugriffsgruppen verwenden, um den Zugriff auf die von ihm bereitgestellten Volumes zu steuern. Wenn CHAP deaktiviert ist, erwartet es, eine Zugriffsgruppe namens „…“ zu finden. `trident` es sei denn, Sie geben in der Konfiguration eine oder mehrere Zugriffsgruppen-IDs an.

Während Trident neue Volumes den konfigurierten Zugriffsgruppen zuordnet, erstellt oder verwaltet es selbst keine Zugriffsgruppen. Die Zugriffsgruppen müssen existieren, bevor das Speicher-Backend zu Trident hinzugefügt wird, und sie müssen die iSCSI-IQNs von jedem Knoten im Kubernetes-Cluster enthalten, der potenziell die von diesem Backend bereitgestellten Volumes einbinden könnte. In den meisten Installationen umfasst dies jeden Worker-Knoten im Cluster.

Für Kubernetes-Cluster mit mehr als 64 Knoten sollten Sie mehrere Zugriffsgruppen verwenden. Jede Zugriffsgruppe kann bis zu 64 IQNs enthalten, und jedes Volume kann zu vier Zugriffsgruppen gehören. Bei der Konfiguration von maximal vier Zugriffsgruppen kann jeder Knoten in einem Cluster mit bis zu 256 Knoten auf jedes beliebige Volume zugreifen. Die aktuellsten Beschränkungen für Datenträgerzugriffsgruppen finden Sie unter "hier," .

Wenn Sie die Konfiguration von einer Konfiguration ändern, die die Standardeinstellungen verwendet.

`trident` Zugriffsgruppe auf eine Gruppe, die auch andere verwendet, einschließlich der ID für die `trident` Zugriffsgruppe in der Liste.

Schnellstart für Trident

Sie können Trident in wenigen Schritten installieren und mit der Verwaltung von Speicherressourcen beginnen. Bevor Sie beginnen, überprüfen Sie Folgendes:"Trident Anforderungen" .

Informationen zu Docker finden Sie unter"Trident für Docker" .

1 Bereiten Sie den Worker-Knoten vor.

Alle Worker-Knoten im Kubernetes-Cluster müssen in der Lage sein, die für Ihre Pods bereitgestellten Volumes einzubinden.

"Bereiten Sie den Worker-Knoten vor."

2 Installieren Sie Trident

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

"Trident installieren"

3 Erstelle ein Backend

Ein Backend definiert die Beziehung zwischen Trident und einem Speichersystem. Es teilt Trident mit, wie mit diesem Speichersystem kommuniziert werden soll und wie Trident Datenträger daraus bereitstellen soll.

"Backend konfigurieren"für Ihr Speichersystem

4 Erstellen einer Kubernetes-Speicherklasse

Das Kubernetes StorageClass-Objekt legt Trident als Provisionierungsdienst fest und ermöglicht es Ihnen, eine Speicherklasse zu erstellen, um Volumes mit anpassbaren Attributen bereitzustellen. Trident erstellt eine passende Speicherklasse für Kubernetes-Objekte, die den Trident Provisioner angeben.

"Erstellen einer Speicherklasse"

5 Bereitstellung eines Volumens

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

Erstellen Sie ein PersistentVolume (PV) und einen PersistentVolumeClaim (PVC), der die konfigurierte Kubernetes StorageClass verwendet, um Zugriff auf das PV anzufordern. Anschließend können Sie die PV-

Anlage an einem Pod montieren.

"Bereitstellung eines Volumens"

Wie geht es weiter?

Sie können nun zusätzliche Backends hinzufügen, Speicherklassen verwalten, Backends verwalten und Volumenoperationen durchführen.

Anforderungen

Vor der Installation von Trident sollten Sie diese allgemeinen Systemvoraussetzungen überprüfen. Bestimmte Backends können zusätzliche Anforderungen haben.

Wichtige Informationen zu Trident

Sie müssen die folgenden wichtigen Informationen über Trident unbedingt lesen.

Wichtige Informationen zu Trident

- Kubernetes 1.34 wird jetzt in Trident unterstützt. Aktualisieren Sie Trident , bevor Sie Kubernetes aktualisieren.
- Trident setzt die Verwendung von Multipathing-Konfigurationen in SAN-Umgebungen strikt durch, mit einem empfohlenen Wert von `find_multipaths: no` in der Datei multipath.conf.

  Verwendung einer Nicht-Multipathing-Konfiguration oder Verwendung von `find_multipaths: yes` oder `find_multipaths: smart` Ein falscher Wert in der Datei multipath.conf führt zu Mount-Fehlern. Trident hat die Verwendung von `find_multipaths: no` seit der Version vom 21.07.

Unterstützte Frontends (Orchestratoren)

Trident unterstützt mehrere Container-Engines und Orchestratoren, darunter die folgenden:

- Anthos On-Prem (VMware) und Anthos auf Bare Metal 1.16
- Kubernetes 1.27 – 1.34
- OpenShift 4.12, 4.14 – 4.19 (Wenn Sie die iSCSI-Knotenvorbereitung mit OpenShift 4.19 verwenden möchten, wird mindestens die Trident Version 25.06.1 unterstützt.)

  Trident unterstützt weiterhin ältere OpenShift-Versionen in Übereinstimmung mit"Red Hat Extended Update Support (EUS) Release-Lebenszyklus" selbst wenn sie auf Kubernetes-Versionen angewiesen sind, die vom Upstream nicht mehr offiziell unterstützt werden. In solchen Fällen können Sie bei der Installation von Trident alle Warnmeldungen zur Kubernetes-Version getrost ignorieren.

- Rancher Kubernetes Engine 2 (RKE2) v1.27.x - 1.34.x

Während Trident auf Rancher Kubernetes Engine 2 (RKE2) Versionen 1.27.x – 1.34.x unterstützt wird, ist Trident derzeit nur auf RKE2 v1.28.5+rke2r1 qualifiziert.

Trident arbeitet außerdem mit einer Vielzahl anderer vollständig verwalteter und selbstverwalteter Kubernetes-Angebote zusammen, darunter 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 verwendet werden für"KubeVirt" .

Bevor Sie einen Kubernetes-Cluster von Version 1.25 auf 1.26 oder höher aktualisieren, auf dem Trident installiert ist, lesen Sie bitte den folgenden Abschnitt:"Aktualisieren einer Helm-Installation" .

Unterstützte Backends (Speicher)

Um Trident nutzen zu können, benötigen Sie eines oder mehrere der folgenden unterstützten Backends:

- Amazon FSx for NetApp ONTAP
- Azure NetApp Files
- Cloud Volumes ONTAP
- Google Cloud NetApp Volumes
- NetApp All SAN Array (ASA)
- Vor-Ort-Clusterversionen FAS, AFF, Select oder ASA r2 (iSCSI und NVMe/TCP) mit eingeschränktem Support von NetApp. Sehen"Softwareversionsunterstützung" .
- NetApp HCI/ Element Software 11 oder höher

Trident Unterstützung für KubeVirt- und OpenShift-Virtualisierung

Unterstützte Speichertreiber:

Trident unterstützt die folgenden ONTAP -Treiber für KubeVirt und OpenShift Virtualisierung:

- ontap-nas
- ontap-nas-economy
- ontap-san (iSCSI, FCP, NVMe über TCP)
- ontap-san-economy (nur iSCSI)

Zu beachtende Punkte:

- Aktualisieren Sie die Speicherklasse, sodass sie Folgendes enthält: `fsType` Parameter (zum Beispiel: `fsType: "ext4"`) in der OpenShift Virtualisierungsumgebung. Stellen Sie bei Bedarf den Lautstärkemodus explizit auf Blockieren ein, indem Sie die `volumeMode=Block` Parameter im `dataVolumeTemplates` CDI anzuweisen, Block-Datenvolumes zu erstellen.
- *RWX-Zugriffsmodus für Block-Speichertreiber*: Die Treiber ontap-san (iSCSI, NVMe/TCP, FC) und ontap-san-economy (iSCSI) werden nur mit "volumeMode: Block" (rohes Gerät) unterstützt. Für diese Fahrer gilt: `fstype` Dieser Parameter kann nicht verwendet werden, da die Volumes im Rohgerätemodus bereitgestellt werden.
- Für Live-Migrations-Workflows, bei denen der RWX-Zugriffsmodus erforderlich ist, werden folgende Kombinationen unterstützt:

- NFS + `volumeMode=Filesystem`
- iSCSI + `volumeMode=Block` (Rohgerät)
- NVMe/TCP + `volumeMode=Block` (Rohgerät)
- FC + `volumeMode=Block` (Rohgerät)

Funktionsanforderungen

Die folgende Tabelle fasst die in dieser Version von Trident verfügbaren Funktionen und die unterstützten Kubernetes-Versionen zusammen.

Funktion	Kubernetes-Version	Funktionsgatter erforderlich?
Trident	1,27 - 1,34	Nein
Volumen-Momentaufnahmen	1,27 - 1,34	Nein
PVC aus Volumen-Schnappschüssen	1,27 - 1,34	Nein
iSCSI PV-Größenänderung	1,27 - 1,34	Nein
ONTAP Bidirektionales CHAP	1,27 - 1,34	Nein
Dynamische Exportrichtlinien	1,27 - 1,34	Nein
Trident -Betreiber	1,27 - 1,34	Nein
CSI-Topologie	1,27 - 1,34	Nein

Getestete Host-Betriebssysteme

Obwohl Trident keine spezifischen Betriebssysteme offiziell unterstützt, ist bekannt, dass die folgenden funktionieren:

- Von der OpenShift Container Platform unterstützte Red Hat Enterprise Linux CoreOS (RHCOS)-Versionen (AMD64 und ARM64)
- RHEL 8+ (AMD64 und ARM64)

  NVMe/TCP erfordert RHEL 9 oder höher.

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

Standardmäßig läuft Trident in einem Container und kann daher auf jedem Linux-Worker ausgeführt werden. Allerdings müssen diese Worker in der Lage sein, die von Trident bereitgestellten Volumes mithilfe des Standard-NFS-Clients oder des iSCSI-Initiators einzubinden, je nachdem, welche Backends Sie verwenden.

Der `tridentctl` Das Dienstprogramm läuft auch auf allen diesen Linux-Distributionen.

Hostkonfiguration

Alle Worker-Knoten im Kubernetes-Cluster müssen in der Lage sein, die für Ihre Pods bereitgestellten Volumes einzubinden. Zur Vorbereitung der Worker-Knoten müssen Sie je nach Treiberauswahl NFS-, iSCSI- oder NVMe-Tools installieren.

"Bereiten Sie den Worker-Knoten vor."

Speichersystemkonfiguration

Trident erfordert möglicherweise Änderungen am Speichersystem, bevor eine Backend-Konfiguration es nutzen kann.

"Backends konfigurieren"

Trident Häfen

Trident benötigt für die Kommunikation Zugriff auf bestimmte Ports.

"Trident Häfen"

Container-Images und zugehörige Kubernetes-Versionen

Bei Installationen ohne Lufttrennung dient die folgende Liste als Referenz für die Container-Images, die zur Installation von Trident benötigt werden. Verwenden Sie die `tridentctl images` Befehl zum Überprüfen der Liste der benötigten Container-Images.

Für Trident 25.06.2 erforderliche Container-Images

Kubernetes-Versionen	Containerbild
v1.27.0, v1.28.0, v1.29.0, v1.30.0, v1.31.0, v1.32.0, v1.33.0, v1.34.0	• docker.io/netapp/trident:25.06.2 • docker.io/netapp/trident-autosupport:25.06 • registry.k8s.io/sig-storage/csi-provisioner:v5.2.0 • registry.k8s.io/sig-storage/csi-attacher:v4.8.1 • registry.k8s.io/sig-storage/csi-resizer:v1.13.2 • registry.k8s.io/sig-storage/csi-snapshotter:v8.2.1 • registry.k8s.io/sig-storage/csi-node-driver-registrar:v2.13.0 • docker.io/netapp/trident-operator:25.06.2 (optional)

Für Trident 25.06 erforderliche Container-Images

Kubernetes-Versionen	Containerbild
v1.27.0, v1.28.0, v1.29.0, v1.30.0, v1.31.0, v1.32.0, v1.33.0, v1.34.0	• docker.io/netapp/trident:25.06.0 • docker.io/netapp/trident-autosupport:25.06 • registry.k8s.io/sig-storage/csi-provisioner:v5.2.0 • registry.k8s.io/sig-storage/csi-attacher:v4.8.1 • registry.k8s.io/sig-storage/csi-resizer:v1.13.2 • registry.k8s.io/sig-storage/csi-snapshotter:v8.2.1 • registry.k8s.io/sig-storage/csi-node-driver-registrar:v2.13.0 • docker.io/netapp/trident-operator:25.06.0 (optional)

Trident installieren

Installation mit dem Trident Operator

Installation mit tridentctl

Installation mit einem OpenShift-zertifizierten Operator

Benutze den Trident.

Bereiten Sie den Worker-Knoten vor.

Alle Worker-Knoten im Kubernetes-Cluster müssen in der Lage sein, die für Ihre Pods bereitgestellten Volumes einzubinden. Zur Vorbereitung der Worker-Knoten müssen Sie je nach Treiberauswahl NFS-, iSCSI-, NVMe/TCP- oder FC-Tools installieren.

Die richtigen Werkzeuge auswählen

Wenn Sie eine Kombination von Treibern verwenden, sollten Sie alle für Ihre Treiber erforderlichen Tools installieren. Bei den neueren Versionen von Red Hat Enterprise Linux CoreOS (RHCOS) sind die Tools standardmäßig installiert.

NFS -Tools

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

iSCSI-Tools

"Installieren Sie die iSCSI-Tools"wenn Sie Folgendes verwenden: `ontap-san` , `ontap-san-economy` , `solidfire-san` .

NVMe-Tools

"Installieren Sie die NVMe-Tools"wenn Sie verwenden `ontap-san` für das NVMe-over-TCP-Protokoll (NVMe/TCP).

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

SCSI über FC-Tools

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

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

Zu beachtende Punkte: * SCSI over FC wird in OpenShift- und KubeVirt-Umgebungen unterstützt. * SCSI over FC wird unter Docker nicht unterstützt. * Die Selbstheilung von iSCSI ist bei SCSI über FC nicht anwendbar.

Knotendiensterkennung

Trident versucht automatisch zu erkennen, ob der Knoten iSCSI- oder NFS-Dienste ausführen kann.

> Die Knotendiensterkennung identifiziert zwar gefundene Dienste, garantiert aber nicht, dass die Dienste ordnungsgemäß konfiguriert sind. Umgekehrt bedeutet das Fehlen eines erkannten Dienstes nicht zwangsläufig, dass die Volume-Einbindung fehlschlägt.

Ereignisse überprüfen

Trident erstellt Ereignisse für den Knoten, um die gefundenen Dienste zu identifizieren. Um diese Ereignisse anzusehen, führen Sie Folgendes aus:

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

Überprüfung entdeckter Dienste

Trident identifiziert die für jeden Knoten auf dem Trident -Knoten-CR aktivierten Dienste. Um die gefundenen Dienste anzuzeigen, führen Sie Folgendes aus:

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

NFS -Volumes

Installieren Sie die NFS-Tools mithilfe der Befehle für Ihr Betriebssystem. Stellen Sie sicher, dass der NFS-Dienst beim Systemstart gestartet wird.

RHEL 8+

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

Ubuntu

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

Starten Sie Ihre Worker-Knoten nach der Installation der NFS-Tools neu, um Fehler beim Anhängen von Volumes an Container zu vermeiden.

iSCSI-Volumes

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

iSCSI-Selbstheilungsfunktionen

Bei ONTAP -Systemen führt Trident alle fünf Minuten eine iSCSI-Selbstheilung durch, um Folgendes zu erreichen:

1. **Identifizieren Sie** den gewünschten iSCSI-Sitzungsstatus und den aktuellen iSCSI-Sitzungsstatus.
2. Vergleichen Sie den gewünschten Zustand mit dem aktuellen Zustand, um notwendige Reparaturen zu ermitteln. Trident legt die Reparaturprioritäten fest und entscheidet, wann Reparaturen vorgezogen werden.
3. Führen Sie die erforderlichen Reparaturen durch, um den aktuellen iSCSI-Sitzungsstatus in den gewünschten iSCSI-Sitzungsstatus zurückzusetzen.

Protokolle der Selbstheilungsaktivitäten befinden sich in der `trident-main` Container im jeweiligen Daemonset-Pod. Um Protokolle anzuzeigen, müssen Sie Folgendes eingestellt haben: `debug` auf "wahr" während der Trident -Installation.

Die Selbstheilungsfunktionen von Trident iSCSI können dazu beitragen, Folgendes zu verhindern:

- Veraltete oder fehlerhafte iSCSI-Sitzungen, die nach einem Netzwerkverbindungsproblem auftreten können. Im Falle einer inaktiven Sitzung wartet Trident sieben Minuten, bevor es sich abmeldet, um die Verbindung mit einem Portal wiederherzustellen.

  Wenn beispielsweise CHAP-Geheimnisse auf dem Speicherkontroller rotiert wurden und die Netzwerkverbindung abbricht, könnten die alten (veralteten) CHAP-Geheimnisse erhalten bleiben. Die Selbstheilungskräfte können dies erkennen und die Sitzung automatisch wiederherstellen, um die aktualisierten CHAP-Geheimnisse anzuwenden.

- Fehlende iSCSI-Sitzungen
- Fehlende LUNs

Punkte, die Sie vor einem Upgrade von Trident beachten sollten

- Wenn nur pro-Knoten-igroups (eingeführt in 23.04+) verwendet werden, initiiert die iSCSI-Selbstheilung SCSI-Neuscans für alle Geräte im SCSI-Bus.
- Wenn nur Backend-bezogene iGroups verwendet werden (veraltet ab Version 23.04), initiiert die iSCSI-Selbstheilung erneute SCSI-Scans für exakte LUN-IDs im SCSI-Bus.
- Bei Verwendung einer Mischung aus knotenbezogenen igroups und backendbezogenen igroups initiiert die iSCSI-Selbstheilung erneute SCSI-Scans für exakte LUN-IDs im SCSI-Bus.

Installieren Sie die iSCSI-Tools

Installieren Sie die iSCSI-Tools mithilfe der Befehle für Ihr Betriebssystem.

Bevor Sie beginnen

- Jeder Knoten im Kubernetes-Cluster muss einen eindeutigen IQN haben. Dies ist eine notwendige Voraussetzung.
- Bei Verwendung von RHCOS Version 4.5 oder höher oder einer anderen RHEL-kompatiblen Linux-Distribution mit `solidfire-san` Treiber und Element OS 12.5 oder früher, stellen Sie sicher, dass der CHAP-Authentifizierungsalgorithmus auf MD5 eingestellt ist. `/etc/iscsi/iscsid.conf` Mit Element 12.7 stehen sichere, FIPS-konforme CHAP-Algorithmen wie SHA1, SHA-256 und SHA3-256 zur Verfügung.

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

- Bei Verwendung von Worker-Knoten, auf denen RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) mit iSCSI-PVs ausgeführt wird, geben Sie Folgendes an: `discard` mountOption in der StorageClass zur Durchführung der Inline-Speicherplatzfreigabe. Siehe "Red Hat-Dokumentation" .
- Stellen Sie sicher, dass Sie auf die neueste Version aktualisiert haben. `multipath-tools` .

RHEL 8+

1. Installieren Sie die folgenden Systempakete:

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

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

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

3. Scanvorgang auf manuelles Scannen 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
```

> Sicherstellen `/etc/multipath.conf` enthält `find_multipaths no` unter `defaults` .

5. Stellen Sie sicher, dass `iscsid` Und `multipathd` laufen:

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

6. Aktivieren und starten `iscsi` :

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

Ubuntu

1. Installieren Sie die folgenden Systempakete:

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

2. Prüfen Sie, ob die Open-iSCSI-Version 2.0.874-5ubuntu2.10 oder höher (für Bionic) bzw. 2.0.874-7.1ubuntu6.1 oder höher (für Focal) ist:

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

3. Scanvorgang auf manuelles Scannen 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
```

> Sicherstellen `/etc/multipath.conf` enthält `find_multipaths no` unter `defaults`.

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

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

> Für Ubuntu 18.04 müssen Sie die Zielports mit folgendem Befehl ermitteln: `iscsiadm` vor dem Start `open-iscsi` damit der iSCSI-Daemon startet. Alternativ können Sie die `iscsi` Servicestart `iscsid` automatisch.

iSCSI-Selbstheilung konfigurieren oder deaktivieren

Sie können die folgenden Trident iSCSI-Selbstheilungseinstellungen konfigurieren, um veraltete Sitzungen zu reparieren:

- **iSCSI-Selbstheilungsintervall**: Bestimmt die Häufigkeit, mit der die iSCSI-Selbstheilung aufgerufen wird (Standard: 5 Minuten). Sie können die Ausführungshäufigkeit durch Angabe einer kleineren Zahl oder die Ausführungshäufigkeit durch Angabe einer größeren Zahl konfigurieren.

Durch Setzen des iSCSI-Selbstheilungsintervalls auf 0 wird die iSCSI-Selbstheilung vollständig gestoppt. Wir empfehlen nicht, die iSCSI-Selbstheilung zu deaktivieren; sie sollte nur in bestimmten Szenarien deaktiviert werden, wenn die iSCSI-Selbstheilung nicht wie vorgesehen funktioniert oder zu Debugging-Zwecken.

- **iSCSI-Selbstheilungs-Wartezeit**: Bestimmt die Dauer, die die iSCSI-Selbstheilung wartet, bevor sie sich von einer fehlerhaften Sitzung abmeldet und versucht, sich erneut anzumelden (Standard: 7 Minuten). Sie können den Wert auf eine größere Zahl einstellen, sodass Sitzungen, die als fehlerhaft identifiziert werden, länger warten müssen, bevor sie abgemeldet werden und anschließend ein erneuter Anmeldeversuch unternommen wird, oder auf eine kleinere Zahl, um sich früher abzumelden und wieder anzumelden.

Helm

Um die iSCSI-Selbstheilungseinstellungen zu konfigurieren oder zu ändern, übergeben Sie die `iscsiSelfHealingInterval` Und `iscsiSelfHealingWaitTime` Parameter während der Helm-Installation oder des Helm-Updates.

Im folgenden Beispiel wird das iSCSI-Selbstheilungsintervall auf 3 Minuten und die Selbstheilungs-Wartezeit auf 6 Minuten eingestellt:

```
helm install trident trident-operator-100.2506.0.tgz --set
iscsiSelfHealingInterval=3m0s --set iscsiSelfHealingWaitTime=6m0s -n
trident
```

tridentctl

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

Im folgenden Beispiel wird das iSCSI-Selbstheilungsintervall auf 3 Minuten und die Selbstheilungs-Wartezeit 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.

- NVMe erfordert RHEL 9 oder höher.
- Falls die Kernelversion Ihres Kubernetes-Knotens zu alt ist oder das NVMe-Paket für Ihre Kernelversion nicht verfügbar ist, müssen Sie möglicherweise die Kernelversion Ihres Knotens auf eine Version mit dem NVMe-Paket 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 der Installation

Überprüfen Sie nach der Installation mit folgendem Befehl, ob jeder Knoten im Kubernetes-Cluster einen eindeutigen NQN besitzt:

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

Trident modifiziert die `ctrl_device_tmo` Wert, um sicherzustellen, dass NVMe den Pfad nicht aufgibt, falls die Verbindung abbricht. Diese Einstellung darf nicht geändert werden.

SCSI über FC-Volumes

Sie können nun das Fibre Channel (FC)-Protokoll mit Trident verwenden, um Speicherressourcen auf einem ONTAP -System bereitzustellen und zu verwalten.

Voraussetzungen

Konfigurieren Sie die erforderlichen Netzwerk- und Knoteneinstellungen für FC.

Netzwerkeinstellungen

1. Ermitteln Sie die WWPN der Zielschnittstellen. Siehe "Netzwerkschnittstelle anzeigen" für weitere Informationen.
2. Ermitteln Sie die WWPN für die Schnittstellen auf dem Initiator (Host).

   Siehe die entsprechenden Dienstprogramme des Host-Betriebssystems.

3. Konfigurieren Sie die Zoneneinteilung auf dem FC-Switch mithilfe der WWPNs des Hosts und des Ziels.

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

   Weitere Einzelheiten entnehmen Sie bitte der folgenden ONTAP -Dokumentation:

   - "Fibre Channel und FCoE-Zonierungsübersicht"

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

Installieren Sie die FC-Tools

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

- Bei Verwendung von Worker-Knoten, auf denen RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) mit FC-PVs ausgeführt wird, geben Sie Folgendes an: `discard` mountOption in der StorageClass zur Durchführung der Inline-Speicherplatzfreigabe. Siehe "Red Hat-Dokumentation" .

RHEL 8+

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
```

> Sicherstellen `/etc/multipath.conf` enthält `find_multipaths no` unter `defaults` .

3. Stellen Sie sicher, dass `multipathd` läuft:

```
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
```

> Sicherstellen `/etc/multipath.conf` enthält `find_multipaths no` unter `defaults` .

3. Stellen Sie sicher, dass `multipath-tools` ist aktiviert und läuft:

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

Backends konfigurieren und verwalten

Backends konfigurieren

Ein Backend definiert die Beziehung zwischen Trident und einem Speichersystem. Es teilt Trident mit, wie mit diesem Speichersystem kommuniziert werden soll und wie Trident Datenträger daraus bereitstellen soll.

Trident bietet automatisch Speicherpools von Backends an, die den Anforderungen einer Speicherklasse entsprechen. Erfahren Sie, wie Sie das Backend für Ihr Speichersystem konfigurieren.

- "Konfigurieren eines Azure NetApp Files -Backends"
- "Konfigurieren eines Google Cloud NetApp Volumes -Backends"
- "Konfigurieren eines Cloud Volumes Service für das Google Cloud Platform-Backend"
- "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"
- "Trident mit Amazon FSx for NetApp ONTAP verwenden"

Azure NetApp Files

Konfigurieren eines Azure NetApp Files -Backends

Sie können Azure NetApp Files als Backend für Trident konfigurieren. Sie können NFS- und SMB-Volumes mithilfe eines Azure NetApp Files Backends einbinden. Trident unterstützt außerdem die Verwaltung von Anmeldeinformationen mithilfe verwalteter Identitäten für Azure Kubernetes Services (AKS)-Cluster.

Details zum Azure NetApp Files Treiber

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

Treiber	Protokoll	Lautstärke modus	Unterstützte Zugriffsmodi	Unterstützte Dateisysteme
`azure-netapp-files`	NFS SMB	Dateisystem	RWO, ROX, RWX, RWOP	`nfs`, `smb`

Überlegungen

- Der Azure NetApp Files -Dienst unterstützt keine Volumes unter 50 GiB. Trident erstellt automatisch 50-GiB-Volumes, wenn ein kleineres Volume angefordert wird.
- Trident unterstützt SMB-Volumes nur, wenn sie in Pods eingebunden sind, die auf Windows-Knoten laufen.

Verwaltete Identitäten für AKS

Trident unterstützt"verwaltete Identitäten" für Azure Kubernetes Services-Cluster. Um die Vorteile der optimierten Anmeldeinformationsverwaltung durch verwaltete Identitäten nutzen zu können, benötigen Sie Folgendes:

- Ein mit AKS bereitgestellter Kubernetes-Cluster
- Auf dem AKS-Kubernetes-Cluster konfigurierte verwaltete Identitäten
- Trident installiert, das Folgendes beinhaltet: `cloudProvider` um zu spezifizieren `"Azure"` .

Trident -Betreiber

Um Trident mithilfe des Trident -Operators zu installieren, bearbeiten Sie `tridentorchestrator_cr.yaml` einstellen `cloudProvider` Zu `"Azure"` . Beispiel:

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

Helm

Das folgende Beispiel installiert Trident -Sets `cloudProvider` für Azure mithilfe der Umgebungsvariablen `$CP` :

```
helm install trident trident-operator-100.2506.0.tgz --create
-namespace --namespace <trident-namespace> --set cloudProvider=$CP
```

<code>tridentctl</code>

Das folgende Beispiel installiert Trident und konfiguriert die `cloudProvider` Flagge an `Azure` :

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

Cloud-Identität für AKS

Cloud Identity ermöglicht es Kubernetes-Pods, auf Azure-Ressourcen zuzugreifen, indem sie sich als Workload-Identität authentifizieren, anstatt explizite Azure-Anmeldeinformationen anzugeben.

Um die Vorteile der Cloud-Identität in Azure nutzen zu können, benötigen Sie Folgendes:

- Ein mit AKS bereitgestellter Kubernetes-Cluster
- Workload-Identität und OIDC-Aussteller wurden auf dem AKS Kubernetes-Cluster konfiguriert.
- Trident installiert, das Folgendes beinhaltet: `cloudProvider` um zu spezifizieren `"Azure"` Und `cloudIdentity` Angabe der Workload-Identität

Trident -Betreiber

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

Beispiel:

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

Helm

Legen Sie die Werte für die Flags **cloud-provider (CP)** und **cloud-identity (CI)** mithilfe 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 konfiguriert es. `cloudProvider` für Azure mithilfe der Umgebungsvariablen `$CP` und stellt die `cloudIdentity` unter Verwendung der Umgebungsvariablen `$CI` :

```
helm install trident trident-operator-100.6.0.tgz --set cloudProvider=$CP --set cloudIdentity="$CI"
```

<code>tridentctl</code>

Legen Sie die Werte für die Flags **Cloud-Anbieter** und **Cloud-Identität** mithilfe 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 konfiguriert die `cloud-provider` Flagge an `$CP` , Und `cloud-identity` Zu `$CI` :

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

Bereiten Sie die Konfiguration eines Azure NetApp Files -Backends vor.

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 anfängliche Konfiguration 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 "Azure NetApp Files" Im Backend benötigen Sie Folgendes:

- `subscriptionID`, `tenantID` , `clientID` , `location` , Und `clientSecret` sind optional bei der Verwendung von verwalteten Identitäten auf einem AKS-Cluster.
- `tenantID`, `clientID` , Und `clientSecret` sind optional, wenn eine Cloud-Identität auf einem AKS-Cluster verwendet wird.

- Ein Kapazitätspool. Siehe"Microsoft: Erstellen eines Kapazitätspools für Azure NetApp Files" .
- Ein an Azure NetApp Files delegiertes Subnetz. Siehe"Microsoft: Ein Subnetz an Azure NetApp Files delegieren" .
- `subscriptionID`aus 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. Für die App-Registrierung sollte eines der folgenden Verfahren verwendet werden:
  - Die Rolle des Eigentümers oder Mitwirkenden"von Azure vordefiniert" .
  - A"benutzerdefinierte Mitwirkenderrolle" 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." .

Benutzerdefinierte Mitwirkenderrolle

```
{
  "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": []
        }
      ]
    }
}
```

- Die Azur `location` das mindestens eines enthält "delegiertes Subnetz" . Stand Trident 22.01, `location` Der Parameter ist ein Pflichtfeld auf oberster Ebene der Backend-Konfigurationsdatei. In virtuellen Pools angegebene Standortwerte werden ignoriert.
- Anwendung `Cloud Identity` , hol dir die `client ID` von einem "vom Benutzer zugewiesene verwaltete Identität" und geben Sie diese ID an in `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx` .

Zusätzliche Anforderungen für SMB-Volumina

Zum Erstellen eines SMB-Volumes benötigen Sie Folgendes:

- Active Directory ist konfiguriert und mit Azure NetApp Files verbunden. Siehe"Microsoft: Erstellen und Verwalten von Active Directory-Verbindungen für Azure NetApp Files" .
- Ein Kubernetes-Cluster mit einem Linux-Controller-Knoten und mindestens einem Windows-Worker-Knoten, auf dem Windows Server 2022 ausgeführt wird. Trident unterstützt SMB-Volumes nur, wenn sie in Pods eingebunden sind, die auf Windows-Knoten laufen.
- Mindestens ein Trident Geheimnis, das Ihre Active Directory-Anmeldeinformationen enthält, damit Azure NetApp Files sich bei Active Directory authentifizieren kann. Um Geheimnisse zu generieren `smbcreds` :

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

- Ein als Windows-Dienst konfigurierter CSI-Proxy. Um einen zu konfigurieren `csi-proxy` , siehe"GitHub: CSI-Proxy" oder"GitHub: CSI-Proxy für Windows" für Kubernetes-Knoten, die unter Windows laufen.

Konfigurationsoptionen und Beispiele für das Azure NetApp Files Backend

Erfahren Sie mehr über die NFS- und SMB-Backend-Konfigurationsoptionen für Azure

NetApp Files und sehen Sie sich Konfigurationsbeispiele an.

Backend-Konfigurationsoptionen

Trident verwendet Ihre Backend-Konfiguration (Subnetz, virtuelles Netzwerk, Dienstebene und Standort), um Azure NetApp Files Volumes auf Kapazitätspools zu erstellen, die am angeforderten Standort verfügbar sind und der angeforderten Dienstebene und dem Subnetz entsprechen.

* Ab der Version NetApp Trident 25.06 werden manuelle QoS-Kapazitätspools als technische Vorschau unterstützt.*

Die Azure NetApp Files -Backends bieten diese Konfigurationsoptionen.

Parameter	Beschreibung	Standard
`version`		Immer 1
`storageDriverName`	Name des Speichertreibers	"azure-netapp-files"
`backendName`	Benutzerdefinierter Name oder das Speicher-Backend	Fahrername + "_" + zufällige Zeichen
`subscriptionID`	Die Abonnement-ID Ihres Azure-Abonnements. Optional, wenn verwaltete Identitäten in einem AKS-Cluster aktiviert sind.	
`tenantID`	Die Mandanten-ID aus einer App-Registrierung ist optional, wenn verwaltete Identitäten oder Cloud-Identitäten auf einem AKS-Cluster verwendet werden.	
`clientID`	Die Client-ID aus einer App-Registrierung ist optional, wenn verwaltete Identitäten oder Cloud-Identitäten auf einem AKS-Cluster verwendet werden.	
`clientSecret`	Der Client-Schlüssel aus einer App-Registrierung ist optional, wenn verwaltete Identitäten oder Cloud-Identitäten auf einem AKS-Cluster verwendet werden.	
`serviceLevel`	Einer von `Standard` , `Premium` , oder `Ultra`	"" (zufällig)
`location`	Name des Azure-Standorts, an dem die neuen Volumes erstellt werden. Optional, wenn verwaltete Identitäten in einem AKS-Cluster aktiviert sind.	
`resourceGroups`	Liste der Ressourcengruppen zum Filtern der gefundenen Ressourcen	"[]" (kein Filter)

Parameter	Beschreibung	Standard
`netappAccounts`	Liste der NetApp -Konten zum Filtern der gefundenen Ressourcen	"[]" (kein Filter)
`capacityPools`	Liste der Kapazitätspools zum Filtern der gefundenen Ressourcen	"[]" (kein Filter, zufällig)
`virtualNetwork`	Name eines virtuellen Netzwerks mit einem delegierten Subnetz	""
`subnet`	Name eines delegierten Subnetzes an `Microsoft.Netapp/volumes`	""
`networkFeatures`	Satz von VNet-Funktionen für ein Volume, kann sein `Basic` oder `Standard` . Die Netzwerkfunktionen sind nicht in allen Regionen verfügbar und müssen gegebenenfalls im Rahmen eines Abonnements aktiviert werden. Spezifizierung `networkFeatures` Wenn die Funktionalität nicht aktiviert ist, schlägt die Volumenbereitstellung fehl.	""
`nfsMountOptions`	Feingranulare Steuerung der NFS-Mount-Optionen. Wird bei SMB-Volumes ignoriert. Um Volumes mit NFS Version 4.1 einzubinden, fügen Sie Folgendes hinzu: `nfsvers=4` in der durch Kommas getrennten Mount-Optionsliste die Option NFS v4.1 auswählen. Die in einer Speicherklassendefinition festgelegten Mount-Optionen überschreiben die in der Backend-Konfiguration festgelegten Mount-Optionen.	"nfsvers=3"
`limitVolumeSize`	Die Bereitstellung schlägt fehl, wenn die angeforderte Volume-Größe diesen Wert überschreitet.	"" (wird nicht standardmäßig erzwungen)
`debugTraceFlags`	Debug-Flags zur Verwendung bei der Fehlersuche. Beispiel, `\{"api": false, "method": true, "discovery": true}` . Verwenden Sie diese Funktion nur, wenn Sie eine Fehlerbehebung durchführen und einen detaillierten Protokollauszug benötigen.	null

Parameter	Beschreibung	Standard
`nasType`	Konfiguration der Erstellung von NFS- oder SMB-Volumes. Optionen sind `nfs` , `smb` oder null. Bei der Einstellung „null“ werden standardmäßig NFS-Volumes verwendet.	`nfs`
`supportedTopologies`	Stellt eine Liste der Regionen und Zonen dar, die von diesem Backend unterstützt werden. Weitere Informationen finden Sie unter"CSI-Topologie verwenden" .	
`qosType`	Stellt den QoS-Typ dar: Automatisch oder Manuell. **Technische Vorschau für Trident 25.06**	Automatisch
`maxThroughput`	Legt den maximal zulässigen Durchsatz in MiB/Sek. fest. Wird nur für manuelle QoS-Kapazitätspools unterstützt. **Technische Vorschau für Trident 25.06**	`4 MiB/sec`

Weitere Informationen zu Netzwerkfunktionen finden Sie unter"Konfigurieren von Netzwerkfunktionen für ein Azure NetApp Files Volume" .

Erforderliche Berechtigungen und Ressourcen

Wenn beim Erstellen eines PVC die Fehlermeldung „Keine Kapazitätspools gefunden“ angezeigt wird, liegt es wahrscheinlich daran, dass Ihrer App-Registrierung die erforderlichen Berechtigungen und Ressourcen (Subnetz, virtuelles Netzwerk, Kapazitätspool) nicht zugeordnet sind. Wenn der Debug-Modus aktiviert ist, protokolliert Trident die beim Erstellen des Backends erkannten Azure-Ressourcen. Vergewissern Sie sich, dass die richtige Rolle verwendet wird.

Die Werte für `resourceGroups` , `netappAccounts` , `capacityPools` , `virtualNetwork` , Und `subnet` können mit Kurznamen oder vollständigen Namen angegeben werden. In den meisten Fällen empfiehlt es sich, vollständige Namen anzugeben, da kurze Namen zu mehreren Ressourcen mit demselben Namen führen können.

Der `resourceGroups` , `netappAccounts` , Und `capacityPools` Die Werte sind Filter, die die Menge der gefundenen Ressourcen auf diejenigen beschränken, die diesem Speichersystem zur Verfügung stehen, und können in beliebiger Kombination angegeben werden. Vollständige Namen folgen diesem Format:

Typ	Format
Ressourcengruppe	<Ressourcengruppe>
NetApp Konto	<Ressourcengruppe>/<NetApp-Konto>
Kapazitätspool	<Ressourcengruppe>/<NetApp-Konto>/<Kapazitätspool>

Typ	Format
Virtuelles Netzwerk	<Ressourcengruppe>/<virtuelles Netzwerk>
Subnetz	<Ressourcengruppe>/<virtuelles Netzwerk>/<Subnetz>

Volume-Bereitstellung

Sie können die Standard-Volume-Bereitstellung steuern, indem Sie die folgenden Optionen in einem speziellen Abschnitt der Konfigurationsdatei angeben. Siehe Beispielkonfigurationen für Details.

Parameter	Beschreibung	Standard
`exportRule`	Exportregeln für neue Volumes. `exportRule` Es muss sich um eine durch Kommas getrennte Liste beliebiger Kombinationen von IPv4-Adressen oder IPv4-Subnetzen in CIDR-Notation handeln. Wird bei SMB-Volumes ignoriert.	"0.0.0.0/0"
`snapshotDir`	Steuert die Sichtbarkeit des .snapshot-Verzeichnisses	"true" für NFSv4, "false" für NFSv3
`size`	Die Standardgröße neuer Volumes	"100G"
`unixPermissions`	Die Unix-Berechtigungen für neue Datenträger (4 Oktalstellen). Wird bei SMB-Volumes ignoriert.	"" (Vorschaufunktion, erfordert Whitelisting im Abonnement)

Beispielkonfigurationen

Die folgenden Beispiele zeigen Basiskonfigurationen, bei denen die meisten Parameter auf Standardwerte eingestellt bleiben. Dies ist die einfachste Möglichkeit, ein Backend zu definieren.

Minimale Konfiguration

Dies ist die absolute Minimalkonfiguration im Backend. Mit dieser Konfiguration erkennt Trident alle Ihre NetApp -Konten, Kapazitätspools und Subnetze, die an Azure NetApp Files delegiert sind, am konfigurierten Speicherort und platziert neue Volumes zufällig auf einem dieser Pools und Subnetze. Weil `nasType` wird ausgelassen, die `nfs` Es gelten die Standardeinstellungen, und das Backend stellt NFS-Volumes bereit.

Diese Konfiguration ist ideal, wenn Sie gerade erst mit Azure NetApp Files beginnen und verschiedene Funktionen ausprobieren möchten. In der Praxis werden Sie jedoch eine zusätzliche Bereichsdefinition für die von Ihnen bereitgestellten Volumes wünschen.

```
---
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 lässt Folgendes aus: `subscriptionID` , `tenantID` , `clientID` , Und `clientSecret` , die bei der Verwendung verwalteter 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 lässt Folgendes aus: `tenantID` , `clientID` , Und `clientSecret` , die bei Verwendung einer Cloud-Identität optional sind.

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

Spezifische Service-Level-Konfiguration mit Kapazitätspoolfiltern

Diese Backend-Konfiguration platziert Volumes in Azure `eastus` Ort in einem `Ultra` Kapazitätspool. Trident erkennt automatisch alle an Azure NetApp Files delegierten Subnetze an diesem Standort und platziert ein neues Volume zufällig auf 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
```

Backend-Beispiel mit manuellen QoS-Kapazitätspools

Diese Backend-Konfiguration platziert Volumes in Azure `eastus` Standort mit manuellen QoS-Kapazitätspools. **Technische Vorschau in NetApp Trident 25.06**.

```
---
version: 1
storageDriverName: azure-netapp-files
backendName: anf1
location: eastus
labels:
  clusterName: test-cluster-1
  cloud: anf
  nasType: nfs
defaults:
  qosType: Manual
storage:
  - serviceLevel: Ultra
    labels:
      performance: gold
    defaults:
      maxThroughput: 10
  - serviceLevel: Premium
    labels:
      performance: silver
    defaults:
      maxThroughput: 5
  - serviceLevel: Standard
    labels:
      performance: bronze
    defaults:
      maxThroughput: 3
```

Erweiterte Konfiguration

Diese Backend-Konfiguration beschränkt den Umfang der Volume-Platzierung weiter auf ein einzelnes Subnetz und ändert außerdem einige Standardeinstellungen 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 eines virtuellen Pools

Diese Backend-Konfiguration definiert mehrere Speicherpools in einer einzigen Datei. Dies ist nützlich, wenn Sie mehrere Kapazitätspools haben, die unterschiedliche Servicelevel unterstützen, und Sie Speicherklassen in Kubernetes erstellen möchten, die diese repräsentieren. Virtuelle Poolbezeichnungen wurden verwendet, um die Pools anhand folgender Kriterien zu unterscheiden: `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
```

Unterstützte Topologiekonfiguration

Trident ermöglicht die Bereitstellung von Volumes für Workloads basierend auf Regionen und Verfügbarkeitszonen. Der `supportedTopologies` Der Block in dieser Backend-Konfiguration dient dazu, eine Liste von Regionen und Zonen pro Backend bereitzustellen. Die hier angegebenen Regions- und Zonenwerte müssen mit den Regions- und Zonenwerten der Labels auf jedem Kubernetes-Clusterknoten übereinstimmen. Diese Regionen und Zonen stellen die Liste der zulässigen Werte dar, die in einer Speicherklasse angegeben werden können. Für Speicherklassen, die eine Teilmenge der im Backend bereitgestellten Regionen und Zonen enthalten, erstellt Trident Volumes in der genannten Region und Zone. Weitere Informationen finden Sie unter"CSI-Topologie verwenden" .

```
---
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
```

Speicherklassendefinitionen

Die folgende `StorageClass` Die Definitionen beziehen sich auf die oben genannten Speicherpools.

Beispieldefinitionen mit `parameter.selector` Feld

Verwenden `parameter.selector` Sie können für jedes einzelne festlegen `StorageClass` Der virtuelle Pool, der zum Hosten eines Volumes verwendet wird. Das Volumen wird die im gewählten Pool definierten Aspekte aufweisen.

```
---
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

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

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"
```

Verwendung unterschiedlicher Geheimnisse pro Namensraum

```
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}
```

Verwendung unterschiedlicher Geheimnisse pro Band

```
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.

Backend erstellen

Nachdem Sie die Backend-Konfigurationsdatei erstellt haben, führen Sie folgenden Befehl aus:

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

Wenn die Backend-Erstellung fehlschlägt, stimmt etwas mit der Backend-Konfiguration nicht. Sie können die Protokolle einsehen, um die Ursache zu ermitteln, indem Sie folgenden Befehl ausführen:

```
tridentctl logs
```

Nachdem Sie das Problem mit der Konfigurationsdatei identifiziert und behoben haben, können Sie den Befehl zum Erstellen erneut ausführen.

Google Cloud NetApp Volumes

Konfigurieren eines Google Cloud NetApp Volumes -Backends

Sie können jetzt Google Cloud NetApp Volumes als Backend für Trident konfigurieren. Sie können NFS- und SMB-Volumes über ein Google Cloud NetApp Volumes -Backend einbinden.

Details zum Google Cloud NetApp Volumes -Treiber

Trident bietet die `google-cloud-netapp-volumes` Der Treiber soll mit dem Cluster kommunizieren. Unterstützte Zugriffsmodi sind: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

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

Cloud-Identität für GKE

Cloud Identity ermöglicht es Kubernetes-Pods, auf Google Cloud-Ressourcen zuzugreifen, indem sie sich als Workload-Identität authentifizieren, anstatt explizite Google Cloud-Anmeldeinformationen anzugeben.

Um die Cloud-Identität in Google Cloud nutzen zu können, benötigen Sie Folgendes:

- Ein mit GKE bereitgestellter Kubernetes-Cluster.
- Die Workload-Identität wurde im GKE-Cluster konfiguriert und der GKE MetaData Server auf den Knotenpools.
- Ein GCP-Dienstkonto mit der Rolle „Google Cloud NetApp Volumes Admin“ (roles/netapp.admin) oder

einer benutzerdefinierten Rolle.

- Trident wurde installiert, einschließlich des Cloud-Providers, der "GCP" angibt, und der Cloud-Identität, die das neue GCP-Dienstkonto angibt. Nachfolgend ein Beispiel.

Trident -Betreiber

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

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 die Flags **cloud-provider (CP)** und **cloud-identity (CI)** mithilfe 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 konfiguriert es. `cloudProvider` für GCP unter Verwendung der Umgebungsvariablen `$CP` und stellt die `cloudIdentity` unter Verwendung der Umgebungsvariablen `$ANNOTATION` :

```
helm install trident trident-operator-100.6.0.tgz --set
cloudProvider=$CP --set cloudIdentity="$ANNOTATION"
```

<code>tridentctl</code>

Legen Sie die Werte für die Flags **Cloud-Anbieter** und **Cloud-Identität** mithilfe 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 konfiguriert die `cloud-provider` Flagge an `$CP` , Und `cloud-identity` Zu `$ANNOTATION` :

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

Bereiten Sie die Konfiguration eines Google Cloud NetApp Volumes Backends vor.

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

Voraussetzungen für NFS-Volumes

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

Stellen Sie sicher, dass Sie Folgendes haben, bevor Sie das Google Cloud NetApp Volumes -Backend konfigurieren:

- Ein Google Cloud-Konto, das mit dem Google Cloud NetApp Volumes -Dienst konfiguriert ist. Siehe"Google Cloud NetApp Volumes" .
- Projektnummer Ihres Google Cloud-Kontos. Siehe"Projekte identifizieren" .
- Ein Google Cloud-Dienstkonto mit NetApp Volumes-Administratorrechten(`roles/netapp.admin`) Rolle. Siehe"Rollen und Berechtigungen für Identitäts- und Zugriffsmanagement" .
- API-Schlüsseldatei für Ihr GCNV-Konto. Siehe"Erstellen Sie einen Dienstkontoschlüssel"
- Ein Speicherbecken. Siehe"Übersicht der Speicherpools" .

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

Google Cloud NetApp Volumes Backend-Konfigurationsoptionen und Beispiele

Erfahren Sie mehr über die Backend-Konfigurationsoptionen für Google Cloud NetApp Volumes und sehen Sie sich Konfigurationsbeispiele an.

Backend-Konfigurationsoptionen

Jedes Backend stellt Volumes in einer einzelnen Google Cloud-Region bereit. Um Volumes in anderen Regionen zu erstellen, können Sie zusätzliche Backends definieren.

Parameter	Beschreibung	Standard
`version`		Immer 1
`storageDriverName`	Name des Speichertreibers	Der Wert von `storageDriverName` muss als "google-cloud-netapp-volumes" angegeben werden.

Parameter	Beschreibung	Standard
`backendName`	(Optional) Benutzerdefinierter Name des Speicher-Backends	Fahrername + "_" + Teil des API-Schlüssels
`storagePools`	Optionaler Parameter zur Angabe von Speicherpools für die Volume-Erstellung.	
`projectNumber`	Google Cloud-Konto-Projektnummer. Den Wert finden Sie auf der Startseite des Google Cloud-Portals.	
`location`	Der Google Cloud-Speicherort, an dem Trident GCNV-Volumes erstellt. Beim Erstellen regionsübergreifender Kubernetes-Cluster werden Volumes, die in einem `location` kann in Workloads verwendet werden, die auf Knoten in mehreren Google Cloud-Regionen geplant sind. Für den Verkehr über Regionen hinweg fallen zusätzliche Kosten an.	
`apiKey`	API-Schlüssel für das Google Cloud-Dienstkonto mit dem `netapp.admin` Rolle. Sie enthält den JSON-formatierten Inhalt der privaten Schlüsseldatei eines Google Cloud-Dienstkontos (wörtlich in die Backend-Konfigurationsdatei kopiert). Der `apiKey` müssen 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`	Feingranulare Steuerung der NFS-Mount-Optionen.	"nfsvers=3"
`limitVolumeSize`	Die Bereitstellung schlägt fehl, wenn die angeforderte Volume-Größe diesen Wert überschreitet.	"" (wird nicht standardmäßig erzwungen)
`serviceLevel`	Der Servicegrad eines Speicherpools und seiner Volumes. Die Werte sind `flex` , `standard` , `premium` , oder `extreme` .	
`labels`	Satz beliebiger JSON-formatierter Bezeichnungen, die auf Datenträger angewendet werden sollen	""
`network`	Das Google Cloud-Netzwerk wird für GCNV-Volumes verwendet.	
`debugTraceFlags`	Debug-Flags zur Verwendung bei der Fehlersuche. Beispiel, `{"api":false, "method":true}` . Verwenden Sie diese Funktion nur, wenn Sie eine Fehlerbehebung durchführen und einen detaillierten Protokollauszug benötigen.	null
`nasType`	Konfiguration der Erstellung von NFS- oder SMB-Volumes. Optionen sind `nfs` , `smb` oder null. Bei der Einstellung „null“ werden standardmäßig NFS-Volumes verwendet.	`nfs`

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

Volumenbereitstellungsoptionen

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

Parameter	Beschreibung	Standard
`exportRule`	Die Ausfuhrbestimmungen für neue Bände. Es muss sich um eine durch Kommas getrennte Liste beliebiger Kombinationen von IPv4-Adressen handeln.	"0.0.0.0/0"
`snapshotDir`	Zugang zu `.snapshot` Verzeichnis	"true" für NFSv4, "false" für NFSv3
`snapshotReserve`	Prozentsatz des für Snapshots reservierten Speichervolumens	"" (Standardwert 0 akzeptieren)
`unixPermissions`	Die Unix-Berechtigungen für neue Datenträger (4 Oktalstellen).	""

Beispielkonfigurationen

Die folgenden Beispiele zeigen Basiskonfigurationen, bei denen die meisten Parameter auf Standardwerte eingestellt bleiben. Dies ist die einfachste Möglichkeit, ein Backend zu definieren.

Minimale Konfiguration

Dies ist die absolute Minimalkonfiguration im Backend. Mit dieser Konfiguration erkennt Trident alle Ihre Speicherpools, die an Google Cloud NetApp Volumes delegiert sind, am konfigurierten Speicherort und platziert neue Volumes zufällig in einem dieser Pools. Weil `nasType` wird ausgelassen, die `nfs` Es gelten die Standardeinstellungen, und das Backend stellt NFS-Volumes bereit.

Diese Konfiguration ist ideal, wenn Sie gerade erst mit Google Cloud NetApp Volumes beginnen und verschiedene Funktionen ausprobieren möchten. In der Praxis werden Sie jedoch höchstwahrscheinlich zusätzliche Einschränkungen für die von Ihnen bereitgestellten Volumes benötigen.

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

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  serviceLevel: premium
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Konfiguration für SMB-Volumes

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv1
  namespace: trident
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123456789"
  location: asia-east1
  serviceLevel: flex
  nasType: smb
  apiKey:
    type: service_account
    project_id: cloud-native-data
    client_email: trident-sample@cloud-native-
data.iam.gserviceaccount.com
    client_id: "123456789737813416734"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/trident-
sample%40cloud-native-data.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Konfiguration mit StoragePools-Filter

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

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  serviceLevel: premium
  storagePools:
    - premium-pool1-europe-west6
    - premium-pool2-europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Konfiguration eines virtuellen Pools

Diese Backend-Konfiguration definiert mehrere virtuelle Pools in einer einzigen Datei. Virtuelle Pools werden definiert in der `storage` Abschnitt. Sie sind nützlich, wenn Sie mehrere Speicherpools haben, die unterschiedliche Servicelevel unterstützen, und Sie Speicherklassen in Kubernetes erstellen möchten, die diese repräsentieren. Virtuelle Poolbezeichnungen dienen zur Unterscheidung der Pools. Zum Beispiel im folgenden Beispiel `performance` Etikett und `serviceLevel` Der Typ wird verwendet, um virtuelle Pools zu unterscheiden.

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

Weitere Informationen finden Sie unter"Virtuelle Pools" .

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

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

```
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
  defaults:
    snapshotReserve: "10"
    exportRule: 10.0.0.0/24
  storage:
    - labels:
        performance: extreme
      serviceLevel: extreme
      defaults:
        snapshotReserve: "5"
        exportRule: 0.0.0.0/0
    - labels:
        performance: premium
      serviceLevel: premium
    - labels:
        performance: standard
      serviceLevel: standard
```

Cloud-Identität für GKE

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

Unterstützte Topologiekonfiguration

Trident ermöglicht die Bereitstellung von Volumes für Workloads basierend auf Regionen und Verfügbarkeitszonen. Der `supportedTopologies` Der Block in dieser Backend-Konfiguration dient dazu, eine Liste von Regionen und Zonen pro Backend bereitzustellen. Die hier angegebenen Regions- und Zonenwerte müssen mit den Regions- und Zonenwerten der Labels auf jedem Kubernetes-Clusterknoten übereinstimmen. Diese Regionen und Zonen stellen die Liste der zulässigen Werte dar, die in einer Speicherklasse angegeben werden können. Für Speicherklassen, die eine Teilmenge der im Backend bereitgestellten Regionen und Zonen enthalten, erstellt Trident Volumes in der genannten Region und Zone. Weitere Informationen finden Sie unter"CSI-Topologie verwenden" .

```
---
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
```

Wie geht es weiter?

Nachdem Sie die Backend-Konfigurationsdatei erstellt haben, führen Sie folgenden Befehl aus:

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

Um zu überprüfen, ob das Backend erfolgreich erstellt wurde, führen Sie folgenden Befehl aus:

```
kubectl get tridentbackendconfig

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

Wenn die Backend-Erstellung fehlschlägt, stimmt etwas mit der Backend-Konfiguration nicht. Sie können das Backend mithilfe des folgenden beschreiben: `kubectl get tridentbackendconfig <backend-name>` Um die Ursache zu ermitteln, führen Sie den folgenden Befehl aus oder überprüfen Sie die Protokolle:

```
tridentctl logs
```

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

Speicherklassendefinitionen

Im Folgenden finden Sie eine grundlegende `StorageClass` Definition, die sich auf das oben genannte Backend 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 unter Verwendung der `parameter.selector` Feld:

Verwenden `parameter.selector` Sie können für jedes einzelne festlegen `StorageClass` Die"virtueller Pool" Das wird zum Hosten eines Volumes verwendet. Das Volumen wird die im gewählten Pool definierten Aspekte aufweisen.

```
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 Einzelheiten zu Speicherklassen finden Sie unter"Erstellen einer Speicherklasse" .

Beispieldefinitionen für SMB-Volumes

Verwenden `nasType` , `node-stage-secret-name` , Und `node-stage-secret-namespace` Sie können ein SMB-Volume angeben und die erforderlichen Active Directory-Anmeldeinformationen bereitstellen. Für das Node-Stage-Secret kann jeder Active Directory-Benutzer/jedes Passwort mit beliebigen/keinen Berechtigungen verwendet werden.

Grundkonfiguration im Standard-Namespace

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

Verwendung unterschiedlicher Geheimnisse pro Namensraum

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

Verwendung unterschiedlicher Geheimnisse pro Band

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

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

PVC-Definitionsbeispiel

```
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-Verbindung hergestellt ist, führen Sie 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
```

Konfigurieren eines Cloud Volumes Service für das Google Cloud-Backend

Erfahren Sie anhand der bereitgestellten Beispielkonfigurationen, wie Sie den NetApp Cloud Volumes Service für Google Cloud als Backend für Ihre Trident -Installation konfigurieren.

Details zum Google Cloud-Treiber

Trident bietet die `gcp-cvs` Der Treiber soll mit dem Cluster kommunizieren. Unterstützte Zugriffsmodi sind: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

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

Erfahren Sie mehr über die Trident Unterstützung für den Cloud Volumes Service für Google Cloud.

Trident kann Cloud Volumes Service Volumes in einem von zwei Formaten erstellen."Servicearten" :

- **CVS-Performance**: Der standardmäßige Trident Diensttyp. Dieser leistungsoptimierte Servicetyp eignet sich am besten für Produktionsworkloads, bei denen Leistung wichtig ist. Der Servicetyp CVS-Performance ist eine Hardwareoption, die Volumes mit einer Mindestgröße von 100 GiB unterstützt. Sie können eines auswählen"drei Serviceebenen" :
  - `standard`
  - `premium`
  - `extreme`
- **CVS**: Der CVS-Diensttyp bietet eine hohe zonale Verfügbarkeit bei begrenzter bis mäßiger Leistungsfähigkeit. Der CVS-Diensttyp ist eine Softwareoption, die Speicherpools nutzt, um Datenmengen ab 1 GiB zu unterstützen. Der Speicherpool kann bis zu 50 Volumes enthalten, wobei sich alle Volumes die Kapazität und Leistung des Pools teilen. Sie können eines auswählen"zwei Serviceebenen" :
  - `standardsw`
  - `zoneredundantstandardsw`

Was du brauchst

Um die "Cloud Volumes Service für Google Cloud" Im Backend benötigen Sie Folgendes:

- Ein mit dem NetApp Cloud Volumes Service konfiguriertes Google Cloud-Konto
- Projektnummer Ihres Google Cloud-Kontos
- Google Cloud-Dienstkonto mit dem `netappcloudvolumes.admin` Rolle
- API-Schlüsseldatei für Ihr Cloud Volumes Service -Konto

Backend-Konfigurationsoptionen

Jedes Backend stellt Volumes in einer einzelnen Google Cloud-Region bereit. Um Volumes in anderen Regionen zu erstellen, können Sie zusätzliche Backends definieren.

Parameter	Beschreibung	Standard
`version`		Immer 1
`storageDriverName`	Name des Speichertreibers	"gcp-cvs"
`backendName`	Benutzerdefinierter Name oder das Speicher-Backend	Fahrername + "_" + Teil des API-Schlüssels
`storageClass`	Optionaler Parameter zur Angabe des CVS-Diensttyps. Verwenden `software` um den CVS-Diensttyp auszuwählen. Andernfalls geht Trident vom Diensttyp CVS-Performance aus.(`hardware`).	
`storagePools`	Nur CVS-Dienstleistungstyp. Optionaler Parameter zur Angabe von Speicherpools für die Volume-Erstellung.	
`projectNumber`	Google Cloud-Konto-Projektnummer. Den Wert finden Sie auf der Startseite des Google Cloud-Portals.	

Parameter	Beschreibung	Standard
`hostProjectNumber`	Erforderlich bei Verwendung eines gemeinsam genutzten VPC-Netzwerks. In diesem Szenario `projectNumber` ist das Serviceprojekt, und `hostProjectNumber` ist das Host-Projekt.	
`apiRegion`	Die Google Cloud-Region, in der Trident Cloud Volumes Service Volumes erstellt. Beim Erstellen regionsübergreifender Kubernetes-Cluster werden Volumes, die in einem `apiRegion` kann in Workloads verwendet werden, die auf Knoten in mehreren Google Cloud-Regionen geplant sind. Für den Verkehr über Regionen hinweg fallen zusätzliche Kosten an.	
`apiKey`	API-Schlüssel für das Google Cloud-Dienstkonto mit dem `netappcloudvolumes.admin` Rolle. Sie enthält den JSON-formatierten Inhalt der privaten Schlüsseldatei eines Google Cloud-Dienstkontos (wörtlich in die Backend-Konfigurationsdatei kopiert).	
`proxyURL`	Proxy-URL, falls ein Proxy-Server zur Verbindung mit dem CVS-Konto erforderlich ist. Der Proxy-Server kann entweder ein HTTP-Proxy oder ein HTTPS-Proxy sein. Bei einem HTTPS-Proxy wird die Zertifikatsvalidierung übersprungen, um die Verwendung selbstsignierter Zertifikate auf dem Proxy-Server zu ermöglichen. Proxyserver mit aktivierter Authentifizierung werden nicht unterstützt.	
`nfsMountOptions`	Feingranulare Steuerung der NFS-Mount-Optionen.	"nfsvers=3"
`limitVolumeSize`	Die Bereitstellung schlägt fehl, wenn die angeforderte Volume-Größe diesen Wert überschreitet.	"" (wird nicht standardmäßig erzwungen)
`serviceLevel`	Die CVS-Performance- oder CVS-Serviceebene für neue Volumina. Die CVS-Performance-Werte sind `standard` , `premium` , oder `extreme` . CVS-Werte sind `standardsw` oder `zoneredundantstandardsw` .	Die Standardeinstellung für CVS-Performance ist "Standard". Der CVS-Standardwert ist "standardsw".
`network`	Das Google Cloud-Netzwerk wird für Cloud Volumes Service Volumes verwendet.	"Standard"
`debugTraceFlags`	Debug-Flags zur Verwendung bei der Fehlersuche. Beispiel, `\{"api":false, "method":true}` . Verwenden Sie diese Funktion nur, wenn Sie eine Fehlerbehebung durchführen und einen detaillierten Protokollauszug benötigen.	null

Parameter	Beschreibung	Standard

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

Volumenbereitstellungsoptionen

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

Parameter	Beschreibung	Standard
`exportRule`	Die Ausfuhrbestimmungen für neue Bände. Es muss sich um eine durch Kommas getrennte Liste beliebiger Kombinationen von IPv4-Adressen oder IPv4-Subnetzen in CIDR-Notation handeln.	"0.0.0.0/0"
`snapshotDir`	Zugang zu `.snapshot` Verzeichnis	"FALSCH"
`snapshotReserve`	Prozentsatz des für Snapshots reservierten Speichervolumens	"" (CVS-Standardwert 0 akzeptieren)
`size`	Der Umfang der neuen Bände. Die Mindestgröße für CVS-Performance beträgt 100 GiB. Die Mindestgröße für CVS beträgt 1 GiB.	Der CVS-Performance-Diensttyp ist standardmäßig auf „100 GiB“ eingestellt. Der CVS-Diensttyp legt keinen Standardwert fest, erfordert aber ein Minimum von 1 GiB.

Beispiele für CVS-Performance-Diensttypen

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

Beispiel 1: Minimale Konfiguration

Dies ist die minimale Backend-Konfiguration unter Verwendung des Standard-CVS-Performance-Diensttyps mit dem Standard-Dienstlevel „standard“.

```
---
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: Konfiguration auf Serviceebene

Dieses Beispiel veranschaulicht die Backend-Konfigurationsoptionen, einschließlich Servicelevel und Volumenstandardeinstellungen.

```
---
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 eines virtuellen Pools

Dieses Beispiel verwendet `storage` virtuelle Pools und die `StorageClasses` die sich auf sie beziehen. SieheSpeicherklassendefinitionen um zu sehen, wie die Speicherklassen definiert wurden.

Hier werden spezifische Standardwerte für alle virtuellen Pools festgelegt, die die `snapshotReserve` bei 5 % und der `exportRule` zu 0.0.0.0/0. Die virtuellen Pools sind definiert in der `storage` Abschnitt. Jeder einzelne virtuelle Pool definiert seine eigenen `serviceLevel` und einige Pools überschreiben die Standardwerte. Virtuelle Poolbezeichnungen wurden verwendet, um die Pools anhand folgender Kriterien zu unterscheiden: `performance` Und `protection` .

```
---
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
```

Speicherklassendefinitionen

Die folgenden StorageClass-Definitionen gelten für das Konfigurationsbeispiel des virtuellen Pools. Verwenden `parameters.selector` Sie können für jede StorageClass den virtuellen Pool angeben, der zum Hosten eines Volumes verwendet wird. Das Volumen wird die im gewählten Pool definierten Aspekte aufweisen.

Beispiel für eine Speicherklasse

```
---
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 Speicherklasse(`cvs-extreme-extra-protection`) wird dem ersten virtuellen Pool zugeordnet. Dies ist der einzige Pool, der extreme Leistung mit einer Momentaufnahme-Reserve von 10% bietet.
- Die letzte Speicherklasse(`cvs-extra-protection`) kennzeichnet jeden Speicherpool, der eine Snapshot-Reserve von 10% bereitstellt. Trident entscheidet, welcher virtuelle Pool ausgewählt wird und stellt sicher, dass die Snapshot-Reservierungsanforderung erfüllt wird.

Beispiele für CVS-Diensttypen

Die folgenden Beispiele enthalten Beispielkonfigurationen für den CVS-Diensttyp.

Beispiel 1: Minimale Konfiguration

Dies ist die minimale Backend-Konfiguration mit `storageClass` um den CVS-Diensttyp und den Standardwert anzugeben `standardsw` Servicelevel.

```
---
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 Speicherpools

Diese Beispielkonfiguration für das Backend verwendet `storagePools` einen Speicherpool 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
```

Wie geht es weiter?

Nachdem Sie die Backend-Konfigurationsdatei erstellt haben, führen Sie folgenden Befehl aus:

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

Wenn die Backend-Erstellung fehlschlägt, stimmt etwas mit der Backend-Konfiguration nicht. Sie können die Protokolle einsehen, um die Ursache zu ermitteln, indem Sie folgenden Befehl ausführen:

```
tridentctl logs
```

Nachdem Sie das Problem mit der Konfigurationsdatei identifiziert und behoben haben, können Sie den Befehl zum Erstellen erneut ausführen.

Konfigurieren Sie ein NetApp HCI oder SolidFire Backend

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

Element-Treiberdetails

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

Der `solidfire-san` Der Speichertreiber unterstützt die Volume-Modi *file* und *block*. Für die `Filesystem` Im Modus volumeMode erstellt Trident ein Volume und ein Dateisystem. Der Dateisystemtyp wird durch die StorageClass festgelegt.

Treiber	Protokoll	Lautstärkemodus	Unterstützte Zugriffsmodi	Unterstützte Dateisysteme
`solidfire-san`	iSCSI	Block	RWO, ROX, RWX, RWOP	Kein Dateisystem. Rohblockgerät.
`solidfire-san`	iSCSI	Dateisystem	RWO, RWOP	`xfs`, `ext3` , `ext4`

Bevor Sie beginnen

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

- Ein unterstütztes Speichersystem, auf dem die Element-Software läuft.
- Anmeldeinformationen für einen NetApp HCI/ SolidFire Cluster-Administrator oder Mandantenbenutzer, der Volumes verwalten kann.
- Auf allen Ihren Kubernetes-Worker-Knoten sollten die entsprechenden iSCSI-Tools installiert sein. Siehe"Informationen zur Vorbereitung der Worker-Knoten" .

Backend-Konfigurationsoptionen

Die folgenden Tabellen enthalten die Backend-Konfigurationsoptionen:

Parameter	Beschreibung	Standard
`version`		Immer 1
`storageDriverName`	Name des Speichertreibers	Immer "solidfire-san"
`backendName`	Benutzerdefinierter Name oder das Speicher-Backend	"solidfire_" + Speicher (iSCSI) IP-Adresse

Parameter	Beschreibung	Standard
`Endpoint`	MVIP für den SolidFire -Cluster mit Mandantenanmeldeinformationen	
`SVIP`	Speicher (iSCSI) IP-Adresse und Port	
`labels`	Satz beliebiger, im JSON-Format vorliegender Bezeichnungen, die auf Datenträger angewendet werden sollen.	""
`TenantName`	Zu verwendender Mandantenname (wird erstellt, falls nicht gefunden)	
`InitiatorIFace`	iSCSI-Datenverkehr auf eine bestimmte Hostschnittstelle beschränken	"Standard"
`UseCHAP`	Verwenden Sie CHAP zur Authentifizierung von iSCSI. Trident verwendet CHAP.	true
`AccessGroups`	Liste der zu verwendenden Zugriffsgruppen-IDs	Findet die ID einer Zugriffsgruppe mit dem Namen "trident"
`Types`	QoS-Spezifikationen	
`limitVolumeSize`	Die Bereitstellung schlägt fehl, wenn die angeforderte Volume-Größe diesen Wert überschreitet.	"" (wird nicht standardmäßig erzwungen)
`debugTraceFlags`	Debug-Flags zur Verwendung bei der Fehlersuche. Beispiel: {"api":false, "method":true}	null

Nicht verwenden `debugTraceFlags` Es sei denn, Sie befinden sich in der Fehlersuche und benötigen einen detaillierten Protokollauszug.

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

Dieses Beispiel zeigt eine Backend-Datei, die CHAP-Authentifizierung verwendet und drei Volumentypen mit spezifischen QoS-Garantien modelliert. Höchstwahrscheinlich würden Sie dann Speicherklassen definieren, um jede dieser Klassen mithilfe der folgenden Methode zu nutzen: `IOPS` Speicherklassenparameter.

```
---
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: Backend- und Speicherklassenkonfiguration für `solidfire-san` Fahrer mit virtuellen Pools

Dieses Beispiel zeigt die Backend-Definitionsdatei, die mit virtuellen Pools konfiguriert ist, sowie StorageClasses, die auf diese verweisen.

Trident kopiert die auf einem Speicherpool vorhandenen Labels beim Provisioning auf die Backend-Speicher-LUN. Zur Vereinfachung können Speicheradministratoren Bezeichnungen pro virtuellem Pool definieren und Volumes nach Bezeichnung gruppieren.

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

```
---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0
SVIP: <svip>:3260
```

```
TenantName: <tenant>
UseCHAP: true
Types:
  - Type: Bronze
    Qos:
      minIOPS: 1000
      maxIOPS: 2000
      burstIOPS: 4000
  - Type: Silver
    Qos:
      minIOPS: 4000
      maxIOPS: 6000
      burstIOPS: 8000
  - Type: Gold
    Qos:
      minIOPS: 6000
      maxIOPS: 8000
      burstIOPS: 10000
type: Silver
labels:
  store: solidfire
  k8scluster: dev-1-cluster
region: us-east-1
storage:
  - labels:
      performance: gold
      cost: "4"
    zone: us-east-1a
    type: Gold
  - labels:
      performance: silver
      cost: "3"
    zone: us-east-1b
    type: Silver
  - labels:
      performance: bronze
      cost: "2"
    zone: us-east-1c
    type: Bronze
  - labels:
      performance: silver
      cost: "1"
    zone: us-east-1d
```

Die folgenden StorageClass-Definitionen beziehen sich auf die oben genannten virtuellen Pools. Verwenden des `parameters.selector` Im Feld „StorageClass“ wird jeweils angegeben, welcher virtuelle Pool (oder

welche virtuellen Pools) zum Hosten eines Volumes verwendet werden kann. Das Volumen wird die im gewählten virtuellen Pool definierten Aspekte aufweisen.

Die erste Speicherklasse(`solidfire-gold-four` wird dem ersten virtuellen Pool zugeordnet. Dies ist der einzige Pool, der Gold-Leistung mit einem bietet `Volume Type QoS` aus Gold. Die letzte Speicherklasse(`solidfire-silver`) nennt jeden Speicherpool, der eine Silber-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

- "Volumenzugriffsgruppen"

ONTAP SAN-Treiber

ONTAP SAN-Treiberübersicht

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

ONTAP SAN-Treiberdetails

Trident stellt die folgenden SAN-Speichertreiber zur Verfügung, um mit dem ONTAP Cluster zu kommunizieren. Unterstützte Zugriffsmodi sind: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Treiber	Protokoll	Lautstärke modus	Unterstützte Zugriffsmodi	Unterstützte Dateisysteme
`ontap-san`	iSCSI SCSI über FC	Block	RWO, ROX, RWX, RWOP	Kein Dateisystem; rohes Blockgerät
`ontap-san`	iSCSI SCSI über FC	Dateisystem	RWO, RWOP ROX und RWX sind im Dateisystem-Volume-Modus nicht verfügbar.	`xfs`, `ext3` , `ext4`
`ontap-san`	NVMe/TCP SieheWeitere Überlegungen zu NVMe/TCP.	Block	RWO, ROX, RWX, RWOP	Kein Dateisystem; rohes Blockgerät

Treiber	Protokoll	Lautstärke modus	Unterstützte Zugriffsmodi	Unterstützte Dateisysteme
`ontap-san`	NVMe/TCP SieheWeitere Überlegungen zu NVMe/TCP.	Dateisystem	RWO, RWOP ROX und RWX sind im Dateisystem-Volume-Modus nicht verfügbar.	`xfs`, `ext3` , `ext4`
`ontap-san-economy`	iSCSI	Block	RWO, ROX, RWX, RWOP	Kein Dateisystem; rohes Blockgerät
`ontap-san-economy`	iSCSI	Dateisystem	RWO, RWOP ROX und RWX sind im Dateisystem-Volume-Modus nicht verfügbar.	`xfs`, `ext3` , `ext4`

- Verwenden `ontap-san-economy` nur wenn die Anzahl der dauerhaften Speichernutzungen voraussichtlich höher sein wird als"Unterstützte ONTAP Lautstärkebegrenzungen" .
- Verwenden `ontap-nas-economy` nur wenn die Anzahl der dauerhaften Speichernutzungen voraussichtlich höher sein wird als"Unterstützte ONTAP Lautstärkebegrenzungen" und die `ontap-san-economy` Der Treiber kann nicht verwendet werden.
- Nicht verwenden `ontap-nas-economy` wenn Sie mit einem Bedarf an Datenschutz, Notfallwiederherstellung oder Mobilität rechnen.
- NetApp empfiehlt die Verwendung von Flexvol Autogrow nicht in allen ONTAP -Treibern, außer ontap-san. Als Ausweichlösung unterstützt Trident die Verwendung von Snapshot-Reserven und skaliert Flexvol-Volumes entsprechend.

Benutzerberechtigungen

Trident wird voraussichtlich entweder als ONTAP oder SVM-Administrator ausgeführt, typischerweise unter Verwendung von `admin` Clusterbenutzer oder ein `vsadmin` SVM-Benutzer oder ein Benutzer mit einem anderen Namen, der die gleiche Rolle hat. Bei Amazon FSx for NetApp ONTAP Bereitstellungen erwartet Trident , dass es entweder als ONTAP oder SVM-Administrator ausgeführt wird und den Cluster nutzt. `fsxadmin` Benutzer oder ein `vsadmin` SVM-Benutzer oder ein Benutzer mit einem anderen Namen, der die gleiche Rolle hat. Der `fsxadmin` Der Benutzer ist ein eingeschränkter Ersatz für den Cluster-Administratorbenutzer.

Wenn Sie die `limitAggregateUsage` Für diesen Parameter sind Cluster-Administratorrechte erforderlich. Bei der Verwendung von Amazon FSx for NetApp ONTAP mit Trident `limitAggregateUsage` Der Parameter funktioniert nicht mit dem `vsadmin` Und `fsxadmin` Benutzerkonten. Die Konfiguration schlägt fehl, wenn Sie diesen Parameter angeben.

Es ist zwar möglich, innerhalb von ONTAP eine restriktivere Rolle zu erstellen, die ein Trident -Treiber

verwenden kann, wir empfehlen dies jedoch nicht. Die meisten neuen Versionen von Trident werden zusätzliche APIs aufrufen, die berücksichtigt werden müssen, was Aktualisierungen schwierig und fehleranfällig macht.

Weitere Überlegungen zu NVMe/TCP

Trident unterstützt das NVMe-Protokoll (Non-Volatile Memory Express) mithilfe des `ontap-san` Fahrer einschließlich:

- IPv6
- Snapshots und Klone von NVMe-Volumes
- Ändern der Größe eines NVMe-Volumes
- Importieren eines NVMe-Volumes, das außerhalb von Trident erstellt wurde, damit sein Lebenszyklus von Trident verwaltet werden kann.
- NVMe-natives Multipathing
- Geordnetes oder ungeordnetes Herunterfahren der K8s-Knoten (24.06)

Trident unterstützt Folgendes nicht:

- DH-HMAC-CHAP, das nativ von NVMe unterstützt wird
- Gerätemapper (DM) Multipathing
- LUKS-Verschlüsselung

> NVMe wird nur mit ONTAP REST APIs unterstützt und nicht mit ONTAPI (ZAPI).

Bereiten Sie die Konfiguration des Backends mit ONTAP SAN-Treibern vor.

Machen Sie sich mit den Anforderungen und Authentifizierungsoptionen für die Konfiguration eines ONTAP -Backends mit ONTAP SAN-Treibern vertraut.

Anforderungen

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

> "ASA r2-Systeme"Sie unterscheiden sich von anderen ONTAP Systemen (ASA, AFF und FAS) in der Implementierung ihrer Speicherschicht. In ASA r2-Systemen werden Speicherverfügbarkeitszonen anstelle von Aggregaten verwendet. Siehe"Das" Wissensdatenbankartikel zur Zuordnung von Aggregaten zu SVMs in ASA r2-Systemen.

Denken Sie daran, dass Sie auch mehrere Treiber gleichzeitig ausführen und Speicherklassen erstellen können, die auf den einen oder anderen Treiber verweisen. Beispielsweise könnten Sie Folgendes konfigurieren: `san-dev` Klasse, die die `ontap-san` Fahrer und ein `san-default` Klasse, die die `ontap-san-economy` eins.

Auf allen Ihren Kubernetes-Worker-Knoten müssen die entsprechenden iSCSI-Tools installiert sein. Siehe "Bereiten Sie den Worker-Knoten vor." für Details.

Authentifizieren Sie das ONTAP Backend

Trident bietet zwei Modi zur Authentifizierung eines ONTAP Backends.

- Anmeldeinformationsbasiert: Benutzername und Passwort eines ONTAP Benutzers mit den erforderlichen Berechtigungen. Es wird empfohlen, eine vordefinierte Sicherheitsanmelderolle zu verwenden, wie zum Beispiel `admin` oder `vsadmin` um maximale Kompatibilität mit ONTAP Versionen zu gewährleisten.
- Zertifikatsbasiert: Trident kann auch mit einem ONTAP Cluster über ein auf dem Backend installiertes Zertifikat kommunizieren. Hierbei müssen in der Backend-Definition Base64-kodierte Werte des Client-Zertifikats, des Schlüssels und des vertrauenswürdigen CA-Zertifikats (falls verwendet, empfohlen) enthalten sein.

Sie können bestehende Backends aktualisieren, um zwischen anmeldeinformationsbasierten und zertifikatsbasierten Methoden zu wechseln. Es wird jedoch jeweils nur eine Authentifizierungsmethode unterstützt. Um zu einer anderen Authentifizierungsmethode zu wechseln, müssen Sie die bestehende Methode aus der Backend-Konfiguration entfernen.

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

Aktivieren Sie die anmeldeinformationsbasierte Authentifizierung

Trident benötigt die Anmeldeinformationen eines SVM-/Cluster-Administrators, um mit dem ONTAP Backend zu kommunizieren. Es wird empfohlen, standardisierte, vordefinierte Rollen zu verwenden, wie zum Beispiel `admin` oder `vsadmin` . Dies gewährleistet die Vorwärtskompatibilität mit zukünftigen ONTAP Versionen, die möglicherweise Feature-APIs zur Verwendung durch zukünftige Trident Versionen bereitstellen. Eine benutzerdefinierte Sicherheitsanmelderolle kann erstellt und mit Trident verwendet werden, dies wird jedoch nicht empfohlen.

Eine beispielhafte Backend-Definition sieht folgendermaßen aus:

YAML

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

JSON

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

Beachten Sie, dass die Backend-Definition der einzige Ort ist, an dem die Anmeldeinformationen im Klartext gespeichert werden. Nach der Erstellung des Backends werden Benutzernamen und Passwörter mit Base64 kodiert und als Kubernetes-Secrets gespeichert. Die Erstellung oder Aktualisierung eines Backends ist der einzige Schritt, der Kenntnisse der Zugangsdaten erfordert. Daher handelt es sich um eine ausschließlich für Administratoren zulässige Operation, die vom Kubernetes-/Speicheradministrator durchgeführt werden muss.

Zertifikatbasierte Authentifizierung aktivieren

Neue und bestehende Backends können ein Zertifikat verwenden und mit dem ONTAP Backend kommunizieren. Für die Backend-Definition werden drei Parameter benötigt.

- clientCertificate: Base64-kodierter Wert des Clientzertifikats.
- clientPrivateKey: Base64-kodierter Wert des zugehörigen privaten Schlüssels.
- trustedCACertificate: Base64-kodierter Wert des vertrauenswürdigen CA-Zertifikats. Bei Verwendung einer vertrauenswürdigen Zertifizierungsstelle muss dieser Parameter angegeben werden. Dies kann ignoriert werden, wenn keine vertrauenswürdige Zertifizierungsstelle verwendet wird.

Ein typischer Arbeitsablauf umfasst die folgenden Schritte.

Schritte

1. Generieren Sie ein Clientzertifikat und einen Schlüssel. Beim Generieren muss der allgemeine Name (CN) auf den ONTAP Benutzer gesetzt werden, der sich authentifizieren soll.

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

2. Fügen Sie dem ONTAP -Cluster ein vertrauenswürdiges CA-Zertifikat hinzu. Dies könnte bereits vom Speicheradministrator erledigt werden. Ignorieren, falls keine vertrauenswürdige Zertifizierungsstelle 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 Clientzertifikat und den Schlüssel (aus Schritt 1) auf dem ONTAP Cluster.

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

4. Bestätigen Sie, dass die ONTAP Sicherheitsanmeldungsrolle die folgenden Funktionen unterstützt: `cert` Authentifizierungsmethode.

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

5. Testen Sie die Authentifizierung mit dem generierten Zertifikat. Ersetzen Sie < ONTAP Management LIF> und <vserver name> durch die Management LIF IP-Adresse und den SVM-Namen.

```
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. Zertifikat, Schlüssel und vertrauenswürdiges CA-Zertifikat mit Base64 kodieren.

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

7. Erstellen Sie ein Backend unter Verwendung der im vorherigen Schritt erhaltenen Werte.

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

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

Aktualisieren Sie die Authentifizierungsmethoden oder ändern Sie die Anmeldeinformationen.

Sie können ein bestehendes Backend aktualisieren, um eine andere Authentifizierungsmethode zu verwenden oder um die Anmeldeinformationen zu ändern. Dies funktioniert in beide Richtungen: Backends, die Benutzername/Passwort verwenden, können auf die Verwendung von Zertifikaten umgestellt werden; Backends, die Zertifikate verwenden, können auf Benutzername/Passwort-basiert umgestellt werden. Dazu müssen Sie die bestehende Authentifizierungsmethode entfernen und die neue Authentifizierungsmethode hinzufügen. Verwenden Sie anschließend die aktualisierte Datei backend.json, die die erforderlichen Parameter enthält, um die Ausführung durchzuführen. `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 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Beim Ändern von Passwörtern muss der Speicheradministrator zuerst das Passwort für den Benutzer auf ONTAP aktualisieren. Anschließend erfolgt ein Backend-Update. Bei der Zertifikatsrotation können dem Benutzer mehrere Zertifikate hinzugefügt werden. Anschließend wird das Backend aktualisiert, um das neue Zertifikat zu verwenden. Danach kann das alte Zertifikat aus dem ONTAP Cluster gelöscht werden.

Durch die Aktualisierung des Backends wird der Zugriff auf bereits erstellte Volumes nicht beeinträchtigt, und auch später hergestellte Volume-Verbindungen werden nicht beeinträchtigt. Ein erfolgreiches Backend-Update zeigt an, dass Trident mit dem ONTAP -Backend kommunizieren und zukünftige Volumenoperationen bewältigen kann.

Erstellen einer benutzerdefinierten ONTAP Rolle für Trident

Sie können eine ONTAP Clusterrolle mit minimalen Berechtigungen erstellen, sodass Sie für Operationen in Trident nicht die ONTAP Administratorrolle verwenden müssen. Wenn Sie den Benutzernamen in einer Trident Backend-Konfiguration angeben, verwendet Trident die von Ihnen erstellte ONTAP Clusterrolle, um die Operationen durchzuführen.

Siehe"Trident -Benutzerrollengenerator" Weitere Informationen zum Erstellen benutzerdefinierter Trident -Rollen finden Sie hier.

Verwendung der ONTAP Befehlszeile

1. Erstellen Sie eine neue Rolle mit folgendem 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\>
```

Systemmanager verwenden

Führen Sie die folgenden Schritte im ONTAP System Manager aus:

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

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

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

Weitere Informationen finden Sie auf den folgenden Seiten:

- "Benutzerdefinierte Rollen für die Administration von ONTAP"oder"Benutzerdefinierte Rollen definieren"
- "Mit Rollen und Benutzern arbeiten"

Authentifizieren Sie Verbindungen mit bidirektionalem CHAP

Trident kann iSCSI-Sitzungen mit bidirektionalem CHAP authentifizieren. `ontap-san` Und `ontap-san-economy` Fahrer. Dies erfordert die Aktivierung der `useCHAP` Option in Ihrer Backend-Definition. Wenn eingestellt auf `true` Trident konfiguriert die Standard-Initiator-Sicherheit der SVM auf bidirektionales CHAP

und legt den Benutzernamen und die Geheimnisse aus der Backend-Datei fest. NetApp empfiehlt die Verwendung von bidirektionalem CHAP zur Authentifizierung von Verbindungen. Siehe die folgende Beispielkonfiguration:

```
---
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` Der Parameter ist eine boolesche Option, die nur einmal konfiguriert werden kann. Es ist standardmäßig auf „false“ eingestellt. Sobald Sie den Wert auf „true“ gesetzt haben, können Sie ihn nicht mehr auf „false“ setzen.

Zusätzlich zu `useCHAP=true` , Die `chapInitiatorSecret` , `chapTargetInitiatorSecret` , `chapTargetUsername` , Und `chapUsername` Die Felder müssen in der Backend-Definition enthalten sein. Die Geheimnisse können nach der Erstellung eines Backends durch Ausführen von geändert werden. `tridentctl update` .

So funktioniert es

Durch die Einstellung `useCHAP` Wenn dies der Fall ist, weist der Speicheradministrator Trident an, CHAP auf dem Speicher-Backend zu konfigurieren. Hierzu gehört Folgendes:

- CHAP auf der SVM einrichten:
  - Wenn der Standard-Initiator-Sicherheitstyp der SVM „Keine“ ist (Standardeinstellung) **und** keine LUNs im Volume vorhanden sind, legt Trident den Standard-Sicherheitstyp auf „Keine“ fest. `CHAP` und fahren Sie mit der Konfiguration des CHAP-Initiators sowie des Zielbenutzernamens und der zugehörigen Geheimnisse fort.
  - Wenn die SVM LUNs enthält, wird Trident CHAP auf der SVM nicht aktivieren. Dadurch wird sichergestellt, dass der Zugriff auf LUNs, die bereits auf der SVM vorhanden sind, nicht eingeschränkt wird.
- Konfiguration des CHAP-Initiators und des Zielbenutzernamens sowie der Geheimnisse; diese Optionen müssen in der Backend-Konfiguration angegeben werden (wie oben gezeigt).

Nachdem das Backend erstellt wurde, erstellt Trident ein entsprechendes `tridentbackend` CRD speichert die CHAP-Geheimnisse und Benutzernamen als Kubernetes-Geheimnisse. Alle von Trident auf diesem Backend erstellten PVs werden über CHAP eingebunden und angehängt.

Rotieren Sie Anmeldeinformationen und aktualisieren Sie Backends

Sie können die CHAP-Zugangsdaten aktualisieren, indem Sie die CHAP-Parameter in der `backend.json` Datei. Dies erfordert eine Aktualisierung der CHAP-Geheimnisse und die Verwendung von `tridentctl update` Befehl, um diese Änderungen widerzuspiegeln.

> Beim Aktualisieren der CHAP-Geheimnisse für ein Backend müssen Sie Folgendes verwenden: `tridentctl` um das Backend zu aktualisieren. Aktualisieren Sie die Anmeldeinformationen des Speicherclusters nicht über die ONTAP CLI oder den ONTAP System Manager, da Trident diese Änderungen nicht erkennen 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 unberührt; sie bleiben aktiv, wenn die Anmeldeinformationen von Trident auf der SVM aktualisiert werden. Neue Verbindungen verwenden die aktualisierten Zugangsdaten, bestehende Verbindungen bleiben weiterhin aktiv. Durch das Trennen und erneute Verbinden alter PV-Geräte werden diese mit den aktualisierten Zugangsdaten verwendet.

ONTAP SAN-Konfigurationsoptionen und Beispiele

Erfahren Sie, wie Sie ONTAP SAN-Treiber mit Ihrer Trident -Installation erstellen und

verwenden. Dieser Abschnitt enthält Beispiele für die Backend-Konfiguration und Details zur Zuordnung von Backends zu StorageClasses.

"ASA r2-Systeme"Sie unterscheiden sich von anderen ONTAP Systemen (ASA, AFF und FAS) in der Implementierung ihrer Speicherschicht. Diese Abweichungen wirken sich wie angegeben auf die Verwendung bestimmter Parameter aus. "Erfahren Sie mehr über die Unterschiede zwischen ASA r2-Systemen und anderen ONTAP Systemen.".

> Nur die `ontap-san` Der Treiber (mit iSCSI- und NVMe/TCP-Protokollen) wird für ASA r2-Systeme unterstützt.

In der Trident Backend-Konfiguration müssen Sie nicht angeben, dass Ihr System ASA r2 ist. Wenn Sie auswählen `ontap-san` als die `storageDriverName` Trident erkennt automatisch das ASA r2- oder das herkömmliche ONTAP System. Einige Backend-Konfigurationsparameter sind für ASA r2-Systeme nicht anwendbar, wie in der folgenden Tabelle vermerkt.

Backend-Konfigurationsoptionen

Die folgenden Tabellen enthalten die Backend-Konfigurationsoptionen:

Parameter	Beschreibung	Standard
`version`		Immer 1
`storageDriverName`	Name des Speichertreibers	`ontap-san`oder `ontap-san-economy`
`backendName`	Benutzerdefinierter Name oder das Speicher-Backend	Fahrername + "_" + dataLIF
`managementLIF`	IP-Adresse eines Cluster- oder SVM-Management-LIF. Es kann ein vollqualifizierter 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]` . Für einen nahtlosen MetroCluster Wechsel sieheMetroCluster Beispiel . > Wenn Sie die Anmeldeinformationen „vsadmin“ verwenden, `managementLIF` muss die des SVM sein; bei Verwendung von "Admin"-Anmeldeinformationen, `managementLIF` muss die des Clusters sein.	„10.0.0.1“, „[2001:1234:abcd::fefe]“

Parameter	Beschreibung	Standard
`dataLIF`	IP-Adresse des Protokolls LIF. Kann so eingestellt werden, dass IPv6-Adressen verwendet werden, wenn Trident mit dem IPv6-Flag installiert wurde. IPv6-Adressen müssen in eckigen Klammern definiert werden, z. B. `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . **Nicht für iSCSI angeben.** Trident Anwendungen"ONTAP Selective LUN Map" um die iSCSI LIFs zu ermitteln, die zum Aufbau einer Multipath-Sitzung benötigt werden. Es wird eine Warnung generiert, wenn `dataLIF` ist explizit definiert. **Für Metrocluster auslassen.** Siehe dieMetroCluster Beispiel .	Abgeleitet durch die SVM
`svm`	Zu verwendende virtuelle Speichermaschine **Für Metrocluster auslassen.** Siehe dieMetroCluster Beispiel .	Abgeleitet, wenn eine SVM `managementLIF` wird angegeben
`useCHAP`	Verwenden Sie CHAP zur Authentifizierung von iSCSI für ONTAP SAN-Treiber [Boolesch]. Auf einstellen `true` damit Trident bidirektionales CHAP als Standardauthentifizierung für die im Backend angegebene SVM konfiguriert und verwendet. Siehe "Bereiten Sie die Konfiguration des Backends mit ONTAP SAN-Treibern vor." für Details. **Nicht unterstützt für FCP oder NVMe/TCP.**	`false`
`chapInitiatorSecret`	Geheimnis des CHAP-Initiators. Erforderlich, wenn `useCHAP=true`	""
`labels`	Satz beliebiger JSON-formatierter Bezeichnungen, die auf Datenträger angewendet werden sollen	""
`chapTargetInitiatorSecret`	Geheimnis des CHAP-Zielinitiators. Erforderlich, wenn `useCHAP=true`	""
`chapUsername`	Eingehender Benutzername. Erforderlich, wenn `useCHAP=true`	""
`chapTargetUsername`	Zielbenutzername. Erforderlich, wenn `useCHAP=true`	""
`clientCertificate`	Base64-kodierter Wert des Clientzertifikats. Wird für zertifikatsbasierte Authentifizierung verwendet	""
`clientPrivateKey`	Base64-kodierter Wert des privaten Client-Schlüssels. Wird für zertifikatsbasierte Authentifizierung verwendet	""
`trustedCACertificate`	Base64-kodierter Wert des vertrauenswürdigen CA-Zertifikats. Optional. Wird für die zertifikatsbasierte Authentifizierung verwendet.	""

Parameter	Beschreibung	Standard
`username`	Für die Kommunikation mit dem ONTAP -Cluster ist ein Benutzername erforderlich. Wird für die auf Anmeldeinformationen basierende Authentifizierung verwendet. Informationen zur Active Directory-Authentifizierung finden Sie unter "Authentifizieren Sie Trident bei einem Backend-SVM mithilfe von Active Directory-Anmeldeinformationen".	""
`password`	Für die Kommunikation mit dem ONTAP -Cluster ist ein Passwort erforderlich. Wird für die auf Anmeldeinformationen basierende Authentifizierung verwendet. Informationen zur Active Directory-Authentifizierung finden Sie unter "Authentifizieren Sie Trident bei einem Backend-SVM mithilfe von Active Directory-Anmeldeinformationen".	""
`svm`	Zu verwendende virtuelle Speichermaschine	Abgeleitet, wenn eine SVM `managementLIF` wird angegeben
`storagePrefix`	Präfix, das beim Bereitstellen neuer Volumes in der SVM verwendet wird. Kann später nicht geändert werden. Um diesen Parameter zu aktualisieren, müssen Sie ein neues Backend erstellen.	`trident`
`aggregate`	Aggregat für die Bereitstellung (optional; falls festgelegt, muss es der SVM zugewiesen werden). Für die `ontap-nas-flexgroup` Treiber, diese Option wird ignoriert. Falls kein Aggregat zugewiesen ist, kann jedes der verfügbaren Aggregate zur Bereitstellung eines FlexGroup Volumes verwendet werden. Wenn das Aggregat in SVM aktualisiert wird, wird es in Trident automatisch durch Abfrage von SVM aktualisiert, ohne dass der Trident Controller neu gestartet werden muss. Wenn Sie in Trident ein bestimmtes Aggregat zur Bereitstellung von Volumes konfiguriert haben und dieses Aggregat umbenannt oder aus der SVM verschoben wird, wechselt das Backend in Trident in den Fehlerzustand, während es das SVM-Aggregat abfragt. Sie müssen entweder das Aggregat in ein auf der SVM vorhandenes ändern oder es vollständig entfernen, um das Backend wieder online zu bringen. **Nicht für ASA r2-Systeme angeben**.	""

Parameter	Beschreibung	Standard
`limitAggregateUsage`	Die Bereitstellung schlägt fehl, wenn die Auslastung diesen Prozentsatz überschreitet. Wenn Sie ein Amazon FSx for NetApp ONTAP -Backend verwenden, geben Sie dies nicht an. `limitAggregateUsage` . Die bereitgestellten `fsxadmin` Und `vsadmin` enthalten nicht die erforderlichen Berechtigungen, um die aggregierte Nutzung abzurufen und sie mit Trident einzuschränken. **Nicht für ASA r2-Systeme angeben**.	"" (wird nicht standardmäßig erzwungen)
`limitVolumeSize`	Die Bereitstellung schlägt fehl, wenn die angeforderte Volume-Größe diesen Wert überschreitet. Außerdem wird die maximale Größe der von ihm verwalteten Volumes für LUNs beschränkt.	"" (wird nicht standardmäßig erzwungen)
`lunsPerFlexvol`	Die maximale Anzahl an LUNs pro Flexvol muss im Bereich [50, 200] liegen.	`100`
`debugTraceFlags`	Debug-Flags zur Verwendung bei der Fehlersuche. Beispiel: {"api":false, "method":true} Verwenden Sie dies nur, wenn Sie eine Fehlerbehebung durchführen und einen detaillierten Protokollauszug benötigen.	`null`

Parameter	Beschreibung	Standard
`useREST`	Boolescher Parameter zur Verwendung von ONTAP REST-APIs. `useREST`Wenn eingestellt auf `true` Trident verwendet ONTAP REST-APIs zur Kommunikation mit dem Backend; wenn eingestellt auf `false` Trident verwendet ONTAPI (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 `ontapi` Anwendung. Dies wird durch die vordefinierte Bedingung erfüllt. `vsadmin` Und `cluster-admin` Rollen. Ab der Trident Version 24.06 und ONTAP 9.15.1 oder höher, `useREST` ist eingestellt auf `true` Standardmäßig; ändern `useREST` Zu `false` ONTAPI (ZAPI)-Aufrufe verwenden. `useREST`ist vollständig für NVMe/TCP qualifiziert. NVMe wird nur mit ONTAP REST APIs unterstützt und nicht mit ONTAPI (ZAPI). **Falls angegeben, immer auf setzen `true` für ASA r2-Systeme**.	`true`für ONTAP 9.15.1 oder höher, andernfalls `false`.
`sanType`	Zur Auswahl verwenden `iscsi` für iSCSI, `nvme` für NVMe/TCP oder `fcp` für SCSI über Fibre Channel (FC).	`iscsi`falls leer

Parameter	Beschreibung	Standard
`formatOptions`	Verwenden `formatOptions` um Befehlszeilenargumente für die `mkfs` Befehl, der immer dann angewendet wird, wenn ein Datenträger formatiert wird. Dies ermöglicht es Ihnen, die Lautstärke nach Ihren Wünschen zu formatieren. Stellen Sie sicher, dass Sie die Formatoptionen analog zu den Optionen des Befehls mkfs angeben, jedoch ohne den Gerätepfad. Beispiel: "-E nodiscard" **Unterstützt für `ontap-san` Und `ontap-san-economy` Treiber mit iSCSI-Protokoll. Zusätzlich wird dies für ASA r2-Systeme bei Verwendung der iSCSI- und NVMe/TCP-Protokolle unterstützt.**	
`limitVolumePoolSize`	Maximal anforderbare FlexVol Größe bei Verwendung von LUNs im ontap-san-economy-Backend.	"" (wird nicht standardmäßig erzwungen)
`denyNewVolumePools`	Beschränkt `ontap-san-economy` Backends daran zu hindern, neue FlexVol -Volumes zu erstellen, die ihre LUNs enthalten. Für die Bereitstellung neuer PVs werden ausschließlich bereits vorhandene Flexvols verwendet.	

Empfehlungen zur Verwendung von formatOptions

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

-E nodiscard:

- Blöcke sollten beim Erstellen des Dateisystems (mkfs) nicht verworfen werden (das anfängliche Verwerfen von Blöcken ist bei Solid-State-Geräten und dünn bereitgestellten Speichern sinnvoll). Dies ersetzt die veraltete Option "-K" und ist auf alle Dateisysteme (xfs, ext3 und ext4) anwendbar.

Authentifizieren Sie Trident bei einem Backend-SVM mithilfe von Active Directory-Anmeldeinformationen

Sie können Trident so konfigurieren, dass es sich mit Active Directory (AD)-Anmeldeinformationen bei einem Back-End-SVM authentifiziert. Bevor ein AD-Konto auf die SVM zugreifen kann, müssen Sie den AD-Domänencontrollerzugriff auf den Cluster oder die SVM konfigurieren. Für die Clusterverwaltung mit einem AD-Konto müssen Sie einen Domänentunnel erstellen. Siehe "Konfigurieren des Active Directory-Domänencontrollerzugriffs in ONTAP" für Details.

Schritte

1. Konfigurieren Sie die DNS-Einstellungen (Domain Name System) für eine Back-End-SVM:

   ```
   vserver services dns create -vserver <svm_name> -dns-servers
   <dns_server_ip1>,<dns_server_ip2>
   ```

2. Führen Sie den folgenden Befehl aus, um ein Computerkonto für die SVM in Active Directory zu erstellen:

   ```
   vserver active-directory create -vserver DataSVM -account-name ADSERVER1
   -domain demo.netapp.com
   ```

3. Verwenden Sie diesen Befehl, um einen AD-Benutzer oder eine AD-Gruppe zum Verwalten des Clusters oder SVM zu erstellen

   ```
   security login create -vserver <svm_name> -user-or-group-name
   <ad_user_or_group> -application <application> -authentication-method domain
   -role vsadmin
   ```

4. Legen Sie in der Trident Backend-Konfigurationsdatei Folgendes fest: `username` Und `password` Parameter auf den AD-Benutzer- oder Gruppennamen bzw. das Kennwort.

Backend-Konfigurationsoptionen für die Bereitstellung von Volumes

Sie können die Standardbereitstellung mithilfe dieser Optionen steuern. `defaults` Abschnitt der Konfiguration. Ein Beispiel finden Sie in den folgenden Konfigurationsbeispielen.

Parameter	Beschreibung	Standard
`spaceAllocation`	Speicherplatzzuweisung für LUNs	"true" **Falls angegeben, auf setzen `true` für ASA r2-Systeme**.
`spaceReserve`	Platzreservierungsmodus; "keine" (dünn) oder "Volumen" (dick). **Einstellen auf `none` für ASA r2** -Systeme.	"keiner"
`snapshotPolicy`	Zu verwendende Snapshot-Richtlinie. **Einstellen auf `none` für ASA r2-Systeme**.	"keiner"
`qosPolicy`	Die QoS-Richtliniengruppe soll den erstellten Volumes zugewiesen werden. Wählen Sie pro Speicherpool/Backend entweder qosPolicy oder adaptiveQosPolicy. Die Verwendung von QoS-Richtliniengruppen mit Trident erfordert ONTAP 9.8 oder höher. Sie sollten eine nicht gemeinsam genutzte QoS-Richtliniengruppe verwenden und sicherstellen, dass die Richtliniengruppe auf jeden einzelnen Bestandteil angewendet wird. Eine gemeinsam genutzte QoS-Richtliniengruppe setzt die Obergrenze für den Gesamtdurchsatz aller Workloads durch.	""
`adaptiveQosPolicy`	Adaptive QoS-Richtliniengruppe, die den erstellten Volumes zugewiesen werden soll. Wählen Sie pro Speicherpool/Backend entweder qosPolicy oder adaptiveQosPolicy aus.	""
`snapshotReserve`	Prozentsatz des für Snapshots reservierten Speichervolumens. **Nicht für ASA r2-Systeme angeben**.	"0" wenn `snapshotPolicy` ist "keine", ansonsten ""
`splitOnClone`	Beim Erstellen eines Klons diesen von seinem Elternklon trennen	"FALSCH"

Parameter	Beschreibung	Standard
`encryption`	Aktivieren Sie die NetApp Volumeverschlüsselung (NVE) auf dem neuen Volume; Standardwert ist `false` . Um diese Option nutzen zu können, muss NVE auf dem Cluster lizenziert und aktiviert sein. Wenn NAE im Backend aktiviert ist, wird jedes in Trident bereitgestellte Volume NAE-fähig sein. Weitere Informationen finden Sie unter:"Wie Trident mit NVE und NAE zusammenarbeitet" .	"false" **Falls angegeben, auf setzen. `true` für ASA r2-Systeme**.
`luksEncryption`	LUKS-Verschlüsselung aktivieren. Siehe"Verwenden Sie Linux Unified Key Setup (LUKS)." .	**Einstellen auf `false` für ASA r2-Systeme**.
`tieringPolicy`	Tiering-Richtlinie auf "keine" setzen **Für ASA r2-Systeme nicht angeben** .	
`nameTemplate`	Vorlage zum Erstellen benutzerdefinierter Datenträgernamen.	""

Beispiele für die Volumenbereitstellung

Hier ist ein Beispiel mit vordefinierten Standardwerten:

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

Für alle mit der `ontap-san` Der Trident -Treiber erweitert die FlexVol -Kapazität um zusätzliche 10 Prozent, um die LUN-Metadaten aufzunehmen. Die LUN wird mit der exakten Größe bereitgestellt, die der Benutzer im PVC anfordert. Trident erhöht den FlexVol um 10 Prozent (wird in ONTAP als verfügbare Größe angezeigt). Die Nutzer erhalten nun die von ihnen angeforderte nutzbare Speicherkapazität. Diese Änderung verhindert auch, dass LUNs schreibgeschützt werden, es sei denn, der verfügbare Speicherplatz wird vollständig genutzt. Dies gilt nicht für ontap-san-economy.

Für Backends, die definieren `snapshotReserve` Trident berechnet die Größe von Volumina wie folgt:

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

Die 1.1 sind die zusätzlichen 10 Prozent, die Trident zum FlexVol hinzufügt, um die LUN-Metadaten unterzubringen. Für `snapshotReserve` = 5% und PVC-Anforderung = 5 GiB, die Gesamtvolumengröße beträgt 5,79 GiB und die verfügbare Größe beträgt 5,5 GiB. Der `volume show` Der Befehl sollte ähnliche Ergebnisse wie in diesem Beispiel liefern:

Aktuell ist die Größenänderung die einzige Möglichkeit, die neue Berechnung für ein bestehendes Volumen zu nutzen.

Beispiele für minimale Konfigurationen

Die folgenden Beispiele zeigen Basiskonfigurationen, bei denen die meisten Parameter auf Standardwerte eingestellt bleiben. Dies ist die einfachste Möglichkeit, ein Backend zu definieren.

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

ONTAP SAN-Beispiel

Dies ist eine Basiskonfiguration unter Verwendung der `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>
```

MetroCluster Beispiel

Sie können das Backend so konfigurieren, dass eine manuelle Aktualisierung der Backend-Definition nach einem Switchover und Switchback vermieden wird."SVM-Replikation und -Wiederherstellung" .

Für einen nahtlosen Übergang und Rückwechsel geben Sie die SVM wie folgt an: `managementLIF` und lassen Sie die `svm` Parameter. Beispiel:

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

ONTAP SAN Wirtschaftsbeispiel

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

Beispiel für zertifikatsbasierte Authentifizierung

In diesem Beispiel für eine einfache Konfiguration `clientCertificate` , `clientPrivateKey` , Und `trustedCACertificate` (optional, falls eine vertrauenswürdige Zertifizierungsstelle verwendet wird) werden in `backend.json` und nehmen Sie die Base64-kodierten Werte des Clientzertifikats, des privaten Schlüssels bzw. des vertrauenswürdigen CA-Zertifikats.

```
---
version: 1
storageDriverName: ontap-san
backendName: DefaultSANBackend
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
```

Beispiele für bidirektionale CHAP-Programme

Diese Beispiele erstellen ein Backend mit `useCHAP` eingestellt auf `true` .

ONTAP SAN CHAP Beispiel

```
---
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>
```

ONTAP SAN Wirtschaft CHAP Beispiel

```
---
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>
```

NVMe/TCP-Beispiel

Sie benötigen eine SVM, die mit NVMe auf Ihrem ONTAP Backend konfiguriert ist. 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
```

SCSI über FC (FCP) Beispiel

Sie benötigen eine SVM, die mit FC auf Ihrem ONTAP Backend konfiguriert ist. Dies ist eine grundlegende Backend-Konfiguration für FC.

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

Backend-Konfigurationsbeispiel mit nameTemplate

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

Beispiel für formatOptions für den ontap-san-economy-Treiber

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

Beispiele für Backends mit virtuellen Pools

In diesen Beispiel-Backend-Definitionsdateien sind spezifische Standardwerte für alle Speicherpools festgelegt, wie zum Beispiel `spaceReserve` bei keiner, `spaceAllocation` bei falsch und `encryption` bei falsch. Die virtuellen Pools werden im Speicherbereich definiert.

Trident legt Bereitstellungsbezeichnungen im Feld „Kommentare“ fest. Kommentare werden auf dem FlexVol volume festgelegt. Trident kopiert bei der Bereitstellung alle auf einem virtuellen Pool vorhandenen Labels auf das Speichervolume. Zur Vereinfachung können Speicheradministratoren Bezeichnungen pro virtuellem Pool definieren und Volumes nach Bezeichnung gruppieren.

In diesen Beispielen legen einige der Speicherpools ihre eigenen Einstellungen fest. `spaceReserve` , `spaceAllocation` , Und `encryption` Werte, und einige Pools überschreiben die Standardwerte.

ONTAP SAN-Beispiel

```
---
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"
```

ONTAP SAN Wirtschaftsbeispiel

```
---
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"
```

NVMe/TCP-Beispiel

```
---
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"
```

Backends StorageClasses zuordnen

Die folgenden StorageClass-Definitionen beziehen sich auf dieBeispiele für Backends mit virtuellen Pools . Verwenden des `parameters.selector` Im Feld „StorageClass“ wird für jede StorageClass angegeben, welche virtuellen Pools zum Hosten eines Volumes verwendet werden können. Das Volumen wird die im gewählten virtuellen Pool definierten Aspekte aufweisen.

- Der `protection-gold` Die StorageClass wird dem ersten virtuellen Pool im `ontap-san` Backend. Dies ist der einzige Pool, der Schutz auf Goldniveau bietet.

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

- Der `protection-not-gold` Die StorageClass wird dem zweiten und dritten virtuellen Pool zugeordnet. `ontap-san` Backend. Dies sind die einzigen Pools, die ein anderes Schutzniveau als Gold bieten.

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

- Der `app-mysqldb` Die StorageClass wird dem dritten virtuellen Pool zugeordnet. `ontap-san-economy` Backend. Dies ist der einzige Pool, der eine Speicherpoolkonfiguration für Anwendungen vom Typ mysqldb bietet.

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

- Der `protection-silver-creditpoints-20k` Die StorageClass wird dem zweiten virtuellen Pool zugeordnet. `ontap-san` Backend. Dies ist der einzige Pool, der Schutz auf Silber-Niveau und 20000 Kreditpunkte bietet.

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

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

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

- Der `my-test-app-sc` Die StorageClass wird der folgenden zugeordnet: `testAPP` virtueller Pool im `ontap-san` Fahrer mit `sanType: nvme` . Dies ist das einzige Poolangebot `testApp` .

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

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

ONTAP NAS-Treiber

ONTAP NAS-Treiberübersicht

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

ONTAP NAS-Treiberdetails

Trident stellt die folgenden NAS-Speichertreiber zur Verfügung, um mit dem ONTAP Cluster zu kommunizieren. Unterstützte Zugriffsmodi sind: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Treiber	Protokoll	Lautstärke modus	Unterstützte Zugriffsmodi	Unterstützte Dateisysteme
`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 `ontap-san-economy` nur wenn die Anzahl der dauerhaften Speichernutzungen voraussichtlich höher sein wird als"Unterstützte ONTAP Lautstärkebegrenzungen" .
- Verwenden `ontap-nas-economy` nur wenn die Anzahl der dauerhaften Speichernutzungen voraussichtlich höher sein wird als"Unterstützte ONTAP Lautstärkebegrenzungen" und die `ontap-san-economy` Der Treiber kann nicht verwendet werden.
- Nicht verwenden `ontap-nas-economy` wenn Sie mit einem Bedarf an Datenschutz, Notfallwiederherstellung oder Mobilität rechnen.
- NetApp empfiehlt die Verwendung von Flexvol Autogrow nicht in allen ONTAP -Treibern, außer ontap-san. Als Ausweichlösung unterstützt Trident die Verwendung von Snapshot-Reserven und skaliert Flexvol-Volumes entsprechend.

Benutzerberechtigungen

Trident wird voraussichtlich entweder als ONTAP oder SVM-Administrator ausgeführt, typischerweise unter Verwendung von `admin` Clusterbenutzer oder ein `vsadmin` SVM-Benutzer oder ein Benutzer mit einem anderen Namen, der die gleiche Rolle hat.

Bei Amazon FSx for NetApp ONTAP Bereitstellungen erwartet Trident , dass es entweder als ONTAP oder SVM-Administrator ausgeführt wird und den Cluster nutzt. `fsxadmin` Benutzer oder ein `vsadmin` SVM-Benutzer oder ein Benutzer mit einem anderen Namen, der die gleiche Rolle hat. Der `fsxadmin` Der Benutzer ist ein eingeschränkter Ersatz für den Cluster-Administratorbenutzer.

Wenn Sie die `limitAggregateUsage` Für diesen Parameter sind Cluster-Administratorrechte erforderlich. Bei der Verwendung von Amazon FSx for NetApp ONTAP mit Trident `limitAggregateUsage` Der Parameter funktioniert nicht mit dem `vsadmin` Und `fsxadmin` Benutzerkonten. Die Konfiguration schlägt fehl, wenn Sie diesen Parameter angeben.

Es ist zwar möglich, innerhalb von ONTAP eine restriktivere Rolle zu erstellen, die ein Trident -Treiber verwenden kann, wir empfehlen dies jedoch nicht. Die meisten neuen Versionen von Trident werden zusätzliche APIs aufrufen, die berücksichtigt werden müssen, was Aktualisierungen schwierig und fehleranfällig macht.

Bereiten Sie die Konfiguration eines Backends mit ONTAP NAS-Treibern vor.

Machen Sie sich mit den Anforderungen, Authentifizierungsoptionen und Exportrichtlinien für die Konfiguration eines ONTAP Backends mit ONTAP -NAS-Treibern vertraut.

Anforderungen

- Für alle ONTAP Backends verlangt Trident , dass mindestens ein Aggregat dem SVM zugewiesen wird.
- Sie können mehrere Treiber gleichzeitig ausführen und Speicherklassen erstellen, die auf den einen oder anderen Treiber verweisen. Beispielsweise könnten Sie eine Gold-Klasse konfigurieren, die Folgendes verwendet: `ontap-nas` Fahrer und eine Bronze-Klasse, die den `ontap-nas-economy` eins.
- Auf allen Ihren Kubernetes-Worker-Knoten müssen die entsprechenden NFS-Tools installiert sein. Siehe"hier," für weitere Details.
- Trident unterstützt SMB-Volumes nur, wenn sie in Pods eingebunden sind, die auf Windows-Knoten laufen. Siehe Bereiten Sie die Bereitstellung von SMB-Volumes vor für Details.

Authentifizieren Sie das ONTAP Backend

Trident bietet zwei Modi zur Authentifizierung eines ONTAP Backends.

- Anmeldeinformationsbasiert: Dieser Modus erfordert ausreichende Berechtigungen für das ONTAP Backend. Es wird empfohlen, ein Konto zu verwenden, das einer vordefinierten Sicherheitsanmelderolle zugeordnet ist, wie zum Beispiel `admin` oder `vsadmin` um maximale Kompatibilität mit ONTAP Versionen zu gewährleisten.
- Zertifikatsbasiert: In diesem Modus ist ein auf dem Backend installiertes Zertifikat erforderlich, damit Trident mit einem ONTAP Cluster kommunizieren kann. Hierbei müssen in der Backend-Definition Base64-kodierte Werte des Client-Zertifikats, des Schlüssels und des vertrauenswürdigen CA-Zertifikats (falls verwendet, empfohlen) enthalten sein.

Sie können bestehende Backends aktualisieren, um zwischen anmeldeinformationsbasierten und zertifikatsbasierten Methoden zu wechseln. Es wird jedoch jeweils nur eine Authentifizierungsmethode unterstützt. Um zu einer anderen Authentifizierungsmethode zu wechseln, müssen Sie die bestehende Methode aus der Backend-Konfiguration entfernen.

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

Aktivieren Sie die anmeldeinformationsbasierte Authentifizierung

Trident benötigt die Anmeldeinformationen eines SVM-/Cluster-Administrators, um mit dem ONTAP Backend zu kommunizieren. Es wird empfohlen, standardisierte, vordefinierte Rollen zu verwenden, wie zum Beispiel `admin` oder `vsadmin` . Dies gewährleistet die Vorwärtskompatibilität mit zukünftigen ONTAP Versionen, die möglicherweise Feature-APIs zur Verwendung durch zukünftige Trident Versionen bereitstellen. Eine benutzerdefinierte Sicherheitsanmelderolle kann erstellt und mit Trident verwendet werden, dies wird jedoch nicht empfohlen.

Eine beispielhafte Backend-Definition sieht folgendermaßen aus:

YAML

```yaml
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
credentials:
  name: secret-backend-creds
```

JSON

```json
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "credentials": {
        "name": "secret-backend-creds"
    }
}
```

Beachten Sie, dass die Backend-Definition der einzige Ort ist, an dem die Anmeldeinformationen im Klartext gespeichert werden. Nach der Erstellung des Backends werden Benutzernamen und Passwörter mit Base64 kodiert und als Kubernetes-Secrets gespeichert. Die Erstellung/Aktualisierung eines Backends ist der einzige Schritt, der Kenntnisse der Zugangsdaten erfordert. Daher handelt es sich um eine ausschließlich für Administratoren zulässige Operation, die vom Kubernetes-/Speicheradministrator durchgeführt werden muss.

Zertifikatsbasierte Authentifizierung aktivieren

Neue und bestehende Backends können ein Zertifikat verwenden und mit dem ONTAP Backend kommunizieren. Für die Backend-Definition werden drei Parameter benötigt.

- clientCertificate: Base64-kodierter Wert des Clientzertifikats.
- clientPrivateKey: Base64-kodierter Wert des zugehörigen privaten Schlüssels.
- trustedCACertificate: Base64-kodierter Wert des vertrauenswürdigen CA-Zertifikats. Bei Verwendung einer vertrauenswürdigen Zertifizierungsstelle muss dieser Parameter angegeben werden. Dies kann ignoriert werden, wenn keine vertrauenswürdige Zertifizierungsstelle verwendet wird.

Ein typischer Arbeitsablauf umfasst die folgenden Schritte.

Schritte

1. Generieren Sie ein Clientzertifikat und einen Schlüssel. Beim Generieren muss der allgemeine Name (CN) auf den ONTAP Benutzer gesetzt werden, der sich authentifizieren soll.

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

2. Fügen Sie dem ONTAP -Cluster ein vertrauenswürdiges CA-Zertifikat hinzu. Dies könnte bereits vom Speicheradministrator erledigt werden. Ignorieren, falls keine vertrauenswürdige Zertifizierungsstelle 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 Clientzertifikat und den Schlüssel (aus Schritt 1) auf dem ONTAP Cluster.

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

4. Bestätigen Sie, dass die ONTAP Sicherheitsanmeldungsrolle die folgenden Funktionen unterstützt: `cert` Authentifizierungsmethode.

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

5. Testen Sie die Authentifizierung mit dem generierten Zertifikat. Ersetzen Sie < ONTAP Management LIF> und <vserver name> durch die Management LIF IP-Adresse und den SVM-Namen. Sie müssen sicherstellen, dass die Servicerichtlinie des LIF auf Folgendes eingestellt 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. Zertifikat, Schlüssel und vertrauenswürdiges CA-Zertifikat mit Base64 kodieren.

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

7. Erstellen Sie ein Backend unter Verwendung der im vorherigen Schritt erhaltenen Werte.

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

#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Aktualisieren Sie die Authentifizierungsmethoden oder ändern Sie die Anmeldeinformationen.

Sie können ein bestehendes Backend aktualisieren, um eine andere Authentifizierungsmethode zu verwenden oder um die Anmeldeinformationen zu ändern. Dies funktioniert in beide Richtungen: Backends, die Benutzername/Passwort verwenden, können auf die Verwendung von Zertifikaten umgestellt werden; Backends, die Zertifikate verwenden, können auf Benutzername/Passwort-basiert umgestellt werden. Dazu müssen Sie die bestehende Authentifizierungsmethode entfernen und die neue Authentifizierungsmethode hinzufügen. Verwenden Sie anschließend die aktualisierte Datei backend.json, die die erforderlichen Parameter enthält, um die Ausführung durchzuführen. `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 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Beim Ändern von Passwörtern muss der Speicheradministrator zuerst das Passwort für den Benutzer auf ONTAP aktualisieren. Anschließend erfolgt ein Backend-Update. Bei der Zertifikatsrotation können dem Benutzer mehrere Zertifikate hinzugefügt werden. Anschließend wird das Backend aktualisiert, um das neue Zertifikat zu verwenden. Danach kann das alte Zertifikat aus dem ONTAP Cluster gelöscht werden.

Durch die Aktualisierung des Backends wird der Zugriff auf bereits erstellte Volumes nicht beeinträchtigt, und auch später hergestellte Volume-Verbindungen werden nicht beeinträchtigt. Ein erfolgreiches Backend-Update zeigt an, dass Trident mit dem ONTAP -Backend kommunizieren und zukünftige Volumenoperationen bewältigen kann.

Erstellen einer benutzerdefinierten ONTAP Rolle für Trident

Sie können eine ONTAP Clusterrolle mit minimalen Berechtigungen erstellen, sodass Sie für Operationen in Trident nicht die ONTAP Administratorrolle verwenden müssen. Wenn Sie den Benutzernamen in einer Trident Backend-Konfiguration angeben, verwendet Trident die von Ihnen erstellte ONTAP Clusterrolle, um die

Operationen durchzuführen.

Siehe"Trident -Benutzerrollengenerator" Weitere Informationen zum Erstellen benutzerdefinierter Trident -Rollen finden Sie hier.

Verwendung der ONTAP Befehlszeile

1. Erstellen Sie eine neue Rolle mit folgendem 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\>
```

Systemmanager verwenden

Führen Sie die folgenden Schritte im ONTAP System Manager aus:

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

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

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

Weitere Informationen finden Sie auf den folgenden Seiten:

- "Benutzerdefinierte Rollen für die Administration von ONTAP"oder"Benutzerdefinierte Rollen definieren"
- "Mit Rollen und Benutzern arbeiten"

NFS-Exportrichtlinien verwalten

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

Trident bietet zwei Optionen für die Arbeit mit Exportrichtlinien:

- Trident kann die Exportrichtlinie selbst dynamisch verwalten; in diesem Betriebsmodus gibt der Speicheradministrator eine Liste von CIDR-Blöcken an, die zulässige IP-Adressen darstellen. Trident fügt bei der Veröffentlichung automatisch die entsprechenden Knoten-IPs, die in diese Bereiche fallen, zur Exportrichtlinie hinzu. Alternativ werden, wenn keine CIDRs angegeben sind, alle globalen Unicast-IPs, die auf dem Knoten gefunden werden, auf dem das Volume veröffentlicht wird, der Exportrichtlinie hinzugefügt.
- Speicheradministratoren können eine Exportrichtlinie erstellen und Regeln manuell hinzufügen. Trident verwendet die Standardexportrichtlinie, es sei denn, in der Konfiguration ist ein anderer Exportrichtlinienname angegeben.

Exportrichtlinien dynamisch verwalten

Trident bietet die Möglichkeit, Exportrichtlinien für ONTAP Backends dynamisch zu verwalten. Dies gibt dem Speicheradministrator die Möglichkeit, einen zulässigen Adressraum für Worker-Knoten-IPs festzulegen, anstatt explizite Regeln manuell zu definieren. Es vereinfacht die Verwaltung der Exportrichtlinien erheblich; Änderungen an den Exportrichtlinien erfordern keinen manuellen Eingriff mehr in den Speichercluster. Darüber hinaus trägt dies dazu bei, den Zugriff auf den Speichercluster auf Worker-Knoten zu beschränken, die Volumes einbinden und über IPs im angegebenen Bereich verfügen, wodurch eine feingranulare und automatisierte Verwaltung unterstützt wird.

> ⓘ Verwenden Sie keine Netzwerkadressübersetzung (NAT), wenn Sie dynamische Exportrichtlinien verwenden. Bei Verwendung von NAT sieht der Speicherkontroller die Frontend-NAT-Adresse und nicht die tatsächliche IP-Hostadresse. Daher wird der Zugriff verweigert, wenn in den Exportregeln keine Übereinstimmung gefunden wird.

Beispiel

Es gibt zwei Konfigurationsoptionen, die verwendet werden müssen. Hier ist ein Beispiel für eine 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
```

Bei Verwendung dieser Funktion müssen Sie sicherstellen, dass für den Root-Junction in Ihrer SVM eine zuvor erstellte Exportrichtlinie mit einer Exportregel existiert, die den CIDR-Block des Knotens zulässt (z. B. die Standardexportrichtlinie). Befolgen Sie stets die von NetApp empfohlenen Best Practices, um eine SVM für Trident zu dedizieren.

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

- `autoExportPolicy`ist eingestellt auf `true`. Dies deutet darauf hin, dass Trident für jedes mit diesem Backend bereitgestellte Volume eine Exportrichtlinie erstellt. `svm1` SVM und handhaben das Hinzufügen und Löschen von Regeln mithilfe von `autoexportCIDRs` Adressblöcke. Solange ein Volume nicht an einen Knoten angehängt ist, verwendet das Volume eine leere Exportrichtlinie ohne Regeln, um unerwünschten Zugriff auf dieses Volume zu verhindern. Wenn ein Volume auf einem Knoten veröffentlicht wird, erstellt Trident eine Exportrichtlinie mit demselben Namen wie der zugrunde liegende Qtree, der die Knoten-IP innerhalb des angegebenen CIDR-Blocks enthält. Diese IPs werden auch der Exportrichtlinie hinzugefügt, die vom übergeordneten FlexVol volume verwendet wird.
  - Beispiel:
    - Backend-UUID 403b5326-8482-40db-96d0-d83fb3f4daec
    - `autoExportPolicy`eingestellt auf `true`
    - Speicherpräfix `trident`
    - PVC UUID a79bcf5f-7b6d-4a40-9876-e2551f159c1c
    - Ein Qtree mit dem Namen trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c erstellt eine Exportrichtlinie für den FlexVol mit dem Namen `trident-403b5326-8482-40db96d0-d83fb3f4daec` , eine Exportrichtlinie für den Qtree namens `trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c` und eine leere Exportpolitik namens `trident_empty` auf der SVM. Die Regeln für die FlexVol -Exportrichtlinie bilden eine Obermenge aller in den Qtree-Exportrichtlinien enthaltenen Regeln. Die leere Exportrichtlinie wird von allen Volumes wiederverwendet, die nicht angehängt sind.
- `autoExportCIDRs`enthält eine Liste von Adressblöcken. Dieses Feld ist optional und hat standardmäßig den Wert ["0.0.0.0/0", "::/0"]. Falls nicht definiert, fügt Trident alle global gültigen Unicast-Adressen hinzu, die auf den Worker-Knoten mit Veröffentlichungen gefunden werden.

In diesem Beispiel, `192.168.0.0/24` Ein Adressraum wird bereitgestellt. Dies bedeutet, dass Kubernetes-Knoten-IPs, die in diesen Adressbereich fallen und Veröffentlichungen enthalten, der 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 überprüft sie anhand der bereitgestellten Adressblöcke. `autoExportCIDRs` Zum Zeitpunkt der Veröffentlichung erstellt Trident nach dem Filtern der IPs die Exportrichtlinienregeln für die Client-IPs des Knotens, auf dem die Veröffentlichung erfolgt.

Sie können aktualisieren `autoExportPolicy` Und `autoExportCIDRs` für Backends, nachdem Sie diese erstellt haben. Sie können neue CIDRs für ein automatisch verwaltetes Backend hinzufügen oder bestehende CIDRs löschen. Beim Löschen von CIDRs ist darauf zu achten, dass bestehende Verbindungen nicht unterbrochen werden. Sie können diese Option auch deaktivieren. `autoExportPolicy` für ein Backend und greifen Sie auf eine manuell erstellte Exportrichtlinie zurück. Dies erfordert die Einstellung der `exportPolicy` Parameter in Ihrer Backend-Konfiguration.

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

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

Wenn ein Knoten entfernt wird, überprüft Trident alle Exportrichtlinien, um die dem Knoten entsprechenden Zugriffsregeln zu entfernen. Durch das Entfernen dieser Knoten-IP aus den Exportrichtlinien der verwalteten Backends verhindert Trident unerwünschte Mounts, es sei denn, diese IP wird von einem neuen Knoten im Cluster wiederverwendet.

Bei bereits bestehenden Backends wird das Backend aktualisiert mit `tridentctl update backend` stellt sicher, dass Trident die Exportrichtlinien automatisch verwaltet. Dadurch werden bei Bedarf zwei neue Exportrichtlinien erstellt, die nach der UUID und dem Qtree-Namen des Backends benannt sind. Auf dem Backend vorhandene Volumes verwenden nach dem Aushängen und erneuten Einhängen die neu erstellten Exportrichtlinien.

Das Löschen eines Backends mit automatisch verwalteten Exportrichtlinien löscht die dynamisch erstellte Exportrichtlinie. Wird das Backend neu erstellt, wird es als neues Backend behandelt und führt zur Erstellung einer neuen Exportrichtlinie.

Wenn die IP-Adresse eines aktiven Knotens aktualisiert wird, müssen Sie den Trident Pod auf dem Knoten neu starten. Trident wird anschließend die Exportrichtlinie für die von ihm verwalteten Backends aktualisieren, um diese IP-Änderung widerzuspiegeln.

Bereiten Sie die Bereitstellung von SMB-Volumes vor

Mit ein wenig zusätzlicher Vorbereitung können Sie SMB-Volumes bereitstellen mithilfe von `ontap-nas` Fahrer.

Sie müssen sowohl das NFS- als auch das SMB/CIFS-Protokoll auf der SVM konfigurieren, um eine `ontap-nas-economy` SMB-Volume für ONTAP On-Premises-Cluster. Wenn eines dieser Protokolle nicht konfiguriert wird, schlägt die Erstellung des SMB-Volumes fehl.

`autoExportPolicy`wird für SMB-Volumes nicht unterstützt.

Bevor Sie beginnen

Bevor Sie SMB-Volumes bereitstellen können, benötigen Sie Folgendes.

- Ein Kubernetes-Cluster mit einem Linux-Controller-Knoten und mindestens einem Windows-Worker-Knoten, auf dem Windows Server 2022 ausgeführt wird. Trident unterstützt SMB-Volumes nur, wenn sie in Pods eingebunden sind, die auf Windows-Knoten laufen.
- Mindestens ein Trident Geheimnis, das Ihre Active Directory-Anmeldeinformationen enthält. Um Geheimnisse zu generieren `smbcreds` :

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

- Ein als Windows-Dienst konfigurierter CSI-Proxy. Um einen zu konfigurieren `csi-proxy` , siehe"GitHub: CSI-Proxy" oder"GitHub: CSI-Proxy für Windows" für Kubernetes-Knoten, die unter Windows laufen.

Schritte

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

   Für Amazon FSx for ONTAP werden SMB-Freigaben benötigt.

   Sie können die SMB-Administratorfreigaben auf zwei Arten erstellen, entweder mithilfe von"Microsoft Management Console" Über das Snap-In „Freigegebene Ordner“ oder über die ONTAP -Befehlszeilenschnittstelle. So erstellen Sie die SMB-Freigaben mithilfe der ONTAP -Befehlszeilenschnittstelle:

   a. Erstellen Sie gegebenenfalls die Verzeichnispfadstruktur für die Freigabe.

      Der `vserver cifs share create` Der Befehl überprüft den Pfad, der bei der Erstellung der Freigabe in der Option -path angegeben wurde. Wenn der angegebene Pfad nicht existiert, schlägt der Befehl fehl.

   b. Erstellen Sie eine SMB-Freigabe, die dem angegebenen SVM zugeordnet ist:

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

   c. Überprüfen Sie, ob die Freigabe erstellt wurde:

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

      Siehe"Erstellen einer SMB-Freigabe" Für alle Details.

2. Bei der Erstellung des Backends müssen Sie Folgendes konfigurieren, um SMB-Volumes anzugeben. Alle Konfigurationsoptionen für das FSx for ONTAP Backend finden Sie unter"FSx für ONTAP: Konfigurationsoptionen und Beispiele" .

Parameter	Beschreibung	Beispiel
`smbShare`	Sie können eine der folgenden Optionen angeben: den Namen einer SMB-Freigabe, die mit der Microsoft Management Console oder der ONTAP CLI erstellt wurde; einen Namen, unter dem Trident die SMB-Freigabe erstellen kann; oder Sie können den Parameter leer lassen, um den Zugriff auf Volumes durch die gemeinsame Freigabe zu verhindern. Dieser Parameter ist für On-Premises ONTAP optional. Dieser Parameter ist für Amazon FSx for ONTAP -Backends erforderlich und darf nicht leer sein.	`smb-share`
`nasType`	**Muss eingestellt werden auf `smb` .** Wenn null, wird standardmäßig der Wert verwendet. `nfs` .	`smb`
`securityStyle`	Sicherheitsstil für neue Bände. **Muss eingestellt sein auf `ntfs` oder `mixed` für SMB-Volumes.**	`ntfs`oder `mixed` für SMB-Volumes
`unixPermissions`	Modus für neue Volumes. **Muss bei SMB-Volumes leer bleiben.**	""

Sichere SMB-Verbindungen aktivieren

Ab Version 25.06 unterstützt NetApp Trident die sichere Bereitstellung von SMB-Volumes, die mit `ontap-nas` Und `ontap-nas-economy` Backends. Wenn Secure SMB aktiviert ist, können Sie Active Directory (AD)-Benutzern und Benutzergruppen mithilfe von Zugriffssteuerungslisten (ACLs) einen kontrollierten Zugriff auf die SMB-Freigaben gewähren.

Wichtige Punkte

- Importieren `ontap-nas-economy` Volumen werden nicht unterstützt.
- Es werden nur schreibgeschützte Klone unterstützt für `ontap-nas-economy` Bände.
- Wenn Secure SMB aktiviert ist, ignoriert Trident die im Backend angegebene SMB-Freigabe.
- Das Aktualisieren der PVC-Annotation, der Speicherklassenannotation und des Backend-Felds aktualisiert nicht die SMB-Freigabe-ACL.
- Die in der Annotation des Klon-PVC angegebene SMB-Freigabe-ACL hat Vorrang vor denjenigen im Quell-PVC.
- Stellen Sie sicher, dass Sie gültige AD-Benutzer angeben, während Sie Secure SMB aktivieren. Ungültige Benutzer werden nicht zur Zugriffskontrollliste (ACL) hinzugefügt.
- Wenn Sie dem gleichen AD-Benutzer im Backend, in der Speicherklasse und im PVC unterschiedliche Berechtigungen zuweisen, ergibt sich folgende Berechtigungspriorität: PVC, Speicherklasse und dann Backend.
- Secure SMB wird unterstützt für `ontap-nas` Gilt für verwaltete Volume-Importe und nicht für nicht verwaltete Volume-Importe.

Schritte

1. Geben Sie adAdminUser in TridentBackendConfig wie im folgenden Beispiel gezeigt an:

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.193.176.x
  svm: svm0
  useREST: true
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

2. Fügen Sie die Annotation in der Speicherklasse hinzu.

   Füge die `trident.netapp.io/smbShareAdUser` Annotation der Speicherklasse, um sicheres SMB ohne Ausfall zu ermöglichen. Der für die Annotation angegebene Benutzerwert `trident.netapp.io/smbShareAdUser` sollte mit dem im `smbcreds` Geheimnis. Sie können eine der folgenden Optionen auswählen: `smbShareAdUserPermission` : `full_control` , `change` , oder `read` . Die Standardberechtigung ist `full_control` .

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

1. Erstellen Sie ein PVC.

   Das folgende Beispiel erzeugt eine PVC:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/snapshotDirectory: "true"
    trident.netapp.io/smbShareAccessControl: |
      read:
        - tridentADtest
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

ONTAP NAS-Konfigurationsoptionen und Beispiele

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

Backend-Konfigurationsoptionen

Die folgenden Tabellen enthalten die Backend-Konfigurationsoptionen:

Parameter	Beschreibung	Standard
`version`		Immer 1
`storageDriverName`	Name des Speichertreibers	`ontap-nas`, `ontap-nas-economy` , oder `ontap-nas-flexgroup`
`backendName`	Benutzerdefinierter Name oder das Speicher-Backend	Fahrername + "_" + dataLIF
`managementLIF`	IP-Adresse eines Clusters oder SVM-Management-LIF. Es kann ein vollqualifizierter 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]` . Für einen nahtlosen MetroCluster Wechsel siehe MetroCluster Beispiel .	„10.0.0.1“, „[2001:1234:abcd::fefe]“

Parameter	Beschreibung	Standard
`dataLIF`	IP-Adresse des Protokolls LIF. NetApp empfiehlt die Angabe `dataLIF` . Falls keine Daten angegeben werden, ruft Trident die dataLIFs vom SVM ab. Sie können einen vollqualifizierten Domänennamen (FQDN) angeben, der für die NFS-Mount-Operationen verwendet werden soll. Dadurch können Sie ein Round-Robin-DNS erstellen, um die Last auf mehrere DataLIFs zu verteilen. Kann nach der Ersteinrichtung geändert werden. Siehe . Kann so eingestellt werden, dass IPv6-Adressen verwendet werden, wenn Trident mit dem IPv6-Flag installiert wurde. IPv6-Adressen müssen in eckigen Klammern definiert werden, z. B. `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . **Für Metrocluster auslassen.** Siehe dieMetroCluster Beispiel .	Angegebene Adresse oder abgeleitet von SVM, falls nicht angegeben (nicht empfohlen).
`svm`	Zu verwendende virtuelle Speichermaschine **Für Metrocluster auslassen.** Siehe dieMetroCluster Beispiel .	Abgeleitet, wenn eine SVM `managementLIF` wird angegeben
`autoExportPolicy`	Automatische Erstellung und Aktualisierung von Exportrichtlinien aktivieren [Boolesch]. Verwenden des `autoExportPolicy` Und `autoExportCIDRs` Optionen: Trident kann Exportrichtlinien automatisch verwalten.	FALSCH
`autoExportCIDRs`	Liste der CIDRs, anhand derer die Kubernetes-Knoten-IPs gefiltert werden sollen, wenn `autoExportPolicy` ist aktiviert. Verwenden des `autoExportPolicy` Und `autoExportCIDRs` Optionen: Trident kann Exportrichtlinien automatisch verwalten.	["0.0.0.0/0", "::/0"]`
`labels`	Satz beliebiger JSON-formatierter Bezeichnungen, die auf Datenträger angewendet werden sollen	""
`clientCertificate`	Base64-kodierter Wert des Clientzertifikats. Wird für zertifikatsbasierte Authentifizierung verwendet	""
`clientPrivateKey`	Base64-kodierter Wert des privaten Client-Schlüssels. Wird für zertifikatsbasierte Authentifizierung verwendet	""
`trustedCACertificate`	Base64-kodierter Wert des vertrauenswürdigen CA-Zertifikats. Optional. Wird für zertifikatsbasierte Authentifizierung verwendet	""
`username`	Benutzername für die Verbindung mit dem Cluster/SVM. Wird für die Authentifizierung auf Basis von Anmeldeinformationen verwendet. Informationen zur Active Directory-Authentifizierung finden Sie unter "Authentifizieren Sie Trident bei einem Backend-SVM mithilfe von Active Directory-Anmeldeinformationen".	

Parameter	Beschreibung	Standard
`password`	Kennwort für die Verbindung mit dem Cluster/SVM. Wird für die Authentifizierung auf Basis von Anmeldeinformationen verwendet. Informationen zur Active Directory-Authentifizierung finden Sie unter "Authentifizieren Sie Trident bei einem Backend-SVM mithilfe von Active Directory-Anmeldeinformationen".	
`storagePrefix`	Präfix, das beim Bereitstellen neuer Volumes in der SVM verwendet wird. Kann nach der Konfiguration nicht mehr aktualisiert werden. Bei Verwendung von ontap-nas-economy und einem storagePrefix mit 24 oder mehr Zeichen wird das storagePrefix nicht in die Qtrees eingebettet, sondern nur im Volume-Namen.	"Dreizack"
`aggregate`	Aggregat für die Bereitstellung (optional; falls festgelegt, muss es der SVM zugewiesen werden). Für die `ontap-nas-flexgroup` Treiber, diese Option wird ignoriert. Falls kein Aggregat zugewiesen ist, kann jedes der verfügbaren Aggregate zur Bereitstellung eines FlexGroup Volumes verwendet werden. Wenn das Aggregat in SVM aktualisiert wird, wird es in Trident automatisch durch Abfrage von SVM aktualisiert, ohne dass der Trident Controller neu gestartet werden muss. Wenn Sie in Trident ein bestimmtes Aggregat zur Bereitstellung von Volumes konfiguriert haben und dieses Aggregat umbenannt oder aus der SVM verschoben wird, wechselt das Backend in Trident in den Fehlerzustand, während es das SVM-Aggregat abfragt. Sie müssen entweder das Aggregat in ein auf der SVM vorhandenes ändern oder es vollständig entfernen, um das Backend wieder online zu bringen.	""
`limitAggregateUsage`	Die Bereitstellung schlägt fehl, wenn die Auslastung diesen Prozentsatz überschreitet. **Gilt nicht für Amazon FSx für ONTAP**.	"" (wird nicht standardmäßig erzwungen)

Parameter	Beschreibung	Standard
flexgroupAggregateList	Liste der Aggregate für die Bereitstellung (optional; falls festgelegt, muss sie der SVM zugewiesen werden). Alle dem SVM zugewiesenen Aggregate werden zur Bereitstellung eines FlexGroup Volumes verwendet. Unterstützt für den Speichertreiber **ontap-nas-flexgroup**. Wenn die Aggregatliste in SVM aktualisiert wird, wird die Liste in Trident automatisch durch Abfrage von SVM aktualisiert, ohne dass der Trident Controller neu gestartet werden muss. Wenn Sie in Trident eine bestimmte Aggregatliste für die Bereitstellung von Volumes konfiguriert haben und diese Aggregatliste umbenannt oder aus SVM verschoben wird, wechselt das Backend in Trident beim Abfragen des SVM-Aggregats in den Fehlerzustand. Sie müssen entweder die Aggregatliste durch eine auf der SVM vorhandene Liste ersetzen oder sie vollständig entfernen, um das Backend wieder online zu bringen.	""
`limitVolumeSize`	Die Bereitstellung schlägt fehl, wenn die angeforderte Volume-Größe diesen Wert überschreitet. Beschränkt außerdem die maximale Größe der von ihm verwalteten Volumina für Qtrees, und die `qtreesPerFlexvol` Diese Option ermöglicht die Anpassung der maximalen Anzahl von Qtrees pro FlexVol volume.	"" (wird nicht standardmäßig erzwungen)
`debugTraceFlags`	Debug-Flags zur Verwendung bei der Fehlersuche. Beispiel: {"api":false, "method":true} Nicht verwenden `debugTraceFlags` Es sei denn, Sie befinden sich in der Fehlersuche und benötigen einen detaillierten Protokollauszug.	null
`nasType`	Konfiguration der Erstellung von NFS- oder SMB-Volumes. Optionen sind `nfs` , `smb` oder null. Bei der Einstellung „null“ werden standardmäßig NFS-Volumes verwendet.	`nfs`

Parameter	Beschreibung	Standard
`nfsMountOptions`	Durch Kommas getrennte Liste der NFS-Mount-Optionen. Die Mount-Optionen für Kubernetes-persistente Volumes werden normalerweise in Speicherklassen angegeben. Wenn jedoch in einer Speicherklasse keine Mount-Optionen angegeben sind, greift Trident auf die in der Konfigurationsdatei des Speicher-Backends angegebenen Mount-Optionen zurück. Wenn in der Speicherklasse oder der Konfigurationsdatei keine Mount-Optionen angegeben sind, setzt Trident keine Mount-Optionen auf einem zugehörigen persistenten Volume.	""
`qtreesPerFlexvol`	Die maximale Anzahl an Qtrees 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 Management Console oder der ONTAP CLI erstellt wurde; einen Namen, unter dem Trident die SMB-Freigabe erstellen kann; oder Sie können den Parameter leer lassen, um den Zugriff auf Volumes durch die gemeinsame Freigabe zu verhindern. Dieser Parameter ist für On-Premises ONTAP optional. Dieser Parameter ist für Amazon FSx for ONTAP -Backends erforderlich und darf nicht leer sein.	`smb-share`
`useREST`	Boolescher Parameter zur Verwendung von ONTAP REST-APIs. `useREST`Wenn eingestellt auf `true` Trident verwendet ONTAP REST-APIs zur Kommunikation mit dem Backend; wenn eingestellt auf `false` Trident verwendet ONTAPI (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 `ontapi` Anwendung. Dies wird durch die vordefinierte Bedingung erfüllt. `vsadmin` Und `cluster-admin` Rollen. Ab der Trident Version 24.06 und ONTAP 9.15.1 oder höher, `useREST` ist eingestellt auf `true` Standardmäßig; ändern `useREST` Zu `false` ONTAPI (ZAPI)-Aufrufe verwenden.	`true`für ONTAP 9.15.1 oder höher, andernfalls `false`.
`limitVolumePoolSize`	Maximal anforderbare FlexVol Größe bei Verwendung von Qtrees im ontap-nas-economy-Backend.	"" (wird nicht standardmäßig erzwungen)
`denyNewVolumePools`	Beschränkt `ontap-nas-economy` Backends daran zu hindern, neue FlexVol -Volumes zu erstellen, die ihre Qtrees enthalten. Für die Bereitstellung neuer PVs werden ausschließlich bereits vorhandene Flexvols verwendet.	

Parameter	Beschreibung	Standard
adAdminUser	Active Directory-Administratorbenutzer oder Benutzergruppe mit vollem Zugriff auf SMB-Freigaben. Verwenden Sie diesen Parameter, um Administratorrechte für die SMB-Freigabe mit voller Kontrolle zu erteilen.	

Backend-Konfigurationsoptionen für die Bereitstellung von Volumes

Sie können die Standardbereitstellung mithilfe dieser Optionen steuern. `defaults` Abschnitt der Konfiguration. Ein Beispiel finden Sie in den folgenden Konfigurationsbeispielen.

Parameter	Beschreibung	Standard
spaceAllocation	Speicherplatzbelegung für Qtrees	"WAHR"
spaceReserve	Platzreservierungsmodus; "keine" (dünn) oder "Volumen" (dick)	"keiner"
snapshotPolicy	Zu verwendende Snapshot-Richtlinie	"keiner"
qosPolicy	Die QoS-Richtliniengruppe soll den erstellten Volumes zugewiesen werden. Wählen Sie pro Speicherpool/Backend entweder qosPolicy oder adaptiveQosPolicy aus.	""
adaptiveQosPolicy	Adaptive QoS-Richtliniengruppe, die den erstellten Volumes zugewiesen werden soll. Wählen Sie pro Speicherpool/Backend entweder qosPolicy oder adaptiveQosPolicy. Wird von ontap-nas-economy nicht unterstützt.	""
snapshotReserve	Prozentsatz des für Snapshots reservierten Speichervolumens	"0" wenn `snapshotPolicy` ist "keine", ansonsten ""
splitOnClone	Beim Erstellen eines Klons diesen von seinem Elternklon trennen	"FALSCH"
encryption	Aktivieren Sie die NetApp Volumeverschlüsselung (NVE) auf dem neuen Volume; Standardwert ist `false` . Um diese Option nutzen zu können, muss NVE auf dem Cluster lizenziert und aktiviert sein. Wenn NAE im Backend aktiviert ist, wird jedes in Trident bereitgestellte Volume NAE-fähig sein. Weitere Informationen finden Sie unter:"Wie Trident mit NVE und NAE zusammenarbeitet" .	"FALSCH"
tieringPolicy	Stufenrichtlinie: "keine" verwenden	
unixPermissions	Modus für neue Volumes	„777“ für NFS-Volumes; leer (nicht zutreffend) für SMB-Volumes
snapshotDir	Steuert den Zugriff auf die `.snapshot` Verzeichnis	"true" für NFSv4, "false" für NFSv3

Parameter	Beschreibung	Standard
`exportPolicy`	Exportrichtlinie zu verwenden	"Standard"
`securityStyle`	Sicherheitsstil für neue Bände. NFS unterstützt `mixed` Und `unix` Sicherheitsstile. SMB-Unterstützung `mixed` Und `ntfs` Sicherheitsstile.	NFS-Standard ist `unix` . SMB-Standard ist `ntfs` .
`nameTemplate`	Vorlage zum Erstellen benutzerdefinierter Datenträgernamen.	""

Die Verwendung von QoS-Richtliniengruppen mit Trident erfordert ONTAP 9.8 oder höher. Sie sollten eine nicht gemeinsam genutzte QoS-Richtliniengruppe verwenden und sicherstellen, dass die Richtliniengruppe auf jeden einzelnen Bestandteil angewendet wird. Eine gemeinsam genutzte QoS-Richtliniengruppe setzt die Obergrenze für den Gesamtdurchsatz aller Workloads durch.

Beispiele für die Volumenbereitstellung

Hier ist ein Beispiel mit vordefinierten 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` Trident verwendet nun eine neue Berechnung, um sicherzustellen, dass FlexVol mit dem SnapshotReserve-Prozentsatz und PVC korrekt dimensioniert wird.

Wenn der Benutzer ein PVC anfordert, erstellt Trident das ursprüngliche FlexVol mit mehr Speicherplatz mithilfe der neuen Berechnung. Diese Berechnung stellt sicher, dass der Benutzer den im PVC angeforderten beschreibbaren Speicherplatz erhält und nicht weniger. Vor Version 21.07 erhielt der Benutzer, wenn er ein PVC (z. B. 5 GiB) mit einem SnapshotReserve von 50 Prozent anforderte, nur 2,5 GiB beschreibbaren Speicherplatz. Dies liegt daran, dass der Benutzer das gesamte Volumen angefordert hat. `snapshotReserve` ist ein Prozentsatz davon. Mit Trident 21.07 fordert der Benutzer den beschreibbaren Speicherplatz an, und Trident definiert diesen. `snapshotReserve` Zahl als Prozentsatz des Gesamtvolumens. Dies gilt nicht für `ontap-nas-economy` . Wie das funktioniert, sehen Sie im folgenden Beispiel:

Die Berechnung erfolgt wie folgt:

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

Bei SnapshotReserve = 50 % und PVC-Anforderung = 5 GiB beträgt die Gesamtgröße des Volumes 5/.5 = 10 GiB und die verfügbare Größe beträgt 5 GiB, was der vom Benutzer in der PVC-Anforderung angeforderten Größe entspricht. Der `volume show` Der Befehl sollte ähnliche Ergebnisse wie in diesem Beispiel liefern:

Vorhandene Backends aus früheren Installationen stellen beim Upgrade von Trident Volumes wie oben beschrieben bereit. Für Volumes, die Sie vor dem Upgrade erstellt haben, sollten Sie die Größe der Volumes anpassen, damit die Änderung berücksichtigt wird. Zum Beispiel ein 2 GiB PVC mit `snapshotReserve=50` Das vorherige Ergebnis war ein Volumen mit 1 GiB beschreibbarem Speicherplatz. Wenn Sie die Größe des Volumes beispielsweise auf 3 GiB ändern, stehen der Anwendung 3 GiB beschreibbarer Speicherplatz auf einem 6-GiB-Volume zur Verfügung.

Beispiele für minimale Konfigurationen

Die folgenden Beispiele zeigen Basiskonfigurationen, bei denen die meisten Parameter auf Standardwerte eingestellt bleiben. Dies ist die einfachste Möglichkeit, ein Backend zu definieren.

> Wenn Sie Amazon FSx auf NetApp ONTAP mit Trident verwenden, wird empfohlen, DNS-Namen für LIFs anstelle von IP-Adressen anzugeben.

ONTAP NAS Wirtschaftsbeispiel

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

ONTAP NAS Flexgroup-Beispiel

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

MetroCluster Beispiel

Sie können das Backend so konfigurieren, dass eine manuelle Aktualisierung der Backend-Definition nach einem Switchover und Switchback vermieden wird."SVM-Replikation und -Wiederherstellung" .

Für einen nahtlosen Übergang und Rückwechsel geben Sie die SVM wie folgt an: `managementLIF` und lassen Sie die `dataLIF` Und `svm` Parameter. Beispiel:

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

Beispiel für 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 zertifikatsbasierte Authentifizierung

Dies ist ein minimales Beispiel für eine Backend-Konfiguration. `clientCertificate` , `clientPrivateKey` , Und `trustedCACertificate` (optional, falls eine vertrauenswürdige Zertifizierungsstelle verwendet wird) werden in `backend.json` und nehmen Sie die Base64-kodierten Werte des Clientzertifikats, des privaten Schlüssels bzw. des vertrauenswürdigen CA-Zertifikats.

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

Beispiel für eine automatische Exportrichtlinie

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

```
---
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` unter 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` Dieser 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_
```

Backend-Konfigurationsbeispiel mit nameTemplate

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

Beispiele für Backends mit virtuellen Pools

In den unten gezeigten Beispiel-Backend-Definitionsdateien sind spezifische Standardwerte für alle Speicherpools festgelegt, wie zum Beispiel: `spaceReserve` bei keiner, `spaceAllocation` bei falsch und `encryption` bei falsch. Die virtuellen Pools werden im Speicherbereich definiert.

Trident legt Bereitstellungsbezeichnungen im Feld „Kommentare“ fest. Kommentare sind auf FlexVol für `ontap-nas` oder FlexGroup für `ontap-nas-flexgroup` . Trident kopiert bei der Bereitstellung alle im

virtuellen Pool vorhandenen Labels auf das Speichervolume. Zur Vereinfachung können Speicheradministratoren Bezeichnungen pro virtuellem Pool definieren und Volumes nach Bezeichnung gruppieren.

In diesen Beispielen legen einige der Speicherpools ihre eigenen Einstellungen fest. `spaceReserve` , `spaceAllocation` , Und `encryption` Werte, und einige Pools überschreiben die Standardwerte.

ONTAP NAS-Beispiel

```
---
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"
```

ONTAP NAS FlexGroup Beispiel

```
---
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"
```

ONTAP NAS Wirtschaftsbeispiel

```
---
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"
```

Backends StorageClasses zuordnen

Die folgenden StorageClass-Definitionen beziehen sich auf Beispiele für Backends mit virtuellen Pools . Verwenden des `parameters.selector` Im Feld „StorageClass“ wird für jede StorageClass angegeben, welche virtuellen Pools zum Hosten eines Volumes verwendet werden können. Das Volumen wird die im gewählten virtuellen Pool definierten Aspekte aufweisen.

- Der `protection-gold` Die StorageClass wird dem ersten und zweiten virtuellen Pool zugeordnet. `ontap-nas-flexgroup` Backend. Dies sind die einzigen Pools, die einen Schutz auf Goldniveau bieten.

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

- Der `protection-not-gold` Die StorageClass wird dem dritten und vierten virtuellen Pool zugeordnet. `ontap-nas-flexgroup` Backend. Dies sind die einzigen Pools, die ein anderes Schutzniveau als Gold bieten.

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

- Der `app-mysqldb` Die StorageClass wird dem vierten virtuellen Pool zugeordnet. `ontap-nas` Backend. Dies ist der einzige Pool, der eine Speicherpoolkonfiguration für Anwendungen vom Typ mysqldb 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` Die StorageClass wird dem dritten virtuellen Pool zugeordnet. `ontap-nas-flexgroup` Backend. Dies ist der einzige Pool, der Schutz auf Silber-Niveau und 20000 Kreditpunkte bietet.

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

- Der `creditpoints-5k` Die StorageClass wird dem dritten virtuellen Pool zugeordnet. `ontap-nas` Backend und der zweite virtuelle Pool im `ontap-nas-economy` Backend. 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.

Aktualisieren `dataLIF` nach der ersten Konfiguration

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

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

Wenn PVCs an einem oder mehreren Pods angeschlossen sind, müssen Sie alle entsprechenden Pods herunterfahren und anschließend wieder hochfahren, damit die neue dataLIF-Regelung wirksam wird.

Beispiele für sichere KMU

Backend-Konfiguration mit dem ONTAP-NAS-Treiber

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Backend-Konfiguration mit dem ontap-nas-economy-Treiber

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas-economy
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Backend-Konfiguration mit Speicherpool

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm0
  useREST: false
  storage:
  - labels:
      app: msoffice
    defaults:
      adAdminUser: tridentADuser
  nasType: smb
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Speicherklassenbeispiel mit dem ONTAP-NAS-Treiber

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADtest
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

Stellen Sie sicher, dass Sie hinzufügen `annotations` um sichere KMU zu ermöglichen. Secure SMB funktioniert nicht ohne die Annotationen, unabhängig von den im Backend oder PVC festgelegten Konfigurationen.

Speicherklassenbeispiel mit dem Treiber ontap-nas-economy

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser3
parameters:
  backendType: ontap-nas-economy
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

PVC-Beispiel mit einem einzelnen AD-Benutzer

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      change:
        - tridentADtest
      read:
        - tridentADuser
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

PVC-Beispiel mit mehreren AD-Benutzern

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-test-pvc
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      full_control:
        - tridentTestuser
        - tridentuser
        - tridentTestuser1
        - tridentuser1
      change:
        - tridentADuser
        - tridentADuser1
        - tridentADuser4
        - tridentTestuser2
      read:
        - tridentTestuser2
        - tridentTestuser3
        - tridentADuser2
        - tridentADuser3
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

Amazon FSx for NetApp ONTAP

Trident mit Amazon FSx for NetApp ONTAP verwenden

"Amazon FSx for NetApp ONTAP"ist ein vollständig verwalteter AWS-Service, der es Kunden ermöglicht, Dateisysteme zu starten und auszuführen, die auf dem Speicherbetriebssystem NetApp ONTAP basieren. FSx for ONTAP ermöglicht es Ihnen, die Ihnen vertrauten Funktionen, die Leistung und die administrativen Möglichkeiten von NetApp zu nutzen und gleichzeitig die Einfachheit, Agilität, Sicherheit und Skalierbarkeit der Datenspeicherung auf AWS in Anspruch zu nehmen. FSx für ONTAP unterstützt die Funktionen des ONTAP -Dateisystems und die Administrations-APIs.

Sie können Ihr Amazon FSx for NetApp ONTAP Dateisystem mit Trident integrieren, um sicherzustellen, dass Kubernetes-Cluster, die im Amazon Elastic Kubernetes Service (EKS) ausgeführt werden, persistente Block- und Dateivolumes bereitstellen können, die von ONTAP unterstützt werden.

Das 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. Dabei handelt es sich um

Datencontainer, in denen die Dateien und Ordner Ihres Dateisystems gespeichert werden. Mit Amazon FSx for NetApp ONTAP wird ein verwaltetes Dateisystem in der Cloud bereitgestellt. Der neue Dateisystemtyp heißt * NetApp ONTAP*.

Durch die Verwendung 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 Dateivolumes bereitstellen können, die von ONTAP unterstützt werden.

Anforderungen

Zusätzlich zu"Trident Anforderungen" Um FSx für ONTAP mit Trident zu integrieren, benötigen Sie:

- Ein bestehender Amazon EKS-Cluster oder ein selbstverwalteter Kubernetes-Cluster mit `kubectl` installiert.
- Ein vorhandenes Amazon FSx for NetApp ONTAP Dateisystem und eine Storage Virtual Machine (SVM), die von den Worker-Knoten Ihres Clusters aus erreichbar ist.
- Worker-Knoten, die vorbereitet sind für"NFS oder iSCSI" .

> Stellen Sie sicher, dass Sie die für Amazon Linux und Ubuntu erforderlichen Schritte zur Knotenvorbereitung befolgen. "Amazon Machine Images" (AMIs) abhängig von Ihrem EKS-AMI-Typ.

Überlegungen

- SMB-Volumes:
  - SMB-Volumes werden mithilfe von `ontap-nas` Nur für den Fahrer.
  - SMB-Volumes werden vom Trident EKS-Add-on nicht unterstützt.
  - Trident unterstützt SMB-Volumes nur, wenn sie in Pods eingebunden sind, die auf Windows-Knoten laufen. Siehe "Bereiten Sie die Bereitstellung von SMB-Volumes vor" für Details.
- Vor Trident 24.02 konnten Volumes, die auf Amazon FSx Dateisystemen mit aktivierter automatischer Datensicherung erstellt wurden, von Trident nicht gelöscht werden. Um dieses Problem in Trident 24.02 oder höher zu vermeiden, geben Sie Folgendes an: `fsxFilesystemID` AWS `apiRegion` AWS `apikey` und AWS `secretKey` in der Backend-Konfigurationsdatei für AWS FSx für ONTAP.

> Wenn Sie Trident eine IAM-Rolle zuweisen, können Sie die Angabe der `apiRegion` , `apiKey` , Und `secretKey` Felder explizit an Trident übergeben. Weitere Informationen finden Sie unter"FSx für ONTAP: Konfigurationsoptionen und Beispiele" .

Gleichzeitige Nutzung von Trident SAN/iSCSI und EBS-CSI-Treiber

Wenn Sie Ontap-San-Treiber (z. B. iSCSI) mit AWS (EKS, ROSA, EC2 oder einer anderen Instanz) verwenden möchten, kann es bei der auf den Knoten erforderlichen Multipath-Konfiguration zu Konflikten mit dem CSI-Treiber von Amazon Elastic Block Store (EBS) kommen. Um sicherzustellen, dass Multipathing funktioniert, ohne EBS-Festplatten auf demselben Knoten zu beeinträchtigen, müssen Sie EBS aus Ihrem Multipathing-Setup ausschließen. Dieses Beispiel zeigt ein `multipath.conf` Datei, die die erforderlichen Trident Einstellungen enthält und gleichzeitig EBS-Festplatten vom Multipathing ausschließt:

```
defaults {
    find_multipaths no
}
blacklist {
    device {
        vendor "NVME"
        product "Amazon Elastic Block Store"
    }
}
```

Authentifizierung

Trident bietet zwei Authentifizierungsmodi an.

- Anmeldeinformationsbasiert (Empfohlen): Speichert Anmeldeinformationen sicher im AWS Secrets Manager. Sie können die `fsxadmin` Benutzer für Ihr Dateisystem oder die `vsadmin` Benutzerkonfiguriert für Ihre SVM.

> Trident geht davon aus, als ein `vsadmin` SVM-Benutzer oder als Benutzer mit einem anderen Namen, der die gleiche Rolle hat. Amazon FSx for NetApp ONTAP hat ein `fsxadmin` Benutzer, der einen eingeschränkten Ersatz für ONTAP darstellt. `admin` Clusterbenutzer. Wir empfehlen dringend die Verwendung `vsadmin` mit Trident.

- Zertifikatsbasiert: Trident kommuniziert mit der SVM auf Ihrem FSx-Dateisystem mithilfe eines auf Ihrer SVM installierten Zertifikats.

Einzelheiten zur Aktivierung der Authentifizierung finden Sie in der Dokumentation zur Authentifizierung für Ihren Treibertyp:

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

Getestete Amazon Machine Images (AMIs)

Der EKS-Cluster unterstützt verschiedene Betriebssysteme, aber AWS hat bestimmte Amazon Machine Images (AMIs) für Container und EKS optimiert. Die folgenden AMIs wurden mit NetApp Trident 25.02 getestet.

AMI	NAS	NAS-Wirtschaft	iSCSI	iSCSI-Economy
AL2023_x86_64_STANDARD	Ja	Ja	Ja	Ja
AL2_x86_64	Ja	Ja	Ja*	Ja*
BOTTLEROCKET_x86_64	Ja**	Ja	k. A.	k. A.
AL2023_ARM_64_STANDARD	Ja	Ja	Ja	Ja
AL2_ARM_64	Ja	Ja	Ja*	Ja*

BOTTLEROCKET_ARM_64	Ja**	Ja	k. A.	k. A.

- * Das Löschen des PV ist ohne Neustart des Knotens nicht möglich
- ** Funktioniert nicht mit NFSv3 mit Trident Version 25.02.

Wenn Ihr gewünschtes AMI hier nicht aufgeführt ist, bedeutet das nicht, dass es nicht unterstützt wird; es bedeutet lediglich, dass es nicht getestet wurde. Diese Liste dient als Leitfaden für AMIs, von denen bekannt ist, dass sie funktionieren.

Tests durchgeführt mit:

- EKS-Version: 1.32
- Installationsmethode: Helm 25.06 und als AWS-Add-On 25.06
- Für NAS wurden sowohl NFSv3 als auch NFSv4.1 getestet.
- Für SAN wurde nur iSCSI getestet, nicht NVMe-oF.

Durchgeführte Tests:

- Erstellen: Speicherklasse, PVC, Kapsel
- Löschen: Pod, PVC (regulär, Qtree/LUN – Economy, NAS mit AWS-Backup)

Weitere Informationen

- "Amazon FSx for NetApp ONTAP -Dokumentation"
- "Blogbeitrag über Amazon FSx for NetApp ONTAP"

Erstellen Sie eine IAM-Rolle und ein AWS-Geheimnis.

Sie können Kubernetes-Pods so konfigurieren, dass sie auf AWS-Ressourcen zugreifen, indem sie sich als AWS-IAM-Rolle authentifizieren, anstatt explizite AWS-Anmeldeinformationen anzugeben.

Zur Authentifizierung mit einer AWS IAM-Rolle benötigen Sie einen Kubernetes-Cluster, der mit EKS bereitgestellt wurde.

AWS Secrets Manager-Geheimnis erstellen

Da Trident APIs an einen FSx vServer ausgibt, um den Speicher für Sie zu verwalten, benötigt es hierfür Anmeldeinformationen. Die sicherste Methode zur Übermittlung dieser Zugangsdaten ist die Verwendung eines AWS Secrets Manager-Geheimnisses. Wenn Sie also noch keines haben, müssen Sie ein AWS Secrets Manager-Geheimnis erstellen, das die Anmeldeinformationen für das vsadmin-Konto enthält.

Dieses Beispiel erstellt ein AWS Secrets Manager-Geheimnis zum Speichern von Trident CSI-Anmeldeinformationen:

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

IAM-Richtlinie erstellen

Trident benötigt außerdem AWS-Berechtigungen, um korrekt ausgeführt werden zu können. Daher müssen Sie eine Richtlinie erstellen, die Trident die benötigten Berechtigungen erteilt.

Die folgenden Beispiele erstellen eine IAM-Richtlinie mithilfe der AWS CLI:

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

Beispiel für eine Richtlinien-JSON-Datei:

```
{
  "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"
}
```

Pod-Identität oder IAM-Rolle für die Dienstkontozuordnung (IRSA) erstellen

Sie können ein Kubernetes-Servicekonto so konfigurieren, dass es eine AWS Identity and Access Management (IAM)-Rolle mit EKS Pod Identity oder IAM role for Service account association (IRSA) übernimmt. Alle Pods, die für die Verwendung des Servicekontos konfiguriert sind, können dann auf jeden AWS-Service zugreifen, für den die Rolle Berechtigungen besitzt.

Pod-Identität

Amazon EKS Pod Identity-Zuordnungen bieten die Möglichkeit, Anmeldeinformationen für Ihre Anwendungen zu verwalten, ähnlich wie Amazon EC2-Instanzprofile Anmeldeinformationen für Amazon EC2-Instanzen bereitstellen.

Installieren Sie Pod Identity auf Ihrem EKS-Cluster:

Sie können eine Pod-Identität über die AWS-Konsole oder mithilfe des folgenden AWS CLI-Befehls erstellen:

```
aws eks create-addon --cluster-name <EKS_CLUSTER_NAME> --addon-name
eks-pod-identity-agent
```

Weitere Informationen finden Sie unter"Amazon EKS Pod Identity Agent einrichten" .

Erstelle trust-relationship.json:

Erstellen Sie eine trust-relationship.json-Datei, um dem EKS-Dienstprinzipal zu ermöglichen, diese Rolle für die Pod-Identität zu übernehmen. Erstellen Sie anschließend eine Rolle mit dieser Vertrauensrichtlinie:

```
aws iam create-role \
  --role-name fsxn-csi-role --assume-role-policy-document file://trust-
relationship.json \
  --description "fsxn csi pod identity role"
```

trust-relationship.json-Datei:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "pods.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

Die Rollenrichtlinie der IAM-Rolle zuordnen:

Weisen Sie der erstellten IAM-Rolle die Rollenrichtlinie aus dem vorherigen Schritt zu:

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:111122223333:policy/fsxn-csi-policy \
  --role-name fsxn-csi-role
```

Eine Pod-Identitätszuordnung erstellen:

Erstellen einer Pod-Identitätszuordnung zwischen der IAM-Rolle und dem Trident -Dienstkonto (trident-controller).

```
aws eks create-pod-identity-association \
  --cluster-name <EKS_CLUSTER_NAME> \
  --role-arn arn:aws:iam::111122223333:role/fsxn-csi-role \
  --namespace trident --service-account trident-controller
```

IAM-Rolle für die Dienstkontozuordnung (IRSA)

Verwendung der AWS CLI:

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

trust-relationship.json-Datei:

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

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

- **<account_id>** - Ihre AWS-Konto-ID
- **<oidc_provider>** - Der OIDC Ihres EKS-Clusters. Sie können den oidc_provider durch Ausführen folgender Befehl erhalten:

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

Verknüpfen Sie die IAM-Rolle mit der IAM-Richtlinie:

Sobald die Rolle erstellt wurde, ordnen Sie die (im vorherigen Schritt erstellte) Richtlinie der Rolle mit diesem Befehl zu:

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

Überprüfen Sie, ob der OICD-Anbieter zugeordnet ist:

Vergewissern Sie sich, dass Ihr OIDC-Anbieter mit Ihrem Cluster verknüpft ist. Sie können dies mit diesem Befehl überprüfen:

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

Wenn die Ausgabe leer ist, verwenden Sie den folgenden Befehl, um IAM OIDC mit Ihrem Cluster zu verknüpfen:

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

Wenn Sie eksctl verwenden, nutzen Sie das folgende Beispiel, um eine IAM-Rolle für ein Dienstkonto in EKS zu erstellen:

```
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
```

Trident installieren

Trident optimiert die Amazon FSx for NetApp ONTAP in Kubernetes, damit sich Ihre Entwickler und Administratoren auf die Anwendungsbereitstellung 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 CSI Snapshot Controller Add-on. Siehe"Snapshot-Funktionalität für CSI-Volumes aktivieren" für weitere Informationen.

Installieren Sie Trident über Helm

Pod-Identität

1. Fügen Sie das Trident Helm-Repository hinzu:

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
```

2. Installieren Sie Trident anhand des folgenden Beispiels:

```
helm install trident-operator netapp-trident/trident-operator --version 100.2502.1 --namespace trident --create-namespace
```

Sie können die `helm list` Befehl zum Überprüfen von Installationsdetails wie Name, Namespace, Chart, Status, App-Version und Revisionsnummer.

```
helm list -n trident
```

```
NAME                NAMESPACE   REVISION    UPDATED
STATUS       CHART                           APP VERSION

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

Servicekonto-Zuordnung (IRSA)

1. Fügen Sie das Trident Helm-Repository hinzu:

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
```

2. Legen Sie die Werte für **Cloud-Anbieter** und **Cloud-Identität** fest:

```
helm install trident-operator netapp-trident/trident-operator
--version 100.2502.1  \
--set cloudProvider="AWS" \
--set cloudIdentity="'eks.amazonaws.com/role-arn:
arn:aws:iam::<accountID>:role/<AmazonEKS_FSxN_CSI_DriverRole>'" \
--namespace trident \
--create-namespace
```

Sie können die `helm list` Befehl zum Überprüfen von Installationsdetails wie Name, Namespace, Chart, Status, App-Version und Revisionsnummer.

```
helm list -n trident
```

```
NAME                NAMESPACE   REVISION    UPDATED
STATUS       CHART                          APP VERSION

trident-operator    trident     1           2024-10-14
14:31:22.463122 +0300 IDT    deployed     trident-operator-
100.2506.0    25.06.0
```

Wenn Sie iSCSI verwenden möchten, stellen Sie sicher, dass iSCSI auf Ihrem Client-Rechner aktiviert ist. Wenn Sie AL2023 Worker Node OS verwenden, können Sie die Installation des iSCSI-Clients automatisieren, indem Sie den Parameter „node prep“ in die Helm-Installation einfügen:

```
helm install trident-operator netapp-trident/trident-operator
--version 100.2502.1 --namespace trident --create-namespace --
set nodePrep={iscsi}
```

Installieren Sie Trident über das EKS-Add-on

Das Trident EKS-Add-on enthält die neuesten Sicherheitspatches und Fehlerbehebungen und ist von AWS für die Verwendung mit Amazon EKS validiert. Mit dem EKS-Add-on können Sie sicherstellen, dass Ihre Amazon EKS-Cluster stets sicher und stabil sind und den Aufwand für die Installation, Konfiguration und Aktualisierung von Add-ons reduzieren.

Voraussetzungen

Stellen Sie sicher, dass Sie Folgendes haben, bevor Sie das Trident Add-on für AWS EKS konfigurieren:

- Ein Amazon EKS-Clusterkonto mit Zusatzabonnement
- 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 for NetApp ONTAP Dateisystem

Aktivieren Sie das Trident Add-on für AWS.

Verwaltungskonsole

1. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/home#/clusters .
2. Im linken Navigationsbereich wählen Sie **Cluster** aus.
3. Wählen Sie den Namen des Clusters aus, für den Sie das NetApp Trident CSI-Add-on konfigurieren möchten.
4. Wählen Sie **Add-ons** und anschließend **Weitere Add-ons abrufen**.
5. Gehen Sie wie folgt vor, um das Add-on auszuwählen:
   a. Scrollen Sie nach unten zum Abschnitt **AWS Marketplace Add-ons** und geben Sie **"Trident"** in das Suchfeld ein.
   b. Aktivieren Sie das Kontrollkästchen in der oberen rechten Ecke des Feldes „Trident by NetApp“ .
   c. Wählen Sie **Weiter**.
6. Gehen Sie auf der Einstellungsseite **Ausgewählte Add-ons konfigurieren** wie folgt vor:

   Überspringen Sie diese Schritte, wenn Sie die Pod Identity-Zuordnung verwenden.

   a. Wählen Sie die **Version** aus, die Sie verwenden möchten.
   b. Wenn Sie die IRSA-Authentifizierung verwenden, stellen Sie sicher, dass Sie die in den optionalen Konfigurationseinstellungen verfügbaren Konfigurationswerte festlegen:
      - Wählen Sie die **Version** aus, die Sie verwenden möchten.
      - Folgen Sie dem **Add-on-Konfigurationsschema** und legen Sie den Parameter **configurationValues** im Abschnitt **Konfigurationswerte** auf den Rollen-ARN fest, den Sie im vorherigen Schritt erstellt haben (der Wert sollte folgendes Format haben):

```
{

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

}
```

   +
   Wenn Sie bei der Konfliktlösungsmethode die Option „Überschreiben“ auswählen, können eine oder mehrere Einstellungen des bestehenden Add-ons mit den Einstellungen des Amazon EKS-Add-ons überschrieben werden. Wenn Sie diese Option nicht aktivieren und es zu einem Konflikt mit Ihren bestehenden Einstellungen kommt, schlägt der Vorgang fehl. Sie können die resultierende Fehlermeldung zur Fehlerbehebung des Konflikts verwenden. Bevor Sie diese Option auswählen, vergewissern Sie sich, dass das Amazon EKS-Add-on keine Einstellungen verwaltet, die Sie selbst verwalten müssen.
7. Wählen Sie **Weiter**.
8. Auf der Seite **Überprüfen und hinzufügen** wählen Sie **Erstellen**.

   Nach Abschluss der Add-on-Installation wird Ihnen das installierte Add-on angezeigt.

AWS CLI

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

Für die Pod-Identität verwenden Sie bitte folgendes Format:

```
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v25.6.0-eksbuild.1",
}
```

Für die IRSA-Authentifizierung verwenden Sie bitte folgendes Format:

```
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v25.6.0-eksbuild.1",
  "serviceAccountRoleArn": "<role ARN>",
  "configurationValues": {
    "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'",
    "cloudProvider": "AWS"
  }
}
```

Ersetzen `<role ARN>` mit dem ARN der Rolle, die im vorherigen Schritt erstellt wurde.

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

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

eksctl

Der folgende Beispielbefehl installiert das Trident EKS-Add-on:

```
eksctl create addon --name netapp_trident-operator --cluster
<cluster_name> --force
```

Aktualisieren Sie das Trident EKS-Add-on

Verwaltungskonsole

1. Öffnen Sie die Amazon EKS-Konsole. https://console.aws.amazon.com/eks/home#/clusters .
2. Im linken Navigationsbereich wählen Sie **Cluster** aus.
3. Wählen Sie den Namen des Clusters aus, für den Sie das NetApp Trident CSI-Add-on aktualisieren möchten.
4. Wählen Sie den Reiter **Add-ons**.
5. Wählen Sie * Trident by NetApp* und anschließend **Bearbeiten**.
6. Führen Sie auf der Seite * Trident von NetApp konfigurieren* folgende Schritte aus:
   a. Wählen Sie die **Version** aus, die Sie verwenden möchten.
   b. Erweitern Sie die **Optionalen Konfigurationseinstellungen** und nehmen Sie bei Bedarf Anpassungen vor.
   c. Wählen Sie **Änderungen speichern**.

AWS CLI

Das folgende Beispiel aktualisiert das EKS-Add-on:

```
aws eks update-addon --cluster-name <eks_cluster_name> --addon-name
netapp_trident-operator --addon-version v25.6.0-eksbuild.1 \
  --service-account-role-arn <role-ARN> --resolve-conflict preserve \
  --configuration-values "{\"cloudIdentity\":
\"'eks.amazonaws.com/role-arn: <role ARN>'\"}"
```

eksctl

- Überprüfen Sie die aktuelle Version Ihres FSxN Trident CSI-Add-ons. Ersetzen `my-cluster` mit Ihrem Clusternamen.

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

  Beispielausgabe:

```
NAME                          VERSION              STATUS    ISSUES
IAMROLE    UPDATE AVAILABLE    CONFIGURATION VALUES
netapp_trident-operator    v25.6.0-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 im Ergebnis des vorherigen Schritts unter UPDATE AVAILABLE angezeigt wurde.

```
eksctl update addon --name netapp_trident-operator --version
v25.6.0-eksbuild.1 --cluster my-cluster --force
```

Wenn Sie die `--force` Wenn eine der Optionen und eine der Einstellungen des Amazon EKS-Add-ons mit Ihren bestehenden Einstellungen in Konflikt steht und das Aktualisieren des Amazon EKS-Add-ons fehlschlägt, erhalten Sie eine Fehlermeldung, die Ihnen bei der Behebung des Konflikts hilft. Bevor Sie diese Option angeben, vergewissern Sie sich, dass das Amazon EKS-Add-on keine Einstellungen verwaltet, die Sie selbst verwalten müssen, da diese Einstellungen durch diese Option überschrieben werden. Weitere Informationen zu anderen Optionen für diese Einstellung finden Sie unter"Addons" . Weitere Informationen zur Feldverwaltung von Amazon EKS Kubernetes finden Sie unter"Kubernetes-Feldmanagement" .

Deinstallieren/entfernen Sie das Trident EKS-Add-on.

Sie haben zwei Möglichkeiten, ein Amazon EKS-Add-on zu entfernen:

- **Zusatzsoftware auf Ihrem Cluster beibehalten** – Diese Option entfernt die Verwaltung aller Einstellungen durch Amazon EKS. Außerdem entfällt dadurch die Möglichkeit für Amazon EKS, Sie über Aktualisierungen zu benachrichtigen und das Amazon EKS-Add-on automatisch zu aktualisieren, nachdem Sie eine Aktualisierung initiiert haben. Die Zusatzsoftware auf Ihrem Cluster bleibt jedoch erhalten. Diese Option macht das Add-on zu einer selbstverwalteten Installation und nicht zu einem Amazon EKS-Add-on. Bei dieser Option gibt es keine Ausfallzeiten für das Add-on. Behalten Sie die `--preserve` Option im Befehl zum Beibehalten des Add-ons.
- **Entfernen Sie die Add-on-Software vollständig aus Ihrem Cluster** – NetApp empfiehlt, das Amazon EKS-Add-on nur dann aus Ihrem Cluster zu entfernen, wenn keine Ressourcen in Ihrem Cluster davon abhängig sind. Entfernen Sie die `--preserve` Option aus der `delete` Befehl zum Entfernen des Add-ons.

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

Verwaltungskonsole

1. Öffnen Sie die Amazon EKS-Konsole unter https://console.aws.amazon.com/eks/home#/clusters .
2. Im linken Navigationsbereich wählen Sie **Cluster** aus.
3. Wählen Sie den Namen des Clusters aus, für den Sie das NetApp Trident CSI-Add-on entfernen möchten.
4. Wählen Sie die Registerkarte **Add-ons** und anschließend * Trident by NetApp*.
5. Wählen Sie **Entfernen**.
6. Führen Sie im Dialogfeld „Bestätigung zum Entfernen des netapp_trident-Operators“ folgende Schritte aus:
   a. Wenn Sie nicht möchten, dass Amazon EKS die Einstellungen für das Add-on verwaltet, wählen Sie **Auf Cluster beibehalten**. Tun Sie dies, wenn Sie die Zusatzsoftware auf Ihrem Cluster behalten möchten, um alle Einstellungen des Zusatzes selbst verwalten zu können.
   b. Geben Sie **netapp_trident-operator** ein.
   c. Wählen Sie **Entfernen**.

AWS CLI

Ersetzen `my-cluster` mit dem Namen Ihres Clusters und führen Sie dann den folgenden Befehl aus.

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

eksctl

Der folgende Befehl deinstalliert das Trident EKS-Add-on:

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

Konfigurieren des Speicher-Backends

ONTAP SAN- und NAS-Treiberintegration

Um ein Speicher-Backend zu erstellen, müssen Sie eine Konfigurationsdatei entweder im JSON- oder im YAML-Format erstellen. Die Datei muss den gewünschten Speichertyp (NAS oder SAN), das Dateisystem und die SVM, von der die Daten bezogen werden sollen, sowie die Art der Authentifizierung angeben. Das folgende Beispiel zeigt, wie Sie NAS-basierten Speicher definieren und ein AWS-Secret verwenden, um die Anmeldeinformationen für die gewünschte SVM zu speichern:

YAML

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

JSON

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

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

- Erstellen Sie eine Trident-Backend-Konfiguration (TBC) aus einer YAML-Datei und führen Sie folgenden Befehl aus:

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

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

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

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

```
NAME                          BACKEND NAME          BACKEND UUID
PHASE   STATUS

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

FSx für ONTAP Treiberdetails

Sie können Trident mit Amazon FSx for NetApp ONTAP mithilfe der folgenden Treiber integrieren:

- `ontap-san`: Jedes bereitgestellte PV ist eine LUN innerhalb eines eigenen Amazon FSx for NetApp ONTAP Volumes. Empfohlen für Blockspeicherung.
- `ontap-nas`Jedes bereitgestellte PV ist ein vollständiges Amazon FSx for NetApp ONTAP -Volume. Empfohlen für NFS und SMB.
- `ontap-san-economy`Jedes bereitgestellte PV ist eine LUN mit einer konfigurierbaren Anzahl von LUNs pro Amazon FSx for NetApp ONTAP Volume.
- `ontap-nas-economy`Jedes bereitgestellte PV ist ein Qtree, wobei die Anzahl der Qtrees pro Amazon FSx for NetApp ONTAP Volume konfigurierbar ist.
- `ontap-nas-flexgroup`Jedes bereitgestellte PV ist ein vollständiges Amazon FSx for NetApp ONTAP FlexGroup Volume.

Weitere Fahrerdetails finden Sie unter"NAS-Treiber" Und"SAN-Treiber" .

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

```
kubectl create -f configuration_file
```

Um den Status zu überprüfen, führen Sie folgenden Befehl aus:

```
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 Backend-Konfiguration und Beispiele

Die folgenden Tabellen enthalten die Backend-Konfigurationsoptionen:

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 Speicher-Backend	Fahrername + "_" + dataLIF
`managementLIF`	IP-Adresse eines Clusters oder SVM-Management-LIF. Es kann ein vollqualifizierter 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 angegeben werden, z. B. [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]. Wenn Sie die `fsxFilesystemID` unter dem `aws` Das Feld müssen Sie nicht angeben. `managementLIF` weil Trident die SVM abruft `managementLIF` Informationen von AWS. Sie müssen also Anmeldeinformationen für einen Benutzer unter der SVM angeben (z. B. vsadmin), und dieser Benutzer muss über die folgenden Berechtigungen verfügen: `vsadmin` Rolle.	„10.0.0.1“, „[2001:1234:abcd::fefe]“

Parameter	Beschreibung	Beispiel
`dataLIF`	IP-Adresse des Protokolls LIF. * ONTAP NAS-Treiber*: NetApp empfiehlt die Angabe von dataLIF. Falls keine Daten angegeben werden, ruft Trident die dataLIFs vom SVM ab. Sie können einen vollqualifizierten Domänennamen (FQDN) angeben, der für die NFS-Mount-Operationen verwendet werden soll. Dadurch können Sie ein Round-Robin-DNS erstellen, um die Last auf mehrere DataLIFs zu verteilen. Kann nach der Ersteinrichtung geändert werden. Siehe . * ONTAP SAN-Treiber*: Nicht für iSCSI angeben. Trident verwendet ONTAP Selective LUN Map, um die iSCI LIFs zu ermitteln, die zum Aufbau einer Multipath-Sitzung benötigt werden. Es wird eine Warnung generiert, 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 angegeben werden, z. B. [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	
`autoExportPolicy`	Automatische Erstellung und Aktualisierung von Exportrichtlinien aktivieren [Boolesch]. Verwenden des `autoExportPolicy` Und `autoExportCIDRs` Optionen: Trident kann Exportrichtlinien automatisch verwalten.	`false`
`autoExportCIDRs`	Liste der CIDRs, anhand derer die Kubernetes-Knoten-IPs gefiltert werden sollen, wenn `autoExportPolicy` ist aktiviert. Verwenden des `autoExportPolicy` Und `autoExportCIDRs` Optionen: Trident kann Exportrichtlinien automatisch verwalten.	"["0.0.0.0/0", "::/0"]"
`labels`	Satz beliebiger JSON-formatierter Bezeichnungen, die auf Datenträger angewendet werden sollen	""

Parameter	Beschreibung	Beispiel
`clientCertificate`	Base64-kodierter Wert des Clientzertifikats. Wird für zertifikatsbasierte Authentifizierung verwendet	""
`clientPrivateKey`	Base64-kodierter Wert des privaten Client-Schlüssels. Wird für zertifikatsbasierte Authentifizierung verwendet	""
`trustedCACertificate`	Base64-kodierter Wert des vertrauenswürdigen CA-Zertifikats. Optional. Wird für die zertifikatsbasierte Authentifizierung verwendet.	""
`username`	Benutzername für die Verbindung zum Cluster oder zur SVM. Wird für die auf Anmeldeinformationen basierende Authentifizierung verwendet. Zum Beispiel vsadmin.	
`password`	Passwort zum Verbinden mit dem Cluster oder der SVM. Wird für die auf Anmeldeinformationen basierende Authentifizierung verwendet.	
`svm`	Zu verwendende virtuelle Speichermaschine	Wird abgeleitet, wenn ein SVM managementLIF angegeben ist.
`storagePrefix`	Präfix, das beim Bereitstellen neuer Volumes in der SVM verwendet wird. Kann nach der Erstellung nicht mehr geändert werden. Um diesen Parameter zu aktualisieren, müssen Sie ein neues Backend erstellen.	`trident`
`limitAggregateUsage`	**Nicht für Amazon FSx for NetApp ONTAP angeben.** Die bereitgestellten `fsxadmin` Und `vsadmin` enthalten nicht die erforderlichen Berechtigungen, um die aggregierte Nutzung abzurufen und sie mit Trident einzuschränken.	Nicht verwenden.

Parameter	Beschreibung	Beispiel
`limitVolumeSize`	Die Bereitstellung schlägt fehl, wenn die angeforderte Volume-Größe diesen Wert überschreitet. Beschränkt außerdem die maximale Größe der von ihm verwalteten Volumes für Qtrees und LUNs, und die `qtreesPerFlexvol` Diese Option ermöglicht die Anpassung der maximalen Anzahl von Qtrees pro FlexVol volume.	"" (wird nicht standardmäßig erzwungen)
`lunsPerFlexvol`	Die maximale Anzahl an LUNs pro Flexvol-Volume muss im Bereich [50, 200] liegen. Nur SAN.	“100”
`debugTraceFlags`	Debug-Flags zur Verwendung bei der Fehlersuche. Beispiel: {"api":false, "method":true} Nicht verwenden `debugTraceFlags` Es sei denn, Sie befinden sich in der Fehlersuche und benötigen einen detaillierten Protokollauszug.	null
`nfsMountOptions`	Durch Kommas getrennte Liste der NFS-Mount-Optionen. Die Mount-Optionen für Kubernetes-persistente Volumes werden normalerweise in Speicherklassen angegeben. Wenn jedoch in einer Speicherklasse keine Mount-Optionen angegeben sind, greift Trident auf die in der Konfigurationsdatei des Speicher-Backends angegebenen Mount-Optionen zurück. Wenn in der Speicherklasse oder der Konfigurationsdatei keine Mount-Optionen angegeben sind, setzt Trident keine Mount-Optionen auf einem zugehörigen persistenten Volume.	""
`nasType`	Konfiguration der Erstellung von NFS- oder SMB-Volumes. Optionen sind `nfs` , `smb` oder null. **Muss eingestellt werden auf `smb` für SMB-Volumes.** Bei der Einstellung „null“ werden standardmäßig NFS-Volumes verwendet.	`nfs`
`qtreesPerFlexvol`	Die maximale Anzahl an Qtrees pro FlexVol volume muss im Bereich [50, 300] liegen.	`"200"`

Parameter	Beschreibung	Beispiel
`smbShare`	Sie können entweder den Namen einer SMB-Freigabe angeben, die mit der Microsoft Management Console oder der ONTAP CLI erstellt wurde, oder einen Namen, unter dem Trident die SMB-Freigabe erstellen kann. Dieser Parameter ist für Amazon FSx for ONTAP -Backends erforderlich.	`smb-share`
`useREST`	Boolescher Parameter zur Verwendung von ONTAP REST-APIs. Wenn eingestellt auf `true` Trident wird ONTAP REST APIs zur Kommunikation mit dem Backend verwenden. Diese Funktion erfordert ONTAP 9.11.1 und höher. Darüber hinaus muss die verwendete ONTAP Anmelderolle Zugriff auf die `ontap` Anwendung. Dies wird durch die vordefinierte Bedingung erfüllt. `vsadmin` Und `cluster-admin` Rollen.	`false`
`aws`	In der Konfigurationsdatei für AWS FSx für ONTAP können Sie Folgendes 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 im AWS Secrets Manager gespeichert werden sollen. - `name` : Amazon Resource Name (ARN) des Geheimnisses, das die Anmeldeinformationen von SVM enthält. - `type` : Aufstellen `awsarn` . Siehe"Erstellen Sie ein AWS Secrets Manager-Geheimnis" für weitere Informationen.	

Backend-Konfigurationsoptionen für die Bereitstellung von Volumes

Sie können die Standardbereitstellung mithilfe dieser Optionen steuern. `defaults` Abschnitt der Konfiguration. Ein Beispiel finden Sie in den folgenden Konfigurationsbeispielen.

Parameter	Beschreibung	Standard
`spaceAllocation`	Speicherplatzzuweisung für LUNs	`true`
`spaceReserve`	Platzreservierungsmodus; "keine" (dünn) oder "Volumen" (dick)	`none`
`snapshotPolicy`	Zu verwendende Snapshot-Richtlinie	`none`
`qosPolicy`	Die QoS-Richtliniengruppe soll den erstellten Volumes zugewiesen werden. Wählen Sie pro Speicherpool oder Backend entweder qosPolicy oder adaptiveQosPolicy aus. Die Verwendung von QoS-Richtliniengruppen mit Trident erfordert ONTAP 9.8 oder höher. Sie sollten eine nicht gemeinsam genutzte QoS-Richtliniengruppe verwenden und sicherstellen, dass die Richtliniengruppe auf jeden einzelnen Bestandteil angewendet wird. Eine gemeinsam genutzte QoS-Richtliniengruppe setzt die Obergrenze für den Gesamtdurchsatz aller Workloads durch.	""
`adaptiveQosPolicy`	Adaptive QoS-Richtliniengruppe, die den erstellten Volumes zugewiesen werden soll. Wählen Sie pro Speicherpool oder Backend entweder qosPolicy oder adaptiveQosPolicy aus. Wird von ontap-nas-economy nicht unterstützt.	""
`snapshotReserve`	Prozentsatz des für Snapshots reservierten Speichervolumens „0“	Wenn `snapshotPolicy` Ist `none` , `else` ""
`splitOnClone`	Beim Erstellen eines Klons diesen von seinem Elternklon trennen	`false`
`encryption`	Aktivieren Sie die NetApp Volumeverschlüsselung (NVE) auf dem neuen Volume; Standardwert ist `false` . Um diese Option nutzen zu können, muss NVE auf dem Cluster lizenziert und aktiviert sein. Wenn NAE im Backend aktiviert ist, wird jedes in Trident bereitgestellte Volume NAE-fähig sein. Weitere Informationen finden Sie unter:"Wie Trident mit NVE und NAE zusammenarbeitet" .	`false`

Parameter	Beschreibung	Standard
`luksEncryption`	LUKS-Verschlüsselung aktivieren. Siehe"Verwenden Sie Linux Unified Key Setup (LUKS)." . Nur SAN.	""
`tieringPolicy`	zu verwendende Stufenrichtlinie `none`	
`unixPermissions`	Modus für neue Volumes. **Für SMB-Volumes leer lassen.**	""
`securityStyle`	Sicherheitsstil für neue Bände. NFS unterstützt `mixed` Und `unix` Sicherheitsstile. SMB-Unterstützung `mixed` Und `ntfs` Sicherheitsstile.	NFS-Standard ist `unix` . SMB-Standard ist `ntfs` .

Bereiten Sie die Bereitstellung von SMB-Volumes vor

Sie können SMB-Volumes mithilfe von … bereitstellen. `ontap-nas` Treiber. Bevor Sie fertigstellenONTAP SAN- und NAS-Treiberintegration Führen Sie die folgenden Schritte aus.

Bevor Sie beginnen

Bevor Sie SMB-Volumes mithilfe von `ontap-nas` Als Fahrer benötigen Sie Folgendes:

- Ein Kubernetes-Cluster mit einem Linux-Controller-Knoten und mindestens einem Windows-Worker-Knoten, auf dem Windows Server 2019 ausgeführt wird. Trident unterstützt SMB-Volumes nur, wenn sie in Pods eingebunden sind, die auf Windows-Knoten laufen.
- Mindestens ein Trident Geheimnis, das Ihre Active Directory-Anmeldeinformationen enthält. Um Geheimnisse zu generieren `smbcreds` :

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

- Ein als Windows-Dienst konfigurierter CSI-Proxy. Um einen zu konfigurieren `csi-proxy` , siehe"GitHub: CSI-Proxy" oder"GitHub: CSI-Proxy für Windows" für Kubernetes-Knoten, die unter Windows laufen.

Schritte

1. SMB-Freigaben erstellen. Sie können die SMB-Administratorfreigaben auf zwei Arten erstellen, entweder mithilfe von"Microsoft Management Console" Über das Snap-In „Freigegebene Ordner“ oder über die ONTAP -Befehlszeilenschnittstelle. So erstellen Sie die SMB-Freigaben mithilfe der ONTAP -Befehlszeilenschnittstelle:
   a. Erstellen Sie gegebenenfalls die Verzeichnispfadstruktur für die Freigabe.

      Der `vserver cifs share create` Der Befehl überprüft den Pfad, der bei der Erstellung der Freigabe in der Option -path angegeben wurde. Wenn der angegebene Pfad nicht existiert, schlägt der Befehl fehl.

   b. Erstellen Sie eine SMB-Freigabe, die dem angegebenen SVM zugeordnet ist:

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

c. Überprüfen Sie, ob die Freigabe erstellt wurde:

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

Siehe"Erstellen einer SMB-Freigabe" Für alle Details.

2. Bei der Erstellung des Backends müssen Sie Folgendes konfigurieren, um SMB-Volumes anzugeben. Alle Konfigurationsoptionen für das FSx for ONTAP Backend finden Sie unter"FSx für ONTAP: Konfigurationsoptionen und Beispiele" .

Parameter	Beschreibung	Beispiel
`smbShare`	Sie können entweder den Namen einer SMB-Freigabe angeben, die mit der Microsoft Management Console oder der ONTAP CLI erstellt wurde, oder einen Namen, unter dem Trident die SMB-Freigabe erstellen kann. Dieser Parameter ist für Amazon FSx for ONTAP -Backends erforderlich.	`smb-share`
`nasType`	**Muss eingestellt werden auf `smb` .** Wenn null, wird standardmäßig der Wert verwendet. `nfs` .	`smb`
`securityStyle`	Sicherheitsstil für neue Bände. **Muss eingestellt sein auf `ntfs` oder `mixed` für SMB-Volumes.**	`ntfs`oder `mixed` für SMB-Volumes
`unixPermissions`	Modus für neue Volumes. **Muss bei SMB-Volumes leer bleiben.**	""

Konfigurieren Sie eine Speicherklasse und einen PVC.

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

Erstellen einer Speicherklasse

Konfigurieren eines Kubernetes StorageClass-Objekts

Der "Kubernetes StorageClass-Objekt" Das Objekt identifiziert Trident als den für diese Klasse verwendeten Provisionierer und weist Trident an, wie ein Volume zu provisionieren ist. Verwenden Sie dieses Beispiel, um die Speicherklasse für Volumes mit NFS einzurichten (die vollständige Liste der Attribute finden Sie im Abschnitt „Trident -Attribute“ weiter unten):

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

Verwenden Sie dieses Beispiel, um die Speicherklasse für Volumes mit iSCSI einzurichten:

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

Um NFSv3-Volumes auf AWS Bottlerocket bereitzustellen, fügen Sie die erforderlichen Komponenten hinzu. `mountOptions` zur Speicherklasse:

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

Siehe"Kubernetes- und Trident Objekte" Einzelheiten darüber, wie Speicherklassen mit dem interagieren, finden Sie hier. `PersistentVolumeClaim` und Parameter zur Steuerung der Volumenbereitstellung Trident .

Erstellen einer Speicherklasse

Schritte

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

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

2. Sie sollten nun sowohl in Kubernetes als auch in Trident eine **basic-csi**-Speicherklasse sehen, und Trident sollte die Pools im Backend erkannt haben.

   ```
   kubectl get sc basic-csi
   ```

```
NAME         PROVISIONER             AGE
basic-csi    csi.trident.netapp.io   15h
```

PVC erstellen

A "*PersistentVolumeClaim*" (PVC) ist eine Anfrage für den Zugriff auf das PersistentVolume im Cluster.

Das PVC kann so konfiguriert werden, dass es Speicherplatz einer bestimmten Größe oder einen bestimmten Zugriffsmodus anfordert. Mithilfe der zugehörigen StorageClass kann der Clusteradministrator mehr als nur die Größe und den Zugriffsmodus des PersistentVolumes steuern – beispielsweise die Leistung oder das Servicelevel.

Nachdem Sie das PVC erstellt haben, können Sie das Volumen in einem Gehäuse montieren.

Beispielmanifeste

Beispielmanifeste für PersistentVolumeClaim

Diese Beispiele zeigen grundlegende PVC-Konfigurationsoptionen.

PVC mit RWX-Zugang

Dieses Beispiel zeigt eine einfache PVC mit RWX-Zugriff, die einer StorageClass namens zugeordnet ist. `basic-csi` .

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

PVC mit iSCSI-Beispiel

Dieses Beispiel zeigt eine einfache PVC für iSCSI mit RWO-Zugriff, die einer StorageClass namens zugeordnet ist. `protection-gold` .

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

PVC erstellen

Schritte

1. Erstellen Sie die PVC.

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

2. Überprüfen Sie den PVC-Status.

```
kubectl get pvc
```

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

Siehe"Kubernetes- und Trident Objekte" Einzelheiten darüber, wie Speicherklassen mit dem interagieren, finden Sie hier. `PersistentVolumeClaim` und Parameter zur Steuerung der Volumenbereitstellung Trident .

Trident Eigenschaften

Diese Parameter legen fest, welche von Trident verwalteten Speicherpools zur Bereitstellung von Volumes eines bestimmten Typs verwendet werden sollen.

Attribut	Typ	Werte	Angebot	Anfrage	Unterstützt von
media[1]	Schnur	HDD, Hybrid, SSD	Der Pool enthält Medien dieses Typs; hybrid bedeutet beides	Medientyp angegeben	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san
Bereitstellungstyp	Schnur	dünn, dick	Pool unterstützt diese Bereitstellungsmethode	Bereitstellungsmethode angegeben	dick: alles vom Fass; dünn: alles vom Fass & Solidfire-San
Backend-Typ	Schnur	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san, gcp-cvs, azure-netapp-files, ontap-san-economy	Pool gehört zu dieser Art von Backend.	Backend spezifiziert	Alle Fahrer
Momentaufnahmen	bool	wahr, falsch	Pool unterstützt Volumes mit Snapshots	Volume mit aktivierten Snapshots	ontap-nas, ontap-san, solidfire-san, gcp-cvs
Klone	bool	wahr, falsch	Pool unterstützt das Klonen von Volumes	Volume mit aktivierten Klonen	ontap-nas, ontap-san, solidfire-san, gcp-cvs

Attribut	Typ	Werte	Angebot	Anfrage	Unterstützt von
Verschlüsselung	bool	wahr, falsch	Pool unterstützt verschlüsselte Volumes	Volume mit aktivierter Verschlüsselung	ontap-nas, ontap-nas-economy, ontap-nas-flexgroups, ontap-san
IOPS	int	positive ganze Zahl	Pool ist in der Lage, IOPS in diesem Bereich zu garantieren.	Volumen garantiert diese IOPS	solidfire-san

[1]: Wird von ONTAP Select -Systemen nicht unterstützt.

Beispielanwendung bereitstellen

Sobald die Speicherklasse und das PVC erstellt sind, können Sie das PV an einem Pod montieren. Dieser Abschnitt listet den Beispielbefehl und die Konfiguration zum Anhängen des PV an einen Pod auf.

Schritte

1. Montieren Sie das Volume in einem Gehäuse.

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

Diese Beispiele zeigen grundlegende Konfigurationen zum Anbringen des PVC an eine Kapsel:
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 überwachen mit `kubectl get pod --watch`.

2. Überprüfen Sie, ob das Volume eingebunden 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-Anwendung wird nicht mehr existieren, das Volume bleibt jedoch erhalten.

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

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

NetApp Trident optimiert die Amazon FSx for NetApp ONTAP in Kubernetes, damit sich Ihre Entwickler und Administratoren auf die Anwendungsbereitstellung konzentrieren können. Das NetApp Trident EKS-Add-on enthält die neuesten Sicherheitspatches und Fehlerbehebungen und ist von AWS für die Verwendung mit Amazon EKS validiert. Mit dem EKS-Add-on können Sie sicherstellen, dass Ihre Amazon EKS-Cluster stets sicher und stabil sind und den Aufwand für die Installation, Konfiguration und Aktualisierung von Add-ons reduzieren.

Voraussetzungen

Stellen Sie sicher, dass Sie Folgendes haben, bevor Sie das Trident Add-on für AWS EKS konfigurieren:

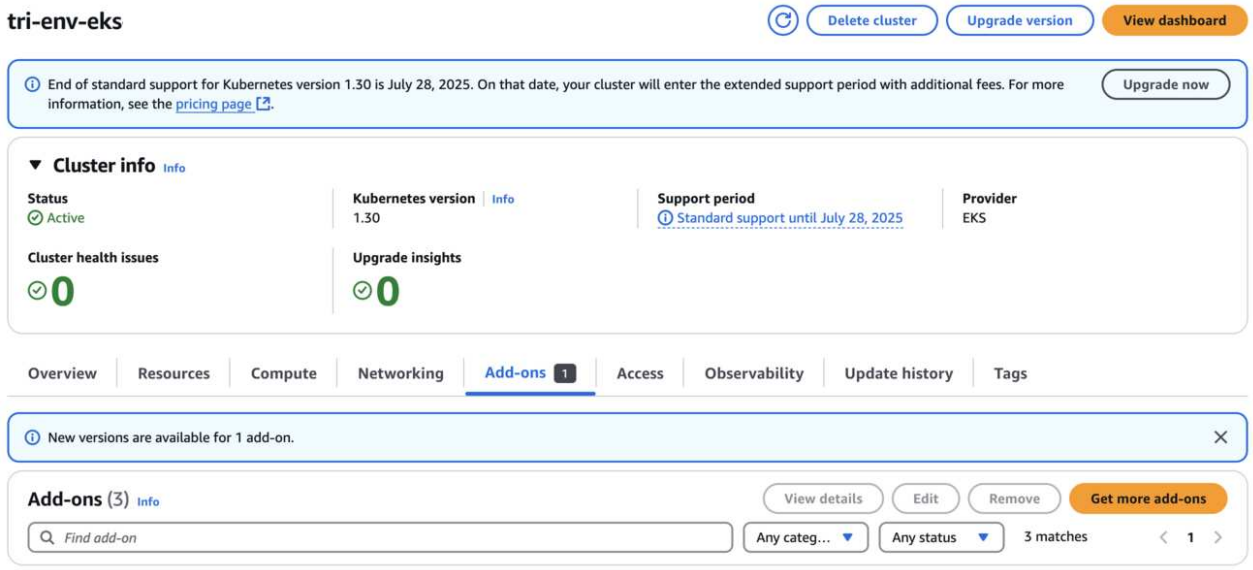
- Ein Amazon EKS-Cluster-Konto mit Berechtigungen zur Arbeit mit Add-ons. Siehe"Amazon EKS-Add-ons" .
- AWS-Berechtigungen für den AWS Marketplace:
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- AMI-Typ: Amazon Linux 2 (AL2_x86_64) oder Amazon Linux 2 Arm (AL2_ARM_64)
- Knotentyp: AMD oder ARM
- Ein bestehendes Amazon FSx for NetApp ONTAP Dateisystem

Schritte

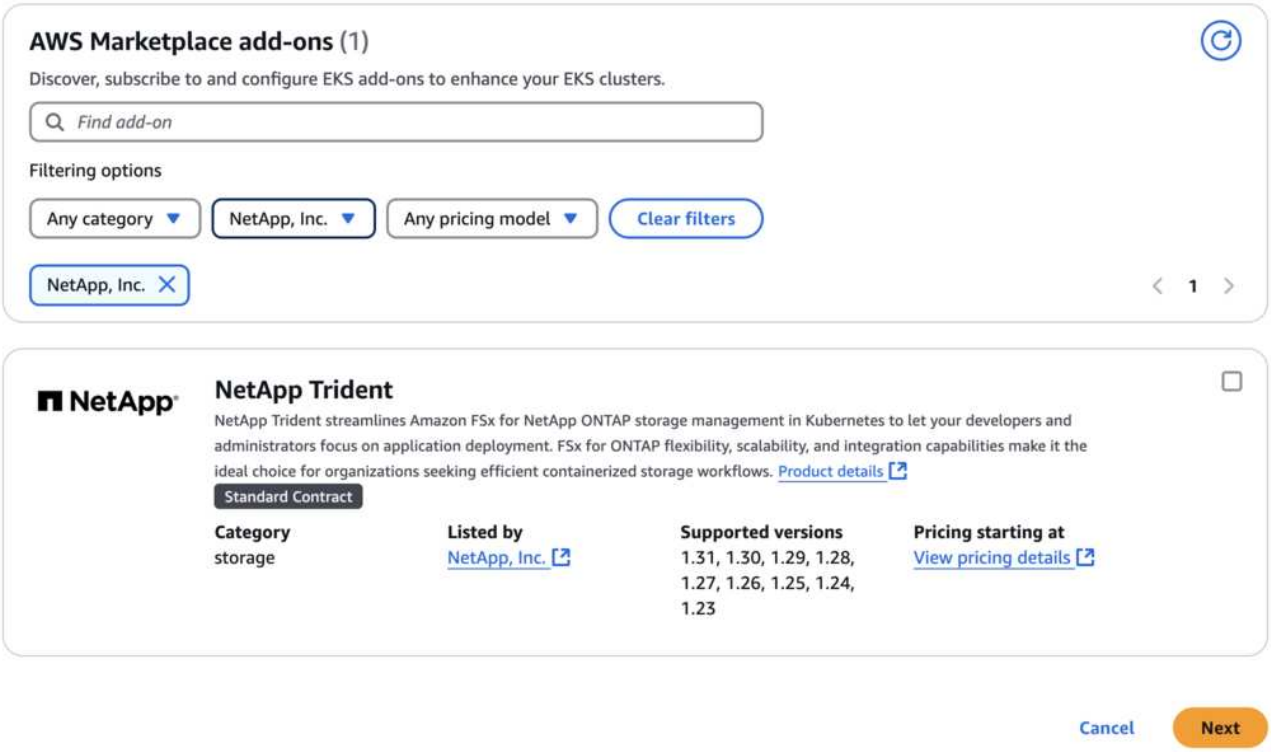
1. Stellen Sie sicher, dass Sie eine IAM-Rolle und ein AWS-Secret erstellen, damit EKS-Pods auf AWS-Ressourcen zugreifen können. Anweisungen finden Sie unter"Erstellen Sie eine IAM-Rolle und ein AWS-

Geheimnis." .

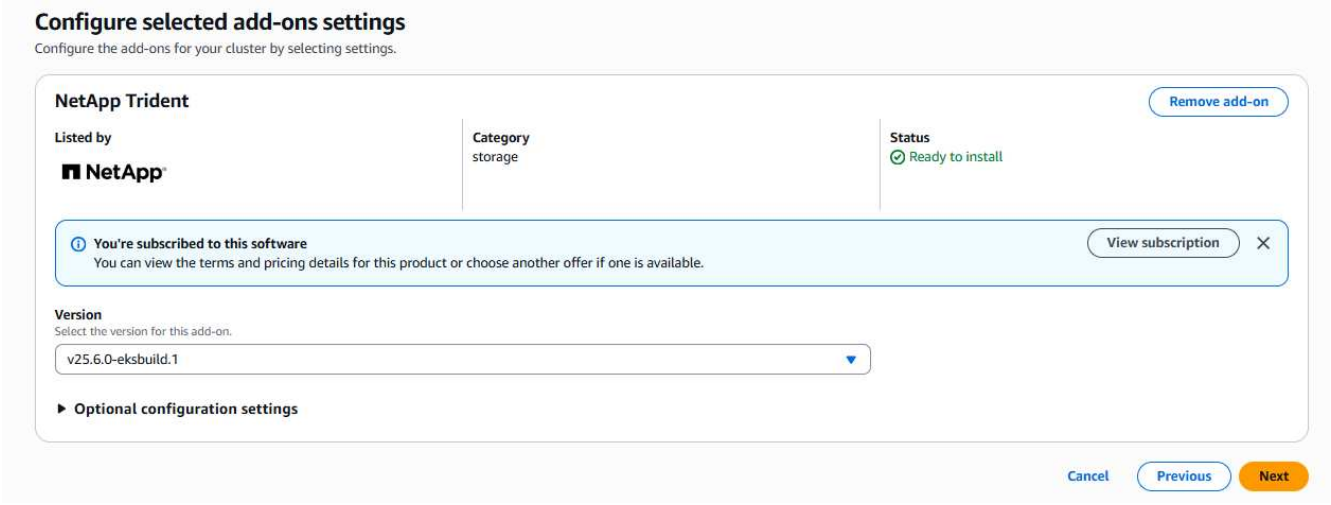
2. Navigieren Sie in Ihrem EKS Kubernetes-Cluster zum Tab **Add-ons**.



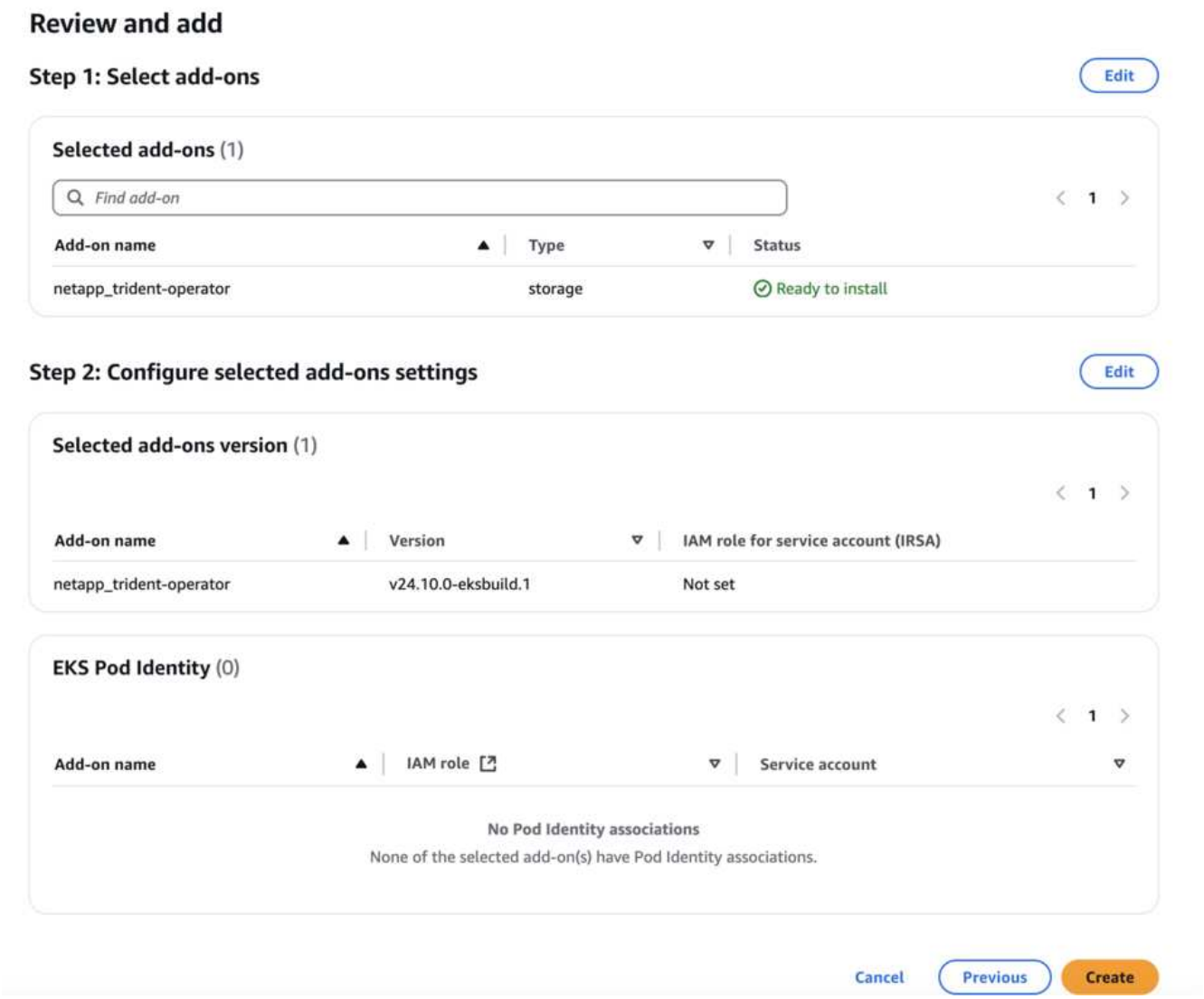
3. Gehen Sie zu **AWS Marketplace Add-ons** und wählen Sie die Kategorie *Speicher* aus.



4. Suchen Sie * NetApp Trident* und aktivieren Sie das Kontrollkästchen für das Trident Add-on. Klicken Sie anschließend auf **Weiter**.
5. Wählen Sie die gewünschte Version des Add-ons.

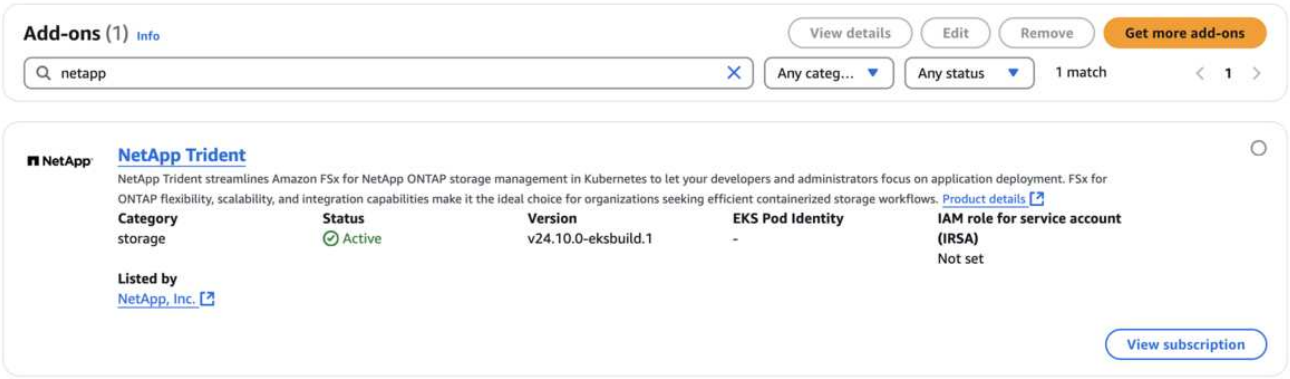


6. Konfigurieren Sie die erforderlichen Add-On-Einstellungen.



7. Wenn Sie IRSA (IAM-Rollen für Dienstkonten) verwenden, beachten Sie die zusätzlichen Konfigurationsschritte."hier," .
8. Wählen Sie **Erstellen**.

9. Überprüfen Sie, ob der Status des Add-ons *Aktiv* 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 Speicher-Backend. Weitere Informationen finden Sie unter"Konfigurieren des Speicher-Backends" .

Installation/Deinstallation des Trident EKS-Add-ons über die Befehlszeile

Installieren Sie das NetApp Trident EKS-Add-on über die Befehlszeile:

Der folgende Beispielbefehl installiert das Trident EKS-Add-on: `eksctl create addon --cluster clusterName --name netapp_trident-operator --version v25.6.0-eksbuild.1` (mit einer speziellen Version)

Deinstallieren Sie das NetApp Trident EKS-Add-on über die Befehlszeile:

Der folgende Befehl deinstalliert das Trident EKS-Add-on:

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

Backends mit kubectl erstellen

Ein Backend definiert die Beziehung zwischen Trident und einem Speichersystem. Es teilt Trident mit, wie mit diesem Speichersystem kommuniziert werden soll und wie Trident Datenträger daraus bereitstellen soll. Nach der Installation von Trident besteht der nächste Schritt darin, ein Backend zu erstellen. Der `TridentBackendConfig` Mit Custom Resource Definition (CRD) können Sie Trident -Backends direkt über die Kubernetes-Schnittstelle erstellen und verwalten. Dies können Sie tun, indem Sie `kubectl` oder das entsprechende CLI-Tool für Ihre Kubernetes-Distribution.

`TridentBackendConfig`

`TridentBackendConfig` (`tbc`, `tbconfig` , `tbackendconfig`) ist ein Frontend mit Namensräumen, das es Ihnen ermöglicht, Trident -Backends zu verwalten. `kubectl` . Kubernetes- und Speicheradministratoren

können Backends jetzt direkt über die Kubernetes-CLI erstellen und verwalten, ohne dass ein separates Befehlszeilenprogramm erforderlich ist.(`tridentctl`).

Bei der Schaffung eines `TridentBackendConfig` Wenn man sich das Objekt ansieht, geschieht Folgendes:

- Ein Backend wird von Trident automatisch auf Basis der von Ihnen bereitgestellten Konfiguration erstellt. Dies wird intern dargestellt als `TridentBackend` (`tbe` , `tridentbackend`) CR.
- Der `TridentBackendConfig` ist auf einzigartige Weise an ein `TridentBackend` Das wurde von Trident entwickelt.

Jede `TridentBackendConfig` pflegt eine Eins-zu-Eins-Zuordnung mit einem `TridentBackend` Ersteres ist die dem Benutzer zur Verfügung gestellte Schnittstelle zum Entwerfen und Konfigurieren von Backends; letzteres ist die Art und Weise, wie Trident das eigentliche Backend-Objekt darstellt.

> TridentBackend`CRs werden von Trident automatisch erstellt. Sie **sollten** sie nicht verändern. Wenn Sie Aktualisierungen an den Backends vornehmen möchten, tun Sie dies durch Ändern der `TridentBackendConfig` Objekt.

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

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

Sie können sich auch die Beispiele in der "Dreizack-Installateur" Verzeichnis mit Beispielkonfigurationen für die gewünschte Speicherplattform/den gewünschten Speicherdienst.

Der `spec` benötigt backendspezifische Konfigurationsparameter. In diesem Beispiel verwendet das Backend die `ontap-san` Der Speichertreiber verwendet die hier tabellarisch aufgeführten Konfigurationsparameter. Eine Liste der Konfigurationsoptionen für Ihren gewünschten Speichertreiber finden Sie unter"Backend-Konfigurationsinformationen für Ihren Speichertreiber" .

Der `spec` Dieser Abschnitt umfasst auch `credentials` Und `deletionPolicy` Felder, die neu eingeführt wurden in der `TridentBackendConfig` CR:

- `credentials`Dieser Parameter ist ein Pflichtfeld und enthält die Anmeldeinformationen, die zur Authentifizierung beim Speichersystem/-dienst verwendet werden. Dies ist auf ein vom Benutzer erstelltes Kubernetes-Secret eingestellt. Die Zugangsdaten dürfen nicht im Klartext übermittelt werden und führen zu

einem Fehler.

- `deletionPolicy`Dieses Feld definiert, was geschehen soll, wenn `TridentBackendConfig` wird gelöscht. Es kann einen von zwei möglichen Werten annehmen:
  - `delete`Dies führt zur Löschung beider `TridentBackendConfig` CR und das zugehörige Backend. Dies ist der Standardwert.
  - `retain`Wenn ein `TridentBackendConfig` Wenn CR gelöscht wird, bleibt die Backend-Definition weiterhin vorhanden und kann verwaltet werden mit `tridentctl` . Die Löschrichtlinie festlegen `retain` Ermöglicht es Benutzern, auf eine frühere Version (vor 21.04) zurückzustufen und die erstellten Backends beizubehalten. Der Wert dieses Feldes kann nach einem `TridentBackendConfig` wird erstellt.

Der Name eines Backends wird wie folgt festgelegt: `spec.backendName` . Wenn kein Name angegeben ist, wird der Name des Backends auf den Namen des Backends gesetzt. `TridentBackendConfig` Objekt (metadata.name). Es wird empfohlen, Backend-Namen explizit festzulegen. `spec.backendName` .

Backends, die mit `tridentctl` haben keine zugehörige `TridentBackendConfig` Objekt. Sie können diese Backends mit folgender Funktion verwalten: `kubectl` durch die Erstellung eines `TridentBackendConfig` CR. Es ist darauf zu achten, dass identische Konfigurationsparameter angegeben werden (wie z. B. `spec.backendName` , `spec.storagePrefix` , `spec.storageDriverName` , und so weiter). Trident wird das neu erstellte automatisch binden. `TridentBackendConfig` mit dem bereits bestehenden Backend.

Schrittübersicht

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

1. Erstellen Sie ein "Kubernetes-Geheimnis" Das Geheimnis enthält die Anmeldeinformationen, die Trident für die Kommunikation mit dem Speichercluster/-dienst benötigt.
2. Erstellen Sie ein `TridentBackendConfig` Objekt. Dieser Eintrag enthält spezifische Informationen zum Speichercluster/Speicherdienst und verweist auf das im vorherigen Schritt erstellte Geheimnis.

Nachdem Sie ein Backend erstellt haben, können Sie dessen Status mithilfe von [Name des Dienstes/der Funktion] beobachten. `kubectl get tbc <tbc-name> -n <trident-namespace>` und weitere Details einholen.

Schritt 1: Erstellen Sie ein Kubernetes-Secret.

Erstellen Sie ein Geheimnis, das die Zugangsdaten für das Backend enthält. Dies ist für jeden Speicherdienst bzw. jede Speicherplattform individuell. Hier ist 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
```

Diese Tabelle fasst die Felder zusammen, die für jede Speicherplattform im Secret enthalten sein müssen:

Beschreibung der geheimen Felder der Speicherplattform	Geheimnis	Feldbeschreibung
Azure NetApp Files	Client-ID	Die Client-ID aus einer App-Registrierung
Cloud Volumes Service für GCP	private_key_id	ID des privaten Schlüssels. Teil des API-Schlüssels für ein GCP-Dienstkonto mit CVS-Administratorrolle
Cloud Volumes Service für GCP	privater_key	Privater Schlüssel. Teil des API-Schlüssels für ein GCP-Dienstkonto mit CVS-Administratorrolle
Element (NetApp HCI/ SolidFire)	Endpunkt	MVIP für den SolidFire -Cluster mit Mandantenanmeldeinformationen
ONTAP	Benutzername	Benutzername für die Verbindung mit dem Cluster/SVM. Wird für die anmeldeinformationsbasierte Authentifizierung verwendet
ONTAP	Passwort	Kennwort für die Verbindung mit dem Cluster/SVM. Wird für die anmeldeinformationsbasierte Authentifizierung verwendet
ONTAP	Client-Privatschlüssel	Base64-kodierter Wert des privaten Client-Schlüssels. Wird für die zertifikatsbasierte Authentifizierung verwendet

Beschreibung der geheimen Felder der Speicherplattform	Geheimnis	Feldbeschreibung
ONTAP	chapUsername	Eingehender Benutzername. Erforderlich, wenn useCHAP=true. Für `ontap-san` Und `ontap-san-economy`
ONTAP	KapitelInitiatorGeheimnis	Geheimnis des CHAP-Initiators. Erforderlich, wenn useCHAP=true. Für `ontap-san` Und `ontap-san-economy`
ONTAP	chapTargetUsername	Zielbenutzername. Erforderlich, wenn useCHAP=true. Für `ontap-san` Und `ontap-san-economy`
ONTAP	Kapitel „Zielinitiatorgeheimnis“	Geheimnis des CHAP-Zielinitiators. Erforderlich, wenn useCHAP=true. Für `ontap-san` Und `ontap-san-economy`

Das in diesem Schritt erstellte Geheimnis wird in der `spec.credentials` Feld des `TridentBackendConfig` Objekt, das im nächsten Schritt erstellt wird.

Schritt 2: Erstellen Sie die `TridentBackendConfig` CR

Sie können nun Ihre eigene erstellen. `TridentBackendConfig` CR. In diesem Beispiel verwendet ein Backend die `ontap-san` Der Treiber wird mithilfe des `TridentBackendConfig` Das unten abgebildete Objekt:

```
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 nun die `TridentBackendConfig` CR, Sie können den Status überprüfen. Siehe das folgende Beispiel:

```
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
```

Es wurde erfolgreich ein Backend erstellt und angebunden. `TridentBackendConfig` CR.

Die Phase kann einen der folgenden Werte annehmen:

- `Bound`: Der `TridentBackendConfig` CR ist mit einem Backend verknüpft, und dieses Backend enthält `configRef` auf die `TridentBackendConfig` CR's uid.
- `Unbound`: Dargestellt durch `""` . Der `TridentBackendConfig` Das Objekt ist nicht an ein Backend gebunden. Alle neu erstellt `TridentBackendConfig` CRs befinden sich standardmäßig in dieser Phase. Nach den Phasenübergängen kann der Zustand „Ungebunden“ nicht mehr rückgängig gemacht werden.
- `Deleting`: Der `TridentBackendConfig` CRs `deletionPolicy` wurde zum Löschen ausgewählt. Wenn die `TridentBackendConfig` Wenn CR gelöscht wird, wechselt es in den Status „Wird gelöscht“.
  - Wenn im Backend keine persistenten Volumenansprüche (PVCs) vorhanden sind, wird deren gelöscht. `TridentBackendConfig` wird dazu führen, dass Trident sowohl das Backend als auch das `TridentBackendConfig` CR.
  - Wenn im Backend ein oder mehrere PVCs vorhanden sind, wechselt das System in den Löschzustand. Der `TridentBackendConfig` Anschließend tritt CR auch in die Löschphase ein. Das Backend und `TridentBackendConfig` werden erst gelöscht, nachdem alle PVCs gelöscht wurden.
- `Lost`: Das zugehörige Backend `TridentBackendConfig` CR wurde versehentlich oder absichtlich gelöscht und die `TridentBackendConfig` CR enthält noch einen Verweis auf das gelöschte Backend.

Der `TridentBackendConfig` CR kann unabhängig davon weiterhin gelöscht werden. `deletionPolicy` Wert.

- `Unknown`Trident kann den Status oder die Existenz des zugehörigen Backends nicht feststellen. `TridentBackendConfig` CR. Wenn beispielsweise der API-Server nicht antwortet oder wenn der `tridentbackends.trident.netapp.io` CRD fehlt. Dies könnte ein Eingreifen erfordern.

In diesem Stadium ist das Backend erfolgreich erstellt worden! Es gibt mehrere weitere Vorgänge, die zusätzlich durchgeführt werden können, wie zum Beispiel"Backend-Aktualisierungen und Backend-Löschungen" .

(Optional) Schritt 4: Weitere Details einholen.

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

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

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

Darüber hinaus können Sie auch einen YAML/JSON-Dump erhalten von `TridentBackendConfig` .

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

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

`backendInfo`enthält `backendName` und die `backendUUID` des Backends, das als Reaktion auf die `TridentBackendConfig` CR. Der `lastOperationStatus` Das Feld repräsentiert den Status der letzten Operation des `TridentBackendConfig` CR, die durch einen Benutzer ausgelöst werden kann (z. B. wenn ein Benutzer etwas geändert hat in `spec` oder ausgelöst durch Trident (zum Beispiel während eines Trident -Neustarts). Es kann entweder ein Erfolg oder ein Misserfolg sein. `phase` stellt den Status der Beziehung zwischen den dar `TridentBackendConfig` CR und das Backend. Im obigen Beispiel `phase` hat den Wert „Gebunden“, was bedeutet, dass `TridentBackendConfig` CR ist mit dem Backend verbunden.

Sie können die `kubectl -n trident describe tbc <tbc-cr-name>` Befehl zum Abrufen von Details zu den Ereignisprotokollen.

> Ein Backend, das ein zugehöriges Backend enthält, kann nicht aktualisiert oder gelöscht werden. `TridentBackendConfig` Objekt verwenden `tridentctl` . Um die einzelnen Schritte beim Wechsel zwischen `tridentctl` Und `TridentBackendConfig` ,"siehe hier" .

Backends verwalten

Führen Sie die Backend-Verwaltung mit kubectl durch.

Erfahren Sie, wie Sie Backend-Verwaltungsvorgänge durchführen, indem Sie `kubectl` .

Backend löschen

Durch das Löschen eines `TridentBackendConfig` Sie weisen Trident an, Backends zu löschen/beizubehalten (basierend auf `deletionPolicy`). Um ein Backend zu löschen, stellen Sie sicher, dass `deletionPolicy` ist auf Löschen eingestellt. Nur das löschen `TridentBackendConfig` , sicherstellen, dass `deletionPolicy` ist auf Beibehaltung eingestellt. Dadurch wird sichergestellt, dass das Backend weiterhin vorhanden ist und mithilfe von verwaltet werden kann. `tridentctl` .

Führen Sie den folgenden Befehl aus:

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

Trident löscht die Kubernetes-Secrets nicht, die von diesem Unternehmen verwendet wurden. `TridentBackendConfig` . Der Kubernetes-Benutzer ist für das Bereinigen von Geheimnissen verantwortlich. Beim Löschen von Geheimnissen ist Vorsicht geboten. Geheimnisse sollten nur dann gelöscht werden, wenn sie von den Backends nicht verwendet werden.

Die vorhandenen Backends ansehen

Führen Sie den folgenden Befehl aus:

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

Sie können auch ausführen `tridentctl get backend -n trident` oder `tridentctl get backend -o yaml -n trident` um eine Liste aller existierenden Backends zu erhalten. Diese Liste wird auch Backends enthalten, die mit `tridentctl` .

Aktualisieren Sie ein Backend

Es kann mehrere Gründe für ein Backend-Update geben:

- Die Zugangsdaten zum Speichersystem haben sich geändert. Um die Anmeldeinformationen zu aktualisieren, muss das Kubernetes-Secret, das im `TridentBackendConfig` Das Objekt muss aktualisiert werden. Trident aktualisiert das Backend automatisch mit den zuletzt angegebenen Zugangsdaten. Führen Sie folgenden Befehl aus, um das Kubernetes-Secret zu aktualisieren:

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

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

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

- Alternativ können Sie Änderungen an der bestehenden Datei vornehmen. `TridentBackendConfig` CR mit folgendem Befehl:

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

- Wenn ein Backend-Update fehlschlägt, bleibt das Backend in seiner zuletzt bekannten Konfiguration. Sie können die Protokolle einsehen, um die Ursache zu ermitteln, indem Sie Folgendes ausführen: `kubectl get tbc <tbc-name> -o yaml -n trident` oder `kubectl describe tbc <tbc-name> -n trident`.
- Nachdem Sie das Problem mit der Konfigurationsdatei identifiziert und behoben haben, können Sie den Aktualisierungsbefehl erneut ausführen.

Führen Sie die Backend-Verwaltung mit tridentctl durch.

Erfahren Sie, wie Sie Backend-Verwaltungsvorgänge durchführen, indem Sie `tridentctl`.

Erstelle ein Backend

Nachdem Sie ein"Backend-Konfigurationsdatei" Führen Sie folgenden Befehl aus:

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

Wenn die Backend-Erstellung fehlschlägt, gab es ein Problem mit der Backend-Konfiguration. Sie können die Protokolle einsehen, um die Ursache zu ermitteln, indem Sie folgenden Befehl ausführen:

```
tridentctl logs -n trident
```

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

Backend löschen

Um ein Backend aus Trident zu löschen, gehen Sie wie folgt vor:

1. Den Backend-Namen abrufen:

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

2. Backend löschen:

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

Falls Trident über dieses Backend Volumes und Snapshots bereitgestellt hat, die noch existieren, verhindert das Löschen des Backends, dass neue Volumes über dieses Backend bereitgestellt werden. Das Backend wird weiterhin im Status „Löschen“ verbleiben.

Die vorhandenen Backends ansehen

Um die von Trident bekannten Backends anzuzeigen, gehen Sie wie folgt vor:

- Um eine Zusammenfassung zu erhalten, führen Sie folgenden Befehl aus:

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

- Um alle Details zu erhalten, führen Sie folgenden Befehl aus:

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

Aktualisieren Sie ein Backend

Nachdem Sie eine neue Backend-Konfigurationsdatei erstellt haben, führen Sie folgenden Befehl aus:

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

Wenn das Backend-Update fehlschlägt, lag entweder ein Fehler in der Backend-Konfiguration vor oder Sie haben versucht, ein ungültiges Update durchzuführen. Sie können die Protokolle einsehen, um die Ursache zu ermitteln, indem Sie folgenden Befehl ausführen:

```
tridentctl logs -n trident
```

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

Identifizieren Sie die Speicherklassen, die ein Backend verwenden.

Dies ist ein Beispiel für die Art von Fragen, die Sie mit JSON beantworten können. `tridentctl` Ausgaben für Backend-Objekte. Dies verwendet die `jq` Dienstprogramm, 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 [fehlende Information] erstellt wurden.

`TridentBackendConfig` .

Wechseln Sie zwischen verschiedenen Backend-Verwaltungsoptionen.

Erfahren Sie mehr über die verschiedenen Möglichkeiten zur Backend-Verwaltung in Trident.

Optionen zur Verwaltung von Backends

Mit der Einführung von `TridentBackendConfig` Administratoren haben nun zwei unterschiedliche Möglichkeiten, Backends zu verwalten. Dies wirft folgende Fragen auf:

- Können Backends erstellt werden mit `tridentctl` verwaltet werden mit `TridentBackendConfig` ?
- Können Backends erstellt werden mit `TridentBackendConfig` verwaltet werden mit `tridentctl` ?

Verwalten `tridentctl` Backends verwenden `TridentBackendConfig`

Dieser Abschnitt beschreibt die erforderlichen Schritte zur Verwaltung von Backends, die mit folgendem Werkzeug erstellt wurden: `tridentctl` direkt über die Kubernetes-Schnittstelle durch Erstellen `TridentBackendConfig` Objekte.

Dies gilt für folgende Szenarien:

- Vorhandene Backends, die keine haben `TridentBackendConfig` weil sie mit `tridentctl` .
- Neue Backends, die mit `tridentctl` während andere `TridentBackendConfig` Objekte existieren.

In beiden Szenarien bleiben die Backend-Systeme weiterhin vorhanden, über die Trident die Datenmengen plant und verarbeitet. Administratoren haben hier zwei Möglichkeiten:

- Weiter verwenden `tridentctl` zur Verwaltung von Backends, die damit erstellt wurden.
- Bind-Backends, die mit `tridentctl` zu einem neuen `TridentBackendConfig` Objekt. Dies würde bedeuten, dass die Backends wie folgt verwaltet werden: `kubectl` und nicht `tridentctl` .

Um ein bereits bestehendes Backend zu verwalten `kubectl` Sie müssen einen erstellen `TridentBackendConfig` das an das bestehende Backend angebunden wird. Hier ist eine Übersicht, wie das funktioniert:

1. Erstelle ein Kubernetes-Secret. Das Geheimnis enthält die Anmeldeinformationen, die Trident für die Kommunikation mit dem Speichercluster/-dienst benötigt.
2. Erstellen Sie ein `TridentBackendConfig` Objekt. Dieser Eintrag enthält spezifische Informationen zum Speichercluster/Speicherdienst und verweist auf das im vorherigen Schritt erstellte Geheimnis. Es ist darauf zu achten, dass identische Konfigurationsparameter angegeben werden (wie z. B. `spec.backendName` , `spec.storagePrefix` , `spec.storageDriverName` , und so weiter). `spec.backendName` muss auf den Namen des bestehenden Backends eingestellt werden.

Schritt 0: Backend identifizieren

Um ein `TridentBackendConfig` Um eine Verbindung zu einem bestehenden Backend herzustellen, müssen Sie die Backend-Konfiguration abrufen. In diesem Beispiel gehen wir davon aus, dass ein Backend mit 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 1: Erstellen Sie ein Kubernetes-Secret.

Erstellen Sie ein Geheimnis, das die Anmeldeinformationen für das Backend enthält, wie in diesem Beispiel gezeigt:

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

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

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

Schritt 2: Erstellen Sie ein `TridentBackendConfig` CR

Der nächste Schritt besteht darin, ein/e zu erstellen `TridentBackendConfig` CR, das automatisch an das bereits vorhandene gebunden wird `ontap-nas-backend` (wie in diesem Beispiel). Stellen Sie sicher, dass die folgenden Anforderungen erfüllt sind:

- Derselbe Backend-Name ist definiert in `spec.backendName` .
- Die Konfigurationsparameter sind identisch mit denen des ursprünglichen Backends.
- Virtuelle Pools (sofern vorhanden) müssen die gleiche Reihenfolge wie im ursprünglichen Backend beibehalten.
- Die Zugangsdaten werden über ein Kubernetes-Secret und nicht im Klartext bereitgestellt.

In diesem Fall `TridentBackendConfig` wird folgendermaßen aussehen:

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

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

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

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

Nach dem `TridentBackendConfig` wurde erstellt, seine Phase muss sein `Bound` . Es sollte außerdem denselben Backend-Namen und dieselbe UUID wie das bestehende Backend aufweisen.

```
kubectl get tbc tbc-ontap-nas-backend -n trident
NAME                   BACKEND NAME          BACKEND UUID
PHASE   STATUS
tbc-ontap-nas-backend  ontap-nas-backend     52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7   Bound   Success

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

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

Verwalten `TridentBackendConfig` Backends verwenden `tridentctl`

```
`tridentctl`kann verwendet werden, um Backends aufzulisten, die mit
folgendem Werkzeug erstellt wurden: `TridentBackendConfig` .  Darüber
hinaus können Administratoren diese Backends auch vollständig selbst
verwalten durch `tridentctl` durch Löschen `TridentBackendConfig` und
sicherstellen `spec.deletionPolicy` ist eingestellt auf `retain` .
```

Schritt 0: Backend identifizieren

Nehmen wir beispielsweise an, das folgende Backend wurde erstellt mit `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 geht hervor, dass `TridentBackendConfig` wurde erfolgreich erstellt und ist an ein Backend gebunden [siehe die UUID des Backends].

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

Werfen wir einen Blick auf den Wert von `deletionPolicy` . Dies muss auf eingestellt werden `retain` . Dies stellt sicher, dass, wenn ein `TridentBackendConfig` Wenn CR gelöscht wird, bleibt die Backend-Definition weiterhin vorhanden und kann verwaltet werden mit `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 `deletionPolicy` ist eingestellt auf `retain` .

Schritt 2: Löschen Sie die `TridentBackendConfig` CR

Der letzte Schritt besteht darin, die `TridentBackendConfig` CR. Nach Bestätigung der `deletionPolicy` ist eingestellt auf `retain` Sie können die Löschung nun durchführen:

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

Nach der Löschung der `TridentBackendConfig` Trident entfernt das Objekt einfach, ohne das Backend selbst zu löschen.

Speicherklassen erstellen und verwalten

Erstellen einer Speicherklasse

Konfigurieren Sie ein Kubernetes StorageClass-Objekt und erstellen Sie die Storage-Klasse, um Trident anzuweisen, wie Volumes bereitgestellt werden sollen.

Konfigurieren eines Kubernetes StorageClass-Objekts

Der "Kubernetes StorageClass-Objekt" Identifiziert Trident als den für diese Klasse verwendeten Provisionierer und weist Trident an, wie ein Volume zu provisionieren ist. Beispiel:

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

Siehe"Kubernetes- und Trident Objekte" Einzelheiten darüber, wie Speicherklassen mit dem interagieren, finden Sie hier. `PersistentVolumeClaim` und Parameter zur Steuerung der Volumenbereitstellung Trident .

Erstellen einer Speicherklasse

Nachdem Sie das StorageClass-Objekt erstellt haben, können Sie die Speicherklasse erstellen. Lagerklassenmuster bietet einige grundlegende Beispiele, die Sie verwenden oder modifizieren können.

Schritte

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

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

2. Sie sollten nun sowohl in Kubernetes als auch in Trident eine **basic-csi**-Speicherklasse sehen, und Trident sollte die Pools im Backend erkannt haben.

   ```
   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"
        ]
      }
    }
  ]
}
```


Lagerklassenmuster

Trident bietet "einfache Speicherklassendefinitionen für spezifische Backends" .

Alternativ können Sie bearbeiten `sample-input/storage-class-csi.yaml.templ` Die mit dem Installationsprogramm gelieferte Datei ersetzen *`BACKEND_TYPE`* mit dem Namen des Speichertreibers.

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

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

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

Speicherklassen verwalten

Sie können vorhandene Speicherklassen anzeigen, eine Standard-Speicherklasse festlegen, das Backend der Speicherklasse identifizieren und Speicherklassen löschen.

Die vorhandenen Speicherklassen anzeigen

- Um vorhandene Kubernetes-Speicherklassen anzuzeigen, führen Sie folgenden Befehl aus:

```
kubectl get storageclass
```

- Um Details zur Kubernetes-Speicherklasse anzuzeigen, führen Sie folgenden Befehl aus:

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

- Um die synchronisierten Speicherklassen von Trident anzuzeigen, führen Sie folgenden Befehl aus:

```
tridentctl get storageclass
```

- Um die Details der synchronisierten Speicherklasse von Trident anzuzeigen, führen Sie folgenden Befehl aus:

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

Legen Sie eine Standardspeicherklasse fest

Mit Kubernetes 1.6 wurde die Möglichkeit eingeführt, eine Standard-Speicherklasse festzulegen. Dies ist die Speicherklasse, die zur Bereitstellung eines Persistent Volume verwendet wird, wenn ein Benutzer in einem Persistent Volume Claim (PVC) kein Persistent Volume angibt.

- Definieren Sie eine Standard-Speicherklasse durch Festlegen der Annotation. `storageclass.kubernetes.io/is-default-class` in der Definition der Speicherklasse auf „true“ setzen. Gemäß der Spezifikation wird jeder andere Wert oder das Fehlen der Annotation als falsch interpretiert.
- Sie können eine vorhandene Speicherklasse als Standardspeicherklasse konfigurieren, indem Sie den folgenden Befehl verwenden:

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

- Ebenso können Sie die Standard-Speicherklassenannotation mit folgendem Befehl entfernen:

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

Im Trident -Installationspaket sind auch Beispiele enthalten, die diese Annotation beinhalten.

> Es sollte immer nur eine Standard-Speicherklasse in Ihrem Cluster vorhanden sein. Kubernetes verhindert technisch gesehen nicht, dass Sie mehr als eine Speicherklasse haben, verhält sich aber so, als gäbe es überhaupt keine Standard-Speicherklasse.

Identifizieren Sie das Backend für eine Speicherklasse

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

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

Eine Speicherklasse löschen

Um eine Speicherklasse aus Kubernetes zu löschen, führen Sie folgenden Befehl aus:

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

`<storage-class>`sollte durch Ihre Speicherklasse ersetzt werden.

Alle persistenten Volumes, die über diese Speicherklasse erstellt wurden, bleiben unberührt und werden weiterhin von Trident verwaltet.

Trident erzwingt eine leere `fsType` für die dadurch entstehenden Mengen. Für iSCSI-Backends wird empfohlen, Folgendes zu erzwingen `parameters.fsType` in der StorageClass. Sie sollten die vorhandenen StorageClasses löschen und neu erstellen mit `parameters.fsType` angegeben.

Bereitstellung und Verwaltung von Volumina

Bereitstellung eines Volumens

Erstellen Sie einen PersistentVolumeClaim (PVC), der die konfigurierte Kubernetes StorageClass verwendet, um Zugriff auf das PV anzufordern. Anschließend können Sie die PV-Anlage an einem Pod montieren.

Überblick

A "*PersistentVolumeClaim*" (PVC) ist eine Anfrage für den Zugriff auf das PersistentVolume im Cluster.

Das PVC kann so konfiguriert werden, dass es Speicherplatz einer bestimmten Größe oder einen bestimmten Zugriffsmodus anfordert. Mithilfe der zugehörigen StorageClass kann der Clusteradministrator mehr als nur die Größe und den Zugriffsmodus des PersistentVolumes steuern – beispielsweise die Leistung oder das Servicelevel.

Nachdem Sie das PVC erstellt haben, können Sie das Volumen in einem Gehäuse montieren.

PVC erstellen

Schritte

1. Erstellen Sie die PVC.

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

2. Überprüfen Sie den PVC-Status.

```
kubectl get pvc
```

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

1. Montieren Sie das Volume in einem Gehäuse.

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

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

2. Überprüfen Sie, ob das Volume eingebunden ist. `/my/mount/path` .

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

3. Sie können den Pod jetzt löschen. Die Pod-Anwendung wird nicht mehr existieren, das Volume bleibt jedoch erhalten.

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

Beispielmanifeste

Beispielmanifeste für PersistentVolumeClaim

Diese Beispiele zeigen grundlegende PVC-Konfigurationsoptionen.

PVC mit RWO-Zugang

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

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

PVC mit NVMe/TCP

Dieses Beispiel zeigt eine einfache 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-Manifestproben

Diese Beispiele zeigen grundlegende Konfigurationen zur Befestigung des PVC an einer Kapsel.

Basiskonfiguration

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

Grundlegende NVMe/TCP-Konfiguration

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

Siehe"Kubernetes- und Trident Objekte" Einzelheiten darüber, wie Speicherklassen mit dem interagieren, finden Sie hier. `PersistentVolumeClaim` und Parameter zur Steuerung der Volumenbereitstellung Trident .

Volumen erweitern

Trident bietet Kubernetes-Benutzern die Möglichkeit, ihre Volumes nach der Erstellung zu erweitern. Hier finden Sie Informationen zu den Konfigurationen, die zum Erweitern von iSCSI-, NFS-, SMB-, NVMe/TCP- und FC-Volumes erforderlich sind.

Erweitern eines iSCSI-Volumes

Sie können ein iSCSI Persistent Volume (PV) mithilfe des CSI-Provisioners erweitern.

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

Schritt 1: Konfigurieren Sie die StorageClass zur Unterstützung der Volume-Erweiterung.

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

```
cat storageclass-ontapsan.yaml
```

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

Bearbeiten Sie eine bereits vorhandene StorageClass, um Folgendes einzuschließen: `allowVolumeExpansion` Parameter.

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

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

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

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

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

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

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

Schritt 3: Definieren Sie eine Halterung, an der das PVC befestigt wird.

Befestigen Sie das PV an einer Kapsel, damit es in seiner Größe angepasst werden kann. Beim Ändern der Größe eines iSCSI PV gibt es zwei Szenarien:

- Wenn das PV an einen Pod angehängt ist, erweitert Trident das Volume im Speicher-Backend, scannt das Gerät erneut und passt die Größe des Dateisystems an.
- Beim Versuch, die Größe eines nicht zugeordneten PV zu ändern, erweitert Trident das Volumen im Speicher-Backend. Nachdem die PVC an einen Pod gebunden wurde, scannt Trident das Gerät erneut und passt die Größe des Dateisystems an. Kubernetes aktualisiert anschließend die PVC-Größe, nachdem der Erweiterungsvorgang erfolgreich abgeschlossen wurde.

In diesem Beispiel wird ein Pod erstellt, der Folgendes 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: PV erweitern

Um die Größe des erstellten PV von 1 GiB auf 2 GiB zu ändern, bearbeiten Sie die PVC-Definition und aktualisieren Sie die `spec.resources.requests.storage` bis 2 GiB.

```
kubectl edit pvc san-pvc
```

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

Schritt 5: Die Erweiterung validieren

Sie können überprüfen, ob die Erweiterung korrekt funktioniert hat, indem Sie die Größe des PVC, PV und des Trident -Volumens kontrollieren:

```
kubectl get pvc san-pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi
RWO            ontap-san      11m
kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM              STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi        RWO
Delete           Bound    default/san-pvc    ontap-san               12m
tridentctl get volumes -n trident
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
| pvc-8a814d62-bd58-4253-b0d1-82f2885db671 | 2.0 GiB | ontap-san     |
block    | a9b7bfff-0505-4e31-b6c5-59f492e02d33 | online | true    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

FC-Volume erweitern

Sie können ein FC Persistent Volume (PV) mithilfe des CSI-Provisioners erweitern.

> Die FC-Volumenerweiterung wird unterstützt durch `ontap-san` Treiber und erfordert Kubernetes 1.16 oder höher.

Schritt 1: Konfigurieren Sie die StorageClass zur Unterstützung der Volume-Erweiterung.

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

```
cat storageclass-ontapsan.yaml
```

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

Bearbeiten Sie eine bereits vorhandene StorageClass, um Folgendes einzuschließen: `allowVolumeExpansion` Parameter.

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

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

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

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

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

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

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

Schritt 3: Definieren Sie eine Halterung, an der das PVC befestigt wird.

Befestigen Sie das PV an einer Kapsel, damit es in seiner Größe angepasst werden kann. Bei der Größenänderung einer FC-PV gibt es zwei Szenarien:

- Wenn das PV an einen Pod angehängt ist, erweitert Trident das Volume im Speicher-Backend, scannt das Gerät erneut und passt die Größe des Dateisystems an.
- Beim Versuch, die Größe eines nicht zugeordneten PV zu ändern, erweitert Trident das Volumen im Speicher-Backend. Nachdem die PVC an einen Pod gebunden wurde, scannt Trident das Gerät erneut

und passt die Größe des Dateisystems an. Kubernetes aktualisiert anschließend die PVC-Größe, nachdem der Erweiterungsvorgang erfolgreich abgeschlossen wurde.

In diesem Beispiel wird ein Pod erstellt, der Folgendes 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: PV erweitern

Um die Größe des erstellten PV von 1 GiB auf 2 GiB zu ändern, bearbeiten Sie die PVC-Definition und aktualisieren Sie die `spec.resources.requests.storage` bis 2 GiB.

```
kubectl edit pvc san-pvc
```

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

Schritt 5: Die Erweiterung validieren

Sie können überprüfen, ob die Erweiterung korrekt funktioniert hat, indem Sie die Größe des PVC, PV und des Trident -Volumens kontrollieren:

```
kubectl get pvc san-pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi
RWO            ontap-san      11m
kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM             STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi        RWO
Delete           Bound    default/san-pvc   ontap-san               12m
tridentctl get volumes -n trident
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
| pvc-8a814d62-bd58-4253-b0d1-82f2885db671 | 2.0 GiB | ontap-san     |
block    | a9b7bfff-0505-4e31-b6c5-59f492e02d33 | online | true    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

Erweitern eines NFS-Volumes

Trident unterstützt die Volumenerweiterung für NFS-PVs, die auf `ontap-nas` , `ontap-nas-economy` , `ontap-nas-flexgroup` , `gcp-cvs` , Und `azure-netapp-files` Backends.

Schritt 1: Konfigurieren Sie die StorageClass zur Unterstützung der Volume-Erweiterung.

Um ein NFS-PV zu vergrößern oder zu verkleinern, muss der Administrator zunächst die Speicherklasse so konfigurieren, dass eine Volumenerweiterung möglich ist. `allowVolumeExpansion` Feld zu `true` :

```
cat storageclass-ontapnas.yaml
```

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

Wenn Sie bereits eine Speicherklasse ohne diese Option erstellt haben, können Sie die vorhandene

Speicherklasse einfach bearbeiten, indem Sie `kubectl edit storageclass` um eine Volumenausdehnung zu ermöglichen.

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

```
cat pvc-ontapnas.yaml
```

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

Trident sollte für dieses PVC ein 20 MiB NFS PV 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: PV erweitern

Um die neu erstellte 20 MiB PV auf 1 GiB zu skalieren, bearbeiten Sie die PVC und stellen Sie sie entsprechend ein. `spec.resources.requests.storage` bis 1 GiB:

```
kubectl edit pvc ontapnas20mb
```

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

Schritt 4: Die Erweiterung validieren

Sie können überprüfen, ob die Größenänderung korrekt durchgeführt wurde, indem Sie die Größe des PVC, PV und des Trident -Volumens überprüfen:

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

kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM                  STORAGECLASS    REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   1Gi        RWO
Delete           Bound    default/ontapnas20mb   ontapnas
5m35s

tridentctl get volume pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 -n trident
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
| pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 | 1.0 GiB | ontapnas      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | true    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

Importmengen

Sie können vorhandene Speichervolumes als Kubernetes PV importieren, indem Sie `tridentctl import`.

Überblick und Überlegungen

Sie könnten beispielsweise ein Volume in Trident importieren, um:

- Eine Anwendung containerisieren und ihren vorhandenen Datensatz wiederverwenden
- Verwenden Sie eine Kopie eines Datensatzes für eine kurzlebige Anwendung.
- Einen ausgefallenen Kubernetes-Cluster wiederherstellen
- Anwendungsdaten während der Notfallwiederherstellung migrieren

Überlegungen

Bevor Sie ein Volume importieren, beachten Sie bitte die folgenden Hinweise.

- Trident kann nur ONTAP -Volumes vom Typ RW (Read-Write) importieren. DP-Volumes (Datenschutz-Volumes) sind SnapMirror Zielvolumes. Sie sollten die Spiegelungsbeziehung aufheben, bevor Sie das Volume in Trident importieren.

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

Dies ist besonders wichtig für Block-Volumes, da Kubernetes die vorherige Verbindung nicht erkennen würde und leicht ein aktives Volume an einen Pod anhängen könnte. Dies kann zu Datenbeschädigung führen.

- Obwohl `StorageClass` muss auf einem PVC angegeben werden, Trident verwendet diesen Parameter beim Import nicht. Speicherklassen werden bei der Erstellung von Volumes verwendet, um anhand der Speichereigenschaften aus den verfügbaren Pools auszuwählen. Da das Volume bereits existiert, ist während des Imports keine Poolauswahl erforderlich. Daher schlägt der Import auch dann nicht fehl, wenn sich das Volume auf einem Backend oder Pool befindet, der nicht der im PVC angegebenen Speicherklasse entspricht.
- Das vorhandene Volumen wird ermittelt und im PVC festgelegt. Nachdem das Volume vom Speichertreiber importiert wurde, wird das PV mit einem ClaimRef zum PVC erstellt.
  - Die Rückforderungsrichtlinie ist anfänglich so eingestellt, dass `retain` im PV. Nachdem Kubernetes PVC und PV erfolgreich gebunden hat, wird die Rückgewinnungsrichtlinie aktualisiert, um der Rückgewinnungsrichtlinie der Speicherklasse zu entsprechen.
  - Wenn die Rückgewinnungsrichtlinie der Speicherklasse lautet `delete` Das Speichervolumen wird gelöscht, wenn das PV gelöscht wird.
- Standardmäßig verwaltet Trident die PVC und benennt das FlexVol volume und die LUN im Backend um. Du kannst die `--no-manage` Flag zum Importieren eines nicht verwalteten Volumes. Wenn Sie `--no -manage` Trident führt während des gesamten Lebenszyklus der Objekte keine zusätzlichen Bearbeitungsschritte an den PVC- oder PV-Materialien durch. Das Speichervolume wird beim Löschen des PV nicht gelöscht, und auch andere Operationen wie das Klonen von Volumes und das Ändern der Volume-Größe werden ignoriert.

Diese Option ist nützlich, wenn Sie Kubernetes für containerisierte Workloads verwenden möchten, ansonsten aber den Lebenszyklus des Speichervolumens außerhalb von Kubernetes verwalten möchten.

- Dem PVC und PV wird eine Anmerkung hinzugefügt, die einen doppelten Zweck erfüllt: Sie gibt an, dass das Volumen importiert wurde und ob PVC und PV verwaltet werden. Diese Anmerkung darf nicht verändert oder entfernt werden.

Importieren Sie ein Volume

Sie können verwenden `tridentctl import` ein Volume importieren.

Schritte

1. Erstellen Sie die Persistent Volume Claim (PVC)-Datei (zum Beispiel, `pvc.yaml`) das zur Herstellung des PVC verwendet wird. Die PVC-Datei sollte Folgendes enthalten: `name` , `namespace` , `accessModes` , Und `storageClassName` . Optional können Sie Folgendes angeben `unixPermissions` in Ihrer PVC-Definition.

   Nachfolgend ein Beispiel für eine Mindestanforderung:

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

Fügen Sie keine zusätzlichen Parameter wie PV-Name oder Volumengröße hinzu. Dies kann dazu führen, dass der Importbefehl fehlschlägt.

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

```
tridentctl import volume <backendName> <volumeName> -f <path-to-pvc-file>
```

Beispiele

Prüfen Sie die folgenden Beispiele für den Volumenimport auf unterstützte Treiber.

ONTAP NAS und ONTAP NAS FlexGroup

Trident unterstützt den Volumenimport über die `ontap-nas` Und `ontap-nas-flexgroup` Fahrer.

- Trident unterstützt keinen Volumenimport mit dem `ontap-nas-economy` Treiber.
- Der `ontap-nas` Und `ontap-nas-flexgroup` Treiber erlauben keine doppelten Datenträgernamen.

Jeder mit dem `ontap-nas` Der Treiber ist ein FlexVol volume auf dem ONTAP Cluster. Importieren von FlexVol -Volumes mit dem `ontap-nas` Der Treiber funktioniert genauso. Ein FlexVol -Volume, das bereits auf einem ONTAP Cluster vorhanden ist, kann als importiert werden `ontap-nas` PVC. In ähnlicher Weise können FlexGroup Volumes importiert werden als `ontap-nas-flexgroup` PVCs.

ONTAP NAS-Beispiele

Nachfolgend werden Beispiele für den Import eines verwalteten und eines nicht verwalteten Volumes gezeigt.

Verwaltetes Volumen

Das folgende Beispiel importiert ein Volume mit dem Namen `managed_volume` auf einem Backend namens `ontap_nas` :

```
tridentctl import volume ontap_nas managed_volume -f <path-to-pvc-file>

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

Unverwaltetes Volumen

Bei der Verwendung des `--no-manage` Argument: Trident benennt das Volume nicht um.

Das folgende Beispiel importiert `unmanaged_volume` auf der `ontap_nas` Backend:

```
tridentctl import volume nas_blog unmanaged_volume -f <path-to-pvc-
file> --no-manage

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

ONTAP SAN

Trident unterstützt den Volumenimport mithilfe der `ontap-san` (iSCSI, NVMe/TCP und FC) und `ontap-san-economy` Treiber.

Trident kann ONTAP SAN FlexVol -Volumes importieren, die eine einzelne LUN enthalten. Dies steht im Einklang mit der `ontap-san` Treiber, der für jedes PVC ein FlexVol volume und eine LUN innerhalb des FlexVol volume erstellt. Trident importiert das FlexVol volume und verknüpft es mit der PVC-Definition. Trident

kann importieren `ontap-san-economy` Volumes, die mehrere LUNs enthalten.

ONTAP SAN-Beispiele

Nachfolgend werden Beispiele für den Import eines verwalteten und eines nicht verwalteten Volumes gezeigt.

Verwaltetes Volumen

Bei verwalteten Volumes benennt Trident das FlexVol volume um in `pvc-<uuid>` Format und die LUN innerhalb des FlexVol volume zu `lun0` .

Das folgende Beispiel importiert die `ontap-san-managed` FlexVol volume, die auf dem `ontap_san_default` Backend:

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

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

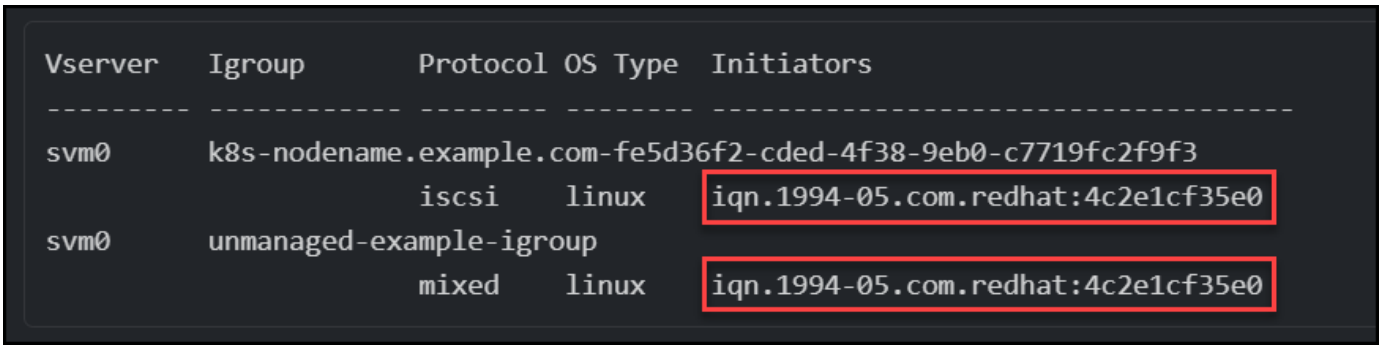
Unverwaltetes Volumen

Das folgende Beispiel importiert `unmanaged_example_volume` auf der `ontap_san` Backend:

```
tridentctl import volume -n trident san_blog unmanaged_example_volume
-f pvc-import.yaml --no-manage
+----------------------------------------+---------+--------------
+---------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------+--------------
+---------+----------------------------------------+--------+---------+
| pvc-1fc999c9-ce8c-459c-82e4-ed4380a4b228 | 1.0 GiB | san-blog      |
block    | e3275890-7d80-4af6-90cc-c7a0759f555a | online | false   |
+----------------------------------------+---------+--------------
+---------+----------------------------------------+--------+---------+
```

Wenn Sie LUNs Igroups zugeordnet haben, die einen IQN mit einem Kubernetes-Knoten-IQN teilen, wie im folgenden Beispiel gezeigt, erhalten Sie die Fehlermeldung: `LUN already mapped to initiator(s) in`

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



Element

Trident unterstützt die NetApp Element Software und den NetApp HCI Volume-Import mithilfe von `solidfire-san` Treiber.

> Der Element-Treiber unterstützt doppelte Datenträgernamen. Trident gibt jedoch eine Fehlermeldung aus, wenn doppelte Datenträgernamen vorhanden sind. Als Ausweichlösung können Sie das Volume klonen, einen eindeutigen Volume-Namen vergeben und das geklonte Volume importieren.

Elementbeispiel

Das folgende Beispiel importiert ein `element-managed` Lautstärke im 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 Volumenimport über die `gcp-cvs` Treiber.

Um ein vom NetApp Cloud Volumes Service unterstütztes Volume in die Google Cloud Platform zu importieren, identifizieren Sie das Volume anhand seines Volume-Pfads. Der Volume-Pfad ist der Teil des Exportpfads des Volumes nach dem `:/` . Wenn beispielsweise der Exportpfad lautet `10.0.0.1:/adroit-jolly-swift` Der Volumenpfad ist `adroit-jolly-swift` .

Beispiel für die Google Cloud Platform

Das folgende Beispiel importiert ein `gcp-cvs` Lautstärke im Backend `gcpcvs_YEppr` mit dem Volumenpfad von `adroit-jolly-swift` .

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

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

Azure NetApp Files

Trident unterstützt den Volumenimport über die `azure-netapp-files` Treiber.

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

Azure NetApp Files -Beispiel

Das folgende Beispiel importiert ein `azure-netapp-files` Lautstärke im Backend `azurenetappfiles_40517` mit dem Volumenpfad `importvol1` .

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

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

Google Cloud NetApp Volumes

Trident unterstützt den Volumenimport über die `google-cloud-netapp-volumes` Treiber.

Google Cloud NetApp Volumes Beispiel

Das folgende Beispiel importiert ein `google-cloud-netapp-volumes` Lautstärke im Backend `backend-tbc-gcnv1` mit dem Volumen `testvoleasiaeast1` .

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

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

Das folgende Beispiel importiert ein `google-cloud-netapp-volumes` Volumen, wenn zwei Volumina im selben Bereich vorhanden sind:

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

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

Datenträgernamen und -bezeichnungen anpassen

Mit Trident können Sie den von Ihnen erstellten Datenträgern aussagekräftige Namen und Bezeichnungen zuweisen. Dies hilft Ihnen, Volumes zu identifizieren und sie einfach ihren jeweiligen Kubernetes-Ressourcen (PVCs) zuzuordnen. Sie können auch auf Backend-Ebene Vorlagen definieren, um benutzerdefinierte Volumennamen und benutzerdefinierte Bezeichnungen zu erstellen; alle Volumen, die Sie erstellen, importieren oder klonen, werden den Vorlagen entsprechen.

Bevor Sie beginnen

Unterstützung für anpassbare Datenträgernamen und -bezeichnungen:

1. Vorgänge zum Erstellen, Importieren und Klonen von Volumes.
2. Im Falle des ontap-nas-economy-Treibers entspricht nur der Name des Qtree-Volumes der Namensvorlage.
3. Im Falle des Treibers ontap-san-economy entspricht nur der LUN-Name der Namensvorlage.

Einschränkungen

1. Anpassbare Volumennamen sind nur mit ONTAP -Treibern für lokale Installationen kompatibel.
2. Anpassbare Volumennamen gelten nicht für bereits vorhandene Volumen.

Wichtige Verhaltensweisen von anpassbaren Volumennamen

1. Wenn aufgrund ungültiger Syntax in einer Namensvorlage ein Fehler auftritt, schlägt die Backend-Erstellung fehl. Falls die Anwendung der Vorlage jedoch fehlschlägt, wird das Volume gemäß der

bestehenden Namenskonvention benannt.

2. Das Speicherpräfix ist nicht anwendbar, wenn ein Volume mithilfe einer Namensvorlage aus der Backend-Konfiguration benannt wird. Der gewünschte Präfixwert kann direkt in die Vorlage eingefügt werden.

Backend-Konfigurationsbeispiele mit Namensvorlage und Bezeichnungen

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

Beispiel auf Wurzelebene

```
{
  "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 Punkte

1. Bei Volumenimporten werden die Etiketten nur dann aktualisiert, wenn das vorhandene Volumen Etiketten in einem bestimmten Format aufweist. Zum Beispiel: `{"provisioning":{"Cluster":"ClusterA", "PVC": "pvcname"}}`.
2. Bei verwalteten Volume-Importen folgt der Volume-Name der Namensvorlage, die auf der obersten Ebene in der Backend-Definition definiert ist.
3. Trident unterstützt die Verwendung eines Slice-Operators mit dem Speicherpräfix nicht.
4. Falls die Vorlagen keine eindeutigen Datenträgernamen ergeben, fügt Trident einige zufällige Zeichen hinzu, um eindeutige Datenträgernamen zu erstellen.
5. Wenn der benutzerdefinierte Name für ein NAS Economy-Volume länger als 64 Zeichen ist, benennt Trident die Volumes gemäß der bestehenden Namenskonvention. Bei allen anderen ONTAP -Treibern schlägt der Volume-Erstellungsprozess fehl, wenn der Volume-Name die Namensbeschränkung überschreitet.

Ein NFS-Volume über Namespaces hinweg freigeben

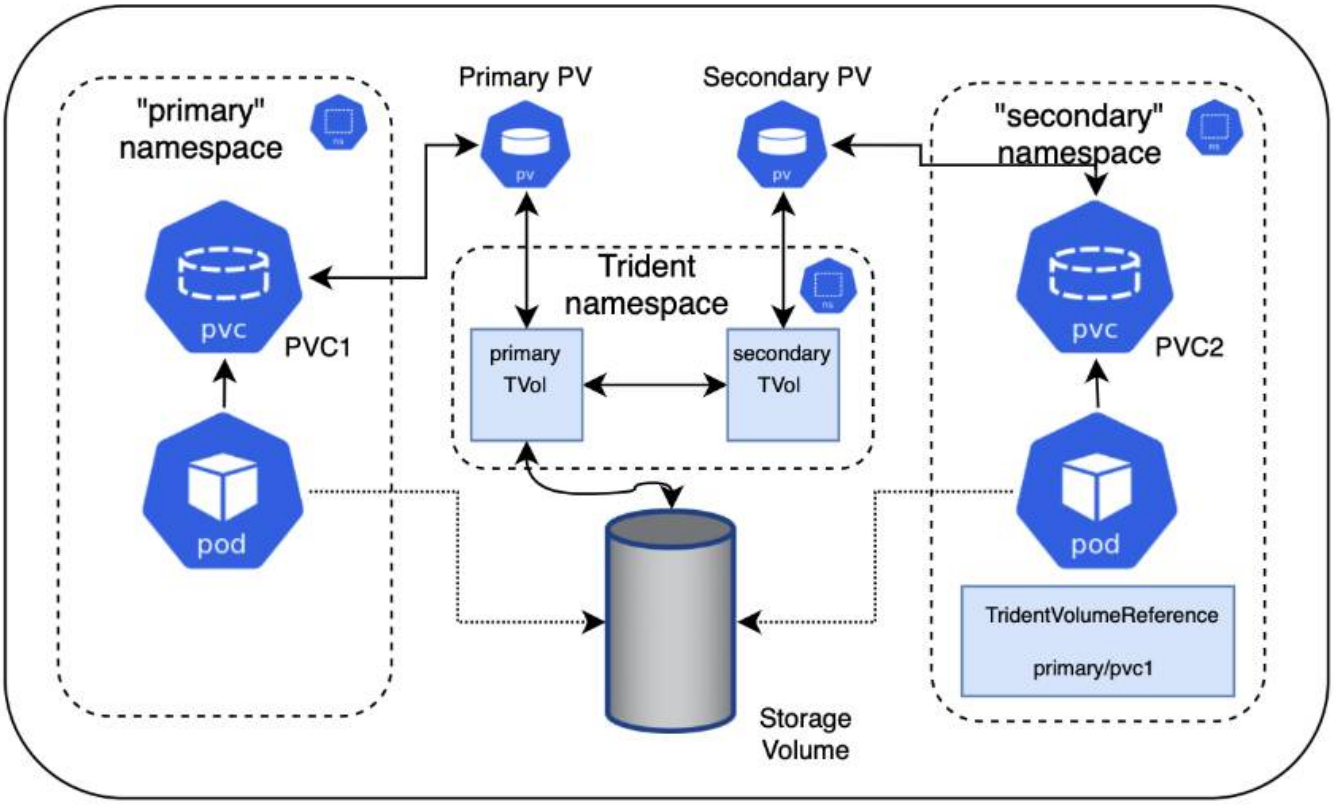
Mit Trident können Sie ein Volume in einem primären Namespace erstellen und es in einem oder mehreren sekundären Namespaces freigeben.

Features

Mit dem TridentVolumeReference CR können Sie ReadWriteMany (RWX) NFS-Volumes sicher über einen oder mehrere Kubernetes-Namespaces hinweg freigeben. Diese Kubernetes-native Lösung bietet folgende Vorteile:

- Mehrere Zugriffskontrollebenen zur Gewährleistung der Sicherheit
- Funktioniert mit allen Trident NFS-Volume-Treibern
- Keine Abhängigkeit von tridentctl oder anderen nicht-nativen Kubernetes-Funktionen.

Dieses Diagramm veranschaulicht die gemeinsame Nutzung von NFS-Volumes über zwei Kubernetes-Namespaces hinweg.



Schnellstart

Die Einrichtung der NFS-Volume-Freigabe ist in wenigen Schritten erledigt.

1 **Konfigurieren Sie den Quell-PVC so, dass er das Volume gemeinsam nutzt.**

Der Inhaber des Quell-Namespace erteilt die Berechtigung zum Zugriff auf die Daten im Quell-PVC.

2 **Erteilen Sie die Berechtigung zum Erstellen eines CR im Ziel-Namespace**

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

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

Der Eigentümer des Ziel-Namespace erstellt die TridentVolumeReference CR, um auf die Quell-PVC zu verweisen.

4 **Erstellen Sie den untergeordneten PVC im Ziel-Namespace**

Der Eigentümer des Ziel-Namensraums erstellt den untergeordneten PVC, um die Datenquelle aus dem Quell-PVC zu verwenden.

Konfigurieren Sie die Quell- und Ziel-Namespaces

Um die Sicherheit zu gewährleisten, erfordert die gemeinsame Nutzung von Namespaces über verschiedene Namensräume hinweg die Zusammenarbeit und das Handeln des Quell-Namespace-Inhabers, des Cluster-Administrators und des Ziel-Namespace-Inhabers. Die Benutzerrolle wird in jedem Schritt festgelegt.

Schritte

1. **Quellnamespace-Inhaber:** PVC erstellen(`pvc1`) im Quell-Namespace, der die Berechtigung zur gemeinsamen Nutzung mit dem Ziel-Namespace erteilt(`namespace2`) unter Verwendung der `shareToNamespace` Anmerkung.

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

   Trident erstellt das PV und sein zugehöriges NFS-Speichervolume.

   - Sie können die PVC mithilfe einer durch Kommas getrennten Liste an mehrere Namensräume weitergeben. Zum Beispiel, `trident.netapp.io/shareToNamespace: namespace2,namespace3,namespace4` .
   - Sie können die Freigabe für alle Namensräume nutzen `*` . Zum Beispiel, `trident.netapp.io/shareToNamespace: *`
   - Sie können das PVC aktualisieren, um Folgendes einzuschließen: `shareToNamespace` Anmerkungen jederzeit möglich.

2. **Clusteradministrator:** Stellen Sie sicher, dass das richtige RBAC vorhanden ist, um dem Besitzer des Zielnamespace die Berechtigung zum Erstellen des TridentVolumeReference CR im Zielnamespace zu erteilen.
3. **Inhaber des Ziel-Namespace:** Erstellen Sie im Ziel-Namespace eine TridentVolumeReference CR, die auf den Quell-Namespace verweist. `pvc1` .

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. **Inhaber des Ziel-Namensraums:** Erstellen Sie eine PVC(`pvc2`) im Ziel-Namespace(`namespace2`) unter Verwendung der `shareFromPVC` Anmerkung zur Kennzeichnung des Quell-PVC.

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

Die Größe des Ziel-PVC-Rohrs muss kleiner oder gleich der Größe des Quell-PVC-Rohrs sein.

Ergebnisse

Trident liest die `shareFromPVC` Es wird eine Annotation auf dem Ziel-PVC vorgenommen und das Ziel-PV als untergeordnetes Volume ohne eigene Speicherressource erstellt, das auf das Quell-PV verweist und die Speicherressource des Quell-PV mitnutzt. Die Ziel-PVC und PV scheinen normal verbunden zu sein.

Ein freigegebenes Volume löschen

Sie können ein Volume löschen, das von mehreren Namespaces gemeinsam genutzt wird. Trident wird den Zugriff auf das Volume im Quell-Namespace entfernen und den Zugriff für andere Namespaces, die das Volume gemeinsam nutzen, aufrechterhalten. Wenn alle Namespaces, die auf das Volume verweisen, entfernt werden, löscht Trident das Volume.

Verwenden `tridentctl get` Abfrage untergeordneter Volumes

Verwenden des[`tridentctl` Mit diesem Dienstprogramm können Sie das ausführen `get` Befehl zum Abrufen

untergeordneter Volumes. Weitere Informationen finden Sie unter folgendem Link:../trident-reference/tridentctl.html[`tridentctl` Befehle und Optionen].

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

Flaggen:

- ``-h, --help`Hilfe für Bände.
- `--parentOfSubordinate string`: Beschränken Sie die Abfrage auf das untergeordnete Quellvolume.
- `--subordinateOf string`: Beschränke die Abfrage auf untergeordnete Elemente des Volumens.

Einschränkungen

- Trident kann nicht verhindern, dass Ziel-Namespaces auf das gemeinsam genutzte Volume schreiben. Sie sollten Dateisperrungen oder andere Verfahren verwenden, um das Überschreiben von gemeinsam genutzten Datenträgerdaten zu verhindern.
- Der Zugriff auf die Quell-PVC kann nicht durch Entfernen der `shareToNamespace` oder `shareFromNamespace` Anmerkungen oder das Löschen der `TridentVolumeReference` CR. Um den Zugriff zu widerrufen, müssen Sie die untergeordnete PVC löschen.
- Snapshots, Klone und Spiegelungen sind auf untergeordneten Volumes nicht möglich.

Weitere Informationen

Um mehr über den Volume-Zugriff über Namespaces hinweg zu erfahren:

- Besuchen"Gemeinsame Nutzung von Volumes zwischen Namespaces: Lernen Sie den Namespace-übergreifenden Volume-Zugriff kennen." .
- Sehen Sie sich die Demo an auf"NetAppTV" .

Volumes über Namespaces hinweg klonen

Mit Trident können Sie neue Volumes erstellen, indem Sie vorhandene Volumes oder Volume-Snapshots aus einem anderen Namespace innerhalb desselben Kubernetes-Clusters verwenden.

Voraussetzungen

Vor dem Klonen von Volumes muss sichergestellt werden, dass Quell- und Ziel-Backend vom gleichen Typ sind und die gleiche Speicherklasse aufweisen.

> Das Klonen über Namensräume hinweg wird nur für die `ontap-san` Und `ontap-nas` Speichertreiber. Schreibgeschützte Klone werden nicht unterstützt.

Schnellstart

Das Klonen von Volumes lässt sich in wenigen Schritten einrichten.

Konfigurieren Sie die Quell-PVC zum Klonen des Volumes

Der Inhaber des Quell-Namespace erteilt die Berechtigung zum Zugriff auf die Daten im Quell-PVC.

Erteilen Sie die Berechtigung zum Erstellen eines CR im Ziel-Namespace

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

Erstellen Sie eine TridentVolumeReference im Ziel-Namespace.

Der Eigentümer des Ziel-Namespace erstellt die TridentVolumeReference CR, um auf die Quell-PVC zu verweisen.

Erstellen Sie den Klon-PVC im Ziel-Namespace

Der Eigentümer des Ziel-Namensraums erstellt einen PVC, um den PVC aus dem Quell-Namensraum zu klonen.

Konfigurieren Sie die Quell- und Ziel-Namespaces

Um die Sicherheit zu gewährleisten, erfordert das Klonen von Volumes über Namespaces hinweg die Zusammenarbeit und das Eingreifen des Quell-Namespace-Besitzers, des Cluster-Administrators und des Ziel-Namespace-Besitzers. Die Benutzerrolle wird in jedem Schritt festgelegt.

Schritte

1. **Quellnamespace-Inhaber:** PVC erstellen(`pvc1`) im Quell-Namespace(`namespace1`) die die Berechtigung zur gemeinsamen Nutzung mit dem Ziel-Namespace erteilt(`namespace2`) unter Verwendung der `cloneToNamespace` Anmerkung.

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

Trident erstellt das PV und sein Backend-Speichervolumen.

- Sie können die PVC mithilfe einer durch Kommas getrennten Liste an mehrere Namensräume weitergeben. Zum Beispiel, `trident.netapp.io/cloneToNamespace: namespace2,namespace3,namespace4` .
- Sie können die Freigabe für alle Namensräume nutzen `*` . Zum Beispiel, `trident.netapp.io/cloneToNamespace: *`
- Sie können das PVC aktualisieren, um Folgendes einzuschließen: `cloneToNamespace` Anmerkungen jederzeit möglich.

2. **Cluster-Administrator:** Stellen Sie sicher, dass die korrekte rollenbasierte Zugriffssteuerung (RBAC) eingerichtet ist, um dem Besitzer des Ziel-Namespace die Berechtigung zum Erstellen der TridentVolumeReference-CR im Ziel-Namespace zu erteilen.(`namespace2`).
3. **Inhaber des Ziel-Namespace:** Erstellen Sie im Ziel-Namespace eine TridentVolumeReference CR, die auf den Quell-Namespace verweist. `pvc1` .

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. **Inhaber des Ziel-Namensraums:** Erstellen Sie eine PVC(`pvc2`) im Ziel-Namespace(`namespace2`) unter Verwendung der `cloneFromPVC` oder `cloneFromSnapshot` , Und `cloneFromNamespace` Anmerkungen zur Kennzeichnung des Quell-PVC.

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

Einschränkungen

- Für PVCs, die mit ontap-nas-economy-Treibern bereitgestellt wurden, werden schreibgeschützte Klone nicht unterstützt.

Volumes mit SnapMirror replizieren

Trident unterstützt Spiegelungsbeziehungen zwischen einem Quellvolume auf einem Cluster und dem Zielvolume auf dem verbundenen Cluster zur Datenreplikation für die Notfallwiederherstellung. Sie können eine namens Trident Mirror Relationship (TMR) mit einem Namensraum verwenden, um die folgenden Operationen durchzuführen:

- Spiegelbeziehungen zwischen Volumina (PVCs) erzeugen
- Spiegelbeziehungen zwischen Volumen entfernen
- Zerstöre die Spiegelbeziehungen
- Förderung des sekundären Datenvolumens während Katastrophensituationen (Ausfallereignisse).
- Führen Sie einen verlustfreien Übergang von Anwendungen von Cluster zu Cluster durch (während geplanter Failover oder Migrationen).

Replikationsvoraussetzungen

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

ONTAP -Cluster

- * Trident *: Auf den Quell- und Ziel-Kubernetes-Clustern, die ONTAP als Backend verwenden, muss Trident Version 22.10 oder höher vorhanden sein.
- **Lizenzen**: ONTAP SnapMirror -Asynchronlizenzen, die das Data Protection-Bundle verwenden, müssen sowohl auf dem Quell- als auch auf dem Ziel ONTAP -Cluster aktiviert sein. Siehe "SnapMirror -Lizenzübersicht in ONTAP" für weitere Informationen.

  Ab ONTAP 9.10.1 werden alle Lizenzen als NetApp Lizenzdatei (NLF) ausgeliefert. Dabei handelt es sich um eine einzelne Datei, die mehrere Funktionen aktiviert. Siehe"In ONTAP One enthaltene Lizenzen" für weitere Informationen.

  Es wird ausschließlich der asynchrone Schutz von SnapMirror unterstützt.

Peering

- **Cluster und SVM**: Die ONTAP -Speicher-Backends müssen per Peering verbunden sein. Siehe "Cluster- und SVM-Peering-Übersicht" für weitere Informationen.

  Stellen Sie sicher, dass die in der Replikationsbeziehung zwischen zwei ONTAP Clustern verwendeten SVM-Namen eindeutig sind.

- * Trident und SVM*: Die verbundenen Remote-SVMs müssen für Trident auf dem Zielcluster verfügbar sein.

Unterstützte Treiber

NetApp Trident unterstützt die Volume-Replikation mit der NetApp SnapMirror -Technologie unter Verwendung von Speicherklassen, die von den folgenden Treibern unterstützt werden: **`ontap-nas`: NFS** `ontap-san`:

iSCSI **`ontap-san`** **FC** `ontap-san` NVMe/TCP (erfordert mindestens ONTAP Version 9.15.1)

Die Volume-Replikation mit SnapMirror wird für ASA r2-Systeme nicht unterstützt. Informationen zu ASA r2-Systemen finden Sie unter"Erfahren Sie mehr über ASA R2-Speichersysteme" .

Erstellen Sie ein spiegelndes PVC

Folgen Sie diesen Schritten und verwenden Sie die CRD-Beispiele, um eine Spiegelbeziehung zwischen primären und sekundären Volumes herzustellen.

Schritte

1. Führen Sie die folgenden Schritte auf dem primären Kubernetes-Cluster durch:
   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. Erstellen Sie eine PVC mit einer zuvor erstellten StorageClass.

      Beispiel

      ```
      kind: PersistentVolumeClaim
      apiVersion: v1
      metadata:
        name: csi-nas
      spec:
        accessModes:
        - ReadWriteMany
        resources:
          requests:
            storage: 1Gi
        storageClassName: csi-nas
      ```

   c. Erstellen Sie eine MirrorRelationship-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 Datenschutzstatus (DP) des Volumes ab und füllt dann das Statusfeld der MirrorRelationship aus.

d. Ermitteln Sie mithilfe der TridentMirrorRelationship CR den internen Namen und die SVM der PVC.

```
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 die folgenden Schritte auf dem sekundären Kubernetes-Cluster durch:

   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 konfigurierten Beziehungsrichtliniennamen (oder dem Standardnamen für ONTAP) und initialisiert diese.

c. Erstellen Sie eine PVC mit einer zuvor erstellten StorageClass, die als sekundäres Ziel (SnapMirror Ziel) fungieren soll.

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 prüft, ob die TridentMirrorRelationship CRD vorhanden ist, und schlägt fehl, das Volume zu erstellen, wenn die Beziehung nicht existiert. Wenn die Beziehung besteht, sorgt Trident dafür, dass das neue FlexVol volume auf einer SVM platziert wird, die mit der in der MirrorRelationship definierten Remote-SVM verbunden ist.

Volumenreplikationszustände

Eine Trident Mirror Relationship (TMR) ist eine CRD, die ein Ende einer Replikationsbeziehung zwischen PVCs darstellt. Das Ziel-TMR verfügt über einen Zustand, der Trident mitteilt, welcher Zustand gewünscht ist. Das Ziel-TMR hat folgende Zustände:

- **Festgestellt**: Das lokale PVC ist das Zielvolumen einer Spiegelbeziehung, und dies ist eine neue Beziehung.
- **Verbessert**: Die lokale PVC ist lesbar und beschreibbar und kann eingebunden werden; derzeit besteht keine Spiegelungsbeziehung.
- **Wiederhergestellt**: Das lokale PVC ist das Zielvolumen einer Spiegelbeziehung und war auch zuvor Teil dieser Spiegelbeziehung.
  - Der wiederhergestellte Zustand muss verwendet werden, wenn das Zielvolume jemals in einer Beziehung zum Quellvolume stand, da er den Inhalt des Zielvolumes überschreibt.
  - Der wiederhergestellte Zustand schlägt fehl, wenn das Volumen zuvor nicht in Beziehung zur Quelle stand.

Förderung der sekundären PVC-Funktion bei einem ungeplanten Ausfall

Führen Sie die folgenden Schritte auf dem sekundären Kubernetes-Cluster durch:

- Aktualisiere das Feld *spec.state* der TridentMirrorRelationship auf `promoted` .

Sekundäre PVC während eines geplanten Ausfalls fördern

Führen Sie während eines geplanten Failovers (Migration) die folgenden Schritte durch, um den sekundären PVC zu fördern:

Schritte

1. Auf dem primären Kubernetes-Cluster wird ein Snapshot des PVC erstellt und gewartet, bis der Snapshot erstellt ist.
2. Erstellen Sie auf dem primären Kubernetes-Cluster die 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 auf dem sekundären Kubernetes-Cluster das Feld *spec.state* der *TridentMirrorRelationship* CR auf *promoted* und *spec.promotedSnapshotHandle* auf den internen Namen

des Snapshots.

4. Bestätigen Sie auf dem sekundären Kubernetes-Cluster, dass der Status (status.state-Feld) der TridentMirrorRelationship auf „promoted“ gesetzt ist.

Nach einem Failover eine Spiegelbeziehung wiederherstellen

Bevor Sie die Spiegelbeziehung wiederherstellen, wählen Sie die Seite aus, die Sie zur neuen primären Seite machen möchten.

Schritte

1. Stellen Sie auf dem sekundären Kubernetes-Cluster sicher, dass die Werte für das Feld *spec.remoteVolumeHandle* in der TridentMirrorRelationship aktualisiert werden.
2. Aktualisieren Sie auf dem sekundären Kubernetes-Cluster das Feld *spec.mirror* der TridentMirrorRelationship auf `reestablished`.

Zusätzliche Operationen

Trident unterstützt die folgenden Operationen auf den primären und sekundären Datenträgern:

Primäres PVC in ein neues sekundäres PVC replizieren

Stellen Sie sicher, dass Sie bereits einen primären PVC-Anschluss und einen sekundären PVC-Anschluss haben.

Schritte

1. Löschen Sie die PersistentVolumeClaim- und TridentMirrorRelationship-CRDs aus dem eingerichteten sekundären (Ziel-)Cluster.
2. Löschen Sie die TridentMirrorRelationship CRD 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 eines gespiegelten, primären oder sekundären PVC

Die PVC-Größe kann wie gewohnt angepasst werden. ONTAP erweitert automatisch alle Ziel-Flevxols, wenn die Datenmenge die aktuelle Größe überschreitet.

Replikation aus einem PVC entfernen

Um die Replikation zu entfernen, führen Sie eine der folgenden Operationen auf dem aktuellen sekundären Volume durch:

- Löschen Sie die MirrorRelationship auf dem sekundären PVC. Dadurch wird die Replikationsbeziehung unterbrochen.
- Oder aktualisieren Sie das Feld spec.state auf *promoted*.

Löschen Sie eine PVC (die zuvor gespiegelt wurde)

Trident prüft, ob PVCs repliziert sind, und gibt die Replikationsbeziehung frei, bevor versucht wird, das Volume zu löschen.

Löschen eines TMR

Das Löschen eines TMR auf einer Seite einer gespiegelten Beziehung führt dazu, dass der verbleibende TMR in den Status *promoted* wechselt, bevor Trident den Löschvorgang abschließt. Wenn sich das zum Löschen ausgewählte TMR bereits im Status *promoted* befindet, besteht keine Spiegelbeziehung und das TMR wird entfernt. Trident wird dann den lokalen PVC auf *ReadWrite* hochstufen. Durch diese Löschung werden SnapMirror Metadaten für das lokale Volume in ONTAP freigegeben. Wenn dieses Volume in Zukunft in einer Spiegelbeziehung verwendet wird, muss beim Erstellen der neuen Spiegelbeziehung ein neues TMR mit einem *etablierten* Volume-Replikationsstatus verwendet werden.

Spiegelbeziehungen aktualisieren, wenn ONTAP online ist

Spiegelbeziehungen können jederzeit nach ihrer Einrichtung aktualisiert werden. Sie können die `state: promoted` oder `state: reestablished` Felder, um die Beziehungen zu aktualisieren. Beim Hochstufen eines Zielvolumes zu einem regulären ReadWrite-Volume können Sie mit *promotedSnapshotHandle* einen bestimmten Snapshot angeben, auf den das aktuelle Volume wiederhergestellt werden soll.

Spiegelbeziehungen aktualisieren, wenn ONTAP offline ist

Sie können eine CRD verwenden, um ein SnapMirror -Update durchzuführen, ohne dass Trident eine direkte Verbindung zum ONTAP -Cluster haben muss. Beachten Sie das folgende Beispielformat für 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`spiegelt den Zustand des TridentActionMirrorUpdate CRD wider. Es kann einen Wert von *Succeeded*, *In Progress* oder *Failed* annehmen.

CSI-Topologie verwenden

Trident kann mithilfe der folgenden Technologie gezielt Volumes erstellen und an Knoten in einem Kubernetes-Cluster anhängen: "CSI-Topologiefunktion" .

Überblick

Mithilfe der CSI-Topologiefunktion kann der Zugriff auf Volumes basierend auf Regionen und Verfügbarkeitszonen auf eine Teilmenge von Knoten beschränkt werden. Cloud-Anbieter ermöglichen es Kubernetes-Administratoren heute, zonenbasierte Knoten zu erzeugen. Knoten können sich in verschiedenen Verfügbarkeitszonen innerhalb einer Region oder über mehrere Regionen verteilt befinden. Um die Bereitstellung von Volumes für Workloads in einer Multizonenarchitektur zu erleichtern, verwendet Trident die CSI-Topologie.

Erfahren Sie mehr über die CSI-Topologiefunktion. "hier," .

Kubernetes bietet zwei einzigartige Volume-Bindungsmodi:

- Mit `VolumeBindingMode` eingestellt auf `Immediate` Trident erzeugt das Volumen ohne Kenntnis der Topologie. Die Volumenbindung und die dynamische Bereitstellung werden bei der Erstellung des PVCs verarbeitet. Dies ist die Standardeinstellung. `VolumeBindingMode` und eignet sich für Cluster, die keine Topologiebeschränkungen erzwingen. Persistente Volumes werden ohne Abhängigkeit von den Planungsanforderungen des anfordernden Pods erstellt.
- Mit `VolumeBindingMode` eingestellt auf `WaitForFirstConsumer` Die Erstellung und Bindung eines Persistent Volume für ein PVC wird verzögert, bis ein Pod, der das PVC verwendet, geplant und erstellt wurde. Auf diese Weise werden Volumes erstellt, die den Planungsbeschränkungen entsprechen, die durch die Topologieanforderungen erzwungen werden.

> Der `WaitForFirstConsumer` Der Bindungsmodus erfordert keine Topologiebezeichnungen. Dies kann unabhängig von der CSI-Topologiefunktion verwendet werden.

Was du brauchst

Um die CSI-Topologie nutzen zu können, benötigen Sie Folgendes:

- Ein Kubernetes-Cluster, der einen"Unterstützte Kubernetes-Version"

```
kubectl version
Client Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:50:19Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:41:49Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
```

- Die Knoten im Cluster sollten Bezeichnungen haben, die die Topologie verdeutlichen.(`topology.kubernetes.io/region` Und `topology.kubernetes.io/zone`). Diese Bezeichnungen **müssen auf den Knoten im Cluster vorhanden sein**, bevor Trident installiert wird, damit Trident topologiebewusst ist.

```
kubectl get nodes -o=jsonpath='{range .items[*]}[{.metadata.name},
{.metadata.labels}]{"\n"}{end}' | grep --color "topology.kubernetes.io"
[node1,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node1","kubernetes.io/
os":"linux","node-
role.kubernetes.io/master":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-a"}]
[node2,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node2","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-b"}]
[node3,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node3","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-c"}]
```

Schritt 1: Erstellen Sie ein topologiebewusstes Backend

Trident -Speicher-Backends können so konzipiert werden, dass sie Volumes selektiv auf Basis von Verfügbarkeitszonen bereitstellen. Jedes Backend kann optional `supportedTopologies` Block, der eine Liste der unterstützten Zonen und Regionen darstellt. Für StorageClasses, die ein solches Backend verwenden, wird ein Volume nur erstellt, wenn es von einer Anwendung angefordert wird, die in einer unterstützten Region/Zone geplant ist.

Hier ist ein Beispiel für eine Backend-Definition:

YAML

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

JSON

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

`supportedTopologies`Dient dazu, eine Liste von Regionen und Zonen pro Backend bereitzustellen. Diese Regionen und Zonen stellen die Liste der zulässigen Werte dar, die in einer StorageClass angegeben werden können. Für StorageClasses, die eine Teilmenge der im Backend bereitgestellten Regionen und Zonen enthalten, erstellt Trident ein Volume im Backend.

Sie können definieren `supportedTopologies` ebenfalls pro Speicherpool. Siehe das folgende Beispiel:

```
---
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, `region` Und `zone` Die Etiketten kennzeichnen den Standort des Speicherbeckens. `topology.kubernetes.io/region` Und `topology.kubernetes.io/zone` vorgeben, von wo aus die Speicherpools genutzt werden können.

Schritt 2: Topologiebewusste Speicherklassen definieren

Basierend auf den Topologiebezeichnungen, die den Knoten im Cluster zugewiesen werden, können StorageClasses definiert werden, die Topologieinformationen enthalten. Dadurch werden die Speicherpools bestimmt, die als Kandidaten für PVC-Anfragen dienen, sowie die Teilmenge der Knoten, die die von Trident bereitgestellten Volumes nutzen können.

Siehe das folgende Beispiel:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata: null
name: netapp-san-us-east1
provisioner: csi.trident.netapp.io
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
  - matchLabelExpressions: null
  - key: topology.kubernetes.io/zone
    values:
      - us-east1-a
      - us-east1-b
  - key: topology.kubernetes.io/region
    values:
      - us-east1
parameters:
  fsType: ext4
```

In der oben angegebenen StorageClass-Definition, `volumeBindingMode` ist eingestellt auf `WaitForFirstConsumer` . PVCs, die mit dieser StorageClass angefordert werden, werden erst dann verarbeitet, wenn sie in einem Pod referenziert werden. Und, `allowedTopologies` gibt die zu verwendenden Zonen und Regionen an. Der `netapp-san-us-east1` StorageClass erstellt PVCs auf dem `san-backend-us-east1` Das Backend ist oben definiert.

Schritt 3: PVC-Rohr erstellen und verwenden

Nachdem die StorageClass erstellt und einem Backend zugeordnet wurde, können Sie nun PVCs erstellen.

Siehe das Beispiel. `spec` unten:

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

Die Erstellung eines PVC mithilfe dieses Manifests hätte folgende Folgen:

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

Um mit Trident ein Volumen zu erzeugen und es an das PVC zu binden, verwenden Sie das PVC in einer Kapsel. Siehe das folgende Beispiel:

```
apiVersion: v1
kind: Pod
metadata:
  name: app-pod-1
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: topology.kubernetes.io/region
            operator: In
            values:
            - us-east1
      preferredDuringSchedulingIgnoredDuringExecution:
      - weight: 1
        preference:
          matchExpressions:
          - key: topology.kubernetes.io/zone
            operator: In
            values:
            - us-east1-a
            - us-east1-b
  securityContext:
    runAsUser: 1000
    runAsGroup: 3000
    fsGroup: 2000
  volumes:
  - name: vol1
    persistentVolumeClaim:
      claimName: pvc-san
  containers:
  - name: sec-ctx-demo
    image: busybox
    command: [ "sh", "-c", "sleep 1h" ]
    volumeMounts:
    - name: vol1
      mountPath: /data/demo
    securityContext:
      allowPrivilegeEscalation: false
```

Diese PodSpec weist Kubernetes an, den Pod auf Knoten zu planen, die im Netzwerk vorhanden sind. `us-east1` Region und wähle einen beliebigen Knoten aus, der in der Region vorhanden ist. `us-east1-a` oder `us-east1-b` Zonen.

Siehe folgende Ausgabe:

```
kubectl get pods -o wide
NAME        READY   STATUS    RESTARTS   AGE   IP               NODE
NOMINATED NODE   READINESS GATES
app-pod-1   1/1     Running   0          19s   192.168.25.131   node2
<none>           <none>
kubectl get pvc -o wide
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS          AGE   VOLUMEMODE
pvc-san   Bound    pvc-ecb1e1a0-840c-463b-8b65-b3d033e2e62b   300Mi
RWO            netapp-san-us-east1   48s   Filesystem
```

Aktualisieren Sie die Backends, um Folgendes einzuschließen `supportedTopologies`

Vorhandene Backends können aktualisiert werden, um eine Liste von `supportedTopologies` mit `tridentctl backend update` . Dies betrifft nicht bereits bereitgestellte Volumes und wird nur für nachfolgende PVCs verwendet.

Weitere Informationen

- "Ressourcen für Container verwalten"
- "Knotenselektor"
- "Affinität und Anti-Affinität"
- "Makel und Duldungen"

Mit Snapshots arbeiten

Kubernetes-Volume-Snapshots von Persistent Volumes (PVs) ermöglichen die Erstellung von zeitpunktbezogenen Kopien von Volumes. Sie können einen Snapshot eines mit Trident erstellten Volumes erstellen, einen außerhalb von Trident erstellten Snapshot importieren, ein neues Volume aus einem vorhandenen Snapshot erstellen und Volume-Daten aus Snapshots wiederherstellen.

Überblick

Volume-Snapshots werden unterstützt von `ontap-nas` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy` , `solidfire-san` , `gcp-cvs` , `azure-netapp-files` , Und `google-cloud-netapp-volumes` Fahrer.

Bevor Sie beginnen

Sie benötigen einen externen Snapshot-Controller und benutzerdefinierte Ressourcendefinitionen (CRDs), um mit Snapshots arbeiten zu können. Dies fällt in den Verantwortungsbereich des Kubernetes-Orchestrators (z. B. Kubeadm, GKE, OpenShift).

Falls Ihre Kubernetes-Distribution den Snapshot-Controller und die CRDs nicht enthält, lesen Sie bitte weiter unterEinen Volume-Snapshot-Controller bereitstellen .

Erstellen Sie keinen Snapshot-Controller, wenn Sie in einer GKE-Umgebung bedarfsgesteuerte Volume-Snapshots erstellen. GKE verwendet einen integrierten, versteckten Snapshot-Controller.

Erstellen Sie einen Volume-Snapshot

Schritte

1. Erstellen Sie ein `VolumeSnapshotClass` Weitere Informationen finden Sie unter"VolumeSnapshotClass" .
   - Der `driver` zeigt auf den Trident CSI-Fahrer.
   - `deletionPolicy`kann sein `Delete` oder `Retain` . Wenn eingestellt auf `Retain` Der zugrunde liegende physische Snapshot des Speicherclusters bleibt auch dann erhalten, wenn `VolumeSnapshot` Das Objekt wurde gelöscht.

     Beispiel

     ```
     cat snap-sc.yaml
     ```

     ```
     apiVersion: snapshot.storage.k8s.io/v1
     kind: VolumeSnapshotClass
     metadata:
       name: csi-snapclass
     driver: csi.trident.netapp.io
     deletionPolicy: Delete
     ```

2. Erstellen Sie einen Snapshot eines bestehenden PVC.

   Beispiele

   - Dieses Beispiel erstellt eine Momentaufnahme eines bestehenden PVC.

     ```
     cat snap.yaml
     ```

     ```
     apiVersion: snapshot.storage.k8s.io/v1
     kind: VolumeSnapshot
     metadata:
       name: pvc1-snap
     spec:
       volumeSnapshotClassName: csi-snapclass
       source:
         persistentVolumeClaimName: pvc1
     ```

   - Dieses Beispiel erstellt ein Volume-Snapshot-Objekt für eine PVC namens `pvc1` und der Name des Snapshots ist festgelegt auf `pvc1-snap` . Ein VolumeSnapshot ist analog zu einem PVC und ist mit einem verknüpft `VolumeSnapshotContent` Objekt, das die tatsächliche Momentaufnahme darstellt.

```
kubectl create -f snap.yaml
volumesnapshot.snapshot.storage.k8s.io/pvc1-snap created

kubectl get volumesnapshots
NAME                    AGE
pvc1-snap               50s
```

- Sie können die `VolumeSnapshotContent` Objekt für das `pvc1-snap` VolumeSnapshot durch Beschreibung. Der `Snapshot Content Name` identifiziert das VolumeSnapshotContent-Objekt, das diesen Snapshot bereitstellt. Der `Ready To Use` Der Parameter gibt an, dass der Snapshot zur Erstellung eines 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 Volumen-Snapshot

Sie können verwenden `dataSource` um ein PVC mithilfe eines VolumeSnapshots zu erstellen, der `<pvc-name>` als Quelle der Daten. Sobald das PVC hergestellt ist, kann es an einer Kapsel befestigt und wie jedes andere PVC verwendet werden.

> Die PVC wird im selben Backend wie das Quellvolume erstellt. Siehe"KB: Die Erstellung eines PVC aus einem Trident PVC-Snapshot ist in einem alternativen Backend nicht möglich." .

Das folgende Beispiel erstellt die PVC-Datei mithilfe von `pvc1-snap` als Datenquelle.

```
cat pvc-from-snap.yaml
```

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: golden
  resources:
    requests:
      storage: 3Gi
  dataSource:
    name: pvc1-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

Importieren eines Volume-Snapshots

Trident unterstützt die"Kubernetes-Vorbereitungs-Snapshot-Prozess" um dem Cluster-Administrator die Möglichkeit zu geben, einen `VolumeSnapshotContent` Objekt- und Import-Snapshots, die außerhalb von Trident erstellt wurden.

Bevor Sie beginnen

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

Schritte

1. **Cluster-Administrator:** Erstellen Sie einen `VolumeSnapshotContent` Objekt, das auf den Backend-Snapshot verweist. Dadurch wird der Snapshot-Workflow in Trident gestartet.
   - Geben Sie den Namen des Backend-Snapshots an in `annotations` als `trident.netapp.io/internalSnapshotName: <"backend-snapshot-name">`.
   - Angeben `<name-of-parent-volume-in-trident>/<volume-snapshot-content-name>` In `snapshotHandle` Dies sind die einzigen Informationen, die Trident vom externen Snapshotter erhält. `ListSnapshots` Anruf.

     > Der `<volumeSnapshotContentName>` Aufgrund von Namensbeschränkungen für CRs kann der Name des Backend-Snapshots nicht immer übereinstimmen.

     Beispiel

     Das folgende Beispiel erstellt ein `VolumeSnapshotContent` Objekt, das auf einen Backend-Snapshot verweist `snap-01`.

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotContent
metadata:
  name: import-snap-content
  annotations:
    trident.netapp.io/internalSnapshotName: "snap-01"  # This is the
name of the snapshot on the backend
spec:
  deletionPolicy: Retain
  driver: csi.trident.netapp.io
  source:
    snapshotHandle: pvc-f71223b5-23b9-4235-bbfe-e269ac7b84b0/import-
snap-content # <import PV name or source PV name>/<volume-snapshot-
content-name>
  volumeSnapshotRef:
    name: import-snap
    namespace: default
```

2. **Cluster-Administrator:** Erstellen Sie die `VolumeSnapshot` CR, das sich auf die `VolumeSnapshotContent` Objekt. Hiermit wird der Zugriff zur Nutzung des/der/der/des `VolumeSnapshot` in einem gegebenen Namensraum.

   Beispiel

   Das folgende Beispiel erstellt ein `VolumeSnapshot` CR benannt `import-snap` das sich auf das bezieht `VolumeSnapshotContent` namens `import-snap-content` .

   ```
   apiVersion: snapshot.storage.k8s.io/v1
   kind: VolumeSnapshot
   metadata:
     name: import-snap
   spec:
     # volumeSnapshotClassName: csi-snapclass (not required for pre-
   provisioned or imported snapshots)
     source:
       volumeSnapshotContentName: import-snap-content
   ```

3. **Interne Verarbeitung (keine Aktion erforderlich):** Der externe Snapshotter erkennt den neu erstellten `VolumeSnapshotContent` und leitet die `ListSnapshots` Anruf. Trident erschafft die `TridentSnapshot` .
   - Der externe Snapshotter setzt die `VolumeSnapshotContent` Zu `readyToUse` und die `VolumeSnapshot` Zu `true` .
   - Trident kehrt zurück `readyToUse=true` .
4. **Jeder Benutzer:** Erstellen Sie ein `PersistentVolumeClaim` um auf das Neue Bezug zu nehmen `VolumeSnapshot` , wo die `spec.dataSource` (oder `spec.dataSourceRef`) Name ist der

`VolumeSnapshot` Name.

Beispiel

Das folgende Beispiel erstellt eine PVC, die auf die `VolumeSnapshot` namens `import-snap` .

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: simple-sc
  resources:
    requests:
      storage: 1Gi
  dataSource:
    name: import-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

Wiederherstellung von Volumendaten mithilfe von Snapshots

Das Snapshot-Verzeichnis ist standardmäßig ausgeblendet, um eine maximale Kompatibilität der mit dem bereitgestellten Volumes zu gewährleisten. `ontap-nas` Und `ontap-nas-economy` Fahrer. Aktivieren Sie die `.snapshot` Verzeichnis zum direkten Wiederherstellen von Daten aus Snapshots.

Verwenden Sie den Befehl „volume snapshot restore ONTAP CLI“, um ein Volume auf einen Zustand zurückzusetzen, der in einem vorherigen Snapshot aufgezeichnet wurde.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

Beim Wiederherstellen einer Snapshot-Kopie wird die bestehende Volume-Konfiguration überschrieben. Änderungen, die an den Volumendaten vorgenommen wurden, nachdem die Snapshot-Kopie erstellt wurde, gehen verloren.

Wiederherstellung des Volumens direkt aus einem Snapshot

Trident ermöglicht die schnelle, direkte Wiederherstellung des Volumens aus einer Momentaufnahme mithilfe der `TridentActionSnapshotRestore` (TASR) CR. Diese CR fungiert als imperative Kubernetes-Aktion und bleibt nach Abschluss der Operation 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 benötigen ein gebundenes PVC-Gehäuse und einen verfügbaren Volumen-Snapshot.

- Prüfen Sie, ob der PVC-Status „gebunden“ ist.

```
kubectl get pvc
```

- Prüfen Sie, ob der Volume-Snapshot einsatzbereit ist.

```
kubectl get vs
```

Schritte

1. TASR CR erstellen. Dieses Beispiel erstellt eine CR für PVC. `pvc1` und Volumen-Snapshot `pvc1-snapshot` .

> Der TASR CR muss sich in einem Namensraum befinden, in dem auch der PVC und der VS existieren.

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

```
apiVersion: trident.netapp.io/v1
kind: TridentActionSnapshotRestore
metadata:
  name: trident-snap
  namespace: trident
spec:
  pvcName: pvc1
  volumeSnapshotName: pvc1-snapshot
```

2. Wenden Sie die CR an, um die Wiederherstellung aus dem Snapshot durchzuführen. Dieses Beispiel stellt eine Wiederherstellung aus einem Snapshot her. `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: pvc1
    volumeSnapshotName: pvc1-snapshot
  status:
    startTime: "2023-04-14T00:20:34Z"
    completionTime: "2023-04-14T00:20:37Z"
    state: Succeeded
kind: List
metadata:
  resourceVersion: ""
```

- In den meisten Fällen wird Trident den Vorgang im Fehlerfall nicht automatisch wiederholen. Sie müssen den Vorgang wiederholen.
- Kubernetes-Benutzer ohne Administratorrechte müssen möglicherweise vom Administrator die Berechtigung erhalten, einen TASR CR in ihrem Anwendungs-Namespace zu erstellen.

Löschen eines PV mit zugehörigen Snapshots

Beim Löschen eines persistenten Volumes mit zugehörigen Snapshots wird das entsprechende Trident Volume auf den Status „Wird gelöscht“ aktualisiert. Um das Trident -Volume zu löschen, müssen die Volume-Snapshots entfernt werden.

Einen Volume-Snapshot-Controller bereitstellen

Falls Ihre Kubernetes-Distribution den Snapshot-Controller und die CRDs nicht enthält, können Sie diese wie folgt bereitstellen.

Schritte

1. Erstellen Sie 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 gegebenenfalls `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` und aktualisieren `namespace` zu Ihrem Namensraum.

Weiterführende Links

- "Volumen-Snapshots"
- "VolumeSnapshotClass"

Arbeiten mit Volumengruppen-Snapshots

Kubernetes-Volume-Gruppen-Snapshots von Persistent Volumes (PVs) NetApp Trident bietet die Möglichkeit, Snapshots mehrerer Volumes (eine Gruppe von Volume-Snapshots) zu erstellen. Dieser Snapshot einer Volumengruppe stellt Kopien von mehreren Volumen dar, die zum gleichen Zeitpunkt erstellt wurden.

VolumeGroupSnapshot ist eine Beta-Funktion in Kubernetes mit Beta-APIs. Für VolumeGroupSnapshot ist mindestens die Version Kubernetes 1.32 erforderlich.

Erstellen von Snapshots von Volumengruppen

Die Snapshot-Funktion für Volumengruppen wird unterstützt. `ontap-san` Treiber nur für das iSCSI-Protokoll, noch nicht unterstützt mit Fibre Channel (FCP) oder NVMe/TCP. Bevor Sie beginnen

- Stellen Sie sicher, dass Ihre Kubernetes-Version K8s 1.32 oder höher ist.
- Sie benötigen einen externen Snapshot-Controller und benutzerdefinierte Ressourcendefinitionen (CRDs), um mit Snapshots arbeiten zu können. Dies fällt in den Verantwortungsbereich des Kubernetes-Orchestrators (z. B. Kubeadm, GKE, OpenShift).

  Falls Ihre Kubernetes-Distribution den externen Snapshot-Controller und die CRDs nicht enthält, lesen Sie bitte weiter unterEinen Volume-Snapshot-Controller bereitstellen .

  > Erstellen Sie keinen Snapshot-Controller, wenn Sie in einer GKE-Umgebung bedarfsgesteuerte Volume-Group-Snapshots erstellen. GKE verwendet einen integrierten, versteckten Snapshot-Controller.

- Im Snapshot-Controller-YAML legen Sie Folgendes fest: `CSIVolumeGroupSnapshot` Feature-Gate auf „true“ setzen, um sicherzustellen, dass die Snapshot-Funktion für Volumengruppen aktiviert ist.
- Erstellen Sie die erforderlichen Volume-Group-Snapshot-Klassen, bevor Sie einen Volume-Group-Snapshot erstellen.
- Stellen Sie sicher, dass sich alle PVCs/Volumes auf derselben SVM befinden, um VolumeGroupSnapshot erstellen zu können.

Schritte

- Erstellen Sie eine VolumeGroupSnapshotClass, bevor Sie einen VolumeGroupSnapshot erstellen. Weitere Informationen finden Sie unter"VolumeGroupSnapshotClass" .

```
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshotClass
metadata:
  name: csi-group-snap-class
  annotations:
    kubernetes.io/description: "Trident group snapshot class"
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

- Erstellen Sie PVCs mit den erforderlichen Etiketten unter Verwendung vorhandener Speicherklassen oder fügen Sie diese Etiketten zu vorhandenen PVCs hinzu.

  Das folgende Beispiel erstellt die PVC-Datei mithilfe von `pvc1-group-snap` als Datenquelle und Bezeichnung `consistentGroupSnapshot: groupA` . Definieren Sie den Label-Schlüssel und den Wert entsprechend Ihren Anforderungen.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1-group-snap
  labels:
    consistentGroupSnapshot: groupA
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Mi
  storageClassName: sc1-1
```

- Erstellen Sie einen VolumeGroupSnapshot mit derselben Bezeichnung(`consistentGroupSnapshot: groupA`) im PVC spezifiziert.

  Dieses Beispiel erstellt einen Snapshot einer Volumengruppe:

```
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshot
metadata:
  name: "vgs1"
  namespace: trident
spec:
  volumeGroupSnapshotClassName: csi-group-snap-class
  source:
    selector:
      matchLabels:
        consistentGroupSnapshot: groupA
```

Wiederherstellung von Volumendaten mithilfe eines Gruppen-Snapshots

Einzelne persistente Volumes können mithilfe der einzelnen Snapshots wiederhergestellt werden, die im Rahmen des Volume Group Snapshots erstellt wurden. Der Volume-Gruppen-Snapshot kann nicht als Einheit wiederhergestellt werden.

Verwenden Sie den Befehl „volume snapshot restore ONTAP CLI“, um ein Volume auf einen Zustand zurückzusetzen, der in einem vorherigen Snapshot aufgezeichnet wurde.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

Beim Wiederherstellen einer Snapshot-Kopie wird die bestehende Volume-Konfiguration überschrieben. Änderungen, die an den Volumendaten vorgenommen wurden, nachdem die Snapshot-Kopie erstellt wurde, gehen verloren.

Wiederherstellung des Volumens direkt aus einem Snapshot

Trident ermöglicht die schnelle, direkte Wiederherstellung des Volumens aus einer Momentaufnahme mithilfe der `TridentActionSnapshotRestore` (TASR) CR. Diese CR fungiert als imperative Kubernetes-Aktion und bleibt nach Abschluss der Operation nicht erhalten.

Weitere Informationen finden Sie unter "Wiederherstellung des Volumens direkt aus einem Snapshot".

Löschen eines PV mit zugehörigen Gruppen-Snapshots

Beim Löschen eines Gruppenvolume-Snapshots:

- Sie können VolumeGroupSnapshots als Ganzes löschen, nicht einzelne Snapshots innerhalb der Gruppe.
- Wenn PersistentVolumes gelöscht werden, während für dieses PersistentVolume ein Snapshot existiert, versetzt Trident dieses Volume in den Status "wird gelöscht", da der Snapshot entfernt werden muss, bevor das Volume sicher gelöscht werden kann.
- Wenn ein Klon mithilfe eines gruppierten Snapshots erstellt wurde und die Gruppe anschließend gelöscht werden soll, wird ein Split-on-Clone-Vorgang gestartet, und die Gruppe kann erst gelöscht werden, wenn der Split abgeschlossen ist.

Einen Volume-Snapshot-Controller bereitstellen

Falls Ihre Kubernetes-Distribution den Snapshot-Controller und die CRDs nicht enthält, können Sie diese wie folgt bereitstellen.

Schritte

1. Erstellen Sie 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-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotcl
asses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotco
ntents.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshots.
yaml
```

2. Erstellen Sie den Snapshot-Controller.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-8.2/deploy/kubernetes/snapshot-
controller/rbac-snapshot-controller.yaml
```

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-8.2/deploy/kubernetes/snapshot-
controller/setup-snapshot-controller.yaml
```

Öffnen Sie gegebenenfalls `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` und aktualisieren `namespace` zu Ihrem Namensraum.

Weiterführende Links

- "VolumeGroupSnapshotClass"
- "Volumen-Snapshots"

Trident verwalten und überwachen

Upgrade Trident

Upgrade Trident

Seit der Veröffentlichung am 24.02. folgt Trident einem viermonatigen Veröffentlichungsrhythmus und bringt jedes Kalenderjahr drei größere Versionen heraus. Jede neue Version baut auf den vorherigen Versionen auf und bietet neue Funktionen, Leistungsverbesserungen, Fehlerbehebungen und Optimierungen. Wir empfehlen Ihnen, mindestens einmal im Jahr ein Upgrade durchzuführen, um die neuen Funktionen von Trident nutzen zu können.

Zu beachtende Punkte vor dem Upgrade

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

- In einem Kubernetes-Cluster sollte nur eine einzige Trident Instanz über alle Namespaces hinweg installiert sein.
- Trident 23.07 und höher benötigen v1-Volume-Snapshots und unterstützen keine Alpha- oder Beta-Snapshots mehr.
- Wenn Sie den Cloud Volumes Service für Google Cloud in der"CVS-Dienstleistungstyp" Sie müssen die Backend-Konfiguration aktualisieren, um die `standardsw` oder `zoneredundantstandardsw` Servicelevel beim Upgrade von Trident 23.01. Versäumnis, die `serviceLevel` Im Backend könnten Fehler auftreten, die zum Ausfall von Volumes führen. Siehe "CVS-Servicebeispiele" für Details.
- Beim Upgrade ist es wichtig, dass Sie Folgendes angeben: `parameter.fsType` In `StorageClasses` Wird von Trident verwendet. Sie können löschen und neu erstellen `StorageClasses` ohne die bereits bestehenden Mengen zu beeinträchtigen.
  - Dies ist eine **Voraussetzung** für die Durchsetzung "Sicherheitskontexte" für SAN-Volumes.
  - Das Verzeichnis sample input enthält Beispiele, wie zum Beispiel `storage-class-basic.yaml.templ` und Link:https://github.com/NetApp/trident/blob/master/trident-installer/sample-input/storage-class-samples/storage-class-bronze-default.yaml[`storage-class-bronze-default.yaml` ^].
  - Weitere Informationen finden Sie unter"Bekannte Probleme" .

Schritt 1: Wählen Sie eine Version aus

Die Trident -Versionen folgen einem datumsbasierten `YY.MM` Namenskonvention, wobei "YY" die letzten beiden Ziffern des Jahres und "MM" der Monat ist. Dot-Veröffentlichungen folgen einem `YY.MM.X` Konvention, wobei "X" die Patch-Ebene angibt. Sie wählen die Version, auf die Sie aktualisieren möchten, anhand der Version aus, von der Sie aktualisieren.

- Sie können ein direktes Upgrade auf jede Zielversion durchführen, die innerhalb eines Vier-Versions-Fensters Ihrer installierten Version liegt. Beispielsweise können Sie direkt von Version 24.06 (oder einer beliebigen 24.06-Punktversion) auf Version 25.06 aktualisieren.
- Wenn Sie von einer Version außerhalb des Vier-Versions-Fensters aktualisieren, führen Sie ein mehrstufiges Upgrade durch. Befolgen Sie die Upgrade-Anweisungen für die"frühere Version" Sie führen

ein Upgrade von [Version] auf die neueste Version durch, die in den Vier-Versions-Zeitraum passt. Wenn Sie beispielsweise Version 23.07 verwenden und auf Version 25.06 aktualisieren möchten:

a. Erstes Upgrade von 23.07 auf 24.06.

b. Führen Sie anschließend ein Upgrade von Version 24.06 auf 25.06 durch.

> Bei einem Upgrade mit dem Trident -Operator auf der OpenShift Container Platform sollten Sie auf Trident 21.01.1 oder höher aktualisieren. Der mit Version 21.01.0 veröffentlichte Trident Operator enthielt ein bekanntes Problem, das in Version 21.01.1 behoben wurde. Weitere Einzelheiten finden Sie unter "Details zum Problem auf GitHub" .

Schritt 2: Ermitteln Sie die ursprüngliche Installationsmethode

Um herauszufinden, welche Version Sie ursprünglich für die Installation von Trident verwendet haben:

1. Verwenden `kubectl get pods -n trident` um die Schoten zu untersuchen.
   - Falls keine Bedienerkabine vorhanden ist, wurde Trident installiert mit `tridentctl` .
   - Falls ein Operator-Pod vorhanden ist, wurde Trident entweder manuell oder mithilfe von Helm mit dem Trident -Operator installiert.
2. Falls ein Bedienerpod vorhanden ist, verwenden Sie `kubectl describe torc` um festzustellen, ob Trident mit Helm installiert wurde.
   - Wenn ein Helm-Label vorhanden ist, wurde Trident mit Helm installiert.
   - Falls kein Helm-Label vorhanden ist, wurde Trident manuell mit dem Trident -Operator installiert.

Schritt 3: Wählen Sie eine Upgrade-Methode aus

Im Allgemeinen sollten Sie für das Upgrade dieselbe Methode verwenden, die Sie für die Erstinstallation genutzt haben."Wechseln zwischen Installationsmethoden" . Es gibt zwei Möglichkeiten, Trident aufzurüsten.

- "Upgrade mit dem Trident -Operator"

> Wir empfehlen Ihnen, Folgendes zu überprüfen"Den Workflow für die Bediener-Upgrades verstehen" vor dem Upgrade beim Anbieter.

*

Upgrade beim Betreiber

Den Workflow für die Bediener-Upgrades verstehen

Bevor Sie den Trident -Operator zum Aktualisieren von Trident verwenden, sollten Sie die Hintergrundprozesse verstehen, die während der Aktualisierung ablaufen. Dies umfasst Änderungen am Trident -Controller, Controller-Pod und Node-Pods sowie am Node-DaemonSet, die Rolling Updates ermöglichen.

Trident -Bedienungs-Upgrade-Handhabung

Einer der vielen"Vorteile der Verwendung des Trident Bedienelements" Die Installation und Aktualisierung von Trident umfasst die automatische Handhabung von Trident und Kubernetes-Objekten, ohne bestehende

eingebundene Volumes zu beeinträchtigen. Auf diese Weise kann Trident Upgrades ohne Ausfallzeiten unterstützen, oder"*laufende Aktualisierungen*" . Insbesondere kommuniziert der Trident -Operator mit dem Kubernetes-Cluster, um:

- Löschen und erstellen Sie die Trident Controller-Bereitstellung und den Knoten-DaemonSet neu.
- Ersetzen Sie den Trident Controller Pod und die Trident Node Pods durch neue Versionen.
  - Wenn ein Knoten nicht aktualisiert wird, hindert dies die Aktualisierung der übrigen Knoten nicht.
  - Nur Knoten mit einem laufenden Trident Node Pod können Volumes einbinden.

Weitere Informationen zur Trident -Architektur im Kubernetes-Cluster finden Sie unter:"Trident -Architektur" .

Workflow zur Bediener-Upgrade

Wenn Sie ein Upgrade mit dem Trident -Operator starten:

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 DaemonSet für *n*. Der Operator wartet nicht auf die Beendigung des Node Pods.
   b. Erstellt das Trident Node Daemonset für *n+1*.
4. **Kubernetes** erstellt Trident Node Pods auf Knoten, auf denen kein Trident Node Pod *n* ausgeführt wird. Dadurch wird sichergestellt, dass sich auf einem Knoten niemals mehr als ein Trident Node Pod, unabhängig von der Version, befindet.

Aktualisieren Sie eine Trident Installation mit dem Trident -Operator oder Helm.

Sie können Trident mithilfe des Trident -Operators entweder manuell oder mit Helm aktualisieren. Sie können von einer Trident -Bedieninstallation auf eine andere Trident -Bedieninstallation aktualisieren oder von einer `tridentctl` Installation auf einer Trident -Operatorversion. Rezension"Wählen Sie eine Upgrade-Methode aus" vor der Aufrüstung einer Trident -Bedienanlage.

Manuelle Installation aktualisieren

Sie können von einer Trident -Operatorinstallation mit Cluster-Bereich auf eine andere Trident -Operatorinstallation mit Cluster-Bereich aktualisieren. Alle Trident -Versionen verwenden einen Cluster-Scope-Operator.

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

Informationen zu diesem Vorgang

Trident stellt eine Bundle-Datei bereit, mit der Sie den Operator installieren und zugehörige Objekte für Ihre Kubernetes-Version erstellen können.

- Für Cluster, auf denen Kubernetes 1.24 läuft, verwenden Sie"bundle_pre_1_25.yaml" .
- Für Cluster, auf denen Kubernetes 1.25 oder höher ausgeführt wird, verwenden Sie"bundle_post_1_25.yaml" .

Bevor Sie beginnen

Stellen Sie sicher, dass Sie einen laufenden Kubernetes-Cluster verwenden."eine unterstützte Kubernetes-Version" .

Schritte

1. Überprüfen Sie Ihre Trident -Version:

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

2. Aktualisieren Sie die `operator.yaml` , `tridentorchestrator_cr.yaml` , Und `post_1_25_bundle.yaml` mit den Registry- und Image-Pfaden für die Version, auf die Sie aktualisieren (z. B. 25.06), und dem korrekten Geheimnis.
3. Löschen Sie den Trident Operator, der zum Installieren der aktuellen Trident Instanz verwendet wurde. Wenn Sie beispielsweise ein Upgrade von 25.02 durchführen, führen Sie den folgenden Befehl aus:

```
kubectl delete -f 25.02.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

4. Wenn Sie Ihre Erstinstallation angepasst haben mit `TridentOrchestrator` Attribute, die Sie bearbeiten können `TridentOrchestrator` Objekt zum Ändern der Installationsparameter. Dies kann Änderungen umfassen, die vorgenommen werden, um gespiegelte Trident und CSI-Image-Registries für den Offline-Modus festzulegen, Debug-Protokolle zu aktivieren oder Image-Pull-Secrets anzugeben.
5. Installieren Sie Trident mithilfe der korrekten Bundle-YAML-Datei für Ihre Umgebung, wobei *<bundle.yaml>* für `bundle_pre_1_25.yaml` oder `bundle_post_1_25.yaml` basierend auf Ihrer Kubernetes-Version. Wenn Sie beispielsweise Trident 25.06.0 installieren, führen Sie den folgenden Befehl aus:

```
kubectl create -f 25.06.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

6. Bearbeiten Sie den Dreizack-Halsreif, um das Bild 25.06.0 einzufügen.

Aktualisieren einer Helm-Installation

Sie können eine Trident Helm-Installation aktualisieren.

Beim Upgrade eines Kubernetes-Clusters von Version 1.24 auf 1.25 oder höher, auf dem Trident installiert ist, muss die Datei values.yaml aktualisiert werden, um Folgendes festzulegen: `excludePodSecurityPolicy` Zu `true` oder hinzufügen `--set excludePodSecurityPolicy=true` zum `helm upgrade` Führen Sie diesen Befehl aus, bevor Sie den Cluster aktualisieren können.

Wenn Sie Ihren Kubernetes-Cluster bereits von Version 1.24 auf 1.25 aktualisiert haben, ohne Trident Helm zu aktualisieren, schlägt das Helm-Upgrade fehl. Damit das Helm-Upgrade erfolgreich durchgeführt werden kann, müssen folgende Schritte als Voraussetzung ausgeführt werden:

1. Installieren Sie das helm-mapkubeapis-Plugin von https://github.com/helm/helm-mapkubeapis .
2. Führen Sie einen Testlauf 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 Lauf mit helm durch, um die Bereinigung durchzuführen.

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

Schritte

1. Wenn du"Trident wurde mit Helm installiert." Sie können verwenden `helm upgrade trident netapp-trident/trident-operator --version 100.2506.0` Um in einem Schritt ein Upgrade durchzuführen. Falls Sie das Helm-Repository nicht hinzugefügt haben oder es nicht für ein Upgrade verwenden können:
   a. Laden Sie die neueste Trident Version herunter von"der Abschnitt „Assets“ auf GitHub" .
   b. Verwenden Sie die `helm upgrade` Befehl, wo `trident-operator-25.06.0.tgz` Zeigt die Version an, auf die Sie aktualisieren möchten.

```
helm upgrade <name> trident-operator-25.06.0.tgz
```

   Wenn Sie während der Erstinstallation benutzerdefinierte Optionen festlegen (z. B. die Angabe privater, gespiegelter Registrierungen für Trident und CSI-Images), fügen Sie Folgendes hinzu: `helm upgrade` Befehl mit `--set` Um sicherzustellen, dass diese Optionen im Upgrade-Befehl enthalten sind, werden die Werte andernfalls auf die Standardwerte zurückgesetzt.

2. Laufen `helm list` um zu überprüfen, ob sowohl die Chart- als auch die App-Version aktualisiert wurden. Laufen `tridentctl logs` um alle Debug-Meldungen zu überprüfen.

Upgrade von einem `tridentctl` Installation für Trident Bediener

Sie können von einem auf die neueste Version des Trident -Operators aktualisieren. `tridentctl` Installation. Die bestehenden Backends und PVCs stehen automatisch zur Verfügung.

Bevor Sie zwischen den Installationsmethoden wechseln, überprüfen Sie Folgendes:"Wechsel zwischen Installationsmethoden" .

Schritte

1. Laden Sie die neueste Trident Version herunter.

```
# Download the release required [25.06.0]
mkdir 25.06.0
cd 25.06.0
wget
https://github.com/NetApp/trident/releases/download/v25.06.0/trident-
installer-25.06.0.tar.gz
tar -xf trident-installer-25.06.0.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. Den Cluster-Operator im selben Namespace bereitstellen.

```
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 Trident auf die vorgesehene Version aktualisiert wurde.

```
kubectl describe torc trident | grep Message -A 3

Message:                Trident installed
Namespace:              trident
Status:                 Installed
Version:                v25.06.0
```

Upgrade mit tridentctl

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

Informationen zu diesem Vorgang

Die Deinstallation und Neuinstallation von Trident fungiert als Upgrade. Bei der Deinstallation von Trident werden der Persistent Volume Claim (PVC) und das Persistent Volume (PV), die von der Trident -Bereitstellung verwendet wurden, nicht gelöscht. Bereits bereitgestellte PVs bleiben auch während der Offline-Phase von Trident verfügbar, und Trident wird nach der Wiederherstellung der Online-Verbindung Volumen für alle in der Zwischenzeit erstellten PVCs bereitstellen.

Bevor Sie beginnen

Rezension"Wählen Sie eine Upgrade-Methode aus" vor dem Upgrade mit `tridentctl`.

Schritte

1. Führen Sie den Deinstallationsbefehl aus in `tridentctl` Alle mit Trident verbundenen Ressourcen außer den CRDs und zugehörigen Objekten sollen entfernt werden.

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

2. Trident neu installieren. Siehe"Installieren Sie Trident mit tridentctl" .

> Unterbrechen Sie den Aktualisierungsprozess nicht. Stellen Sie sicher, dass das Installationsprogramm vollständig durchläuft.

Trident mit tridentctl verwalten

Der "Trident -Installationspaket" beinhaltet die `tridentctl` Kommandozeilenprogramm für den einfachen Zugriff auf Trident. Kubernetes-Benutzer mit ausreichenden Berechtigungen können es verwenden, um Trident zu installieren oder den Namespace zu verwalten, der den Trident Pod enthält.

Befehle und globale Flaggen

Du kannst laufen `tridentctl help` um eine Liste der verfügbaren Befehle zu erhalten für `tridentctl` oder fügen Sie die `--help` Verwenden Sie den Parameter `--flag` für einen beliebigen Befehl, um eine Liste der Optionen und Parameter für diesen spezifischen Befehl zu erhalten.

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

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

Befehle

`create`
Füge eine Ressource zu Trident hinzu.

`delete`
Entferne eine oder mehrere Ressourcen aus Trident.

`get`
Laden Sie eine oder mehrere Ressourcen von Trident herunter.

`help`
Hilfe zu jedem Befehl.

`images`
Erstellen Sie eine Tabelle mit den Containerbildern, die Trident benötigt.

`import`
Eine vorhandene Ressource in Trident importieren.

`install`
Installieren Sie Trident.

`logs`
Die Protokolle von Trident ausdrucken.

`send`
Senden Sie eine Ressource von Trident.

`uninstall`
Deinstallieren Sie Trident.

`update`
Eine Ressource in Trident bearbeiten.

`update backend state`
Die Backend-Operationen werden vorübergehend ausgesetzt.

`upgrade`
Eine Ressource in Trident aufwerten.

`version`
Drucken Sie die Version von Trident.

Weltflaggen

-d, --debug
Debug-Ausgabe.

-h, --help
Hilfe für `tridentctl` .

-k, --kubeconfig string
Geben Sie die `KUBECONFIG` Pfad zum Ausführen von Befehlen lokal oder von einem Kubernetes-Cluster zu einem anderen.

Alternativ können Sie die exportieren `KUBECONFIG` Variable, die auf einen bestimmten Kubernetes-Cluster verweist, und Problem `tridentctl` Befehle an diesen Cluster.

-n, --namespace string
Namespace der Trident -Bereitstellung.

-o, --output string
Ausgabeformat. Einer der folgenden Typen: json|yaml|name|wide|ps (Standard).

-s, --server string
Adresse/Port der Trident REST-Schnittstelle.

Die Trident REST-Schnittstelle kann so konfiguriert werden, dass sie nur unter 127.0.0.1 (für IPv4) oder [::1] (für IPv6) lauscht und Anfragen beantwortet.

Befehlsoptionen und Flags

erstellen

Verwenden Sie die `create` Befehl zum Hinzufügen einer Ressource zu Trident.

```
tridentctl create [option]
```

Optionen

`backend`: Füge Trident ein Backend hinzu.

löschen

Verwenden Sie die `delete` Befehl zum Entfernen einer oder mehrerer Ressourcen aus Trident.

```
tridentctl delete [option]
```

Optionen

`backend`: Löschen Sie ein 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` : Ein oder mehrere Speichervolumes aus Trident löschen.

erhalten

Verwenden Sie die `get` Befehl zum Abrufen einer oder mehrerer Ressourcen von Trident.

```
tridentctl get [option]
```

Optionen

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

Flaggen

`-h`, `--help` Hilfe für Bände.
`--parentOfSubordinate string` : Beschränken Sie die Abfrage auf das untergeordnete Quellvolume.
`--subordinateOf string` : Beschränke die Abfrage auf untergeordnete Elemente des Volumens.

Bilder

Verwenden `images` Flaggen zum Drucken einer Tabelle der Containerbilder, die Trident benötigt.

```
tridentctl images [flags]
```

Flaggen

`-h`, `--help` Hilfe zu Bildern.
`-v` , `--k8s-version string` : Semantische Version eines Kubernetes-Clusters.

Importvolumen

Verwenden Sie die `import volume` Befehl zum Importieren eines vorhandenen Volumes in Trident.

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

Aliase

`volume`, `v`

Flaggen

`-f`, `--filename string` Pfad zur YAML- oder JSON-PVC-Datei.
`-h` , `--help` Hilfe zur Lautstärkeregelung.
`--no-manage` : Nur PV/PVC erstellen. Gehen Sie nicht von einem automatisierten Lebenszyklusmanagement aus.

installieren

Verwenden Sie die `install` Flags zur Installation von Trident.

```
tridentctl install [flags]
```

Flaggen

`--autosupport-image string`: Das Container-Image für Autosupport Telemetry (Standardwert "netapp/trident autosupport:<aktuelle-Version>").
`--autosupport-proxy string` Die Adresse/der Port eines Proxys zum Senden von Autosupport-Telemetriedaten.
`--enable-node-prep` : Versuch, die erforderlichen Pakete auf den Knoten zu installieren.
`--generate-custom-yaml` : YAML-Dateien generieren, ohne etwas installieren zu müssen.
`-h` , `--help` Hilfe zur Installation.
`--http-request-timeout` : Überschreiben des HTTP-Anfrage-Timeouts für die REST-API des Trident -Controllers (Standardwert: 1 Minute 30 Sekunden).
`--image-registry string` Die Adresse/der Port einer internen Image-Registry.
`--k8s-timeout duration` : Das Timeout für alle Kubernetes-Operationen (Standardwert 3m0s).
`--kubelet-dir string` : Der Host-Speicherort des internen Zustands von kubelet (Standardwert "/var/lib/kubelet").
`--log-format string` : Das Trident -Protokollierungsformat (text, json) (Standardwert "text").
`--node-prep` : Ermöglicht es Trident , die Knoten des Kubernetes-Clusters für die Verwaltung von Volumes unter Verwendung des angegebenen Datenspeicherprotokolls vorzubereiten. **Momentan, `iscsi` ist der einzige unterstützte Wert. Ab OpenShift 4.19 ist die minimale Trident Version, die für diese Funktion unterstützt wird, 25.06.1.**
`--pv string` : Der Name des von Trident verwendeten Legacy-PV, um sicherzustellen, dass dieser nicht existiert (Standardwert "trident").
`--pvc string` : Der Name des von Trident verwendeten Legacy-PVCs, um sicherzustellen, dass dieser nicht existiert (Standardwert "trident").
`--silence-autosupport` : Automatisches Senden von Autosupport-Bundles an NetApp verhindern (Standardeinstellung: true).
`--silent` : Deaktiviere die meisten Ausgaben während der Installation.
`--trident-image string` : Das zu installierende Trident Image.
`--k8s-api-qps` : Die maximale Anzahl an Abfragen pro Sekunde (QPS) für Kubernetes-API-Anfragen (Standardwert 100; optional).
`--use-custom-yaml` : Verwenden Sie alle vorhandenen YAML-Dateien im Setup-Verzeichnis.
`--use-ipv6` : Trident soll IPv6 für die Kommunikation nutzen.

Protokolle

Verwenden `logs` Optionen zum Drucken der Protokolle von Trident.

```
tridentctl logs [flags]
```

Flaggen

`-a`, `--archive` : Es wird ein Supportarchiv mit allen Protokollen erstellt, sofern nicht anders angegeben.
`-h` , `--help` Hilfe zu Protokollen.
`-l` , `--log string` : Trident Protokoll anzeigen. Einer der folgenden Befehle: trident|auto|trident-operator|all (Standardwert: "auto").
`--node string` Der Name des Kubernetes-Knotens, von dem die Pod-Logs des Knotens erfasst werden sollen.
`-p` , `--previous` : Ruft die Protokolle der vorherigen Containerinstanz ab, falls vorhanden.
`--sidecars` : Rufe die Protokolle für die Sidecar-Container ab.

schicken

Verwenden Sie die `send` Befehl zum Senden einer Ressource von Trident.

```
tridentctl send [option]
```

Optionen

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

deinstallieren

Verwenden `uninstall` Optionen zum Deinstallieren von Trident.

```
tridentctl uninstall [flags]
```

Flaggen

`-h, --help`Hilfe zur Deinstallation.
`--silent` : Deaktiviert die meisten Ausgaben während der Deinstallation.

aktualisieren

Verwenden Sie die `update` Befehl zum Ändern einer Ressource in Trident.

```
tridentctl update [option]
```

Optionen

`backend`: Ein Backend in Trident aktualisieren.

Backend-Status aktualisieren

Verwenden Sie die `update backend state` Befehl zum Anhalten oder Fortsetzen von Backend-Operationen.

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

Zu berücksichtigende Punkte

- Wenn ein Backend mithilfe einer TridentBackendConfig (tbc) erstellt wurde, kann das Backend nicht mithilfe einer aktualisiert werden. `backend.json` Datei.
- Wenn die `userState` wurde in einer TBC-Datei festgelegt und kann nicht mit der `tridentctl update backend state <backend-name> --user-state suspended/normal` Befehl.
- Um die Fähigkeit wiederzuerlangen, die Einstellungen vorzunehmen `userState` über tridentctl, nachdem es über tbc festgelegt wurde, `userState` Das Feld muss aus der TBC entfernt werden. Dies kann mithilfe von … erfolgen. `kubectl edit tbc` Befehl. Nach dem `userState` Das Feld wurde entfernt, Sie können nun das verwenden `tridentctl update backend state` Befehl zum Ändern des `userState` eines Backends.
- Verwenden Sie die `tridentctl update backend state` um die `userState` . Sie können auch die `userState` mit `TridentBackendConfig` oder `backend.json` Datei; dies löst eine vollständige Neuinitialisierung des Backends aus und kann zeitaufwändig sein.

  Flaggen

  `-h, --help` Hilfe für den Backend-Status.
  `--user-state` : Aufstellen `suspended` Backend-Operationen anhalten. Auf einstellen `normal` Wiederaufnahme des Backend-Betriebs. Wenn eingestellt auf `suspended` :

- `AddVolume`Und `Import Volume` werden pausiert.
- `CloneVolume`, `ResizeVolume` , `PublishVolume` , `UnPublishVolume` , `CreateSnapshot` , `GetSnapshot` , `RestoreSnapshot` , `DeleteSnapshot` , `RemoveVolume` , `GetVolumeExternal` , `ReconcileNodeAccess` bleiben verfügbar.

Sie können den Backend-Status auch aktualisieren mit `userState` Feld in der Backend-Konfigurationsdatei `TridentBackendConfig` oder `backend.json` . Weitere Informationen finden Sie unter"Optionen zur Verwaltung von Backends" Und"Führen Sie die Backend-Verwaltung mit kubectl durch." .

Beispiel:

JSON

Führen Sie diese Schritte aus, um die `userState` mithilfe der `backend.json` Datei:

1. Bearbeiten Sie die `backend.json` Datei, die die `userState` Feld, dessen Wert auf „suspendiert“ gesetzt ist.
2. Aktualisieren Sie das Backend mithilfe der `tridentctl update backend` Befehl und Pfad zur aktualisierten `backend.json` Datei.

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

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

YAML

Sie können die TBC-Datei nach ihrer Anwendung mit der folgenden Methode bearbeiten: `kubectl edit <tbc-name> -n <namespace>` Befehl. Das folgende Beispiel aktualisiert den Backend-Status mithilfe von … auf „Anhaltend“. `userState: suspended` Option:

```
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 `version` Flaggen zum Drucken der Version von `tridentctl` und den laufenden Trident -Service.

```
tridentctl version [flags]
```

Flaggen

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

Plugin-Unterstützung

Tridentctl unterstützt ähnliche Plugins wie kubectl. Tridentctl erkennt ein Plugin, wenn der Name der Plugin-Binärdatei dem Schema "tridentctl-<Plugin>" entspricht und sich die Binärdatei in einem Ordner befindet, der in der Umgebungsvariablen PATH aufgeführt ist. Alle erkannten Plugins sind im Plugin-Abschnitt der tridentctl-Hilfe aufgelistet. Optional können Sie die Suche auch einschränken, indem Sie einen Plugin-Ordner in der Umgebungsvariablen TRIDENTCTL_PLUGIN_PATH angeben (Beispiel: `TRIDENTCTL_PLUGIN_PATH=~/tridentctl-plugins/`). Wenn die Variable verwendet wird, sucht tridenctl nur im angegebenen Ordner.

Monitor Trident

Trident stellt eine Reihe von Prometheus-Metrikenendpunkten bereit, mit denen Sie die Leistung von Trident überwachen können.

Überblick

Die von Trident bereitgestellten Metriken ermöglichen Ihnen Folgendes:

- Behalten Sie den Zustand und die Konfiguration von Trident im Auge. Sie können überprüfen, wie erfolgreich die Abläufe sind und ob die Kommunikation mit den Backends wie erwartet funktioniert.
- Untersuchen Sie die Backend-Nutzungsinformationen und verstehen Sie, wie viele Volumes auf einem Backend bereitgestellt werden und wie viel Speicherplatz belegt ist usw.
- Pflegen Sie eine Übersicht über die Anzahl der auf den verfügbaren Backends bereitgestellten Volumes.
- Streckenleistung. Sie können sich ansehen, wie lange Trident für die Kommunikation mit den Backends und die Durchführung von Operationen benötigt.

Standardmäßig werden die Metriken von Trident auf dem Zielport angezeigt. `8001` am `/metrics` Endpunkt. Diese Metriken sind **standardmäßig aktiviert**, wenn Trident installiert ist.

Was du brauchst

- Ein Kubernetes-Cluster mit installiertem Trident .
- Eine Prometheus-Instanz. Dies kann ein "containerisierte Prometheus-Bereitstellung" oder Sie können Prometheus als … ausführen "native Anwendung" .

Schritt 1: Definieren Sie ein Prometheus-Ziel

Sie sollten ein Prometheus-Ziel definieren, um die Metriken zu erfassen und Informationen über die von Trident verwalteten Backends, die von ihm erstellten Volumes usw. zu erhalten. Das "Blog" erklärt, wie man

Prometheus und Grafana mit Trident verwenden kann, um Metriken abzurufen. Der Blog erklärt, wie Sie Prometheus als Operator in Ihrem Kubernetes-Cluster ausführen und einen ServiceMonitor erstellen, um Trident -Metriken zu erhalten.

Schritt 2: Erstellen Sie einen Prometheus ServiceMonitor

Um die Trident -Metriken zu nutzen, sollten Sie einen Prometheus ServiceMonitor erstellen, der die folgenden Metriken überwacht: `trident-csi` Service und hört auf dem `metrics` Hafen. Ein Beispiel für einen ServiceMonitor sieht folgendermaßen 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 Metriken ab, die vom ServiceMonitor zurückgegeben werden. `trident-csi` Service und sucht insbesondere nach dem `metrics` Endpunkt des Dienstes. Daher ist Prometheus nun so konfiguriert, dass es die Metriken von Trident versteht.

Zusätzlich zu den direkt von Trident verfügbaren Metriken stellt kubelet viele weitere bereit. `kubelet_volume_*` Metriken über den eigenen Metrik-Endpunkt. Kubelet kann Informationen über die angehängten Volumes sowie über Pods und andere interne Operationen bereitstellen, die es durchführt. Siehe "hier," .

Schritt 3: Trident -Metriken mit PromQL abfragen

PromQL eignet sich gut zum Erstellen von Ausdrücken, die Zeitreihen- oder Tabellendaten zurückgeben.

Hier sind einige PromQL-Abfragen, die Sie verwenden können:

Holen Sie sich Gesundheitsinformationen von 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 den Statuscode**

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

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

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

Informationen zur Trident -Nutzung abrufen

- **Durchschnittliche Volumengröße**

```
trident_volume_allocated_bytes/trident_volume_count
```

- **Von jedem Backend bereitgestellter Gesamtspeicherplatz**

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

Individuelle Volumennutzung abrufen

Diese Funktion ist nur aktiviert, wenn auch Kubelet-Metriken erfasst werden.

- **Prozentualer Anteil des genutzten Platzes pro Volumen**

```
kubelet_volume_stats_used_bytes / kubelet_volume_stats_capacity_bytes *
100
```

Erfahren Sie mehr über die Telemetrie von Trident AutoSupport.

Standardmäßig sendet Trident täglich Prometheus-Metriken und grundlegende Backend-Informationen an NetApp .

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

- Trident kann Containerprotokolle auch bei Bedarf an den NetApp -Support senden über `tridentctl send autosupport` . Sie müssen Trident auffordern, seine Protokolle hochzuladen. Bevor Sie Protokolle einreichen, sollten Sie die Bedingungen von NetApp akzeptieren.https://www.netapp.com/company/legal/privacy-policy/["Datenschutzrichtlinie"^] .
- Sofern nicht anders angegeben, ruft Trident die Protokolle der letzten 24 Stunden ab.
- Sie können den Aufbewahrungszeitraum für Protokolle mit der `--since` Flagge. Zum Beispiel: `tridentctl send autosupport --since=1h` . Diese Informationen werden gesammelt und über ein `trident-autosupport` Container, der zusammen mit Trident installiert wird. Das Container-Image erhalten Sie unter "Trident AutoSupport" .
- Trident AutoSupport sammelt oder übermittelt keine personenbezogenen Daten (PII) oder persönlichen Informationen. Es wird mit einem "EULA" Dies gilt nicht für das Trident -Containerimage selbst. Sie können mehr über NetApps Engagement für Datensicherheit und Vertrauen erfahren. "hier," .

Ein Beispiel für eine von Trident gesendete Nutzlast sieht folgendermaßen aus:

```
---
items:
  - backendUUID: ff3852e1-18a5-4df4-b2d3-f59f829627ed
    protocol: file
    config:
      version: 1
      storageDriverName: ontap-nas
      debug: false
      debugTraceFlags: null
      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 Registry zum Speichern von Container-Images verwenden, können Sie die folgende verwenden: `--image-registry` Flagge.
- Sie können Proxy-URLs auch konfigurieren, indem Sie die Installations-YAML-Dateien generieren. Dies kann durch folgende Verwendung erfolgen: `tridentctl install --generate-custom-yaml` um die YAML-Dateien zu erstellen und die `--proxy-url` Argument für die `trident-autosupport` Container in `trident-deployment.yaml` .

Trident -Metriken deaktivieren

Um die Meldung von Metriken **zu deaktivieren**, sollten Sie benutzerdefinierte YAML-Dateien generieren (mithilfe der `--generate-custom-yaml` Flagge) und bearbeiten Sie sie, um die Flagge zu entfernen `--metrics` Flagge davon abgehalten, für die `trident-main` Container.

Trident deinstallieren

Sie sollten zum Deinstallieren von Trident dieselbe Methode verwenden, die Sie zum Installieren von Trident verwendet haben.

Informationen zu diesem Vorgang

- Falls Sie nach einem Upgrade Fehler beheben müssen, Abhängigkeitsprobleme auftreten oder das Upgrade fehlgeschlagen oder unvollständig war, sollten Sie Trident deinstallieren und die vorherige Version gemäß der entsprechenden Anleitung neu installieren."Version" . Dies ist die einzige empfohlene Methode, um auf eine frühere Version zurückzukehren.
- Um ein einfaches Upgrade und eine Neuinstallation zu ermöglichen, werden bei der Deinstallation von Trident weder die von Trident erstellten CRDs noch zugehörige Objekte entfernt. Falls Sie Trident und alle zugehörigen Daten vollständig entfernen müssen, lesen Sie bitte weiter unter"Trident und CRDs vollständig entfernen" .

Bevor Sie beginnen

Wenn Sie Kubernetes-Cluster außer Betrieb nehmen, müssen Sie vor der Deinstallation alle Anwendungen löschen, die von Trident erstellte Volumes verwenden. Dadurch wird sichergestellt, dass PVCs auf Kubernetes-Knoten deaktiviert werden, bevor sie gelöscht werden.

Ermitteln Sie die ursprüngliche Installationsmethode

Sie sollten zum Deinstallieren von Trident dieselbe Methode verwenden, die Sie zum Installieren verwendet haben. Prüfen Sie vor der Deinstallation, mit welcher Version Sie Trident ursprünglich installiert haben.

1. Verwenden `kubectl get pods -n trident` um die Schoten zu untersuchen.
   - Falls keine Bedienerkabine vorhanden ist, wurde Trident installiert mit `tridentctl` .
   - Falls ein Operator-Pod vorhanden ist, wurde Trident entweder manuell oder mithilfe von Helm mit dem Trident -Operator installiert.
2. Falls ein Bedienerpod vorhanden ist, verwenden Sie `kubectl describe tproc trident` um festzustellen, ob Trident mit Helm installiert wurde.
   - Wenn ein Helm-Label vorhanden ist, wurde Trident mit Helm installiert.
   - Falls kein Helm-Label vorhanden ist, wurde Trident manuell mit dem Trident -Operator installiert.

Deinstallieren einer Trident -Operatorinstallation

Sie können eine Trident-Operator-Installation manuell oder mithilfe von Helm deinstallieren.

Manuelle Installation deinstallieren

Wenn Sie Trident mithilfe des Operators installiert haben, können Sie es auf eine der folgenden Arten deinstallieren:

1. **Bearbeiten `TridentOrchestrator` CR und das Deinstallationsflag setzen**:

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

Wenn die `uninstall` Das Flag ist gesetzt auf `true` Der Trident Operator deinstalliert Trident, entfernt aber nicht den TridentOrchestrator selbst. Sie sollten den TridentOrchestrator bereinigen und einen neuen erstellen, wenn Sie Trident erneut installieren möchten.

2. **Löschen `TridentOrchestrator`** : Durch Entfernen der `TridentOrchestrator` Wenn Sie eine CR verwenden, die zur Bereitstellung von Trident verwendet wurde, weisen Sie den Operator an, Trident zu deinstallieren. Der Bediener verarbeitet die Entfernung von `TridentOrchestrator` und entfernt anschließend die Trident Bereitstellung und das Daemonset und löscht dabei die Trident -Pods, die im Rahmen der Installation erstellt wurden.

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

Helm-Installation deinstallieren

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

```
#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 ein `tridentctl` Installation

Verwenden Sie die `uninstall` Befehl in `tridentctl` Alle mit Trident verbundenen Ressourcen außer den CRDs und zugehörigen Objekten entfernen:

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

Trident für Docker

Voraussetzungen für die Bereitstellung

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

Überprüfen Sie die Anforderungen

- Vergewissern Sie sich, dass Ihre Bereitstellung alle Anforderungen erfüllt."Anforderungen" .
- Vergewissern Sie sich, dass Sie eine unterstützte Version von Docker installiert haben. Wenn Ihre Docker-Version veraltet ist, "installieren oder aktualisieren" .

```
docker --version
```

- Vergewissern Sie sich, dass die Protokollvoraussetzungen auf Ihrem Host installiert und konfiguriert sind.

NFS -Tools

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

RHEL 8+

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

Ubuntu

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

Starten Sie Ihre Worker-Knoten nach der Installation der NFS-Tools neu, um Fehler beim Anhängen von Volumes an Container zu vermeiden.

iSCSI-Tools

Installieren Sie die iSCSI-Tools mithilfe der Befehle für Ihr Betriebssystem.

RHEL 8+

1. Installieren Sie die folgenden Systempakete:

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

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

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

3. Scanvorgang auf manuelles Scannen 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
```

> Sicherstellen `etc/multipath.conf` enthält `find_multipaths no` unter `defaults` .

5. Stellen Sie sicher, dass `iscsid` Und `multipathd` laufen:

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

6. Aktivieren und starten `iscsi` :

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

Ubuntu

1. Installieren Sie die folgenden Systempakete:

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

2. Prüfen Sie, ob die Open-iSCSI-Version 2.0.874-5ubuntu2.10 oder höher (für Bionic) bzw. 2.0.874-7.1ubuntu6.1 oder höher (für Focal) ist:

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

3. Scanvorgang auf manuelles Scannen 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
```

> Sicherstellen `etc/multipath.conf` enthält `find_multipaths no` unter `defaults` .

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

```
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.

- NVMe erfordert RHEL 9 oder höher.
- Falls die Kernelversion Ihres Kubernetes-Knotens zu alt ist oder das NVMe-Paket für Ihre Kernelversion nicht verfügbar ist, müssen Sie möglicherweise die Kernelversion Ihres Knotens auf eine Version mit dem NVMe-Paket 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
```

FC -Tools

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

- Bei Verwendung von Worker-Knoten, auf denen RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) mit FC-PVs ausgeführt wird, geben Sie Folgendes an: `discard` mountOption in der StorageClass zur Durchführung der Inline-Speicherplatzfreigabe. Siehe "Red Hat-Dokumentation" .

RHEL 8+

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
```

> Sicherstellen `etc/multipath.conf` enthält `find_multipaths no` unter `defaults` .

3. Stellen Sie sicher, dass `multipathd` läuft:

```
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
```

> Sicherstellen `etc/multipath.conf` enthält `find_multipaths no` unter `defaults` .

3. Stellen Sie sicher, dass `multipath-tools` ist aktiviert und läuft:

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

Trident einsetzen

Trident for Docker bietet eine direkte Integration mit dem Docker-Ökosystem für NetApp Speicherplattformen. Es unterstützt die Bereitstellung und Verwaltung von Speicherressourcen von der Speicherplattform bis hin zu Docker-Hosts und bietet ein Framework für die zukünftige Integration weiterer Plattformen.

Mehrere Instanzen von Trident können gleichzeitig auf demselben Host ausgeführt werden. Dies ermöglicht die gleichzeitige Anbindung an mehrere Speichersysteme und Speichertypen sowie die Möglichkeit, den für die Docker-Volumes verwendeten Speicher individuell anzupassen.

Was du brauchst

Siehe die"Voraussetzungen für die Bereitstellung" . Sobald Sie sichergestellt haben, dass die Voraussetzungen erfüllt sind, können Sie Trident einsetzen.

Docker-verwaltete Plugin-Methode (Version 1.13/17.03 und höher)

Bevor Sie beginnen

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

1. Alle laufenden Instanzen stoppen:

```
pkill /usr/local/bin/netappdvp
pkill /usr/local/bin/trident
```

2. Docker neu starten.

```
systemctl restart docker
```

3. Stellen Sie sicher, dass Sie Docker Engine 17.03 (neu 1.13) oder höher installiert haben.

```
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 einen beliebigen Namen verwenden, indem Sie ihn angeben. `config` Option mit dem Dateinamen. Die Konfigurationsdatei muss sich im folgenden Verzeichnis befinden: `/etc/netappdvp` Verzeichnis auf dem Hostsystem.
   - `log-level`: Protokollierungsstufe festlegen(`debug` , `info` , `warn` , `error` , `fatal`). Standardmäßig ist `info` .

- `debug`: Geben Sie an, ob die Debug-Protokollierung aktiviert ist. Standardwert ist „false“. Überschreibt den Protokollierungsgrad, falls wahr.

  i. Erstellen Sie einen Speicherort für die Konfigurationsdatei:

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

  ii. Erstellen Sie die Konfigurationsdatei:

  ```
  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 mithilfe des verwalteten Plugin-Systems. Ersetzen `<version>` mit der von Ihnen verwendeten 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 Nutzung von Trident, um Speicherplatz des konfigurierten Systems zu belegen.

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

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

   b. Erstellen Sie ein Standardvolume beim Start des Containers:

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

   c. Entfernen Sie die Lautstärke "firstVolume":

```
docker volume rm firstVolume
```

Traditionelle Methode (Version 1.12 oder früher)

Bevor Sie beginnen

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

   ```
   docker --version
   ```

   Sollte Ihre Version veraltet sein, aktualisieren Sie Ihre Installation.

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

   Oder, "Befolgen Sie die Anweisungen für Ihre Verteilung." .

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

Schritte

1. Installieren und konfigurieren Sie das NetApp Docker Volume Plugin:
   a. Laden Sie die Anwendung herunter und entpacken Sie sie:

      ```
      wget
      https://github.com/NetApp/trident/releases/download/v25.06.0/trident-
      installer-25.06.0.tar.gz
      tar zxf trident-installer-25.06.0.tar.gz
      ```

   b. An eine Position im Papierkorbpfad verschieben:

      ```
      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. Erstellen Sie einen Speicherort für die Konfigurationsdatei:

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

   d. Erstellen Sie die Konfigurationsdatei:

      ```
      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 anders angegeben, lautet der Standardname für den Volume-Treiber "netapp".

Nach dem Start des Daemons können Sie Volumes mithilfe der Docker CLI-Schnittstelle erstellen und verwalten.

3. Ein Volume erstellen:

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

4. Beim Starten eines Containers ein Docker-Volume bereitstellen:

```
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
```

Trident beim Systemstart starten

Eine Beispiel-Unit-Datei für systemd-basierte Systeme finden Sie unter

`contrib/trident.service.example` im Git-Repository. Um die Datei mit RHEL zu verwenden, gehen Sie wie folgt vor:

1. Kopieren Sie die Datei an den richtigen Speicherort.

   Sie sollten eindeutige Namen für die Unit-Dateien verwenden, wenn Sie mehr als eine Instanz ausführen.

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

2. Bearbeiten Sie die Datei, ändern Sie die Beschreibung (Zeile 2) so, dass sie dem Treibernamen entspricht, und den Pfad zur Konfigurationsdatei (Zeile 9) so, dass er Ihre Umgebung widerspiegelt.
3. Systemd muss neu geladen werden, damit die Änderungen übernommen werden:

   ```
   systemctl daemon-reload
   ```

4. Aktivieren Sie den Dienst.

   Dieser Name variiert je nachdem, wie Sie die Datei im `/usr/lib/systemd/system` Verzeichnis.

   ```
   systemctl enable trident
   ```

5. Starten Sie den Dienst.

   ```
   systemctl start trident
   ```

6. Status ansehen.

   ```
   systemctl status trident
   ```

Jedes Mal, wenn Sie die Unit-Datei ändern, führen Sie den Befehl aus. `systemctl daemon-reload` Befehl, damit es die Änderungen erkennt.

Aktualisieren oder deinstallieren Sie Trident.

Sie können Trident für Docker bedenkenlos aktualisieren, ohne dass dies Auswirkungen auf bereits verwendete Volumes hat. Während des Aktualisierungsprozesses wird es eine kurze Zeitspanne geben, in der `docker volume` An das Plugin gerichtete Befehle werden nicht erfolgreich sein, und Anwendungen können keine Volumes einbinden, bis das Plugin wieder ausgeführt wird. Unter den meisten Umständen ist dies eine Frage von Sekunden.

Upgrade

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

Schritte

1. Liste der vorhandenen Bände:

```
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. Aktualisieren Sie das Plugin:

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

> Mit der Version 18.01 von Trident wird nDVP ersetzt. Sie sollten direkt von der `netapp/ndvp-plugin` Bild zum `netapp/trident-plugin` Bild.

4. Aktivieren Sie das Plugin:

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

5. Überprüfen Sie, ob das Plugin aktiviert ist:

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

6. Überprüfen Sie, ob die Volumes sichtbar sind:

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

Beim Upgrade von einer älteren Version von Trident (vor 20.10) auf Trident 20.10 oder neuer kann es zu einem Fehler kommen. Weitere Informationen finden Sie unter"Bekannte Probleme" . Sollte dieser Fehler auftreten, deaktivieren Sie zunächst das Plugin, entfernen Sie es anschließend und installieren Sie dann die benötigte 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 vom Plugin erstellten Volumes.
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 Volumina

Sie können Volumes mithilfe des Standards einfach erstellen, klonen und entfernen. `docker volume` Befehle, bei denen der Name des Trident -Treibers gegebenenfalls angegeben wird.

Erstellen eines Volumes

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

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

- Erstellen Sie ein Volume mit einer bestimmten Trident -Instanz:

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

> Wenn Sie keine Angaben machen"Optionen" Es werden die Standardeinstellungen des Treibers verwendet.

- Überschreiben Sie die Standard-Volumegröße. Im folgenden Beispiel wird gezeigt, wie Sie ein 20-GiB-Volume mit einem Treiber erstellen:

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

> Die Größe des Speichervolumens wird als Zeichenkette angegeben, die einen ganzzahligen Wert mit optionalen Einheiten enthält (Beispiel: 10G, 20GB, 3TiB). Werden keine Einheiten angegeben, ist der Standardwert G. Größenangaben können entweder als Zweierpotenzen (B, KiB, MiB, GiB, TiB) oder als Zehnerpotenzen (B, KB, MB, GB, TB) erfolgen. Kurzschreibweisen verwenden Zweierpotenzen (G = GiB, T = TiB, …).

Ein Volume entfernen

- Entfernen Sie das Volume genau wie jedes andere Docker-Volume:

```
docker volume rm firstVolume
```

> Bei der Verwendung des `solidfire-san` Der Treiber löscht und bereinigt im obigen Beispiel das Volume.

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

Klonen eines Volumes

Bei der Verwendung des `ontap-nas` , `ontap-san` , `solidfire-san` , Und `gcp-cvs storage drivers` Trident kann Volumes klonen. Bei der Verwendung des `ontap-nas-flexgroup` oder `ontap-nas-economy` Treiber, Klonen wird nicht unterstützt. Wenn aus einem bestehenden Volume ein neues Volume erstellt wird, wird ein neuer Snapshot erstellt.

- Untersuchen Sie das Volume, um Snapshots aufzulisten:

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

- Ein neues Volume aus einem vorhandenen Volume erstellen. Dies führt zur Erstellung eines neuen Snapshots:

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

- Erstellen Sie ein neues Volume aus einem vorhandenen Snapshot 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 mit Trident auf extern erstellte Blockgeräte (oder deren Klone) zugreifen, **nur** wenn diese keine Partitionen besitzen und ihr Dateisystem von Trident unterstützt wird (z. B. ein …). `ext4` -formatiert `/dev/sdc1` wird nicht über Trident zugänglich sein).

Fahrerspezifische Lautstärkeoptionen

Jeder Speichertreiber verfügt über unterschiedliche Optionen, die Sie bei der Volume-Erstellung festlegen können, um das Ergebnis anzupassen. Nachfolgend finden Sie die Optionen, die für Ihr konfiguriertes Speichersystem gelten.

Die Verwendung dieser Optionen während der Volume-Erstellung ist einfach. Geben Sie die Option und den Wert mithilfe der `-o` Bediener während der CLI-Operation. Diese Werte überschreiben alle entsprechenden Werte aus der JSON-Konfigurationsdatei.

ONTAP Volume-Optionen

Die Optionen zum Erstellen von Volumes für NFS, iSCSI und FC umfassen Folgendes:

Option	Beschreibung
`size`	Die Größe des Volumes beträgt standardmäßig 1 GiB.
`spaceReserve`	Die Volume-Anpassung erfolgt wahlweise dünn oder dick, standardmäßig wird dünn eingestellt. Gültige Werte sind `none` (Thin Provisioning) und `volume` (Thick Provisioned).
`snapshotPolicy`	Dadurch wird die Snapshot-Richtlinie auf den gewünschten Wert eingestellt. Standardmäßig ist `none` Das bedeutet, dass für das Volume keine Snapshots automatisch erstellt werden. Sofern nicht von Ihrem Speicheradministrator geändert, existiert auf allen ONTAP -Systemen eine Richtlinie mit der Bezeichnung „default“, die sechs stündliche, zwei tägliche und zwei wöchentliche Snapshots erstellt und speichert. Die in einem Snapshot gespeicherten Daten können durch Navigieren zu folgendem Pfad wiederhergestellt werden: `.snapshot` Verzeichnis in einem beliebigen Verzeichnis auf dem Datenträger.

Option	Beschreibung
`snapshotReserve`	Dadurch wird die Snapshot-Reserve auf den gewünschten Prozentsatz eingestellt. Standardmäßig wird kein Wert angegeben. Das bedeutet, dass ONTAP die snapshotReserve (normalerweise 5 %) auswählt, wenn Sie eine snapshotPolicy ausgewählt haben, oder 0 %, wenn keine snapshotPolicy ausgewählt ist. Sie können den Standardwert für snapshotReserve in der Konfigurationsdatei für alle ONTAP -Backends festlegen und ihn als Option zur Volume-Erstellung für alle ONTAP -Backends außer ontap-nas-economy verwenden.
`splitOnClone`	Beim Klonen eines Volumes führt dies dazu, dass ONTAP den Klon sofort vom übergeordneten Volume trennt. Standardmäßig ist `false` . Bei manchen Anwendungsfällen für das Klonen von Volumes ist es am besten, den Klon direkt nach der Erstellung vom übergeordneten Volume zu trennen, da es wahrscheinlich keine Möglichkeit zur Speicheroptimierung gibt. Das Klonen einer leeren Datenbank kann beispielsweise eine große Zeitersparnis, aber nur eine geringe Speicherersparnis bieten, daher ist es am besten, den Klon sofort wieder aufzuteilen.
`encryption`	Aktivieren Sie die NetApp Volumeverschlüsselung (NVE) auf dem neuen Volume; Standardwert ist `false` . Um diese Option nutzen zu können, muss NVE auf dem Cluster lizenziert und aktiviert sein. Wenn NAE im Backend aktiviert ist, wird jedes in Trident bereitgestellte Volume NAE-fähig sein. Weitere Informationen finden Sie unter:"Wie Trident mit NVE und NAE zusammenarbeitet" .
`tieringPolicy`	Legt die für das Volumen anzuwendende Staffelungsrichtlinie fest. Hiermit wird entschieden, ob Daten in die Cloud-Ebene verschoben werden, wenn sie inaktiv (kalt) werden.

Die folgenden zusätzlichen Optionen gelten **nur** für NFS:

Option	Beschreibung
`unixPermissions`	Hiermit wird der Berechtigungssatz für das Volume selbst gesteuert. Standardmäßig sind die Berechtigungen auf folgende Einstellung gesetzt: `---rwxr-xr-x` oder in numerischer Notation 0755, und `root` wird der Eigentümer sein. Sowohl das Text- als auch das Zahlenformat sind geeignet.
`snapshotDir`	Dies einstellen auf `true` wird machen `.snapshot` Verzeichnis, das für Clients sichtbar ist, die auf das Volume zugreifen. Der Standardwert ist `false` , was bedeutet, dass die Sichtbarkeit der `.snapshot` Das Verzeichnis ist standardmäßig deaktiviert. Manche Images, beispielsweise das offizielle MySQL-Image, funktionieren nicht wie erwartet, wenn `.snapshot` Das Verzeichnis ist sichtbar.
`exportPolicy`	Legt die für das Volume zu verwendende Exportrichtlinie fest. Standardmäßig ist `default` .
`securityStyle`	Legt den Sicherheitsstil fest, der für den Zugriff auf das Volume verwendet werden soll. Standardmäßig ist `unix` . Gültige Werte sind `unix` Und `mixed` .

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

Option	Beschreibung
`fileSystemType`	Legt das Dateisystem fest, das zum Formatieren von iSCSI-Volumes verwendet wird. Standardmäßig ist `ext4` . Gültige Werte sind `ext3` , `ext4` , Und `xfs` .
`spaceAllocation`	Dies einstellen auf `false` wird die Speicherplatzzuweisungsfunktion der LUN deaktivieren. Der Standardwert ist `true` Das bedeutet, dass ONTAP den Host benachrichtigt, wenn der Speicherplatz des Volumes erschöpft ist und die LUN im Volume keine Schreibvorgänge mehr annehmen kann. Diese Option ermöglicht es ONTAP außerdem, Speicherplatz automatisch freizugeben, wenn Ihr Host Daten löscht.

Beispiele

Siehe die folgenden Beispiele:

- Erstellen Sie ein 10-GiB-Volume:

```
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 Mindestgröße des Volumes beträgt 20 MiB.

Wenn die Snapshot-Reserve nicht angegeben ist und die Snapshot-Richtlinie `none` Trident verwendet eine Snapshot-Reserve von 0%.

- Erstellen Sie ein Volume ohne Snapshot-Richtlinie und ohne Snapshot-Reservierung:

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

- Erstellen Sie ein Volume ohne Snapshot-Richtlinie und mit einer benutzerdefinierten 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 benutzerdefinierten 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 des ONTAP (normalerweise 5 %):

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

Element Software Lautstärkeoptionen

Die Softwareoptionen von Element legen die Größe und die mit dem Datenvolumen verbundenen QoS-Richtlinien (Quality of Service) offen. Beim Erstellen des Volumes wird die zugehörige QoS-Richtlinie mithilfe der `-o type=service_level` Nomenklatur.

Der erste Schritt zur Definition eines QoS-Dienstlevels mit dem Element-Treiber besteht darin, mindestens einen Typ zu erstellen und die minimalen, maximalen und Burst-IOPS anzugeben, die mit einem Namen in der Konfigurationsdatei verknüpft sind.

Weitere Optionen zur Volumenerstellung mit der Element-Software umfassen Folgendes:

Option	Beschreibung
`size`	Die Größe des Datenträgers beträgt standardmäßig 1 GiB oder Konfigurationseintrag … "defaults": {"size": "5G"}.
`blocksize`	Verwenden Sie entweder 512 oder 4096, standardmäßig wird 512 oder der Konfigurationseintrag DefaultBlockSize verwendet.

Beispiel

Siehe die folgende Beispielkonfigurationsdatei mit 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, Silber und Gold. Diese Namen sind willkürlich.

- Erstellen Sie ein 10 GiB Gold-Volume:

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

- Erstellen Sie ein 100 GiB Bronze-Volume:

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

Protokolle sammeln

Sie können Protokolle sammeln, um bei der Fehlersuche zu helfen. Die Methode, mit der Sie die Protokolle sammeln, hängt davon ab, wie Sie das Docker-Plugin ausführen.

Sammeln Sie Protokolle zur Fehlerbehebung.

Schritte

1. Wenn Sie Trident mit der empfohlenen Methode für verwaltete Plugins ausführen (d. h. mit `docker plugin` Befehle), betrachten Sie sie wie folgt:

```
docker plugin ls
```

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

   Mit dem Standard-Protokollierungsgrad sollten Sie die meisten Probleme diagnostizieren können. Sollte das nicht ausreichen, können Sie die 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 die 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 die Protokolle im Verzeichnis des Hosts verfügbar. `/var/log/netappdvp` Verzeichnis. Um die Debug-Protokollierung zu aktivieren, geben Sie Folgendes an: `-debug` wenn Sie das Plugin ausführen.

Allgemeine Tipps zur Fehlerbehebung

- Das häufigste Problem, auf das neue Benutzer stoßen, ist eine Fehlkonfiguration, die die Initialisierung des Plugins verhindert. Wenn dies passiert, wird Ihnen beim Versuch, das Plugin zu installieren oder zu aktivieren, wahrscheinlich eine Meldung wie die folgende angezeigt:

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

  Dies bedeutet, dass das Plugin nicht gestartet werden konnte. Zum Glück verfügt das Plugin über eine umfassende Protokollierungsfunktion, die Ihnen bei der Diagnose der meisten auftretenden Probleme helfen sollte.

- Falls es Probleme bei der Montage einer Photovoltaikanlage an einem Container gibt, stellen Sie sicher, dass `rpcbind` ist installiert und läuft. Verwenden Sie den erforderlichen Paketmanager für das Host-Betriebssystem und prüfen Sie, ob `rpcbind` läuft. Sie können den Status des rpcbind-Dienstes überprüfen, indem Sie folgenden Befehl ausführen: `systemctl status rpcbind` oder etwas Gleichwertiges.

Mehrere Trident Instanzen verwalten

Mehrere Instanzen von Trident sind erforderlich, wenn Sie mehrere Speicherkonfigurationen gleichzeitig nutzen möchten. Der Schlüssel zu mehreren Instanzen besteht darin, ihnen mithilfe des `--alias` Option mit dem containerisierten Plugin, oder `--volume-driver` Option beim Instanziieren von Trident auf dem Host.

Schritte für das Docker-verwaltete Plugin (Version 1.13/17.03 oder höher)

1. Starten Sie die erste Instanz unter Angabe eines Alias und einer Konfigurationsdatei.

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

2. Starten Sie die zweite Instanz und geben Sie dabei einen anderen Alias und eine andere Konfigurationsdatei an.

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

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

   Zum Beispiel für das Goldvolumen:

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

   Beispiel: Silbervolumen

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

Schritte für die traditionelle Version (Version 1.12 oder älter)

1. Starten Sie das Plugin 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 Plugin mit einer iSCSI-Konfiguration unter Verwendung einer benutzerdefinierten Treiber-ID:

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

3. Docker-Volumes für jede Treiberinstanz bereitstellen:

   Zum Beispiel für NFS:

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

   Zum Beispiel für iSCSI:

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

Speicherkonfigurationsoptionen

Sehen Sie sich die für Ihre Trident -Konfigurationen verfügbaren Konfigurationsoptionen an.

Globale Konfigurationsoptionen

Diese Konfigurationsoptionen gelten für alle Trident -Konfigurationen, unabhängig von der verwendeten Speicherplattform.

Option	Beschreibung	Beispiel
`version`	Versionsnummer der Konfigurationsdatei	`1`

Option	Beschreibung	Beispiel
`storageDriverName`	Name des Speichertreibers	`ontap-nas`, `ontap-san` , `ontap-nas-economy` , `ontap-nas-flexgroup` , `solidfire-san`
`storagePrefix`	Optionales Präfix für Datenträgernamen. Standard: `netappdvp_` .	`staging_`
`limitVolumeSize`	Optionale Beschränkung der Volumengrößen. Standardwert: "" (nicht erzwungen)	`10g`

Nicht verwenden `storagePrefix` (einschließlich des Standard-Backends) für Element-Backends. Standardmäßig ist die `solidfire-san` Der Treiber wird diese Einstellung ignorieren und kein Präfix verwenden. NetApp empfiehlt, entweder eine spezifische Mandanten-ID für das Docker-Volume-Mapping zu verwenden oder die Attributdaten zu nutzen, die mit der Docker-Version, Treiberinformationen und dem Rohnamen von Docker gefüllt sind, falls eine Namensmanipulation vorgenommen wurde.

Es stehen Standardoptionen zur Verfügung, sodass Sie diese nicht bei jedem neu erstellten Volume angeben müssen. Der `size` Diese Option ist für alle Controllertypen verfügbar. Ein Beispiel für die Festlegung der Standard-Volumegröße finden Sie im Abschnitt ONTAP -Konfiguration.

Option	Beschreibung	Beispiel
`size`	Optionale Standardgröße für neue Volumes. Standard: `1G`	`10G`

ONTAP-Konfiguration

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

Option	Beschreibung	Beispiel
`managementLIF`	IP-Adresse des ONTAP -Management-LIF. Sie können einen vollqualifizierten Domänennamen (FQDN) angeben.	`10.0.0.1`

Option	Beschreibung	Beispiel
`dataLIF`	IP-Adresse des Protokolls LIF. • ONTAP NAS-Treiber *: NetApp empfiehlt die Angabe `dataLIF`. Falls keine Daten angegeben werden, ruft Trident die dataLIFs vom SVM ab. Sie können einen vollqualifizierten Domänennamen (FQDN) angeben, der für die NFS-Mount-Operationen verwendet werden soll. Dadurch können Sie ein Round-Robin-DNS erstellen, um die Last auf mehrere DataLIFs zu verteilen. • ONTAP SAN-Treiber*: Nicht für iSCSI oder FC angeben. Trident Anwendungen"ONTAP Selective LUN Map" um die iSCSI- oder FC-LIFs zu ermitteln, die zum Aufbau einer Multipath-Sitzung benötigt werden. Es wird eine Warnung generiert, wenn `dataLIF` ist explizit definiert.	`10.0.0.2`
`svm`	Zu verwendende virtuelle Speichermaschine (erforderlich, wenn es sich bei der Management-LIF um eine Cluster-LIF handelt)	`svm_nfs`
`username`	Benutzername für die Verbindung zum Speichergerät	`vsadmin`
`password`	Passwort zum Verbinden mit dem Speichergerät	`secret`
`aggregate`	Aggregat für die Bereitstellung (optional; falls festgelegt, muss es der SVM zugewiesen werden). Für die `ontap-nas-flexgroup` Treiber, diese Option wird ignoriert. Alle dem SVM zugewiesenen Aggregate werden zur Bereitstellung eines FlexGroup Volumes verwendet.	`aggr1`

Option	Beschreibung	Beispiel
`limitAggregateUsage`	Optional: Die Bereitstellung schlägt fehl, wenn die Nutzung diesen Prozentsatz überschreitet.	`75%`
`nfsMountOptions`	Feingranulare Steuerung der NFS-Mount-Optionen; Standardwert ist "-o nfsvers=3". **Nur verfügbar für die `ontap-nas` Und `ontap-nas-economy` Fahrer**. "Informationen zur NFS-Hostkonfiguration finden Sie hier." .	`-o nfsvers=4`
`igroupName`	Trident erstellt und verwaltet pro Knoten `igroups` als `netappdvp` . Dieser Wert darf weder geändert noch weggelassen werden. **Nur verfügbar für die `ontap-san` Treiber**.	`netappdvp`
`limitVolumeSize`	Maximal anforderbares Volumen.	`300g`
`qtreesPerFlexvol`	Die maximale Anzahl an Qtrees pro FlexVol muss im Bereich [50, 300] liegen; der Standardwert ist 200. **Für die `ontap-nas-economy` Mit dieser Option kann die maximale Anzahl von Qtrees pro FlexVol** angepasst werden.	`300`
`sanType`	**Unterstützt für `ontap-san` Nur für den Fahrer.** Zur Auswahl verwenden `iscsi` für iSCSI, `nvme` für NVMe/TCP oder `fcp` für SCSI über Fibre Channel (FC).	`iscsi`falls leer
`limitVolumePoolSize`	**Unterstützt für `ontap-san-economy` Und `ontap-san-economy` Nur für Fahrer.** Begrenzt die FlexVol -Größen in den ONTAP Treibern ontap-nas-economy und ontap-SAN-economy.	`300g`

Es stehen Standardoptionen zur Verfügung, sodass Sie diese nicht bei jedem erstellten Volume angeben müssen:

Option	Beschreibung	Beispiel
`spaceReserve`	Platzreservierungsmodus; `none` (Thin Provisioning) oder `volume` (dick)	`none`
`snapshotPolicy`	Zu verwendende Snapshot-Richtlinie, Standardwert ist `none`	`none`
`snapshotReserve`	Snapshot-Reserveprozentsatz, Standardwert ist "", um den ONTAP Standardwert zu akzeptieren	`10`
`splitOnClone`	Beim Erstellen eines Klons diesen vom übergeordneten Klon trennen (Standardeinstellung). `false`	`false`
`encryption`	Aktiviert die NetApp Volume-Verschlüsselung (NVE) auf dem neuen Volume; Standardwert: `false` . Um diese Option nutzen zu können, muss NVE auf dem Cluster lizenziert und aktiviert sein. Wenn NAE im Backend aktiviert ist, wird jedes in Trident bereitgestellte Volume NAE-fähig sein. Weitere Informationen finden Sie unter:"Wie Trident mit NVE und NAE zusammenarbeitet" .	true
`unixPermissions`	NAS-Option für bereitgestellte NFS-Volumes, Standardwert `777`	`777`
`snapshotDir`	NAS-Option für den Zugriff auf die `.snapshot` Verzeichnis.	"true" für NFSv4, "false" für NFSv3
`exportPolicy`	NAS-Option für die zu verwendende NFS-Exportrichtlinie, Standardwert `default`	`default`
`securityStyle`	NAS-Option für den Zugriff auf das bereitgestellte NFS-Volume. NFS unterstützt `mixed` Und `unix` Sicherheitsstile. Standardmäßig ist `unix` .	`unix`
`fileSystemType`	SAN-Option zur Auswahl des Dateisystemtyps, Standardwert: `ext4`	`xfs`
`tieringPolicy`	Zu verwendende Tiering-Richtlinie, Standardwert ist `none` .	`none`

Skalierungsoptionen

Der `ontap-nas` Und `ontap-san` Die Treiber erstellen für jedes Docker-Volume ein ONTAP FlexVol . ONTAP unterstützt bis zu 1000 FlexVols pro Clusterknoten, wobei der Cluster maximal 12.000 FlexVol -Volumes umfassen kann. Wenn Ihre Docker-Volume-Anforderungen innerhalb dieser Beschränkung liegen, `ontap-nas` Der Treiber ist aufgrund der zusätzlichen Funktionen von FlexVols, wie Docker-Volume-granularen Snapshots und Klonen, die bevorzugte NAS-Lösung.

Wenn Sie mehr Docker-Volumes benötigen, als die FlexVol Grenzwerte zulassen, wählen Sie die Option `ontap-nas-economy` oder die `ontap-san-economy` Treiber.

Der `ontap-nas-economy` Der Treiber erstellt Docker-Volumes als ONTAP Qtrees innerhalb eines Pools von automatisch verwalteten FlexVol Volumes. Qtrees bieten eine weitaus größere Skalierbarkeit, bis zu 100.000 pro Clusterknoten und 2.400.000 pro Cluster, allerdings auf Kosten einiger Funktionen. Der `ontap-nas-economy` Der Treiber unterstützt weder Docker-Volume-granulare Snapshots noch das Klonen.

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

Der `ontap-san-economy` Der Treiber erstellt Docker-Volumes als ONTAP -LUNs innerhalb eines gemeinsam genutzten Pools von automatisch verwalteten FlexVol Volumes. Auf diese Weise ist jedes FlexVol nicht auf nur eine LUN beschränkt und bietet eine bessere Skalierbarkeit für SAN-Workloads. Abhängig vom Speicherarray unterstützt ONTAP bis zu 16384 LUNs pro Cluster. Da es sich bei den Volumes im Kern um LUNs handelt, unterstützt dieser Treiber Docker-volume-granulare Snapshots und Klonen.

Wählen Sie das Symbol `ontap-nas-flexgroup` Treiber zur Steigerung der Parallelität auf ein einzelnes Volume, das mit Milliarden von Dateien in den Petabyte-Bereich anwachsen kann. Zu den idealen Anwendungsfällen für FlexGroups gehören KI/ML/DL, Big Data und Analytik, Softwareentwicklung, Streaming, Dateispeicher und so weiter. Trident verwendet alle einem SVM zugewiesenen Aggregate bei der Bereitstellung eines FlexGroup -Volumes. Die FlexGroup Unterstützung in Trident erfordert außerdem folgende Überlegungen:

- Erfordert ONTAP Version 9.2 oder höher.
- Zum jetzigen Zeitpunkt unterstützen FlexGroups nur NFS v3.
- Es wird empfohlen, die 64-Bit-NFSv3-Kennungen für die SVM zu aktivieren.
- Die empfohlene Mindestgröße für FlexGroup Mitglieder/Volumes beträgt 100 GiB.
- Das Klonen von FlexGroup -Volumes wird nicht unterstützt.

Weitere Informationen zu FlexGroups und für FlexGroups geeigneten Workloads finden Sie in der "NetApp FlexGroup Volume: Best Practices und Implementierungsleitfaden" .

Um erweiterte Funktionen und große Skalierbarkeit in derselben Umgebung zu nutzen, können Sie mehrere Instanzen des Docker Volume Plugins ausführen, wobei eine davon verwendet wird. `ontap-nas` und ein weiterer, der `ontap-nas-economy` .

Benutzerdefinierte ONTAP Rolle für Trident

Sie können eine ONTAP Clusterrolle mit minimalen Berechtigungen erstellen, sodass Sie für Operationen in Trident nicht die ONTAP Administratorrolle verwenden müssen. Wenn Sie den Benutzernamen in einer Trident Backend-Konfiguration angeben, verwendet Trident die von Ihnen erstellte ONTAP Clusterrolle, um die Operationen durchzuführen.

Siehe"Trident -Benutzerrollengenerator" Weitere Informationen zum Erstellen benutzerdefinierter Trident -Rollen finden Sie hier.

Verwendung der ONTAP Befehlszeile

1. Erstellen Sie eine neue Rolle mit folgendem 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\>
```

Systemmanager verwenden

Führen Sie die folgenden Schritte im ONTAP System Manager aus:

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

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

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

Weitere Informationen finden Sie auf den folgenden Seiten:

- "Benutzerdefinierte Rollen für die Administration von ONTAP"oder"Benutzerdefinierte Rollen definieren"
- "Mit Rollen und Benutzern arbeiten"

Beispielhafte ONTAP Konfigurationsdateien

NFS-Beispiel für den <code>ontap-nas</code>-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 den <code>ontap-nas-flexgroup</code>-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 den <code>ontap-nas-economy</code>-Treiber

```
{
  "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 den <code>ontap-san</code>-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 den <code>ontap-san-economy</code>-Treiber

```
{
  "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 den <code>ontap-san</code>-Treiber

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

SCSI über FC-Beispiel für den <code>ontap-san</code>-Treiber

```
{
  "version": 1,
  "backendName": "ontap-san-backend",
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "sanType": "fcp",
  "svm": "trident_svm",
  "username": "vsadmin",
  "password": "password",
  "useREST": true
}
```

Element-Softwarekonfiguration

Zusätzlich zu den globalen Konfigurationswerten stehen bei Verwendung der Element-Software (NetApp HCI/ SolidFire) folgende Optionen zur Verfügung.

Option	Beschreibung	Beispiel
`Endpoint`	<a href="https://<login>:<password>@<mvip>/json-rpc/<element-version&gt" class="bare">https://<login>:<password>@<mvip>/json-rpc/<element-version&gt;	https://admin:admin@192.168.160.3/json-rpc/8.0

Option	Beschreibung	Beispiel
`SVIP`	iSCSI-IP-Adresse und Port	10.0.0.7:3260
`TenantName`	SolidFireF-Mandant (wird erstellt, falls nicht gefunden)	`docker`
`InitiatorIFace`	Schnittstelle angeben, wenn der iSCSI-Datenverkehr auf eine nicht standardmäßige Schnittstelle beschränkt wird	`default`
`Types`	QoS-Spezifikationen	Siehe Beispiel unten
`LegacyNamePrefix`	Präfix für aktualisierte Trident Installationen. Wenn Sie eine Version von Trident vor 1.3.2 verwendet haben und ein Upgrade mit bestehenden Volumes durchführen, müssen Sie diesen Wert festlegen, um auf Ihre alten Volumes zugreifen zu können, die über die Volume-Name-Methode zugeordnet wurden.	`netappdvp-`

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

Beispiel einer Element-Softwarekonfigurationsdatei

```
{
  "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

Hier finden Sie Informationen zu bekannten Problemen und Einschränkungen bei der Verwendung von Trident mit Docker.

Beim Upgrade des Trident Docker Volume Plugins von älteren Versionen auf Version 20.10 oder höher tritt ein Upgrade-Fehler mit der Fehlermeldung „Datei oder Verzeichnis nicht gefunden“ auf.

Problemumgehung

1. Deaktivieren Sie das Plugin.

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

2. Entfernen Sie das Plugin.

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

3. Installieren Sie das Plugin erneut, indem Sie die zusätzlichen Informationen bereitstellen. `config` Parameter.

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

Die Bandnamen müssen mindestens 2 Zeichen lang sein.

Dies ist eine Einschränkung des Docker-Clients. Der Client interpretiert einen einstelligen Namen als Windows-Pfad. "Siehe Fehler 25773" .

Docker Swarm weist bestimmte Verhaltensweisen auf, die Trident daran hindern, es mit jeder Speicher- und Treiberkombination zu unterstützen.

- Docker Swarm verwendet derzeit den Volumennamen anstelle der Volumen-ID als eindeutigen Volumenbezeichner.
- Volumenanforderungen werden gleichzeitig an jeden Knoten in einem Swarm-Cluster gesendet.
- Volume-Plugins (einschließlich Trident) müssen auf jedem Knoten in einem Swarm-Cluster unabhängig ausgeführt werden. Aufgrund der Funktionsweise von ONTAP und der Art und Weise, wie die `ontap-nas` Und `ontap-san` Wenn Fahrer funktionieren, sind sie die einzigen, die innerhalb dieser Grenzen arbeiten können.

Die übrigen Treiber sind Problemen wie Race Conditions ausgesetzt, die dazu führen können, dass für eine einzelne Anfrage eine große Anzahl von Volumes erstellt wird, ohne dass es einen klaren "Gewinner" gibt; beispielsweise verfügt Element über eine Funktion, die es ermöglicht, dass Volumes den gleichen Namen, aber unterschiedliche IDs haben.

NetApp hat dem Docker-Team Feedback gegeben, es gibt aber keine Hinweise auf weitere Maßnahmen.

Wird eine FlexGroup bereitgestellt, stellt ONTAP keine zweite FlexGroup bereit, wenn die zweite FlexGroup ein oder mehrere Aggregate mit der bereitzustellenden FlexGroup gemeinsam hat.

Bewährte Verfahren und Empfehlungen

Einsatz

Beachten Sie bei der Bereitstellung von Trident die hier aufgeführten Empfehlungen.

In einem dedizierten Namensraum bereitstellen

"Namensräume"Sie sorgen für eine administrative Trennung zwischen verschiedenen Anwendungen und stellen ein Hindernis für die gemeinsame Nutzung von Ressourcen dar. Beispielsweise kann ein PVC aus einem Namespace nicht aus einem anderen Namespace verwendet werden. Trident stellt PV-Ressourcen für alle Namespaces im Kubernetes-Cluster bereit und nutzt daher ein Servicekonto mit erhöhten Berechtigungen.

Darüber hinaus könnte der Zugriff auf den Trident Pod einem Benutzer den Zugriff auf Anmeldeinformationen des Speichersystems und andere sensible Informationen ermöglichen. Es ist wichtig sicherzustellen, dass Anwendungsbenutzer und Verwaltungsanwendungen keinen Zugriff auf die Trident -Objektdefinitionen oder die Pods selbst haben.

Nutzen Sie Quoten und Bereichsgrenzen, um den Speicherverbrauch zu kontrollieren.

Kubernetes verfügt über zwei Funktionen, die in Kombination einen leistungsstarken Mechanismus zur Begrenzung des Ressourcenverbrauchs von Anwendungen bieten. Der "Speicherquotenmechanismus" Ermöglicht dem Administrator die Implementierung globaler und speicherklassenspezifischer Kapazitäts- und Objektanzahlbeschränkungen pro Namespace. Des Weiteren, unter Verwendung eines "Reichweitenbegrenzung" stellt sicher, dass die PVC-Anfragen sowohl einen minimalen als auch einen maximalen Wert nicht überschreiten, bevor die Anfrage an den Provisionierer weitergeleitet wird.

Diese Werte werden pro Namespace definiert, was bedeutet, dass für jeden Namespace Werte definiert sein sollten, die seinen Ressourcenanforderungen entsprechen. Hier finden Sie weitere Informationen zu "wie man Quoten optimal nutzt" .

Storage-Konfiguration

Jede Speicherplattform im NetApp Portfolio verfügt über einzigartige Funktionen, die Anwendungen zugutekommen, unabhängig davon, ob sie containerisiert sind oder nicht.

Plattformübersicht

Trident ist mit ONTAP und Element kompatibel. Es gibt keine Plattform, die für alle Anwendungen und Szenarien besser geeignet ist als eine andere. Bei der Auswahl einer Plattform sollten jedoch die Bedürfnisse der Anwendung und des Teams, das das Gerät verwaltet, berücksichtigt werden.

Sie sollten die grundlegenden Best Practices für das Host-Betriebssystem mit dem von Ihnen verwendeten Protokoll befolgen. Optional können Sie, sofern verfügbar, die Best Practices der Anwendung in die Backend-, Speicherklassen- und PVC-Einstellungen einbeziehen, um den Speicher für bestimmte Anwendungen zu optimieren.

ONTAP und Cloud Volumes ONTAP Best Practices

Lernen Sie die besten Vorgehensweisen für die Konfiguration von ONTAP und Cloud Volumes ONTAP für Trident kennen.

Die folgenden Empfehlungen sind Richtlinien für die Konfiguration von ONTAP für containerisierte Workloads, die Volumes nutzen, die von Trident dynamisch bereitgestellt werden. Jede dieser Optionen sollte auf ihre Eignung für Ihr Umfeld hin geprüft und bewertet werden.

Verwenden Sie SVMs, die speziell für Trident vorgesehen sind.

Storage Virtual Machines (SVMs) bieten Isolation und administrative Trennung zwischen Mandanten auf einem ONTAP System. Die Zuweisung einer SVM zu Anwendungen ermöglicht die Delegation von Berechtigungen und die Anwendung bewährter Verfahren zur Begrenzung des Ressourcenverbrauchs.

Für die Verwaltung der SVM stehen mehrere Optionen zur Verfügung:

- Stellen Sie in der Backend-Konfiguration die Cluster-Management-Schnittstelle zusammen mit den entsprechenden Anmeldeinformationen bereit und geben Sie den SVM-Namen an.
- Erstellen Sie eine dedizierte Verwaltungsschnittstelle für die SVM mithilfe des ONTAP System Manager oder der CLI.
- Teilen Sie die Verwaltungsrolle mit einer NFS-Datenschnittstelle.

In jedem Fall sollte die Schnittstelle im DNS hinterlegt sein, und der DNS-Name sollte bei der Konfiguration von Trident verwendet werden. Dies erleichtert bestimmte DR-Szenarien, beispielsweise SVM-DR ohne Verwendung der Netzwerkidentitätsbeibehaltung.

Es gibt keine Präferenz zwischen einem dedizierten oder einem gemeinsam genutzten Management-LIF für die SVM. Sie sollten jedoch sicherstellen, dass Ihre Netzwerksicherheitsrichtlinien mit dem von Ihnen gewählten Ansatz übereinstimmen. Ungeachtet dessen sollte die Management-LIF über DNS zugänglich sein, um maximale Flexibilität zu gewährleisten. "SVM-DR" wird in Verbindung mit Trident verwendet.

Begrenzen Sie die maximale Anzahl

ONTAP Speichersysteme haben eine maximale Anzahl an Volumes, die je nach Softwareversion und Hardwareplattform variiert. Siehe "NetApp Hardware Universe" Die genauen Grenzwerte hängen von Ihrer spezifischen Plattform und ONTAP Version ab. Wenn die maximale Speicherkapazität erreicht ist, schlagen Bereitstellungsvorgänge nicht nur für Trident, sondern für alle Speicheranforderungen fehl.

Tridents `ontap-nas` Und `ontap-san` Treiber stellen für jedes erstellte Kubernetes Persistent Volume (PV) ein FlexVolume bereit. Der `ontap-nas-economy` Der Treiber erzeugt ungefähr ein FlexVolume pro 200 PVs (konfigurierbar zwischen 50 und 300). Der `ontap-san-economy` Der Treiber erzeugt ungefähr ein FlexVolume pro 100 PVs (konfigurierbar zwischen 50 und 200). Um zu verhindern, dass Trident alle verfügbaren Volumes auf dem Speichersystem verbraucht, sollten Sie ein Limit für die SVM festlegen. Sie können dies über die Befehlszeile tun:

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

Der Wert für `max-volumes` variiert je nach verschiedenen Kriterien, die für Ihre Umgebung spezifisch sind:

- Die Anzahl der vorhandenen Volumes im ONTAP Cluster

- Die Anzahl der Volumes, die Sie voraussichtlich außerhalb von Trident für andere Anwendungen bereitstellen werden.
- Die Anzahl der persistenten Volumes, die voraussichtlich von Kubernetes-Anwendungen belegt werden.

Der `max-volumes` Der Wert bezieht sich auf die Gesamtmenge der auf allen Knoten im ONTAP Cluster bereitgestellten Daten und nicht auf einen einzelnen ONTAP Knoten. Infolgedessen können Bedingungen auftreten, bei denen ein ONTAP Clusterknoten über weitaus mehr oder weniger von Trident bereitgestellte Volumes verfügt als ein anderer Knoten.

Ein ONTAP -Cluster mit zwei Knoten kann beispielsweise maximal 2000 FlexVol Volumes hosten. Die Festlegung der maximalen Lautstärkeanzahl auf 1250 erscheint sehr sinnvoll. Wenn jedoch nur "Aggregate" Wenn von einem Knoten Daten dem SVM zugewiesen werden oder die von einem Knoten zugewiesenen Aggregate nicht bereitgestellt werden können (z. B. aufgrund von Kapazitätsengpässen), wird der andere Knoten zum Ziel für alle von Trident bereitgestellten Volumes. Dies bedeutet, dass die Volumenbegrenzung für diesen Knoten möglicherweise erreicht wird, bevor `max-volumes` Dieser Wert wird erreicht, was sich sowohl auf Trident als auch auf andere Volumenoperationen auswirkt, die diesen Knoten verwenden. **Diese Situation lässt sich vermeiden, indem sichergestellt wird, dass die Aggregate von jedem Knoten im Cluster dem von Trident verwendeten SVM in gleicher Anzahl zugewiesen werden.**

Klonen eines Volumes

NetApp Trident unterstützt das Klonen von Volumes bei Verwendung von `ontap-nas` , `ontap-san` , `solidfire-san` , Und `gcp-cvs` Speichertreiber. Bei der Verwendung des `ontap-nas-flexgroup` oder `ontap-nas-economy` Treiber, Klonen wird nicht unterstützt. Wenn aus einem bestehenden Volume ein neues Volume erstellt wird, wird ein neuer Snapshot erstellt.

> Vermeiden Sie das Klonen eines PVC, das einer anderen StorageClass zugeordnet ist. Um Kompatibilität zu gewährleisten und unerwartetes Verhalten zu vermeiden, sollten Klonvorgänge innerhalb derselben StorageClass durchgeführt werden.

Begrenzen Sie die maximale Größe der von Trident erstellten Volumen.

Um die maximale Größe für von Trident erstellbare Volumes zu konfigurieren, verwenden Sie die `limitVolumeSize` Parameter in Ihrem `backend.json` Definition.

Zusätzlich zur Steuerung der Volumegröße im Speicherarray sollten Sie auch die Kubernetes-Funktionen nutzen.

Die maximale Größe von FlexVols, die von Trident erstellt werden, begrenzen.

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

Konfigurieren Sie Trident für die Verwendung von bidirektionalem CHAP.

Sie können die Benutzernamen und Passwörter für CHAP-Initiator und -Ziel in Ihrer Backend-Definition angeben und Trident veranlassen, CHAP auf der SVM zu aktivieren. Verwenden des `useCHAP` Parameter in Ihrer Backend-Konfiguration, Trident authentifiziert iSCSI-Verbindungen für ONTAP -Backends mit CHAP.

Erstellen und Verwenden einer SVM-QoS-Richtlinie

Durch die Nutzung einer ONTAP QoS-Richtlinie, die auf die SVM angewendet wird, wird die Anzahl der von den bereitgestellten Trident -Volumes verbrauchbaren IOPS begrenzt. Dies hilft dabei "Mobbing verhindern" oder dass ein außer Kontrolle geratener Container Arbeitslasten außerhalb der Trident SVM beeinträchtigt.

Sie können in wenigen Schritten eine QoS-Richtlinie für die SVM erstellen. Die genauesten Informationen finden Sie in der Dokumentation zu Ihrer ONTAP -Version. Das folgende Beispiel erstellt eine QoS-Richtlinie, die die insgesamt für die 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>
```

Zusätzlich können Sie, sofern Ihre ONTAP -Version dies unterstützt, die Verwendung eines QoS-Minimums in Betracht ziehen, um einen gewissen Durchsatz für containerisierte Workloads zu garantieren. Adaptives QoS ist nicht mit einer SVM-Richtlinie kompatibel.

Die Anzahl der für die containerisierten Workloads reservierten IOPS hängt von vielen Aspekten ab. Dazu gehören unter anderem:

- Andere Arbeitslasten, die das Speichersystem nutzen. Falls andere Arbeitslasten, die nicht mit der Kubernetes-Bereitstellung zusammenhängen, die Speicherressourcen nutzen, ist darauf zu achten, dass diese Arbeitslasten nicht versehentlich beeinträchtigt werden.
- Erwartete Workloads, die in Containern ausgeführt werden. Wenn Workloads mit hohen IOPS-Anforderungen in Containern ausgeführt werden, führt eine niedrige QoS-Richtlinie zu einer schlechten Benutzererfahrung.

Wichtig ist zu beachten, dass eine auf SVM-Ebene zugewiesene QoS-Richtlinie dazu führt, dass alle der SVM bereitgestellten Volumes denselben IOPS-Pool nutzen. Wenn eine oder wenige der containerisierten Anwendungen einen hohen IOPS-Bedarf haben, könnte dies zu einer Belastung für die anderen containerisierten Workloads führen. In diesem Fall sollten Sie erwägen, externe Automatisierung zur Zuweisung von QoS-Richtlinien pro Volume zu verwenden.

Sie sollten die QoS-Richtliniengruppe der SVM **nur** dann zuweisen, wenn Ihre ONTAP Version älter als 9.8 ist.

Erstellen Sie QoS-Richtliniengruppen für Trident.

Die Dienstgüte (QoS) gewährleistet, dass die Leistung kritischer Arbeitslasten nicht durch konkurrierende Arbeitslasten beeinträchtigt wird. ONTAP QoS-Richtliniengruppen bieten QoS-Optionen für Volumes und ermöglichen es Benutzern, die Durchsatzobergrenze für eine oder mehrere Workloads zu definieren. Weitere Informationen zu QoS finden Sie unter "Gewährleistung des Durchsatzes mit QoS" . Sie können QoS-Richtliniengruppen im Backend oder in einem Speicherpool festlegen, und diese werden auf jedes in diesem Pool oder Backend erstellte Volume angewendet.

ONTAP kennt zwei Arten von QoS-Richtliniengruppen: traditionelle und adaptive. Traditionelle

Richtliniengruppen bieten einen festen maximalen (bzw. minimalen, in späteren Versionen) Durchsatz in IOPS. Adaptive QoS skaliert den Durchsatz automatisch an die Workload-Größe und hält dabei das Verhältnis von IOPS zu TBs|GBs aufrecht, während sich die Größe der Workload ändert. Dies bietet einen erheblichen Vorteil, wenn Sie Hunderte oder Tausende von Workloads in einer großen Bereitstellung verwalten.

Beachten Sie Folgendes beim Erstellen von QoS-Richtliniengruppen:

- Du solltest die `qosPolicy` Schlüssel im `defaults` Block der Backend-Konfiguration. Siehe das folgende Beispiel für eine Backend-Konfiguration:

```
---
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, durch die Richtliniengruppe festgelegten Durchsatz erhält. Gemeinsame Richtliniengruppen werden nicht unterstützt.

Weitere Informationen zu QoS-Richtliniengruppen finden Sie unter "ONTAP Befehlsreferenz" .

Beschränken Sie den Zugriff auf Speicherressourcen für Kubernetes-Cluster-Mitglieder.

Die Beschränkung des Zugriffs auf die von Trident erstellten NFS-Volumes, iSCSI-LUNs und FC-LUNs ist ein entscheidender Bestandteil der Sicherheitsarchitektur Ihrer Kubernetes-Bereitstellung. Dadurch wird verhindert, dass Hosts, die nicht Teil des Kubernetes-Clusters sind, auf die Volumes zugreifen und möglicherweise Daten unerwartet ändern.

Es ist wichtig zu verstehen, dass Namespaces die logische Grenze für Ressourcen in Kubernetes sind. Die Annahme ist, dass Ressourcen im selben Namensraum gemeinsam genutzt werden können; wichtig ist jedoch, dass es keine Möglichkeit zur Nutzung über verschiedene Namensräume hinweg gibt. Dies bedeutet, dass PVs zwar globale Objekte sind, aber wenn sie an ein PVC gebunden sind, nur von Pods im selben Namespace zugänglich sind. Es ist von entscheidender Bedeutung, sicherzustellen, dass Namensräume gegebenenfalls zur Trennung verwendet werden.

Die größte Sorge der meisten Organisationen hinsichtlich der Datensicherheit im Kubernetes-Kontext besteht

darin, dass ein Prozess in einem Container auf Speicher zugreifen kann, der auf dem Host eingebunden ist, aber nicht für den Container bestimmt ist. "Namensräume" sind so konzipiert, dass diese Art von Kompromiss verhindert wird. Es gibt jedoch eine Ausnahme: privilegierte Container.

Ein privilegierter Container ist ein Container, der mit wesentlich mehr Berechtigungen auf Hostebene als üblich ausgeführt wird. Diese werden nicht standardmäßig deaktiviert. Stellen Sie daher sicher, dass Sie die entsprechende Funktion deaktivieren. "Pod-Sicherheitsrichtlinien" .

Bei Volumes, auf die sowohl von Kubernetes als auch von externen Hosts zugegriffen werden soll, sollte der Speicher auf traditionelle Weise verwaltet werden, wobei das PV vom Administrator eingeführt und nicht von Trident verwaltet wird. Dadurch wird sichergestellt, dass das Speichervolume erst dann zerstört wird, wenn sowohl Kubernetes als auch externe Hosts die Verbindung getrennt haben und das Volume nicht mehr verwenden. Zusätzlich kann eine benutzerdefinierte Exportrichtlinie angewendet werden, die den Zugriff von den Kubernetes-Clusterknoten und von Zielservern außerhalb des Kubernetes-Clusters ermöglicht.

Bei Bereitstellungen mit dedizierten Infrastrukturknoten (z. B. OpenShift) oder anderen Knoten, die keine Benutzeranwendungen einplanen können, sollten separate Exportrichtlinien verwendet werden, um den Zugriff auf Speicherressourcen weiter einzuschränken. Dazu gehört das Erstellen einer Exportrichtlinie für Dienste, die auf diesen Infrastrukturknoten bereitgestellt werden (z. B. die OpenShift-Dienste „Metrics“ und „Logging“), und für Standardanwendungen, die auf Nicht-Infrastrukturknoten bereitgestellt werden.

Nutzen Sie eine spezielle Exportrichtlinie

Sie sollten sicherstellen, dass für jedes Backend eine Exportrichtlinie existiert, die nur den Zugriff auf die im Kubernetes-Cluster vorhandenen Knoten erlaubt. Trident kann Exportrichtlinien automatisch erstellen und verwalten. Auf diese Weise beschränkt Trident den Zugriff auf die Volumes, die es den Knoten im Kubernetes-Cluster bereitstellt, und vereinfacht das Hinzufügen/Löschen von Knoten.

Alternativ können Sie auch manuell eine Exportrichtlinie erstellen und diese mit einer oder mehreren Exportregeln füllen, die jede Knotenzugriffsanfrage verarbeiten:

- Verwenden Sie die `vserver export-policy create` ONTAP CLI-Befehl zum Erstellen der Exportrichtlinie.
- Fügen Sie der Exportrichtlinie Regeln hinzu, indem Sie die `vserver export-policy rule create` ONTAP CLI-Befehl.

Durch Ausführen dieser Befehle können Sie einschränken, welche Kubernetes-Knoten Zugriff auf die Daten haben.

Deaktivieren `showmount` für die Anwendung SVM

Der `showmount` Diese Funktion ermöglicht es einem NFS-Client, die SVM nach einer Liste der verfügbaren NFS-Exporte abzufragen. Ein im Kubernetes-Cluster bereitgestellter Pod kann die folgende Aktion ausführen: `showmount -e` Mit dem Befehl `get` erhalten Sie eine Liste der verfügbaren Mounts, einschließlich derer, auf die es keinen Zugriff hat. Dies stellt zwar für sich genommen keine Sicherheitslücke dar, liefert aber unnötige Informationen, die einem nicht autorisierten Benutzer möglicherweise dabei helfen, eine Verbindung zu einem NFS-Export herzustellen.

Sie sollten deaktivieren `showmount` durch Verwendung des ONTAP CLI-Befehls auf SVM-Ebene:

```
vserver nfs modify -vserver <svm_name> -showmount disabled
```

SolidFire – bewährte Verfahren

Lernen Sie die besten Vorgehensweisen für die Konfiguration von SolidFire -Speicher für Trident kennen.

Solidfire-Konto erstellen

Jedes SolidFire -Konto repräsentiert einen eindeutigen Volume-Inhaber und erhält seine eigenen CHAP-Anmeldeinformationen (Challenge-Handshake Authentication Protocol). Sie können auf einem Konto zugewiesene Datenträger entweder über den Kontonamen und die zugehörigen CHAP-Anmeldeinformationen oder über eine Datenträgerzugriffsgruppe zugreifen. Einem Konto können bis zu zweitausend Volumes zugeordnet werden, aber ein Volume kann nur zu einem Konto gehören.

Erstellen Sie eine QoS-Richtlinie

Verwenden Sie SolidFire Quality of Service (QoS)-Richtlinien, wenn Sie eine standardisierte Servicequalitätseinstellung erstellen und speichern möchten, die auf viele Volumes angewendet werden kann.

Sie können QoS-Parameter pro Volume festlegen. Die Leistungsfähigkeit jedes Volumes kann durch die Festlegung von drei konfigurierbaren Parametern, die die QoS definieren, sichergestellt werden: Min IOPS, Max IOPS und Burst IOPS.

Hier sind die möglichen Minimal-, Maximal- und Burst-IOPS-Werte für die 4-Kb-Blockgröße.

IOPS-Parameter	Definition	Minimalwert	Standardwert	Maximalwert (4 KB)
Min IOPS	Das garantierte Leistungsniveau für ein bestimmtes Volumen.	50	50	15000
Maximale IOPS	Die Leistung wird dieses Limit nicht überschreiten.	50	15000	200.000
Burst IOPS	Maximal zulässige IOPS in einem Kurzzeit-Burst-Szenario.	50	15000	200.000

Obwohl die Werte für Max IOPS und Burst IOPS auf bis zu 200.000 eingestellt werden können, ist die tatsächliche maximale Leistung eines Volumes durch die Clusternutzung und die Leistung pro Knoten begrenzt.

Blockgröße und Bandbreite haben einen direkten Einfluss auf die Anzahl der IOPS. Mit zunehmender Blockgröße erhöht das System die Bandbreite auf ein Niveau, das zur Verarbeitung der größeren Blockgrößen erforderlich ist. Mit zunehmender Bandbreite sinkt die Anzahl der IOPS, die das System erreichen kann. Siehe "SolidFire Servicequalität" Für weitere Informationen zu QoS und Leistung.

SolidFire Authentifizierung

Element unterstützt zwei Authentifizierungsmethoden: CHAP und Volume Access Groups (VAG). CHAP verwendet das CHAP-Protokoll zur Authentifizierung des Hosts gegenüber dem Backend. Volume Access

Groups steuert den Zugriff auf die von ihnen bereitgestellten Volumes. NetApp empfiehlt die Verwendung von CHAP zur Authentifizierung, da es einfacher ist und keine Skalierungsbeschränkungen aufweist.

> Trident mit dem erweiterten CSI-Provisioner unterstützt die Verwendung der CHAP-Authentifizierung. VAGs sollten nur im herkömmlichen Nicht-CSI-Betriebsmodus verwendet werden.

Die CHAP-Authentifizierung (Überprüfung, ob der Initiator der beabsichtigte Volumennutzer ist) wird nur bei kontobasierter Zugriffskontrolle unterstützt. Wenn Sie CHAP zur Authentifizierung verwenden, stehen Ihnen zwei Optionen zur Verfügung: unidirektionales CHAP und bidirektionales CHAP. Unidirektionales CHAP authentifiziert den Zugriff auf das Volume mithilfe des SolidFire -Kontonamens und des Initiator-Geheimnisses. Die bidirektionale CHAP-Option bietet die sicherste Methode zur Authentifizierung des Volumes, da das Volume den Host über den Kontonamen und das Initiator-Geheimnis authentifiziert und der Host anschließend das Volume über den Kontonamen und das Ziel-Geheimnis 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 eine kontobasierte Zugriffskontrolle verwendet. Wenn der iSCSI-Initiator nicht für die Verwendung der CHAP-Authentifizierung konfiguriert ist, wird die Zugriffskontrolle über Volume Access Groups (VAG) verwendet.

Wo finde ich weitere Informationen?

Nachfolgend sind einige der Best-Practice-Dokumentationen aufgeführt. Suchen Sie nach "NetApp Bibliothek" für die aktuellsten Versionen.

- ONTAP*
- "NFS-Leitfaden für bewährte Verfahren und Implementierung"
- "SAN-Verwaltung"(für iSCSI)
- "iSCSI Express-Konfiguration für RHEL"

Element Software

- "SolidFire für Linux konfigurieren"
- NetApp HCI*
- "Voraussetzungen für die NetApp HCI Bereitstellung"
- "Greifen Sie auf die NetApp Bereitstellungs-Engine zu."

Informationen zu bewährten Anwendungspraktiken

- "Bewährte Verfahren für MySQL auf ONTAP"
- "Bewährte Vorgehensweisen für MySQL auf SolidFire"
- "NetApp SolidFire und Cassandra"
- "Oracle Best Practices für SolidFire"
- "PostgreSQL-Best Practices auf SolidFire"

Nicht für alle Anwendungen gibt es spezifische Richtlinien. Es ist wichtig, mit Ihrem NetApp -Team zusammenzuarbeiten und die "NetApp Bibliothek" um die aktuellste Dokumentation zu finden.

Integriere Trident

Für die Integration von Trident ist die Integration der folgenden Design- und Architekturelemente erforderlich: Treiberauswahl und -bereitstellung, Speicherklassendesign, Design virtueller Pools, Auswirkungen von Persistent Volume Claims (PVC) auf die Speicherbereitstellung, Volumenoperationen und die Bereitstellung von OpenShift-Diensten mit Trident.

Fahrerauswahl und -einsatz

Wählen Sie einen Backend-Treiber für Ihr Speichersystem aus und stellen Sie ihn bereit.

ONTAP Backend-Treiber

ONTAP Backend-Treiber unterscheiden sich durch das verwendete Protokoll und die Art und Weise, wie die Volumes auf dem Speichersystem bereitgestellt werden. Überlegen Sie sich daher gut, welchen Treiber Sie einsetzen möchten.

Auf einer höheren Ebene, wenn Ihre Anwendung Komponenten enthält, die gemeinsam genutzten Speicher benötigen (mehrere Pods greifen auf denselben PVC zu), sind NAS-basierte Treiber die Standardwahl, während die blockbasierten iSCSI-Treiber die Anforderungen von nicht gemeinsam genutztem Speicher erfüllen. Wählen Sie das Protokoll basierend auf den Anforderungen der Anwendung und dem Komfortniveau der Speicher- und Infrastrukturteams. Im Allgemeinen gibt es für die meisten Anwendungen kaum einen Unterschied zwischen ihnen, daher basiert die Entscheidung oft darauf, ob gemeinsam genutzter Speicher (auf den mehr als ein Pod gleichzeitig zugreifen muss) benötigt wird oder nicht.

Folgende ONTAP Backend-Treiber stehen zur Verfügung:

- `ontap-nas`Jedes bereitgestellte PV ist ein vollständiges ONTAP FlexVolume.
- `ontap-nas-economy`: Jedes bereitgestellte PV ist ein Qtree, wobei die Anzahl der Qtrees pro FlexVolume konfigurierbar ist (Standardwert ist 200).
- `ontap-nas-flexgroup`: Jeder PV wird als vollständige ONTAP FlexGroup bereitgestellt, und alle einem SVM zugewiesenen Aggregate werden verwendet.
- `ontap-san`Jedes bereitgestellte PV ist eine LUN innerhalb eines eigenen FlexVolumes.
- `ontap-san-economy`Jedes bereitgestellte PV ist eine LUN, wobei die Anzahl der LUNs pro FlexVolume konfigurierbar ist (Standardwert ist 100).

Die Wahl zwischen den drei NAS-Treibern hat Auswirkungen auf die Funktionen, die der Anwendung zur Verfügung stehen.

Beachten Sie, dass in den folgenden Tabellen nicht alle Funktionen über Trident zugänglich sind. Einige dieser Einstellungen müssen vom Speicheradministrator nach der Bereitstellung vorgenommen werden, wenn die gewünschte Funktionalität gewünscht ist. Die hochgestellten Fußnoten unterscheiden die Funktionalität pro Feature und Treiber.

ONTAP NAS-Treiber	Snapshots	Klone	Dynamische Exportrichtlinien	Mehrfachbefestigung	QoS	Größe ändern	Replikation
`ontap-nas`	Ja	Ja	Fußnote Ja:5[]	Ja	Fußnote:1[]	Ja	Fußnote:1[]
`ontap-nas-economy`	NO [3]	NO [3]	Fußnote Ja:5[]	Ja	NO [3]	Ja	NO [3]
`ontap-nas-flexgroup`	Fußnote:1[]	NEIN	Fußnote Ja:5[]	Ja	Fußnote:1[]	Ja	Fußnote:1[]

Trident bietet 2 SAN-Treiber für ONTAP an, deren Funktionen im Folgenden dargestellt werden.

ONTAP SAN-Treiber	Snapshots	Klone	Mehrfachbefestigung	Bidirektionales CHAP	QoS	Größe ändern	Replikation
`ontap-san`	Ja	Ja	Fußnote Ja:4[]	Ja	Fußnote:1[]	Ja	Fußnote:1[]
`ontap-san-economy`	Ja	Ja	Fußnote Ja:4[]	Ja	NO [3]	Ja	NO [3]

> Fußnoten zu den obigen Tabellen: Yes [1]: Nicht von Trident verwaltet Yes [2]: Von Trident verwaltet, aber nicht PV-granular No [3]: Nicht von Trident verwaltet und nicht PV-granular Yes [4]: Für Raw-Block-Volumes unterstützt Yes [5]: Von Trident unterstützt

Die Features, die nicht PV-granular sind, werden auf das gesamte FlexVolume angewendet, und alle PVs (d. h. Qtrees oder LUNs in gemeinsam genutzten FlexVols) teilen sich einen gemeinsamen Zeitplan.

Wie wir aus den obigen Tabellen ersehen können, besteht ein Großteil der Funktionalität zwischen den `ontap-nas` Und `ontap-nas-economy` ist dasselbe. Da jedoch `ontap-nas-economy` Der Treiber schränkt die Möglichkeit ein, den Zeitplan auf Ebene einzelner PVs zu steuern; dies kann sich insbesondere auf Ihre Notfallwiederherstellungs- und Datensicherungsplanung auswirken. Für Entwicklungsteams, die die PVC-Klonfunktion auf ONTAP -Speicher nutzen möchten, ist dies nur möglich, wenn sie Folgendes verwenden: `ontap-nas` , `ontap-san` oder `ontap-san-economy` Fahrer.

> Der `solidfire-san` Der Treiber ist auch in der Lage, PVCs zu klonen.

Cloud Volumes ONTAP Backend-Treiber

Cloud Volumes ONTAP bietet Datenkontrolle sowie Speicherfunktionen der Enterprise-Klasse für verschiedene Anwendungsfälle, darunter Dateifreigaben und Blockspeicher, die NAS- und SAN-Protokolle (NFS, SMB / CIFS und iSCSI) unterstützen. 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 und Cloud Volume ONTAP für GCP.

Amazon FSx für ONTAP Backend-Treiber

Mit Amazon FSx for NetApp ONTAP können Sie die Ihnen vertrauten Funktionen, die Leistung und die administrativen Möglichkeiten von NetApp nutzen und gleichzeitig die Einfachheit, Agilität, Sicherheit und Skalierbarkeit der Datenspeicherung auf AWS in Anspruch nehmen. 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` .

NetApp HCI/ SolidFire Backend-Treiber

Der `solidfire-san` Der Treiber, der mit den NetApp HCI/ SolidFire Plattformen verwendet wird, hilft dem Administrator, ein Element-Backend für Trident auf Basis von QoS-Grenzwerten zu konfigurieren. Wenn Sie Ihr Backend so konfigurieren möchten, dass die spezifischen QoS-Grenzwerte für die von Trident bereitgestellten Volumes festgelegt werden, verwenden Sie die `type` Parameter in der Backend-Datei. Der Administrator kann außerdem die Größe des auf dem Speicher erstellten Volumes mithilfe der `limitVolumeSize` Parameter. Aktuell werden Element-Speicherfunktionen wie die Größenänderung von Datenträgern und die Replikation von Datenträgern nicht unterstützt. `solidfire-san` Treiber. Diese Vorgänge sollten manuell über die Web-Benutzeroberfläche von Element Software durchgeführt werden.

SolidFire -Treiber	Snapshots	Klone	Mehrfachbefestigung	KERL	QoS	Größe ändern	Replikation
`solidfire-san`	Ja	Ja	Fußnote Ja:2[]	Ja	Ja	Ja	Fußnote:1[]

> Fußnote: Yes [1]: Wird nicht von Trident verwaltet Yes [2]: Wird für Raw-Block-Volumes unterstützt

Azure NetApp Files -Backend-Treiber

Trident verwendet die `azure-netapp-files` Fahrer zur Verwaltung des"Azure NetApp Files" Service.

Weitere Informationen zu diesem Treiber und dessen Konfiguration finden Sie in"Trident Backend-Konfiguration für Azure NetApp Files" .

Azure NetApp Files Treiber	Snapshots	Klone	Mehrfachbefestigung	QoS	Expandieren	Replikation
`azure-netapp-files`	Ja	Ja	Ja	Ja	Ja	Fußnote:1[]

> Fußnote: Yes [1]: Nicht von Trident verwaltet

Cloud Volumes Service auf Google Cloud Backend-Treiber

Trident verwendet die `gcp-cvs` Treiber zur Verbindung mit dem Cloud Volumes Service auf Google Cloud.

Der `gcp-cvs` Der Treiber verwendet virtuelle Pools, um das Backend zu abstrahieren und Trident die Bestimmung der Volume-Platzierung zu ermöglichen. Der Administrator definiert die virtuellen Pools im `backend.json` Dateien. Speicherklassen verwenden Selektoren, um virtuelle Pools anhand ihrer Bezeichnung zu identifizieren.

- Wenn im Backend virtuelle Pools definiert sind, versucht Trident , ein Volume in den Google Cloud-Speicherpools zu erstellen, auf das diese virtuellen Pools beschränkt sind.
- Wenn im Backend keine virtuellen Pools definiert sind, wählt Trident einen Google Cloud-Speicherpool aus den in der Region verfügbaren Speicherpools aus.

Um das Google Cloud-Backend auf Trident zu konfigurieren, müssen Sie Folgendes angeben: `projectNumber` , `apiRegion` , Und `apiKey` in der Backend-Datei. Die Projektnummer finden Sie in der Google Cloud Console. Der API-Schlüssel stammt aus der privaten Schlüsseldatei des Dienstkontos, die Sie beim Einrichten des API-Zugriffs für den Cloud Volumes Service auf Google Cloud erstellt haben.

Für Details zum Cloud Volumes Service auf Google Cloud (Diensttypen und Servicelevel) siehe"Erfahren Sie mehr über die Trident Unterstützung für CVS für GCP." .

Cloud Volumes Service für Google Cloud Driver	Snapshots	Klone	Mehrfachbefestigung	QoS	Expandieren	Replikation
`gcp-cvs`	Ja	Ja	Ja	Ja	Ja	Nur verfügbar beim Servicetyp CVS-Performance.

Replikationsnotizen

- Die Replikation wird nicht von Trident verwaltet.
- Der Klon wird im selben Speicherpool wie das Quellvolume erstellt.

Speicherklassendesign

Einzelne Storage-Klassen müssen konfiguriert und angewendet werden, um ein Kubernetes-Storage-Class-Objekt zu erstellen. In diesem Abschnitt wird erläutert, wie Sie eine Speicherklasse für Ihre Anwendung entwerfen.

Spezifische Backend-Nutzung

Mithilfe von Filtern kann innerhalb eines bestimmten Speicherklassenobjekts ermittelt werden, welcher Speicherpool oder welche Gruppe von Speicherpools mit dieser bestimmten Speicherklasse verwendet werden soll. In der Speicherklasse können drei Filtersätze festgelegt werden: `storagePools` , `additionalStoragePools` und/oder `excludeStoragePools` .

Der `storagePools` Mit diesem Parameter kann der Speicher auf die Pools beschränkt werden, die den angegebenen Attributen entsprechen. Der `additionalStoragePools` Der Parameter dient dazu, die Menge der Pools, die Trident für die Bereitstellung verwendet, zusammen mit der Menge der durch die Attribute ausgewählten Pools zu erweitern. `storagePools` Parameter. Sie können entweder den einen oder den anderen Parameter einzeln oder beide zusammen verwenden, um sicherzustellen, dass die richtigen Speicherpools ausgewählt werden.

Der `excludeStoragePools` Der Parameter dient dazu, die aufgelisteten Pools, die den Attributen entsprechen, gezielt auszuschließen.

Emulieren von QoS-Richtlinien

Wenn Sie Speicherklassen entwerfen möchten, um QoS-Richtlinien zu emulieren, erstellen Sie eine Speicherklasse mit der `media` Attribut als `hdd` oder `ssd` . Basierend auf der `media` Trident wählt für jedes in der Speicherklasse angegebene Attribut das passende Backend aus. `hdd` oder `ssd` Aggregate werden erstellt, um das Medienattribut abzugleichen und anschließend die Bereitstellung der Volumes auf das jeweilige Aggregat zu steuern. Daher können wir eine Speicherklasse PREMIUM erstellen, die Folgendes hätte: `media` Attribut festgelegt als `ssd` was als PREMIUM QoS-Richtlinie eingestuft werden könnte. Wir können eine weitere Speicherklasse STANDARD erstellen, deren Medienattribut auf `hdd` gesetzt wäre, die als STANDARD QoS-Richtlinie klassifiziert werden könnte. Wir könnten auch das Attribut „IOPS“ in der Speicherklasse verwenden, um die Bereitstellung auf ein Element-Gerät umzuleiten, das als QoS-Richtlinie definiert werden kann.

Nutzen Sie das Backend basierend auf spezifischen Funktionen

Speicherklassen können so konzipiert werden, dass die Volumenbereitstellung auf einem bestimmten Backend gesteuert wird, auf dem Funktionen wie Thin und Thick Provisioning, Snapshots, Klone und Verschlüsselung aktiviert sind. Um festzulegen, welcher Speicher verwendet werden soll, erstellen Sie Speicherklassen, die das entsprechende Backend mit den erforderlichen aktivierten Funktionen angeben.

Virtuelle Pools

Virtuelle Pools sind für alle Trident -Backends verfügbar. Sie können virtuelle Pools für jedes Backend definieren und dabei jeden von Trident bereitgestellten Treiber verwenden.

Virtuelle Pools ermöglichen es einem Administrator, eine Abstraktionsebene über Backends zu erstellen, auf die über Storage Classes verwiesen werden kann, um eine größere Flexibilität und effizientere Platzierung von Volumes auf Backends zu gewährleisten. Verschiedene Backends können mit derselben Serviceklasse definiert werden. Darüber hinaus können auf demselben Backend mehrere Speicherpools mit unterschiedlichen Eigenschaften erstellt werden. Wenn eine Speicherklasse mit einem Selektor mit den spezifischen Bezeichnungen konfiguriert ist, wählt Trident ein Backend aus, das allen Selektorbezeichnungen entspricht, um das Volume dort zu platzieren. Wenn die Bezeichnungen des Speicherklassenselektors mit mehreren Speicherpools übereinstimmen, wählt Trident einen davon aus, um das Volume bereitzustellen.

Virtuelles Pool-Design

Beim Erstellen eines Backends können Sie im Allgemeinen einen Satz von Parametern angeben. Es war für den Administrator unmöglich, ein anderes Backend mit denselben Speicheranmeldeinformationen und einem anderen Satz von Parametern zu erstellen. Mit der Einführung virtueller Pools wurde dieses Problem behoben. Ein virtueller Pool ist eine Ebenenabstraktion zwischen dem Backend und der Kubernetes-Speicherklasse, sodass der Administrator Parameter zusammen mit Bezeichnungen definieren kann, auf die über Kubernetes-Speicherklassen als Selektor Backend-unabhängig verwiesen werden kann. Virtuelle Pools können mit Trident für alle unterstützten NetApp Backends definiert werden. Diese Liste umfasst SolidFire/ NetApp HCI, ONTAP, Cloud Volumes Service auf GCP sowie Azure NetApp Files.

> Bei der Definition virtueller Pools wird empfohlen, die Reihenfolge bestehender virtueller Pools in einer Backend-Definition nicht zu ändern. Es empfiehlt sich außerdem, die Attribute eines bestehenden virtuellen Pools nicht zu bearbeiten/zu ändern, sondern stattdessen einen neuen virtuellen Pool zu definieren.

Emulation verschiedener Servicelevel/QoS

Es ist möglich, virtuelle Pools zur Emulation von Serviceklassen zu entwerfen. Anhand der virtuellen Pool-Implementierung für Cloud Volume Service für Azure NetApp Files wollen wir untersuchen, wie wir

verschiedene Dienstklassen einrichten können. Konfigurieren Sie das Azure NetApp Files -Backend mit mehreren Bezeichnungen, die unterschiedliche Leistungsstufen darstellen. Satz `servicelevel` den jeweiligen Aspekt dem entsprechenden Leistungsniveau zuordnen und unter jeder Bezeichnung weitere erforderliche Aspekte hinzufügen. Erstellen Sie nun verschiedene Kubernetes-Speicherklassen, die unterschiedlichen virtuellen Pools zugeordnet werden. Verwenden des `parameters.selector` Im Feld „StorageClass“ wird für jede StorageClass angegeben, welche virtuellen Pools zum Hosten eines Volumes verwendet werden können.

Zuweisung eines bestimmten Satzes von Aspekten

Ausgehend von einem einzigen Speicher-Backend können mehrere virtuelle Pools mit jeweils spezifischen Eigenschaften entworfen werden. Konfigurieren Sie dazu das Backend mit mehreren Labels und legen Sie unter jedem Label die erforderlichen Aspekte fest. Erstellen Sie nun verschiedene Kubernetes-Speicherklassen mithilfe der `parameters.selector` Feld, das verschiedenen virtuellen Pools zugeordnet werden würde. Die im Backend bereitgestellten Volumes weisen die im gewählten virtuellen Pool definierten Eigenschaften auf.

PVC-Eigenschaften, die die Speicherkapazität beeinflussen

Bei der Erstellung eines PVC können neben den angeforderten Speicherklassen auch andere Parameter den Trident -Bereitstellungsprozess beeinflussen.

Zugriffsmodus

Bei der Anforderung von Speicherplatz über eine PVC ist eines der Pflichtfelder der Zugriffsmodus. Der gewünschte Modus kann Einfluss darauf haben, welches Backend für die Verarbeitung der Speicheranfrage ausgewählt wird.

Trident wird versuchen, das verwendete Speicherprotokoll mit der angegebenen Zugriffsmethode gemäß der folgenden Matrix abzugleichen. Dies ist unabhängig von der zugrunde liegenden Speicherplattform.

	Einmal lesen und schreiben	ReadOnlyMany	ReadWriteMany
iSCSI	Ja	Ja	Ja (Rohblock)
NFS	Ja	Ja	Ja

Eine Anfrage für ein ReadWriteMany PVC, die an eine Trident Bereitstellung ohne konfiguriertes NFS-Backend gesendet wird, führt dazu, dass kein Volume bereitgestellt wird. Aus diesem Grund sollte der Anfragende den für seine Anwendung geeigneten Zugriffsmodus verwenden.

Volumenoperationen

Persistente Volumes ändern

Persistente Volumes sind, mit zwei Ausnahmen, unveränderliche Objekte in Kubernetes. Nach der Erstellung können die Rückforderungsrichtlinie und die Größe angepasst werden. Dies verhindert jedoch nicht, dass bestimmte Aspekte des Volumes außerhalb von Kubernetes modifiziert werden. Dies kann wünschenswert sein, um das Volume für bestimmte Anwendungen anzupassen, um sicherzustellen, dass die Kapazität nicht versehentlich verbraucht wird, oder um das Volume aus beliebigen Gründen auf einen anderen Speichercontroller zu verschieben.

Die In-Tree-Provisioner von Kubernetes unterstützen derzeit keine Volume-Resize-Operationen für NFS-, iSCSI- oder FC-PVs. Trident unterstützt die Erweiterung von NFS-, iSCSI- und FC-Volumes.

Die Verbindungsdetails des PV können nach der Erstellung nicht mehr geändert werden.

Erstellen Sie bedarfsgesteuerte Volume-Snapshots

Trident unterstützt die Erstellung von Volume-Snapshots bei Bedarf sowie die Erstellung von PVCs aus Snapshots mithilfe des CSI-Frameworks. Snapshots bieten eine komfortable Methode zur Aufbewahrung von zeitpunktbezogenen Kopien der Daten und haben einen vom Quell-PV in Kubernetes unabhängigen Lebenszyklus. Mithilfe dieser Snapshots können PVCs geklont werden.

Volumes aus Snapshots erstellen

Trident unterstützt auch die Erstellung von PersistentVolumes aus Volume-Snapshots. Um dies zu erreichen, erstellen Sie einfach einen PersistentVolumeClaim und geben Sie den entsprechenden Eintrag an. `datasource` als der erforderliche Snapshot, aus dem das Volume erstellt werden muss. Trident wird dieses PVC verarbeiten, indem ein Volume mit den im Snapshot vorhandenen Daten erstellt wird. Mit dieser Funktion können Daten regionsübergreifend dupliziert, Testumgebungen erstellt, ein beschädigtes oder fehlerhaftes Produktionsvolume vollständig ersetzt oder bestimmte Dateien und Verzeichnisse abgerufen und auf ein anderes angeschlossenes Volume übertragen werden.

Verschieben Sie die Volumes im Cluster

Speicheradministratoren haben die Möglichkeit, Datenträger zwischen Aggregaten und Controllern im ONTAP Cluster zu verschieben, ohne den Speicherkonsumenten zu beeinträchtigen. Dieser Vorgang hat keine Auswirkungen auf Trident oder den Kubernetes-Cluster, sofern es sich bei dem Zielaggregat um ein solches handelt, auf das die von Trident verwendete SVM Zugriff hat. Wichtig ist, dass, wenn das Aggregat neu zum SVM hinzugefügt wurde, das Backend aktualisiert werden muss, indem es erneut zu Trident hinzugefügt wird. Dies veranlasst Trident, das SVM neu zu inventarisieren, damit das neue Aggregat erkannt wird.

Das Verschieben von Datenvolumen zwischen verschiedenen Backends wird von Trident jedoch nicht automatisch unterstützt. Dies umfasst SVMs innerhalb desselben Clusters, SVMs zwischen Clustern oder SVMs auf einer anderen Speicherplattform (selbst wenn dieses Speichersystem mit Trident verbunden ist).

Wenn ein Volume an einen anderen Speicherort kopiert wird, kann die Volume-Importfunktion verwendet werden, um die aktuellen Volumes in Trident zu importieren.

Volumen erweitern

Trident unterstützt die Größenänderung von NFS-, iSCSI- und FC-PVs. Dies ermöglicht es Benutzern, ihre Volumes direkt über die Kubernetes-Schicht zu vergrößern oder zu verkleinern. Die Speichererweiterung ist für alle wichtigen NetApp Speicherplattformen möglich, einschließlich ONTAP, SolidFire/ NetApp HCI und Cloud Volumes Service Backends. Um eine spätere Erweiterung zu ermöglichen, setzen Sie `allowVolumeExpansion` Zu `true` in Ihrer StorageClass, die dem Volume zugeordnet ist. Immer wenn die Größe des persistenten Volumes geändert werden muss, bearbeiten Sie die `spec.resources.requests.storage` Anmerkung im Persistent Volume Claim zur erforderlichen Volumengröße. Trident kümmert sich automatisch um die Größenanpassung des Volumes im Speichercluster.

Ein vorhandenes Volume in Kubernetes importieren

Der Volume-Import ermöglicht es, ein vorhandenes Speichervolume in eine Kubernetes-Umgebung zu importieren. Dies wird derzeit unterstützt durch `ontap-nas` , `ontap-nas-flexgroup` , `solidfire-san` ,

`azure-netapp-files` , Und `gcp-cvs` Fahrer. Diese Funktion ist nützlich beim Portieren einer bestehenden Anwendung nach Kubernetes oder in Notfallwiederherstellungsszenarien.

Bei der Verwendung von ONTAP und `solidfire-san` Fahrer, verwenden Sie den Befehl `tridentctl import volume <backend-name> <volume-name> -f /path/pvc.yaml` Ein vorhandenes Volume in Kubernetes importieren, das von Trident verwaltet werden soll. Die im Import-Volume-Befehl verwendete PVC-YAML- oder JSON-Datei verweist auf eine Speicherklasse, die Trident als Provisionierer identifiziert. Bei Verwendung eines NetApp HCI/ SolidFire Backends muss sichergestellt werden, dass die Volume-Namen eindeutig sind. Falls die Volumennamen doppelt vorkommen, klonen Sie das Volumen unter einem eindeutigen Namen, damit die Volumenimportfunktion sie unterscheiden kann.

Wenn die `azure-netapp-files` oder `gcp-cvs` Wenn der Treiber verwendet wird, verwenden Sie den Befehl `tridentctl import volume <backend-name> <volume path> -f /path/pvc.yaml` Das Volume soll in Kubernetes importiert und von Trident verwaltet werden. Dies gewährleistet eine eindeutige Volumenreferenz.

Wenn der obige Befehl ausgeführt wird, findet Trident das Volume im Backend und liest dessen Größe aus. Die konfigurierte PVC-Volumengröße wird automatisch hinzugefügt (und gegebenenfalls überschrieben). Trident erstellt dann das neue PV und Kubernetes bindet das PVC an das PV.

Wenn ein Container so eingesetzt wird, dass er die spezifische importierte PVC benötigt, bleibt er im Status "ausstehend", bis das PVC/PV-Paar über den Volumenimportprozess gebunden wird. Nach dem Verbinden der PVC/PV-Leitungen sollte sich der Behälter hochziehen lassen, sofern keine anderen Probleme auftreten.

Registrierungsdienst

Die Bereitstellung und Verwaltung des Speichers für die Registry wurde dokumentiert auf"netapp.io" im"Blog" .

Protokollierungsdienst

Wie andere OpenShift-Dienste wird auch der Protokollierungsdienst mithilfe von Ansible bereitgestellt, wobei die Konfigurationsparameter aus der Inventardatei (auch Hosts genannt) stammen, die dem Playbook übergeben wird. Es werden zwei Installationsmethoden behandelt: die Bereitstellung der Protokollierung während der Erstinstallation von OpenShift und die Bereitstellung der Protokollierung nach der Installation von OpenShift.

> Ab Red Hat OpenShift Version 3.9 rät die offizielle Dokumentation aufgrund von Bedenken hinsichtlich Datenbeschädigung von NFS für den Protokollierungsdienst ab. Dies basiert auf Tests, die Red Hat mit ihren Produkten durchgeführt hat. Der ONTAP NFS-Server hat diese Probleme nicht und kann problemlos eine Protokollierungsbereitstellung unterstützen. Letztendlich liegt die Wahl des Protokolls für den Protokollierungsdienst bei Ihnen. Sie sollten jedoch wissen, dass beide Protokolle auf NetApp -Plattformen hervorragend funktionieren und es keinen Grund gibt, NFS zu vermeiden, wenn Sie diese Option bevorzugen.

Wenn Sie NFS mit dem Protokollierungsdienst verwenden möchten, müssen Sie die Ansible-Variable festlegen. `openshift_enable_unsupported_configurations` Zu `true` um zu verhindern, dass das Installationsprogramm fehlschlägt.

Erste Schritte

Der Protokollierungsdienst kann optional sowohl für Anwendungen als auch für den Kernbetrieb des OpenShift-Clusters selbst bereitgestellt werden. Wenn Sie die Protokollierung von Vorgängen aktivieren möchten, geben Sie dazu die Variable an. `openshift_logging_use_ops` als `true` Es werden zwei Instanzen des Dienstes erstellt. Die Variablen, die die Protokollierungsinstanz für Operationen steuern,

enthalten "ops" in sich, während dies bei der Instanz für Anwendungen nicht der Fall ist.

Die Konfiguration der Ansible-Variablen entsprechend der Bereitstellungsmethode ist wichtig, um sicherzustellen, dass die zugrunde liegenden Dienste den richtigen Speicher nutzen. Schauen wir uns die Optionen für jede der Bereitstellungsmethoden an.

> Die nachfolgenden Tabellen enthalten nur die Variablen, die für die Speicherkonfiguration im Zusammenhang mit dem Protokollierungsdienst relevant sind. Weitere Optionen finden Sie in"Red Hat OpenShift-Protokollierungsdokumentation" Diese sollten entsprechend Ihrer Bereitstellung geprüft, konfiguriert und verwendet werden.

Die Variablen in der folgenden Tabelle führen dazu, dass das Ansible-Playbook anhand der angegebenen Details ein PV und ein PVC für den Logging-Service erstellt. Diese Methode ist deutlich weniger flexibel als die Verwendung des Komponenteninstallations-Playbooks nach der OpenShift-Installation. Wenn Sie jedoch bereits Volumes zur Verfügung haben, ist sie eine Option.

Variable	Details
`openshift_logging_storage_kind`	Auf einstellen `nfs` Der Installer soll ein NFS PV für den Protokollierungsdienst erstellen.
`openshift_logging_storage_host`	Der Hostname oder die IP-Adresse des NFS-Hosts. Dies sollte auf die dataLIF Ihrer virtuellen Maschine eingestellt werden.
`openshift_logging_storage_nfs_directory`	Der Mount-Pfad für den NFS-Export. Wenn beispielsweise das Volumen als verbunden wird `/openshift_logging` Diesen Pfad würden Sie für diese Variable verwenden.
`openshift_logging_storage_volume_name`	Der Name, z.B. `pv_ose_logs` , der PV zu erzeugen.
`openshift_logging_storage_volume_size`	Die Größe des NFS-Exports, zum Beispiel `100Gi` .

Wenn Ihr OpenShift-Cluster bereits läuft und Trident daher bereitgestellt und konfiguriert wurde, kann das Installationsprogramm die Volumes mithilfe der dynamischen Bereitstellung erstellen. Folgende Variablen müssen konfiguriert werden.

Variable	Details
`openshift_logging_es_pvc_dynamic`	Auf „true“ setzen, um dynamisch bereitgestellte Volumes zu verwenden.
`openshift_logging_es_pvc_storage_class_name`	Die Bezeichnung der Speicherklasse, die im PVC verwendet wird.
`openshift_logging_es_pvc_size`	Die im PVC angeforderte Volumengröße.
`openshift_logging_es_pvc_prefix`	Ein Präfix für die vom Protokollierungsdienst verwendeten PVCs.
`openshift_logging_es_ops_pvc_dynamic`	Auf einstellen `true` Um dynamisch bereitgestellte Volumes für die Ops-Protokollierungsinstanz zu verwenden.
`openshift_logging_es_ops_pvc_storage_class_name`	Der Name der Speicherklasse für die Operationsprotokollierungsinstanz.

Variable	Details
`openshift_logging_es_ops_pvc_size`	Die Größe der Volumenanforderung für die Ops-Instanz.
`openshift_logging_es_ops_pvc_prefix`	Ein Präfix für die PVCs der Operationsinstanz.

Stellen Sie den Logging-Stack bereit.

Wenn Sie die Protokollierung im Rahmen des anfänglichen OpenShift-Installationsprozesses bereitstellen, müssen Sie lediglich dem Standard-Bereitstellungsprozess folgen. Ansible konfiguriert und stellt die benötigten Dienste und OpenShift-Objekte bereit, sodass der Dienst verfügbar ist, sobald Ansible den Vorgang abgeschlossen hat.

Wenn Sie jedoch nach der Erstinstallation eine Bereitstellung durchführen, muss das Komponenten-Playbook von Ansible verwendet werden. Dieser Prozess kann sich je nach OpenShift-Version geringfügig unterscheiden. Lesen und befolgen Sie daher unbedingt die Anweisungen."Red Hat OpenShift Container Platform 3.11 Dokumentation" für Ihre Version.

Metrikdienst

Der Metrikdienst liefert dem Administrator wertvolle Informationen über den Status, die Ressourcennutzung und die Verfügbarkeit des OpenShift-Clusters. Es ist außerdem für die automatische Skalierungsfunktion der Pods erforderlich, und viele Organisationen nutzen Daten aus dem Metrikdienst für ihre Kostenverrechnungs- und/oder Anzeigeanwendungen.

Wie beim Logging-Service und OpenShift insgesamt wird auch beim Metrik-Service Ansible verwendet. Ebenso wie der Protokollierungsdienst kann auch der Metrikendienst während der Ersteinrichtung des Clusters oder nach dessen Inbetriebnahme mithilfe der Komponenteninstallationsmethode bereitgestellt werden. Die folgenden Tabellen enthalten die Variablen, die bei der Konfiguration des persistenten Speichers für den Metrikdienst wichtig sind.

> Die nachfolgenden Tabellen enthalten nur die Variablen, die für die Speicherkonfiguration im Zusammenhang mit dem Metrikdienst relevant sind. Es gibt viele weitere Optionen in der Dokumentation, die geprüft, konfiguriert und entsprechend Ihrer Bereitstellung verwendet werden sollten.

Variable	Details
`openshift_metrics_storage_kind`	Auf einstellen `nfs` Der Installer soll ein NFS PV für den Protokollierungsdienst erstellen.
`openshift_metrics_storage_host`	Der Hostname oder die IP-Adresse des NFS-Hosts. Dies sollte auf den dataLIF für Ihre SVM eingestellt werden.
`openshift_metrics_storage_nfs_directory`	Der Mount-Pfad für den NFS-Export. Wenn beispielsweise das Volumen als verbunden wird `/openshift_metrics` Diesen Pfad würden Sie für diese Variable verwenden.
`openshift_metrics_storage_volume_name`	Der Name, z.B. `pv_ose_metrics` , der PV zu erzeugen.
`openshift_metrics_storage_volume_size`	Die Größe des NFS-Exports, zum Beispiel `100Gi` .

Wenn Ihr OpenShift-Cluster bereits läuft und Trident daher bereitgestellt und konfiguriert wurde, kann das Installationsprogramm die Volumes mithilfe der dynamischen Bereitstellung erstellen. Folgende Variablen müssen konfiguriert werden.

Variable	Details
`openshift_metrics_cassandra_pvc_prefix`	Ein Präfix für die Metrik-PVCs.
`openshift_metrics_cassandra_pvc_size`	Die Größe der anzufordernden Volumina.
`openshift_metrics_cassandra_storage_type`	Der Speichertyp für Metriken muss auf dynamisch eingestellt sein, damit Ansible PVCs mit der entsprechenden Speicherklasse erstellen kann.
`openshift_metrics_cassanda_pvc_storage_class_name`	Der Name der zu verwendenden Speicherklasse.

Stellen Sie den Metrikdienst bereit.

Nachdem die entsprechenden Ansible-Variablen in Ihrer hosts/inventory-Datei definiert wurden, können Sie den Dienst mithilfe von Ansible bereitstellen. Wenn Sie die Bereitstellung zum Zeitpunkt der OpenShift-Installation durchführen, wird das PV automatisch erstellt und verwendet. Wenn Sie die Bereitstellung mithilfe der Komponenten-Playbooks durchführen, erstellt Ansible nach der OpenShift-Installation alle benötigten PVCs und stellt den Dienst bereit, nachdem Trident Speicherplatz dafür bereitgestellt hat.

Die oben genannten Variablen und der Bereitstellungsprozess können sich mit jeder Version von OpenShift ändern. Bitte überprüfen und befolgen Sie die Anweisungen."Red Hats OpenShift-Bereitstellungsleitfaden" für Ihre Version, damit sie für Ihre Umgebung konfiguriert ist.

Datenschutz und Notfallwiederherstellung

Erfahren Sie mehr über Schutz- und Wiederherstellungsoptionen für Trident und mit Trident erstellte Volumes. Sie sollten für jede Anwendung mit Persistenzanforderungen eine Datensicherungs- und -wiederherstellungsstrategie haben.

Trident -Replikation und -Wiederherstellung

Sie können eine Sicherungskopie erstellen, um Trident im Katastrophenfall wiederherzustellen.

Trident -Replikation

Trident verwendet Kubernetes CRDs, um seinen eigenen Zustand zu speichern und zu verwalten, und das Kubernetes-Cluster-etcd, um seine Metadaten zu speichern.

Schritte

1. Sichern Sie den etcd-Speicher des Kubernetes-Clusters mit"Kubernetes: Sichern eines etcd-Clusters" .
2. Platzieren Sie die Sicherungsdateien auf einem FlexVol volume.

> NetApp empfiehlt, die SVM, auf der sich FlexVol befindet, durch eine SnapMirror -Beziehung zu einer anderen SVM zu schützen.

Trident Wiederherstellung

Mithilfe von Kubernetes CRDs und dem etcd-Snapshot des Kubernetes-Clusters lässt sich Trident wiederherstellen.

Schritte

1. Vom Ziel-SVM wird das Volume, das die Kubernetes etcd-Datendateien und Zertifikate enthält, auf dem Host eingebunden, der als Master-Knoten eingerichtet werden soll.
2. Kopieren Sie alle erforderlichen Zertifikate, die sich auf den Kubernetes-Cluster beziehen, unter `/etc/kubernetes/pki` und die etcd-Mitgliedsdateien unter `/var/lib/etcd` .
3. Stellen Sie den Kubernetes-Cluster aus dem etcd-Backup wieder her, indem Sie"Kubernetes: Wiederherstellen eines etcd-Clusters" .
4. Laufen `kubectl get crd` um zu überprüfen, ob alle Trident Benutzerressourcen geladen wurden, und um die Trident -Objekte abzurufen, um zu überprüfen, ob alle Daten verfügbar sind.

SVM-Replikation und -Wiederherstellung

Trident kann keine Replikationsbeziehungen konfigurieren, der Speicheradministrator kann dies jedoch tun. "ONTAP SnapMirror" um eine SVM zu replizieren.

Im Katastrophenfall können Sie die SnapMirror Ziel-SVM aktivieren, um mit der Datenbereitstellung zu beginnen. Sie können wieder auf das primäre System umschalten, sobald die Systeme wiederhergestellt sind.

Informationen zu diesem Vorgang

Beachten Sie Folgendes bei der Verwendung der SnapMirror SVM-Replikationsfunktion:

- Sie sollten für jede SVM mit aktiviertem SVM-DR ein separates Backend erstellen.
- Konfigurieren Sie die Speicherklassen so, dass die replizierten Backends nur bei Bedarf ausgewählt werden, um zu vermeiden, dass Volumes, die keine Replikation benötigen, auf den Backends bereitgestellt werden, die SVM-DR unterstützen.
- Anwendungsadministratoren sollten sich der zusätzlichen Kosten und der Komplexität bewusst sein, die mit der Replikation verbunden sind, und ihren Wiederherstellungsplan sorgfältig prüfen, bevor sie mit diesem Prozess beginnen.

SVM-Replikation

Sie können verwenden"ONTAP: SnapMirror SVM-Replikation" um die SVM-Replikationsbeziehung herzustellen.

SnapMirror ermöglicht es Ihnen, Optionen festzulegen, um zu steuern, was repliziert werden soll. Sie müssen wissen, welche Optionen Sie bei der Durchführung ausgewählt haben.SVM-Wiederherstellung mit Trident .

- "-identity-reserve true"repliziert die gesamte SVM-Konfiguration.
- "-discard-configs Netzwerk"Ausgenommen sind LIFs und zugehörige Netzwerkeinstellungen.
- "-identity-reserve false"Es werden lediglich die Volumes und die Sicherheitskonfiguration repliziert.

SVM-Wiederherstellung mit Trident

Trident erkennt SVM-Fehler nicht automatisch. Im Katastrophenfall kann der Administrator manuell ein Trident Failover auf die neue SVM einleiten.

Schritte

1. Brechen Sie geplante und laufende SnapMirror Übertragungen ab, trennen Sie die Replikationsbeziehung, stoppen Sie die Quell-SVM und aktivieren Sie anschließend die SnapMirror Ziel-SVM.
2. Wenn Sie angegeben haben `-identity-preserve false` oder `-discard-config network` Aktualisieren Sie bei der Konfiguration Ihrer SVM-Replikation die `managementLIF` Und `dataLIF` in der Trident Backend-Definitionsdatei.
3. Bestätigen `storagePrefix` ist in der Trident Backend-Definitionsdatei vorhanden. Dieser Parameter kann nicht geändert werden. Auslassen `storagePrefix` wird dazu führen, dass das Backend-Update fehlschlägt.
4. Aktualisieren Sie alle erforderlichen Backends, um den neuen Ziel-SVM-Namen widerzuspiegeln, indem Sie Folgendes verwenden:

```
./tridentctl update backend <backend-name> -f <backend-json-file> -n
<namespace>
```

5. Wenn Sie angegeben haben `-identity-preserve false` oder `discard-config network` Sie müssen alle Anwendungs-Pods neu starten.

> Wenn Sie angegeben haben `-identity-preserve true` Alle von Trident bereitgestellten Volumes beginnen mit der Datenbereitstellung, sobald die Ziel-SVM aktiviert ist.

Volumenreplikation und Wiederherstellung

Trident kann keine SnapMirror Replikationsbeziehungen konfigurieren, der Speicheradministrator kann jedoch Folgendes verwenden:"ONTAP SnapMirror Replikation und -Wiederherstellung" um von Trident erstellte Volumes zu replizieren.

Anschließend können Sie die wiederhergestellten Volumes in Trident importieren."tridentctl volume import" .

> Der Import wird nicht unterstützt auf `ontap-nas-economy` , `ontap-san-economy` , oder `ontap-flexgroup-economy` Fahrer.

Snapshot-Datenschutz

Sie können Daten schützen und wiederherstellen mit:

- Ein externer Snapshot-Controller und CRDs zur Erstellung von Kubernetes-Volume-Snapshots von Persistent Volumes (PVs).

  "Volumen-Snapshots"

- ONTAP Snapshots dienen zur Wiederherstellung des gesamten Inhalts eines Volumes oder zur Wiederherstellung einzelner Dateien oder LUNs.

  "ONTAP Schnappschüsse"

Sicherheit

Sicherheit

Befolgen Sie die hier aufgeführten Empfehlungen, um eine sichere Installation Ihres Trident zu gewährleisten.

Trident in einem eigenen Namensraum ausführen

Um eine zuverlässige Speicherung zu gewährleisten und potenziell schädliche Aktivitäten zu verhindern, ist es wichtig, Anwendungen, Anwendungsadministratoren, Benutzer und Verwaltungsanwendungen den Zugriff auf Trident -Objektdefinitionen oder die Pods zu verwehren.

Um andere Anwendungen und Benutzer von Trident zu trennen, installieren Sie Trident immer in einem eigenen Kubernetes-Namespace.(`trident`). Durch die Platzierung von Trident in einem eigenen Namespace wird sichergestellt, dass nur das Kubernetes-Administratorpersonal Zugriff auf den Trident -Pod und die in den Namespace-CRD-Objekten gespeicherten Artefakte (wie z. B. Backend- und CHAP-Secrets, falls zutreffend) hat. Sie sollten sicherstellen, dass nur Administratoren Zugriff auf den Trident -Namensraum und somit auf die entsprechenden Funktionen haben. `tridentctl` Anwendung.

CHAP-Authentifizierung mit ONTAP SAN-Backends verwenden

Trident unterstützt die CHAP-basierte Authentifizierung für ONTAP -SAN-Workloads (unter Verwendung der `ontap-san` Und `ontap-san-economy` Fahrer). NetApp empfiehlt die Verwendung von bidirektionalem CHAP mit Trident zur Authentifizierung zwischen einem Host und dem Speicher-Backend.

Für ONTAP Backends, die SAN-Speichertreiber verwenden, kann Trident bidirektionales CHAP einrichten und CHAP-Benutzernamen und -Geheimnisse verwalten. `tridentctl` . Siehe"Bereiten Sie die Konfiguration des Backends mit ONTAP SAN-Treibern vor." um zu verstehen, wie Trident CHAP auf ONTAP Backends konfiguriert.

CHAP-Authentifizierung mit NetApp HCI und SolidFire -Backends verwenden

NetApp empfiehlt den Einsatz von bidirektionalem CHAP, um die Authentifizierung zwischen einem Host und den NetApp HCI und SolidFire -Backends sicherzustellen. Trident verwendet ein geheimes Objekt, das zwei CHAP-Passwörter pro Mandant enthält. Nach der Installation von Trident verwaltet es die CHAP-Geheimnisse und speichert sie in einem `tridentvolume` CR-Objekt für das jeweilige PV. Wenn Sie ein PV erstellen, verwendet Trident die CHAP-Secrets, um eine iSCSI-Sitzung zu initiieren und über CHAP mit dem NetApp HCI und SolidFire -System zu kommunizieren.

Die von Trident erstellten Volumes sind keiner Volume-Zugriffsgruppe zugeordnet.

Verwenden Sie Trident mit NVE und NAE

NetApp ONTAP bietet Datenverschlüsselung im Ruhezustand, um sensible Daten für den Fall zu schützen, dass eine Festplatte gestohlen, zurückgegeben oder anderweitig verwendet wird. Weitere Einzelheiten finden Sie unter"Übersicht über die Konfiguration der NetApp Volume-Verschlüsselung" .

- Wenn NAE im Backend aktiviert ist, wird jedes in Trident bereitgestellte Volume NAE-fähig sein.
  - Sie können das NVE-Verschlüsselungsflag auf „“ setzen. `""` um NAE-fähige Volumes zu erstellen.
- Wenn NAE im Backend nicht aktiviert ist, wird jedes in Trident bereitgestellte Volume NVE-fähig sein, es

sei denn, das NVE-Verschlüsselungsflag ist auf „true“ gesetzt. `false` (der Standardwert) in der Backend-Konfiguration.

> In Trident auf einem NAE-fähigen Backend erstellte Volumes müssen NVE- oder NAE-verschlüsselt sein.
>
> - Sie können das NVE-Verschlüsselungsflag auf „“ setzen. `true` in der Trident -Backend -Konfiguration, um die NAE-Verschlüsselung zu überschreiben und einen spezifischen Verschlüsselungsschlüssel pro Volume zu verwenden.
> - Setzen des NVE-Verschlüsselungsflags auf `false` Auf einem NAE-fähigen Backend wird ein NAE-fähiges Volume erstellt. Die NAE-Verschlüsselung kann nicht durch Setzen des NVE-Verschlüsselungsflags deaktiviert werden. `false` .

- Sie können in Trident manuell ein NVE-Volume erstellen, indem Sie das NVE-Verschlüsselungsflag explizit setzen. `true` .

Weitere Informationen zu den Backend-Konfigurationsoptionen finden Sie unter:

- "ONTAP SAN-Konfigurationsoptionen"
- "ONTAP NAS-Konfigurationsoptionen"

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 von Passphrasen und die Volumenerweiterung für LUKS-verschlüsselte Volumes.

In Trident verwenden LUKS-verschlüsselte Volumes die aes-xts-plain64-Verschlüsselung und den entsprechenden Modus, wie von empfohlen."NIST" .

> LUKS-Verschlüsselung wird für ASA r2-Systeme nicht unterstützt. Informationen zu ASA r2-Systemen finden Sie unter"Erfahren Sie mehr über ASA R2-Speichersysteme" .

Bevor Sie beginnen

- Auf den Worker-Knoten muss cryptsetup 2.1 oder höher (aber niedriger als 3.0) installiert sein. Weitere Informationen finden Sie unter"GitLab: cryptsetup" .
- Aus Performancegründen empfiehlt NetApp , dass Worker-Knoten den Advanced Encryption Standard New Instructions (AES-NI) unterstützen. Um die AES-NI-Unterstützung zu überprüfen, führen Sie folgenden Befehl aus:

```
grep "aes" /proc/cpuinfo
```

Wenn keine Antwort zurückgegeben wird, unterstützt Ihr Prozessor AES-NI nicht. Weitere Informationen zu AES-NI finden Sie unter:"Intel: Advanced Encryption Standard Instructions (AES-NI)" .

LUKS-Verschlüsselung aktivieren

Sie können die volumenbezogene, hostseitige Verschlüsselung mithilfe von Linux Unified Key Setup (LUKS) für ONTAP SAN- und ONTAP SAN ECONOMY-Volumes aktivieren.

Schritte

1. Definieren Sie die LUKS-Verschlüsselungsattribute in der Backend-Konfiguration. Weitere Informationen zu den Backend-Konfigurationsoptionen für ONTAP SAN finden Sie unter"ONTAP SAN-Konfigurationsoptionen" .

```
{
  "storage": [
    {
      "labels": {
        "luks": "true"
      },
      "zone": "us_east_1a",
      "defaults": {
        "luksEncryption": "true"
      }
    },
    {
      "labels": {
        "luks": "false"
      },
      "zone": "us_east_1a",
      "defaults": {
        "luksEncryption": "false"
      }
    }
  ]
}
```

2. Verwenden `parameters.selector` Die Speicherpools werden mithilfe der LUKS-Verschlüsselung definiert. 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 Datenträger können die Deduplizierung und Komprimierung von ONTAP nicht nutzen.

Backend-Konfiguration für den Import von LUKS-Volumes

Um ein LUKS-Volume zu importieren, müssen Sie Folgendes einstellen: `luksEncryption` Zu(`true` im Backend. Der `luksEncryption` Diese Option teilt Trident mit, ob das Volume LUKS-kompatibel 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, legen Sie die Annotation fest. `trident.netapp.io/luksEncryption` Zu `true` und eine LUKS-fähige Speicherklasse in die PVC einbinden, 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
```

LUKS-Passphrase rotieren

Sie können die LUKS-Passphrase ändern und die Änderung bestätigen.

> Vergessen Sie eine Passphrase erst dann, wenn Sie sichergestellt haben, dass sie von keinem Volume, Snapshot oder Geheimnis mehr referenziert wird. Geht die angegebene Passphrase verloren, kann das Volume möglicherweise nicht eingebunden werden und die Daten bleiben verschlüsselt und unzugänglich.

Informationen zu diesem Vorgang

Eine LUKS-Passphrase-Rotation tritt auf, wenn ein Pod, der das Volume einbindet, erstellt wird, nachdem eine neue LUKS-Passphrase angegeben wurde. Wenn ein neuer Pod erstellt wird, vergleicht Trident die LUKS-Passphrase des Volumes mit der aktiven Passphrase im Secret.

- Stimmt die Passphrase des Volumes nicht mit der aktiven Passphrase im Geheimnis überein, findet eine Rotation statt.
- Wenn die Passphrase des Volumes mit der aktiven Passphrase im Geheimnis übereinstimmt, `previous-luks-passphrase` Der Parameter wird ignoriert.

Schritte

1. Füge die `node-publish-secret-name` Und `node-publish-secret-namespace` StorageClass-Parameter. 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 Passphrasen auf dem Volume oder Snapshot.

Volumen

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames:["A"]
```

Schnappschuss

```
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 neue und die vorherige Passphrase anzugeben. Sicherstellen `previous-luke-passphrase-name` Und `previous-luks-passphrase` Die vorherige Passphrase muss übereinstimmen.

```
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. Erstelle einen neuen Pod, der das Volume einbindet. Dies ist erforderlich, um die Rotation einzuleiten.
5. Überprüfen Sie, ob die Passphrase geändert wurde.

Volumen

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames:["B"]
```

Schnappschuss

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames:["B"]
```

Ergebnisse

Die Passphrase wurde geändert, wenn auf dem Volume und im Snapshot nur noch die neue Passphrase angezeigt wird.

> Wenn zwei Passphrasen zurückgegeben werden, zum Beispiel `luksPassphraseNames: ["B", "A"]` Die Rotation ist unvollständig. Sie können eine neue Kapsel auslösen, um zu versuchen, die Rotation abzuschließen.

Volumenerweiterung aktivieren

Sie können die Volumenerweiterung auf einem LUKS-verschlüsselten Volumen aktivieren.

Schritte

1. Aktivieren Sie die `CSINodeExpandSecret` Feature Gate (Beta 1,25+). Siehe "Kubernetes 1.25: Verwendung von Secrets zur knotengesteuerten Erweiterung von CSI-Volumes" für Details.
2. Füge die `node-expand-secret-name` Und `node-expand-secret-namespace` StorageClass-Parameter. 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 starten, übergibt das Kubelet die entsprechenden Anmeldeinformationen an den Treiber.

Kerberos-Verschlüsselung während des Fluges

Durch die Verwendung von Kerberos-Verschlüsselung während der Datenübertragung können Sie die Datensicherheit verbessern, indem Sie die Verschlüsselung für den Datenverkehr zwischen Ihrem verwalteten Cluster und dem Speicher-Backend aktivieren.

Trident unterstützt Kerberos-Verschlüsselung für ONTAP als Speicher-Backend:

- **On-Premise ONTAP** - Trident unterstützt die Kerberos-Verschlüsselung über NFSv3- und NFSv4-Verbindungen von Red Hat OpenShift und Upstream-Kubernetes-Clustern zu On-Premise ONTAP Volumes.

Sie können Volumes erstellen, löschen, ihre Größe ändern, Snapshots erstellen, klonen, schreibgeschützte Klone erstellen und importieren, die NFS-Verschlüsselung verwenden.

Konfigurieren der Kerberos-Verschlüsselung während der Übertragung mit lokalen ONTAP -Volumes

Sie können die Kerberos-Verschlüsselung für den Speicherdatenverkehr zwischen Ihrem verwalteten Cluster und einem lokalen ONTAP -Speicher-Backend aktivieren.

> Die Kerberos-Verschlüsselung für NFS-Datenverkehr mit lokalen ONTAP -Speicher-Backends wird nur mit folgender Konfiguration unterstützt: `ontap-nas` Speichertreiber.

Bevor Sie beginnen

- Stellen Sie sicher, dass Sie Zugriff auf die `tridentctl` Dienstprogramm.
- Stellen Sie sicher, dass Sie über Administratorzugriff auf das ONTAP -Speicher-Backend verfügen.
- Stellen Sie sicher, dass Sie den Namen des oder der Volumes kennen, die Sie vom ONTAP -Speicher -Backend freigeben werden.
- Stellen Sie sicher, dass Sie die ONTAP -Speicher-VM für die Unterstützung der Kerberos-Verschlüsselung für NFS-Volumes vorbereitet haben. Siehe "Kerberos auf einem dataLIF aktivieren" für Anweisungen.
- Stellen Sie sicher, dass alle NFSv4-Volumes, die Sie mit Kerberos-Verschlüsselung verwenden, korrekt konfiguriert sind. Siehe Abschnitt „NetApp NFSv4-Domänenkonfiguration“ (Seite 13) des "NetApp NFSv4-Erweiterungen und Best Practices-Leitfaden" .

ONTAP Exportrichtlinien hinzufügen oder ändern

Sie müssen bestehenden ONTAP Exportrichtlinien Regeln hinzufügen oder neue Exportrichtlinien erstellen, die die Kerberos-Verschlüsselung für das Root-Volume der ONTAP Speicher-VM sowie für alle ONTAP -Volumes unterstützen, die mit dem Upstream-Kubernetes-Cluster geteilt werden. Die von Ihnen hinzugefügten Exportrichtlinienregeln oder die von Ihnen erstellten neuen Exportrichtlinien müssen die folgenden Zugriffsprotokolle und Zugriffsberechtigungen unterstützen:

Zugriffsprotokolle

Konfigurieren Sie die Exportrichtlinie mit den Zugriffsprotokollen NFS, NFSv3 und NFSv4.

Zugangsdaten

Je nach Ihren Anforderungen an das Volume können Sie eine von drei verschiedenen Versionen der Kerberos-Verschlüsselung konfigurieren:

- **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 Exportrichtlinienregel mit den entsprechenden Zugriffsberechtigungen. Wenn Cluster beispielsweise die NFS-Volumes mit einer Mischung aus Kerberos 5i- und Kerberos 5p-Verschlüsselung einbinden, verwenden Sie die folgenden Zugriffseinstellungen:

Typ	Nur-Lese-Zugriff	Lese-/Schreibzugriff	Superuser-Zugriff
UNIX	Ermöglicht	Ermöglicht	Ermöglicht
Kerberos 5i	Ermöglicht	Ermöglicht	Ermöglicht
Kerberos 5p	Ermöglicht	Ermöglicht	Ermöglicht

In der folgenden Dokumentation finden Sie Informationen zum Erstellen von ONTAP Exportrichtlinien und Exportrichtlinienregeln:

- "Erstellen einer Exportrichtlinie"
- "Hinzufügen einer Regel zu einer Exportrichtlinie"

Erstellen Sie ein Speicher-Backend

Sie können eine Trident -Speicher-Backend-Konfiguration erstellen, die die Kerberos-Verschlüsselungsfunktion beinhaltet.

Informationen zu diesem Vorgang

Wenn Sie eine Konfigurationsdatei für das Speicher-Backend erstellen, die die Kerberos-Verschlüsselung konfiguriert, können Sie mithilfe der folgenden Optionen eine von drei verschiedenen Versionen der Kerberos-Verschlüsselung angeben: `spec.nfsMountOptions` Parameter:

- `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-Verschlüsselungsstufe angeben, wird nur die erste Option verwendet.

Schritte

1. Erstellen Sie auf dem verwalteten Cluster eine Speicher-Backend-Konfigurationsdatei anhand des folgenden Beispiels. Ersetzen Sie die Werte in eckigen 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 im vorherigen Schritt erstellte Konfigurationsdatei, um das Backend zu erstellen:

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

Wenn die Backend-Erstellung fehlschlägt, stimmt etwas mit der Backend-Konfiguration nicht. Sie können die Protokolle einsehen, um die Ursache zu ermitteln, indem Sie folgenden Befehl ausführen:

```
tridentctl logs
```

Nachdem Sie das Problem mit der Konfigurationsdatei identifiziert und behoben haben, können Sie den Befehl zum Erstellen erneut ausführen.

Erstellen einer Speicherklasse

Sie können eine Speicherklasse erstellen, um Volumes mit Kerberos-Verschlüsselung bereitzustellen.

Informationen zu diesem Vorgang

Beim Erstellen eines Speicherklassenobjekts können Sie mithilfe der Kerberos-Verschlüsselungsmethode eine von drei verschiedenen Versionen der Kerberos-Verschlüsselung angeben. `mountOptions` Parameter:

- `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-Verschlüsselungsstufe angeben, wird nur die erste Option verwendet. Wenn der in der Speicher-Backend-Konfiguration angegebene Verschlüsselungsgrad von dem im Speicherklassenobjekt angegebenen Grad abweicht, hat das Speicherklassenobjekt Vorrang.

Schritte

1. Erstellen Sie ein StorageClass-Kubernetes-Objekt anhand des folgenden Beispiels:

```
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. Erstellen Sie die Speicherklasse:

```
kubectl create -f sample-input/storage-class-ontap-nas-sc.yaml
```

3. Stellen Sie sicher, dass die Speicherklasse erstellt wurde:

```
kubectl get sc ontap-nas-sc
```

   Die Ausgabe sollte in etwa wie folgt aussehen:

```
NAME          PROVISIONER             AGE
ontap-nas-sc    csi.trident.netapp.io   15h
```

Bereitstellungsmengen

Nachdem Sie ein Speicher-Backend und eine Speicherklasse erstellt haben, können Sie nun ein Volume bereitstellen. Anweisungen hierzu finden Sie unter "Bereitstellung eines Volumens" .

Konfigurieren der Kerberos-Verschlüsselung während der Übertragung mit Azure NetApp Files -Volumes

Sie können die Kerberos-Verschlüsselung für den Speicherdatenverkehr zwischen Ihrem verwalteten Cluster und einem einzelnen Azure NetApp Files -Speicher-Backend oder einem virtuellen Pool von Azure NetApp Files -Speicher-Backends 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 die `tridentctl` Dienstprogramm.
- Stellen Sie sicher, dass Sie das Azure NetApp Files -Speicher-Backend für die Kerberos-Verschlüsselung vorbereitet haben, indem Sie die Anforderungen beachten und die Anweisungen in [Link einfügen] befolgen. "Azure NetApp Files Dokumentation" .
- Stellen Sie sicher, dass alle NFSv4-Volumes, die Sie mit Kerberos-Verschlüsselung verwenden, korrekt konfiguriert sind. Siehe Abschnitt „NetApp NFSv4-Domänenkonfiguration“ (Seite 13) des "NetApp NFSv4-Erweiterungen und Best Practices-Leitfaden" .

Erstellen Sie ein Speicher-Backend

Sie können eine Azure NetApp Files -Speicher-Backend-Konfiguration erstellen, die die Kerberos-Verschlüsselungsfunktion beinhaltet.

Informationen zu diesem Vorgang

Wenn Sie eine Konfigurationsdatei für das Speicher-Backend erstellen, die die Kerberos-Verschlüsselung konfiguriert, können Sie festlegen, dass diese auf einer von zwei möglichen Ebenen angewendet werden soll:

- Die **Speicher-Backend-Ebene** unter Verwendung der `spec.kerberos` Feld
- Die **virtuelle Poolebene** unter Verwendung der `spec.storage.kerberos` Feld

Wenn Sie die Konfiguration auf Ebene des virtuellen Pools definieren, wird der Pool anhand der Bezeichnung 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 Storage-Backend-Konfigurationsdatei anhand eines der folgenden Beispiele, je nachdem, wo Sie das Storage-Backend definieren müssen (Storage-Backend-Ebene oder Virtual-Pool-Ebene). Ersetzen Sie die Werte in eckigen Klammern <> durch Informationen aus Ihrer Umgebung:

Beispiel auf Speicher-Backend-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 für ein virtuelles Schwimmbecken

```
---
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 im vorherigen Schritt erstellte Konfigurationsdatei, um das Backend zu erstellen:

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

Wenn die Backend-Erstellung fehlschlägt, stimmt etwas mit der Backend-Konfiguration nicht. Sie können die Protokolle einsehen, um die Ursache zu ermitteln, indem Sie folgenden Befehl ausführen:

```
tridentctl logs
```

Nachdem Sie das Problem mit der Konfigurationsdatei identifiziert und behoben haben, können Sie den Befehl zum Erstellen erneut ausführen.

Erstellen einer Speicherklasse

Sie können eine Speicherklasse erstellen, um Volumes mit Kerberos-Verschlüsselung bereitzustellen.

Schritte

1. Erstellen Sie ein StorageClass-Kubernetes-Objekt anhand des folgenden Beispiels:

```
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. Erstellen Sie die Speicherklasse:

```
kubectl create -f sample-input/storage-class-sc-nfs.yaml
```

3. Stellen Sie sicher, dass die Speicherklasse erstellt wurde:

```
kubectl get sc -sc-nfs
```

Die Ausgabe sollte in etwa wie folgt aussehen:

```
NAME          PROVISIONER             AGE
sc-nfs        csi.trident.netapp.io   15h
```

Bereitstellungsmengen

Nachdem Sie ein Speicher-Backend und eine Speicherklasse erstellt haben, können Sie nun ein Volume bereitstellen. Anweisungen hierzu finden Sie unter "Bereitstellung eines Volumens" .

Schützen Sie Ihre Anwendungen mit Trident Protect.

Erfahren Sie mehr über Trident Protect.

NetApp Trident Protect bietet erweiterte Funktionen zur Anwendungsdatenverwaltung, die die Funktionalität und Verfügbarkeit von Stateful-Kubernetes-Anwendungen verbessern, die von NetApp ONTAP Speichersystemen und dem NetApp Trident CSI-Speicherbereitsteller unterstützt werden. Trident Protect vereinfacht die Verwaltung, den Schutz und die Verschiebung von Container-Workloads über öffentliche Clouds und lokale Umgebungen hinweg. Es bietet außerdem 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.

Wie geht es weiter?

Sie können sich vor der Installation über die Anforderungen an Trident Protect informieren:

- "Trident Schutzanforderungen"

Installieren Sie Trident Protect.

Trident Schutzanforderungen

Beginnen Sie mit der Überprüfung der Einsatzbereitschaft Ihrer Betriebsumgebung, Anwendungscluster, Anwendungen und Lizenzen. Stellen Sie sicher, dass Ihre Umgebung diese Anforderungen erfüllt, um Trident Protect einsetzen und betreiben zu können.

Trident schützt die Kompatibilität von Kubernetes-Clustern

Trident Protect ist mit einer Vielzahl von vollständig verwalteten und selbstverwalteten Kubernetes-Angeboten kompatibel, darunter:

- Amazon Elastic Kubernetes Service (EKS)
- Google Kubernetes Engine (GKE)
- Microsoft Azure Kubernetes Service (AKS)
- Red Hat OpenShift
- SUSE Rancher
- VMware Tanzu Portfolio
- Upstream Kubernetes

- Trident Protect-Backups werden nur auf Linux-Rechenknoten unterstützt. Windows-Rechenknoten werden für Sicherungsvorgänge nicht unterstützt.
- Stellen Sie sicher, dass der Cluster, auf dem Sie Trident Protect installieren, mit einem laufenden Snapshot-Controller und den zugehörigen CRDs konfiguriert ist. Informationen zur Installation eines Snapshot-Controllers finden Sie unter "diese Anweisungen" .

Kompatibilität des Trident Protect-Speicher-Backends

Trident Protect unterstützt die folgenden Speichersysteme:

- Amazon FSx for NetApp ONTAP
- Cloud Volumes ONTAP
- ONTAP Speicherarrays
- Google Cloud NetApp Volumes
- Azure NetApp Files

Stellen Sie sicher, dass Ihr Speichersystem die folgenden Anforderungen erfüllt:

- Stellen Sie sicher, dass auf dem mit dem Cluster verbundenen NetApp Speicher Trident 24.02 oder neuer verwendet wird (Trident 24.10 wird empfohlen).
- Stellen Sie sicher, dass Sie über ein NetApp ONTAP -Speicher-Backend verfügen.
- Stellen Sie sicher, dass Sie einen Objektspeicher-Bucket für die Speicherung von Backups konfiguriert haben.
- Erstellen Sie alle Anwendungs-Namespaces, die Sie für Anwendungen oder Anwendungsdatenverwaltungsvorgänge verwenden möchten. Trident protect erstellt diese Namespaces nicht für Sie; wenn Sie in einer benutzerdefinierten Ressource einen nicht existierenden Namespace angeben, schlägt der Vorgang fehl.

Anforderungen an die Volumina der Nas-Economy

Trident Protect unterstützt Backup- und Wiederherstellungsvorgänge auf NAS-Economy-Volumes. Snapshots, Klone und die SnapMirror Replikation auf nas-economy-Volumes werden derzeit nicht unterstützt. Sie müssen für jedes NAS-Economy-Volume, das Sie mit Trident Protect verwenden möchten, ein Snapshot-Verzeichnis aktivieren.

Manche Anwendungen sind nicht kompatibel mit Volumes, 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 dabei Folgendes ersetzen: `<volume-UUID>` mit der UUID des Volumes, 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 entsprechende Option in der Trident Backend-Konfiguration festlegen. `snapshotDir` Zu `true` . Bestehende Volumina bleiben unberührt.

Datenschutz mit KubeVirt-VMs

Trident Protect 24.10 und 24.10.1 sowie neuere Versionen verhalten sich unterschiedlich, wenn Sie Anwendungen schützen, die auf KubeVirt VMs ausgeführt werden. Bei beiden Versionen können Sie das Einfrieren und Auftauen des Dateisystems während Datensicherungsvorgängen aktivieren oder deaktivieren.

Während der Wiederherstellungsvorgänge `VirtualMachineSnapshots` Für eine virtuelle Maschine (VM) erstellte Daten werden nicht wiederhergestellt.

Trident 24.10

Trident Protect 24.10 gewährleistet nicht automatisch einen konsistenten Zustand der KubeVirt VM-Dateisysteme während Datensicherungsvorgängen. Wenn Sie Ihre KubeVirt VM-Daten mit Trident protect 24.10 schützen möchten, müssen Sie die Freeze/Unfreeze-Funktionalität für die Dateisysteme vor dem Datensicherungsvorgang 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 Auftauen des VM-Dateisystems während Datensicherungsvorgängen verwaltet wird, indem"Virtualisierung konfigurieren" und anschließend mit folgendem Befehl:

```
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 Datensicherungsvorgängen automatisch ein und wieder auf. Optional können Sie dieses automatische Verhalten mit folgendem Befehl deaktivieren:

```
kubectl set env deployment/trident-protect-controller-manager
NEPTUNE_VM_FREEZE=false -n trident-protect
```

Anforderungen für die SnapMirror Replikation

Die NetApp SnapMirror Replikation ist für die Verwendung mit Trident Protect für die folgenden ONTAP -Lösungen verfügbar:

- Lokale NetApp FAS, AFF und ASA Cluster
- NetApp ONTAP Select
- NetApp Cloud Volumes ONTAP

- Amazon FSx for NetApp ONTAP

ONTAP Clusteranforderungen für die SnapMirror Replikation

Stellen Sie sicher, dass Ihr ONTAP -Cluster die folgenden Anforderungen erfüllt, wenn Sie die SnapMirror Replikation verwenden möchten:

- * NetApp Trident*: NetApp Trident muss sowohl auf dem Quell- als auch auf dem Ziel-Kubernetes-Cluster vorhanden sein, die ONTAP als Backend verwenden. Trident Protect unterstützt die Replikation mit der NetApp SnapMirror -Technologie unter Verwendung von Speicherklassen, die von den folgenden Treibern unterstützt werden:
  - `ontap-nas`: NFS
  - `ontap-san`: iSCSI
  - `ontap-san`FC
  - `ontap-san`NVMe/TCP (erfordert mindestens ONTAP Version 9.15.1)
- **Lizenzen**: ONTAP SnapMirror -Asynchronlizenzen, die das Data Protection-Bundle verwenden, müssen sowohl auf dem Quell- als auch auf dem Ziel ONTAP -Cluster aktiviert sein. Siehe "SnapMirror -Lizenzübersicht in ONTAP" für weitere Informationen.

  Ab ONTAP 9.10.1 werden alle Lizenzen als NetApp Lizenzdatei (NLF) ausgeliefert. Dabei handelt es sich um eine einzelne Datei, die mehrere Funktionen aktiviert. Siehe"In ONTAP One enthaltene Lizenzen" für weitere Informationen.

  > Es wird ausschließlich der asynchrone Schutz von SnapMirror unterstützt.

Peering-Überlegungen für die SnapMirror Replikation

Stellen Sie sicher, dass Ihre Umgebung die folgenden Anforderungen erfüllt, wenn Sie Storage-Backend-Peering nutzen möchten:

- **Cluster und SVM**: Die ONTAP -Speicher-Backends müssen per Peering verbunden sein. Siehe "Cluster- und SVM-Peering-Übersicht" für weitere Informationen.

  > Stellen Sie sicher, dass die in der Replikationsbeziehung zwischen zwei ONTAP Clustern verwendeten SVM-Namen eindeutig sind.

- * NetApp Trident und SVM*: Die verbundenen Remote-SVMs müssen für NetApp Trident auf dem Zielcluster verfügbar sein.
- **Verwaltete Backends**: Sie müssen ONTAP -Speicher-Backends in Trident Protect hinzufügen und verwalten, um eine Replikationsbeziehung herzustellen.

Trident / ONTAP Konfiguration für SnapMirror Replikation

Trident Protect erfordert, dass Sie mindestens ein Storage-Backend konfigurieren, das die Replikation sowohl für den Quell- als auch für den Zielcluster unterstützt. Wenn Quell- und Zielcluster identisch sind, sollte die Zielanwendung für eine optimale Ausfallsicherheit ein anderes Speichersystem als die Quellanwendung verwenden.

Anforderungen an einen Kubernetes-Cluster für die SnapMirror Replikation

Stellen Sie sicher, dass Ihre Kubernetes-Cluster die folgenden Anforderungen erfüllen:

- **AppVault-Zugänglichkeit**: Sowohl der Quell- als auch der Zielcluster müssen über Netzwerkzugriff verfügen, um für die Replikation von Anwendungsobjekten aus dem AppVault lesen und in diesen schreiben zu können.
- **Netzwerkanbindung**: Konfigurieren Sie Firewall-Regeln, Bucket-Berechtigungen und IP-Zulassungslisten, um die Kommunikation zwischen beiden Clustern und dem AppVault über WANs hinweg zu ermöglichen.

> In vielen Unternehmensumgebungen werden strenge Firewall-Richtlinien für WAN-Verbindungen implementiert. Klären Sie diese Netzwerkanforderungen mit Ihrem Infrastrukturteam ab, bevor Sie die Replikation konfigurieren.

Trident Protect installieren und konfigurieren

Wenn Ihre Umgebung die Anforderungen für Trident protect erfüllt, können Sie die folgenden Schritte befolgen, um Trident protect auf Ihrem Cluster zu installieren. Sie können Trident Protect von NetApp beziehen oder es aus Ihrer eigenen privaten Registry installieren. Die Installation aus einer privaten Registry ist hilfreich, wenn Ihr Cluster keinen Internetzugang hat.

Installieren Sie Trident Protect.

Installieren Sie Trident Protect von NetApp

Schritte

1. Fügen Sie das Trident Helm-Repository hinzu:

```
helm repo add netapp-trident-protect
https://netapp.github.io/trident-protect-helm-chart
```

2. Verwenden Sie Helm, um Trident Protect zu installieren. Ersetzen `<name-of-cluster>` mit einem Clusternamen, der dem Cluster zugewiesen wird und zur Identifizierung der Backups und Snapshots des Clusters verwendet wird:

```
helm install trident-protect netapp-trident-protect/trident-protect
--set clusterName=<name-of-cluster> --version 100.2506.0 --create
-namespace --namespace trident-protect
```

Installieren Sie Trident Protect aus einer privaten Registry.

Sie können Trident Protect aus einer privaten Image-Registry installieren, wenn Ihr Kubernetes-Cluster keinen Zugriff auf das Internet hat. Ersetzen Sie in diesen Beispielen die Werte in Klammern durch Informationen aus Ihrer Umgebung:

Schritte

1. Laden Sie die folgenden Images auf Ihren lokalen Rechner herunter, aktualisieren Sie die Tags und laden Sie sie anschließend in Ihre private Registry hoch:

```
netapp/controller:25.06.0
netapp/restic:25.06.0
netapp/kopia:25.06.0
netapp/trident-autosupport:25.06.0
netapp/exechook:25.06.0
netapp/resourcebackup:25.06.0
netapp/resourcerestore:25.06.0
netapp/resourcedelete:25.06.0
bitnami/kubectl:1.30.2
kubebuilder/kube-rbac-proxy:v0.16.0
```

   Beispiel:

```
docker pull netapp/controller:25.06.0
```

```
docker tag netapp/controller:25.06.0 <private-registry-
url>/controller:25.06.0
```

```
docker push <private-registry-url>/controller:25.06.0
```

2. Erstellen Sie den Trident Protect-Systemnamensraum:

```
kubectl create ns trident-protect
```

3. Melden Sie sich bei der Registry an:

```
helm registry login <private-registry-url> -u <account-id> -p <api-token>
```

4. Erstellen Sie ein Pull-Secret zur Verwendung für die private Registry-Authentifizierung:

```
kubectl create secret docker-registry regcred --docker-username=<registry-username> --docker-password=<api-token> -n trident-protect --docker-server=<private-registry-url>
```

5. Fügen Sie das Trident Helm-Repository hinzu:

```
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. Verwenden Sie Helm, um Trident Protect zu installieren. Ersetzen `<name_of_cluster>` mit einem Clusternamen, der dem Cluster zugewiesen wird und zur Identifizierung der Backups und Snapshots des Clusters verwendet wird:

```
helm install trident-protect netapp-trident-protect/trident-protect
--set clusterName=<name_of_cluster> --version 100.2506.0 --create
-namespace --namespace trident-protect -f protectValues.yaml
```

Installieren Sie das Trident Protect CLI-Plugin.

Sie können das Trident Protect-Befehlszeilen-Plugin verwenden, das eine Erweiterung von Trident ist. `tridentctl` Hilfsprogramm zum Erstellen und Interagieren mit Trident Protect Custom Resources (CRs).

Installieren Sie das Trident Protect CLI-Plugin.

Bevor Sie das Befehlszeilenprogramm verwenden können, müssen Sie es auf dem Rechner installieren, mit dem Sie auf Ihren Cluster zugreifen. Befolgen Sie diese Schritte, je nachdem, ob Ihr Rechner eine x64- oder eine ARM CPU verwendet.

Plugin für Linux AMD64-CPUs herunterladen

Schritte

1. Laden Sie das Trident Protect CLI-Plugin herunter:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-linux-amd64
```

Plugin für Linux ARM64-CPUs herunterladen

Schritte

1. Laden Sie das Trident Protect CLI-Plugin herunter:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-linux-arm64
```

Plugin für Mac AMD64-CPUs herunterladen

Schritte

1. Laden Sie das Trident Protect CLI-Plugin herunter:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-macos-amd64
```

Plugin für Mac ARM64-CPUs herunterladen

Schritte

1. Laden Sie das Trident Protect CLI-Plugin herunter:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-macos-arm64
```

1. Aktivieren Sie die Ausführungsberechtigungen für die Plugin-Binärdatei:

```
chmod +x tridentctl-protect
```

2. Kopieren Sie die Plugin-Binärdatei an einen Ort, der in Ihrer PATH-Variablen definiert ist. Zum Beispiel, `/usr/bin` oder `/usr/local/bin` (Möglicherweise benötigen Sie erhöhte Berechtigungen):

```
cp ./tridentctl-protect /usr/local/bin/
```

3. Optional können Sie die Plugin-Binärdatei in ein Verzeichnis in Ihrem Home-Verzeichnis kopieren. In diesem Fall wird empfohlen, sicherzustellen, dass der Speicherort in Ihrer PATH-Variablen enthalten ist:

```
cp ./tridentctl-protect ~/bin/
```

Durch Kopieren des Plugins in einen Ordner Ihrer PATH-Variablen können Sie das Plugin durch Eingabe von … verwenden. `tridentctl-protect` oder `tridentctl protect` von jedem beliebigen Ort aus.

Hilfe zum Trident CLI-Plugin anzeigen

Mithilfe der integrierten Plugin-Hilfefunktionen erhalten Sie detaillierte Informationen zu den Funktionen des Plugins:

Schritte

1. Nutzen Sie die Hilfefunktion, um Hinweise zur Verwendung zu erhalten:

```
tridentctl-protect help
```

Automatische Befehlsvervollständigung aktivieren

Nach der Installation des Trident Protect CLI-Plugins können Sie die automatische Vervollständigung für bestimmte Befehle aktivieren.

Aktivieren Sie die automatische Vervollständigung für die Bash-Shell.

Schritte

1. Laden Sie das Vervollständigungsskript herunter:

```
curl -L -O https://github.com/NetApp/tridentctl-
protect/releases/download/25.06.0/tridentctl-completion.bash
```

2. Erstellen Sie in Ihrem Home-Verzeichnis ein neues Verzeichnis, das das Skript enthalten soll:

```
mkdir -p ~/.bash/completions
```

3. Verschieben Sie das heruntergeladene Skript in den folgenden Ordner: `~/.bash/completions` Verzeichnis:

```
mv tridentctl-completion.bash ~/.bash/completions/
```

4. Fügen Sie die folgende Zeile hinzu: `~/.bashrc` Datei in Ihrem Home-Verzeichnis:

```
source ~/.bash/completions/tridentctl-completion.bash
```

Automatische Vervollständigung für die Z-Shell aktivieren

Schritte

1. Laden Sie das Vervollständigungsskript herunter:

```
curl -L -O https://github.com/NetApp/tridentctl-
protect/releases/download/25.06.0/tridentctl-completion.zsh
```

2. Erstellen Sie in Ihrem Home-Verzeichnis ein neues Verzeichnis, das das Skript enthalten soll:

```
mkdir -p ~/.zsh/completions
```

3. Verschieben Sie das heruntergeladene Skript in den folgenden Ordner: `~/.zsh/completions` Verzeichnis:

```
mv tridentctl-completion.zsh ~/.zsh/completions/
```

4. Fügen Sie die folgende Zeile hinzu: `~/.zprofile` Datei in Ihrem Home-Verzeichnis:

```
source ~/.zsh/completions/tridentctl-completion.zsh
```

Ergebnis

Beim nächsten Login in die Shell können Sie die Befehlsvervollständigung mit dem tridentctl-protect-Plugin nutzen.

Trident Schutzinstallation anpassen

Sie können die Standardkonfiguration von Trident Protect an die spezifischen Anforderungen Ihrer Umgebung anpassen.

Legen Sie die Ressourcengrenzen des Trident Schutzcontainers fest.

Sie können nach der Installation von Trident protect eine Konfigurationsdatei verwenden, um Ressourcenlimits für Trident protect-Container festzulegen. Durch das Festlegen von Ressourcenlimits können Sie steuern, wie viele der Clusterressourcen von Trident Protect-Operationen verbraucht werden.

Schritte

1. Erstellen Sie eine Datei mit dem Namen `resourceLimits.yaml` .
2. Füllen Sie die Datei mit Ressourcenlimitoptionen für Trident Protect-Container entsprechend den Anforderungen Ihrer Umgebung.

   Die folgende Beispielkonfigurationsdatei zeigt die verfügbaren Einstellungen und enthält die Standardwerte 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. Wenden Sie die Werte aus dem `resourceLimits.yaml` Datei:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f resourceLimits.yaml --reuse-values
```

Sicherheitskontextbeschränkungen anpassen

Sie können eine Konfigurationsdatei verwenden, um die OpenShift-Sicherheitskontextbeschränkungen (SCCs) für Trident Protect-Container zu ändern, nachdem Sie Trident Protect installiert haben. Diese Einschränkungen definieren Sicherheitsbeschränkungen für Pods in einem Red Hat OpenShift-Cluster.

Schritte

1. Erstellen Sie eine Datei mit dem Namen `sccconfig.yaml` .
2. Fügen Sie die SCC-Option zur Datei hinzu und passen Sie die Parameter an die Bedürfnisse Ihrer Umgebung an.

   Das folgende Beispiel zeigt die Standardwerte der Parameter für die SCC-Option:

```
scc:
  create: true
  name: trident-protect-job
  priority: 1
```

Diese Tabelle beschreibt die Parameter für die SCC-Option:

Parameter	Beschreibung	Standard
erstellen	Ermittelt, ob eine SCC-Ressource erstellt werden kann. Eine SCC-Ressource wird nur dann erstellt, wenn `scc.create` ist eingestellt auf `true` und der Helm-Installationsprozess erkennt eine OpenShift-Umgebung. Wenn der Betrieb nicht auf OpenShift erfolgt, oder wenn `scc.create` ist eingestellt auf `false` Es wird keine SCC-Ressource erstellt.	true
Name	Gibt den Namen des SCC an.	trident-protect-job
Priorität	Definiert die Priorität des SCC. SCCs mit höheren Prioritätswerten werden vor solchen mit niedrigeren Werten bewertet.	1

3. Wenden Sie die Werte aus dem `sccconfig.yaml` Datei:

```
helm upgrade trident-protect netapp-trident-protect/trident-protect -f
sccconfig.yaml --reuse-values
```

Dadurch werden die Standardwerte durch die in der `sccconfig.yaml` Datei.

Konfigurieren Sie zusätzliche Trident Protect-Helm-Chart-Einstellungen

Sie können die AutoSupport Einstellungen und die Namespace-Filterung an Ihre spezifischen Anforderungen anpassen. In der folgenden Tabelle werden die verfügbaren Konfigurationsparameter beschrieben:

Parameter	Typ	Beschreibung
autoSupport.proxy	Schnur	Konfiguriert eine Proxy-URL für NetApp AutoSupport Verbindungen. Verwenden Sie dies, um Support-Bundle-Uploads über einen Proxyserver zu leiten. Beispiel: `http://my.proxy.url` .

Parameter	Typ	Beschreibung
autoSupport.unsicher	boolescher Wert	Überspringt die TLS-Verifizierung für AutoSupport Proxy-Verbindungen, wenn diese Option aktiviert ist. `true` . Nur für unsichere Proxy-Verbindungen verwenden. (Standard: `false`)
autoSupport.enabled	boolescher Wert	Aktiviert oder deaktiviert tägliche Uploads des Trident Protect AutoSupport Pakets. Wenn eingestellt auf `false` Tägliche Uploads sind deaktiviert, Support-Bundles können aber weiterhin manuell generiert werden. (Standard: `true`)
restoreSkipNamespaceAnnotations	Schnur	Durch Kommas getrennte Liste von Namespace-Anmerkungen, die von Sicherungs- und Wiederherstellungsvorgängen ausgeschlossen werden sollen. Ermöglicht Ihnen, Namespaces basierend auf Anmerkungen zu filtern.
restoreSkipNamespaceLabels	Schnur	Durch Kommas getrennte Liste von Namespace-Bezeichnungen, die von Sicherungs- und Wiederherstellungsvorgängen ausgeschlossen werden sollen. Ermöglicht Ihnen, Namespaces basierend auf Labels zu filtern.

Sie können diese Optionen entweder mithilfe einer YAML-Konfigurationsdatei oder mithilfe von Befehlszeilenflags konfigurieren:

YAML-Datei verwenden

Schritte

1. Erstellen Sie eine Konfigurationsdatei und benennen Sie sie. `values.yaml` .
2. Fügen Sie in der von Ihnen erstellten Datei die Konfigurationsoptionen hinzu, die Sie anpassen möchten.

```
autoSupport:
  enabled: false
  proxy: http://my.proxy.url
  insecure: true
restoreSkipNamespaceAnnotations: "annotation1,annotation2"
restoreSkipNamespaceLabels: "label1,label2"
```

3. Nachdem Sie die `values.yaml` Datei mit den korrekten Werten, Konfigurationsdatei anwenden:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f values.yaml --reuse-values
```

CLI-Flag verwenden

Schritte

1. Verwenden Sie den folgenden Befehl mit dem `--set` Flagge zur Angabe einzelner Parameter:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect \
  --set autoSupport.enabled=false \
  --set autoSupport.proxy=http://my.proxy.url \
  --set restoreSkipNamespaceAnnotations="annotation1,annotation2" \
  --set restoreSkipNamespaceLabels="label1,label2" \
  --reuse-values
```

Trident Schutz-Pods auf bestimmte Knoten beschränken

Mithilfe der Kubernetes-Knotenauswahlbeschränkung nodeSelector können Sie anhand der Knotenbezeichnungen steuern, welche Ihrer Knoten für die Ausführung von Trident Protect Pods geeignet sind. Standardmäßig ist Trident Protect auf Knoten beschränkt, auf denen Linux ausgeführt wird. Sie können diese Einschränkungen je nach Ihren Bedürfnissen weiter anpassen.

Schritte

1. Erstellen Sie eine Datei mit dem Namen `nodeSelectorConfig.yaml` .
2. Fügen Sie die Option nodeSelector zur Datei hinzu und modifizieren Sie die Datei, um Knotenbezeichnungen hinzuzufügen oder zu ändern und diese entsprechend den Anforderungen Ihrer

Umgebung einzuschränken. Die folgende Datei enthält beispielsweise die Standardbeschränkung des Betriebssystems, zielt aber auch auf eine bestimmte Region und einen bestimmten Anwendungsnamen ab:

```
nodeSelector:
  kubernetes.io/os: linux
  region: us-west
  app.kubernetes.io/name: mysql
```

3. Wenden Sie die Werte aus dem `nodeSelectorConfig.yaml` Datei:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f nodeSelectorConfig.yaml --reuse-values
```

Dadurch werden die Standardbeschränkungen durch die von Ihnen angegebenen Beschränkungen ersetzt. `nodeSelectorConfig.yaml` Datei.

Trident Schutz verwalten

Trident Protect-Autorisierung und Zugriffskontrolle verwalten

Trident Protect verwendet das Kubernetes-Modell der rollenbasierten Zugriffskontrolle (RBAC). Standardmäßig stellt Trident Protect einen einzelnen System-Namespace und das zugehörige Standard-Dienstkonto bereit. Wenn Ihre Organisation viele Benutzer oder spezielle Sicherheitsanforderungen hat, können Sie die RBAC-Funktionen von Trident Protect nutzen, um eine genauere Kontrolle über den Zugriff auf Ressourcen und Namespaces zu erhalten.

Der Clusteradministrator hat immer Zugriff auf die Ressourcen im Standardverzeichnis. `trident-protect` Namespace und kann auch auf Ressourcen in allen anderen Namespaces zugreifen. Um den Zugriff auf Ressourcen und Anwendungen zu steuern, müssen Sie zusätzliche Namensräume erstellen und diesen Namensräumen Ressourcen und Anwendungen hinzufügen.

Beachten Sie, dass im Standardmodus keine Benutzer Änderungsanforderungen für die Anwendungsdatenverwaltung erstellen können. `trident-protect` Namespace. Sie müssen Anwendungsdatenverwaltungs-CRs in einem Anwendungs-Namespace erstellen (als Best Practice sollten Anwendungsdatenverwaltungs-CRs im selben Namespace wie die zugehörige Anwendung erstellt werden).

Nur Administratoren sollten Zugriff auf privilegierte , Trident Protect geschützte benutzerdefinierte Ressourcenobjekte haben, darunter:

- **AppVault**: Erfordert Bucket-Anmeldeinformationen
- **AutoSupportBundle**: Erfasst Metriken, Protokolle und andere sensible Trident Protect-Daten
- **AutoSupportBundleSchedule**: Verwaltet die Zeitpläne für die Protokollerfassung

Als bewährte Methode empfiehlt es sich, RBAC zu verwenden, um den Zugriff auf privilegierte Objekte auf Administratoren zu beschränken.

Weitere Informationen darüber, wie RBAC den Zugriff auf Ressourcen und Namensräume regelt, finden Sie in der "Kubernetes RBAC-Dokumentation" .

Weitere Informationen zu Servicekonten finden Sie unter "Dokumentation zum Kubernetes-Dienstkonto" .

Beispiel: Zugriffsverwaltung für zwei Benutzergruppen

Eine Organisation verfügt beispielsweise über einen Cluster-Administrator, eine Gruppe von Entwicklern und eine Gruppe von Marketing-Anwendern. Der Cluster-Administrator würde die folgenden Aufgaben erledigen, um eine Umgebung zu schaffen, in der die Entwicklungsabteilung und die Marketingabteilung jeweils nur Zugriff auf die Ressourcen haben, die ihren jeweiligen Namensräumen zugewiesen sind.

Schritt 1: Erstellen Sie einen Namensraum, der die Ressourcen für jede Gruppe enthält.

Durch die Erstellung eines Namensraums können Sie Ressourcen logisch trennen und besser kontrollieren, wer Zugriff auf diese Ressourcen hat.

Schritte

1. Erstellen Sie einen Namensraum für die Entwicklungsgruppe:

```
kubectl create ns engineering-ns
```

2. Erstellen Sie einen Namensraum für die Marketinggruppe:

```
kubectl create ns marketing-ns
```

Schritt 2: Erstellen Sie neue Dienstkonten, um mit Ressourcen in jedem Namensraum zu interagieren.

Jeder neu erstellte Namespace verfügt über ein Standarddienstkonto. Sie sollten jedoch für jede Benutzergruppe ein Dienstkonto erstellen, um die Berechtigungen bei Bedarf später weiter zwischen den Gruppen aufteilen zu können.

Schritte

1. Erstellen Sie ein Dienstkonto für die Entwicklungsgruppe:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: eng-user
  namespace: engineering-ns
```

2. Erstellen Sie ein Servicekonto für die Marketinggruppe:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: mkt-user
  namespace: marketing-ns
```

Schritt 3: Erstellen Sie ein Geheimnis für jedes neue Dienstkonto.

Ein Dienstkontogeheimnis dient zur Authentifizierung beim Dienstkonto und kann im Falle einer Kompromittierung leicht gelöscht und neu erstellt werden.

Schritte

1. Erstellen Sie ein Geheimnis für das Engineering-Service-Konto:

```
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 Marketing-Service-Konto:

```
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 Dienstkonto zu binden.

Beim Installieren von Trident protect wird ein Standard-ClusterRole-Objekt 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 die ClusterRole an das Marketing-Servicekonto:

```
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: Berechtigungen testen

Prüfen Sie, ob die Berechtigungen korrekt sind.

Schritte

1. Stellen Sie sicher, dass die technischen Benutzer auf die technischen 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 die technischen Benutzer keinen Zugriff auf Marketingressourcen haben:

```
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 Datenverwaltungsaufgaben wie Backups und Snapshots durchzuführen, muss der Clusteradministrator einzelnen Benutzern Zugriff auf AppVault-Objekte gewähren.

Schritte

1. Erstellen und Anwenden einer YAML-Datei mit einer Kombination aus AppVault und Geheimnis, die einem Benutzer Zugriff auf einen AppVault gewährt. Beispielsweise gewährt die folgende CR dem Benutzer 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 wenden Sie eine Rollen-CR an, um Cluster-Administratoren die Möglichkeit zu geben, Zugriff auf bestimmte Ressourcen in einem Namespace zu gewähren. 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 eine 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. Versuch, AppVault-Objektinformationen für alle Namespaces abzurufen:

   ```
   kubectl get appvaults -n trident-protect
   --as=system:serviceaccount:engineering-ns:eng-user
   ```

   Die Ausgabe sollte in etwa wie folgt aussehen:

```
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 abrufen kann, für die er nun Zugriffsberechtigung 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
```

Die Ausgabe sollte in etwa wie folgt aussehen:

```
yes
```

Ergebnis

Die Benutzer, denen Sie AppVault-Berechtigungen erteilt haben, sollten in der Lage sein, autorisierte AppVault-Objekte für Anwendungsdatenverwaltungsvorgänge zu verwenden und sollten nicht in der Lage sein, auf Ressourcen außerhalb der zugewiesenen Namensräume zuzugreifen oder neue Ressourcen zu erstellen, auf die sie keinen Zugriff haben.

Trident überwachen und Ressourcen schützen

Sie können die Open-Source-Tools kube-state-metrics, Prometheus und Alertmanager verwenden, um den Zustand der durch Trident protect geschützten Ressourcen zu überwachen.

Der Dienst kube-state-metrics generiert Metriken aus der Kubernetes-API-Kommunikation. Die Verwendung mit Trident Protect liefert nützliche Informationen über den Zustand der Ressourcen in Ihrer Umgebung.

Prometheus ist ein Toolkit, das die von kube-state-metrics generierten Daten aufnehmen und als leicht lesbare Informationen über diese Objekte darstellen kann. Zusammen bieten kube-state-metrics und Prometheus Ihnen die Möglichkeit, den Zustand und die Integrität der Ressourcen zu überwachen, die Sie mit Trident protect verwalten.

Alertmanager ist ein Dienst, der die von Tools wie Prometheus gesendeten Warnmeldungen aufnimmt und sie an von Ihnen konfigurierte Ziele weiterleitet.

Die in diesen Schritten enthaltenen Konfigurationen und Anleitungen sind nur Beispiele; Sie müssen sie an Ihre Umgebung anpassen. Spezifische Anweisungen und Unterstützung finden Sie in der folgenden offiziellen Dokumentation:

- "kube-state-metrics-Dokumentation"
- "Prometheus-Dokumentation"
- "Alertmanager-Dokumentation"

Schritt 1: Installieren Sie die Überwachungstools

Um die Ressourcenüberwachung in Trident Protect zu aktivieren, müssen Sie kube-state-metrics, Promethus und Alertmanager installieren und konfigurieren.

Installieren Sie kube-state-metrics

Sie können kube-state-metrics mit Helm installieren.

Schritte

1. Füge das Helm-Chart kube-state-metrics hinzu. Beispiel:

```
helm repo add prometheus-community https://prometheus-
community.github.io/helm-charts
helm repo update
```

2. Wenden Sie den Prometheus ServiceMonitor CRD auf den Cluster an:

```
kubectl apply -f https://raw.githubusercontent.com/prometheus-
operator/prometheus-operator/main/example/prometheus-operator-
crd/monitoring.coreos.com_servicemonitors.yaml
```

3. Erstellen Sie eine Konfigurationsdatei für das Helm-Chart (zum Beispiel, `metrics-config.yaml`). Sie können die folgende Beispielkonfiguration an Ihre Umgebung anpassen:

metrics-config.yaml: kube-state-metrics Helm-Chart-Konfiguration

```
---
extraArgs:
  # Collect only custom metrics
  - --custom-resource-state-only=true

customResourceState:
  enabled: true
  config:
    kind: CustomResourceStateMetrics
    spec:
      resources:
      - groupVersionKind:
          group: protect.trident.netapp.io
          kind: "Backup"
          version: "v1"
        labelsFromPath:
          backup_uid: [metadata, uid]
          backup_name: [metadata, name]
          creation_time: [metadata, creationTimestamp]
        metrics:
        - name: backup_info
          help: "Exposes details about the Backup state"
          each:
            type: Info
            info:
              labelsFromPath:
                appVaultReference: ["spec", "appVaultRef"]
                appReference: ["spec", "applicationRef"]
rbac:
  extraRules:
  - apiGroups: ["protect.trident.netapp.io"]
    resources: ["backups"]
    verbs: ["list", "watch"]

# Collect metrics from all namespaces
namespaces: ""

# Ensure that the metrics are collected by Prometheus
prometheus:
  monitor:
    enabled: true
```

4. Installieren Sie kube-state-metrics durch Bereitstellung des Helm-Charts. Beispiel:

```
helm install custom-resource -f metrics-config.yaml prometheus-
community/kube-state-metrics --version 5.21.0
```

5. Konfigurieren Sie kube-state-metrics so, dass Metriken für die von Trident Protect verwendeten benutzerdefinierten Ressourcen generiert werden, indem Sie den Anweisungen in der "Dokumentation der benutzerdefinierten Ressource kube-state-metrics" .

Prometheus installieren

Sie können Prometheus installieren, indem Sie den Anweisungen in der "Prometheus-Dokumentation" .

Installieren Sie Alertmanager

Sie können Alertmanager installieren, indem Sie den Anweisungen in der "Alertmanager-Dokumentation" .

Schritt 2: Konfigurieren Sie die Überwachungstools für die Zusammenarbeit

Nach der Installation der Überwachungstools müssen Sie diese so konfigurieren, dass sie zusammenarbeiten.

Schritte

1. kube-state-metrics mit Prometheus integrieren. Bearbeiten Sie die Prometheus-Konfigurationsdatei(`prometheus.yaml`) und fügen Sie die Informationen zum kube-state-metrics-Dienst hinzu. Beispiel:

   prometheus.yaml: Integration des kube-state-metrics-Dienstes mit Prometheus

```
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: prometheus-config
  namespace: trident-protect
data:
  prometheus.yaml: |
    global:
      scrape_interval: 15s
    scrape_configs:
      - job_name: 'kube-state-metrics'
        static_configs:
          - targets: ['kube-state-metrics.trident-protect.svc:8080']
```

2. Konfigurieren Sie Prometheus so, dass Warnmeldungen an Alertmanager weitergeleitet werden. Bearbeiten Sie die Prometheus-Konfigurationsdatei(`prometheus.yaml`) und fügen Sie den folgenden Abschnitt hinzu:

prometheus.yaml: Benachrichtigungen an Alertmanager senden

```
alerting:
  alertmanagers:
    - static_configs:
        - targets:
            - alertmanager.trident-protect.svc:9093
```

Ergebnis

Prometheus kann nun Metriken von kube-state-metrics erfassen und Warnungen an Alertmanager senden. Sie können nun konfigurieren, unter welchen Bedingungen eine Warnung ausgelöst wird und wohin die Warnungen gesendet werden sollen.

Schritt 3: Benachrichtigungen und Benachrichtigungsziele konfigurieren

Nachdem Sie die Tools so konfiguriert haben, dass sie zusammenarbeiten, müssen Sie festlegen, welche Art von Informationen Warnmeldungen auslösen und wohin die Warnmeldungen gesendet werden sollen.

Beispiel für eine Warnung: Backup-Fehler

Das folgende Beispiel definiert eine kritische Warnung, die ausgelöst wird, wenn der Status der benutzerdefinierten Sicherungsressource auf „“ gesetzt wird. `Error` für 5 Sekunden oder länger. Sie können dieses Beispiel an Ihre Umgebung anpassen und diesen YAML-Ausschnitt in Ihre Datei einfügen. `prometheus.yaml` Konfigurationsdatei:

rules.yaml: Definiere eine Prometheus-Warnung für fehlgeschlagene Backups

```
rules.yaml: |
  groups:
    - name: fail-backup
        rules:
          - alert: BackupFailed
            expr: kube_customresource_backup_info{status="Error"}
            for: 5s
            labels:
              severity: critical
            annotations:
              summary: "Backup failed"
              description: "A backup has failed."
```

Konfigurieren Sie Alertmanager so, dass Benachrichtigungen an andere Kanäle gesendet werden.

Sie können Alertmanager so konfigurieren, dass Benachrichtigungen an andere Kanäle wie E-Mail, PagerDuty, Microsoft Teams oder andere Benachrichtigungsdienste gesendet werden, indem Sie die entsprechende Konfiguration in der `alertmanager.yaml` Datei.

Das folgende Beispiel konfiguriert Alertmanager so, dass Benachrichtigungen an einen Slack-Kanal gesendet werden. Um dieses Beispiel an Ihre Umgebung anzupassen, ersetzen Sie den Wert von `api_url` Schlüssel mit der in Ihrer Umgebung verwendeten Slack-Webhook-URL:

alertmanager.yaml: Benachrichtigungen an einen Slack-Kanal senden

```
data:
  alertmanager.yaml: |
    global:
      resolve_timeout: 5m
    route:
      receiver: 'slack-notifications'
    receivers:
      - name: 'slack-notifications'
        slack_configs:
          - api_url: '<your-slack-webhook-url>'
            channel: '#failed-backups-channel'
            send_resolved: false
```

Generieren Sie ein Trident Protect Support-Paket

Mit Trident Protect können Administratoren Pakete erstellen, die für den NetApp Support nützliche Informationen enthalten, darunter Protokolle, Metriken und Topologieinformationen zu den verwalteten Clustern und Apps. Wenn Sie mit dem Internet verbunden sind, können Sie Support-Pakete mithilfe einer benutzerdefinierten Ressourcendatei (CR) auf die NetApp Support Site (NSS) hochladen.

Erstellen Sie ein Support-Bundle mithilfe eines CR

Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie (zum Beispiel). `trident-protect-support-bundle.yaml`).
2. Konfigurieren Sie die folgenden Attribute:
   - **metadata.name**: (*Erforderlich*) Der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und aussagekräftigen Namen für Ihre Umgebung.
   - **spec.triggerType**: (*Erforderlich*) Legt fest, ob das Support-Bundle sofort oder geplant generiert wird. Die geplante Bündelgenerierung erfolgt um 0:00 Uhr UTC. Mögliche Werte:
     - Geplant
     - Handbuch
   - **spec.uploadEnabled**: (*Optional*) Steuert, ob das Support-Bundle nach seiner Generierung auf die NetApp Support-Website hochgeladen werden soll. Falls nicht anders angegeben, wird standardmäßig der Wert verwendet. `false` . Mögliche Werte:
     - true
     - falsch (Standardeinstellung)
   - **spec.dataWindowStart**: (*Optional*) Eine Datumszeichenfolge im RFC 3339-Format, die Datum und Uhrzeit angibt, zu der das Fenster der im Support-Bundle enthaltenen Daten beginnen soll. Falls nicht anders angegeben, wird standardmäßig der Wert von vor 24 Stunden verwendet. Das früheste Datum, das Sie angeben können, liegt 7 Tage zurück.

     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 `trident-protect-support-bundle.yaml` Datei mit den korrekten Werten, CR anwenden:

```
kubectl apply -f trident-protect-support-bundle.yaml -n trident-
protect
```

Erstellen Sie ein Support-Bundle mithilfe der CLI.

Schritte

1. Erstellen Sie das Support-Bundle und ersetzen Sie die Werte in Klammern durch Informationen aus

Ihrer Umgebung. Der `trigger-type` bestimmt, ob das Paket sofort erstellt wird oder ob die Erstellungszeit durch den Zeitplan vorgegeben ist, und kann sein `Manual` oder `Scheduled` . Die Standardeinstellung ist `Manual` .

Beispiel:

```
tridentctl-protect create autosupportbundle <my-bundle-name>
--trigger-type <trigger-type> -n trident-protect
```

Überwachen und Abrufen des Support-Pakets

Nachdem Sie mit einer der beiden Methoden ein Support-Paket erstellt haben, können Sie den Generierungsfortschritt überwachen und es auf Ihr lokales System abrufen.

Schritte

1. Warten Sie auf die `status.generationState` erreichen `Completed` Zustand. Sie können den Generierungsfortschritt mit dem folgenden Befehl überwachen:

   ```
   kubectl get autosupportbundle trident-protect-support-bundle -n trident-
   protect
   ```

2. Rufen Sie das Support-Paket auf Ihr lokales System ab. Holen Sie sich den Kopierbefehl aus dem vollständigen AutoSupport Paket:

   ```
   kubectl describe autosupportbundle trident-protect-support-bundle -n
   trident-protect
   ```

   Finden Sie die `kubectl cp` Führen Sie den Befehl aus der Ausgabe aus und ersetzen Sie dabei das Zielargument durch Ihr bevorzugtes lokales Verzeichnis.

Verbesserter Trident Schutz

Sie können Trident Protect auf die neueste Version aktualisieren, um von neuen Funktionen oder Fehlerbehebungen zu profitieren.

> Beim Upgrade von Version 24.10 kann es vorkommen, dass während des Upgrades ausgeführte Snapshots fehlschlagen. Dieser Fehler verhindert nicht, dass künftige Snapshots, ob manuell oder geplant, erstellt werden. Falls ein Snapshot während des Upgrades fehlschlägt, können Sie manuell einen neuen Snapshot erstellen, um sicherzustellen, dass Ihre Anwendung geschützt ist.
>
> Um mögliche Fehler zu vermeiden, können Sie vor dem Upgrade alle Snapshot-Zeitpläne deaktivieren und anschließend wieder aktivieren. Dies hat jedoch zur Folge, dass während des Upgrade-Zeitraums geplante Snapshots verpasst werden.

Um Trident Protect zu aktualisieren, führen Sie die folgenden Schritte aus.

Schritte

1. Aktualisieren Sie das Trident Helm-Repository:

```
helm repo update
```

2. Rüsten Sie die Trident Protect CRDs auf:

> Dieser Schritt ist erforderlich, wenn Sie von einer Version vor 25.06 aktualisieren, da die CRDs jetzt im Trident Protect Helm-Chart enthalten sind.

a. Führen Sie diesen Befehl aus, um die Verwaltung von CRDs zu ändern von `trident-protect-crds` Zu `trident-protect` :

```
kubectl get crd | grep protect.trident.netapp.io | awk '{print $1}' |
xargs -I {} kubectl patch crd {} --type merge -p '{"metadata":
{"annotations":{"meta.helm.sh/release-name": "trident-protect"}}}'
```

b. Führen Sie diesen Befehl aus, um das Helm-Secret für die `trident-protect-crds` Diagramm:

> Deinstallieren Sie die `trident-protect-crds` Verwenden Sie Helm nicht für Charts, da dies Ihre CRDs und alle zugehörigen Daten entfernen könnte.

```
kubectl delete secret -n trident-protect -l name=trident-protect-
crds,owner=helm
```

3. Upgrade Trident Protect:

```
helm upgrade trident-protect netapp-trident-protect/trident-protect
--version 100.2506.0 --namespace trident-protect
```

Anwendungen verwalten und schützen

Verwenden Sie Trident Protect AppVault-Objekte, um Buckets zu verwalten.

Die Bucket Custom Resource (CR) für Trident Protect wird als AppVault bezeichnet. AppVault-Objekte sind die deklarative Kubernetes-Workflow-Darstellung eines Speicher-Buckets. Ein AppVault CR enthält die Konfigurationen, die für die Verwendung eines Buckets in Schutzvorgängen wie Backups, Snapshots, Wiederherstellungsvorgängen und SnapMirror Replikation erforderlich sind. Nur Administratoren können AppVaults erstellen.

Sie müssen eine AppVault-CR manuell oder über die Befehlszeile erstellen, wenn Sie Datenschutzvorgänge

für eine Anwendung ausführen. Die AppVault-CR ist spezifisch für Ihre Umgebung. Die Beispiele auf dieser Seite können Ihnen beim Erstellen von AppVault-CRs als Leitfaden dienen.

> Stellen Sie sicher, dass sich der AppVault CR auf dem Cluster befindet, auf dem Trident Protect installiert ist. Wenn der AppVault CR nicht vorhanden ist oder Sie nicht darauf zugreifen können, wird in der Befehlszeile ein Fehler angezeigt.

AppVault-Authentifizierung und Passwörter konfigurieren

Bevor Sie einen AppVault CR erstellen, stellen Sie sicher, dass sich der AppVault und der von Ihnen gewählte Datenverschieber beim Anbieter und allen zugehörigen Ressourcen authentifizieren können.

Passwörter für das Datenübertragungs-Repository

Wenn Sie AppVault-Objekte mit CRs oder dem Trident Protect CLI-Plugin erstellen, können Sie ein Kubernetes-Geheimnis mit benutzerdefinierten Passwörtern für die Restic- und Kopia-Verschlüsselung angeben. Wenn Sie kein Geheimnis angeben, verwendet Trident Protect ein Standardpasswort.

- Beim manuellen Erstellen von AppVault CRs verwenden Sie das Feld **spec.dataMoverPasswordSecretRef**, um das Geheimnis anzugeben.
- Verwenden Sie beim Erstellen von AppVault-Objekten mit der Trident Protect CLI die `--data-mover-password-secret-ref` Argument zur Angabe des Geheimnisses.

Erstellen Sie ein Passwort für das Datenübertragungs-Repository.

Nutzen Sie die folgenden Beispiele, um das Passwortgeheimnis zu erstellen. Wenn Sie AppVault-Objekte erstellen, können Sie Trident Protect anweisen, dieses Geheimnis zur Authentifizierung beim Datenübertragungsrepository zu verwenden.

> - Je nachdem, welchen Datenmigrationsdienst Sie verwenden, müssen Sie nur das entsprechende Passwort für diesen Datenmigrationsdienst angeben. Wenn Sie beispielsweise Restic verwenden und nicht planen, Kopia in Zukunft zu nutzen, können Sie beim Erstellen des Geheimnisses nur das Restic-Passwort angeben.
> - Bewahren Sie das Passwort an einem sicheren Ort auf. Sie benötigen es, um Daten auf demselben oder einem anderen Cluster wiederherzustellen. Wenn der Cluster oder der `trident-protect` Wenn der Namespace gelöscht wird, können Sie Ihre Backups oder Snapshots ohne das Passwort nicht wiederherstellen.

Verwenden Sie einen CR

```
---
apiVersion: v1
data:
  KOPIA_PASSWORD: <base64-encoded-password>
  RESTIC_PASSWORD: <base64-encoded-password>
kind: Secret
metadata:
  name: my-optional-data-mover-secret
  namespace: trident-protect
type: Opaque
```

Verwenden der CLI

```
kubectl create secret generic my-optional-data-mover-secret \
--from-literal=KOPIA_PASSWORD=<plain-text-password> \
--from-literal=RESTIC_PASSWORD=<plain-text-password> \
-n trident-protect
```

S3-kompatible Speicher-IAM-Berechtigungen

Wenn Sie auf S3-kompatiblen Speicher wie Amazon S3, Generic S3, zugreifen "StorageGrid S3" , oder "ONTAP S3" Bei der Verwendung von Trident Protect müssen Sie sicherstellen, dass die von Ihnen angegebenen Benutzeranmeldeinformationen über die erforderlichen Berechtigungen für den Zugriff auf den Bucket verfügen. Nachfolgend ein Beispiel für eine Richtlinie, die die minimal erforderlichen Berechtigungen für den Zugriff mit Trident Protect gewährt. Sie können diese Richtlinie auf den Benutzer anwenden, der S3-kompatible Bucket-Richtlinien verwaltet.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:ListBucket",
        "s3:DeleteObject"
      ],
      "Resource": "*"
    }
  ]
}
```

Weitere Informationen zu Amazon S3-Richtlinien finden Sie in den Beispielen im Abschnitt „…“. "Amazon S3-Dokumentation" .

EKS Pod Identity für Amazon S3 (AWS)-Authentifizierung

Trident Protect unterstützt EKS Pod Identity für Kopia-Datenverschiebungsvorgänge. Diese Funktion ermöglicht den sicheren Zugriff auf S3-Buckets, ohne AWS-Anmeldeinformationen in Kubernetes-Geheimnissen zu speichern.

Anforderungen für EKS Pod Identity mit Trident Protect

Bevor Sie EKS Pod Identity mit Trident Protect verwenden, stellen Sie Folgendes sicher:

- In Ihrem EKS-Cluster ist Pod Identity aktiviert.
- Sie haben eine IAM-Rolle mit den erforderlichen S3-Bucket-Berechtigungen erstellt. Weitere Informationen finden Sie unter "S3-kompatible Speicher-IAM-Berechtigungen".
- Die IAM-Rolle ist mit den folgenden Trident Protect-Dienstkonten verknüpft:
  - `<trident-protect>-controller-manager`
  - `<trident-protect>-resource-backup`
  - `<trident-protect>-resource-restore`
  - `<trident-protect>-resource-delete`

Detaillierte Anweisungen zum Aktivieren von Pod Identity und zum Zuordnen von IAM-Rollen zu Servicekonten finden Sie in der Dokumentation. "AWS EKS Pod Identity-Dokumentation" .

AppVault-Konfiguration Wenn Sie EKS Pod Identity verwenden, konfigurieren Sie Ihre AppVault CR mit der `useIAM: true` Flagge statt expliziter Anmeldeinformationen:

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: eks-protect-vault
  namespace: trident-protect
spec:
  providerType: AWS
  providerConfig:
    s3:
      bucketName: trident-protect-aws
      endpoint: s3.example.com
      useIAM: true
```

AppVault-Schlüsselgenerierungsbeispiele für Cloud-Anbieter

Wenn Sie eine AppVault CR definieren, müssen Sie Anmeldeinformationen für den Zugriff auf die vom Anbieter gehosteten Ressourcen angeben, es sei denn, Sie verwenden die IAM-Authentifizierung. Die Art und Weise, wie Sie die Schlüssel für die Zugangsdaten generieren, unterscheidet sich je nach Anbieter. Nachfolgend finden Sie Beispiele für die Schlüsselgenerierung über die Befehlszeile für mehrere Anbieter. Sie können die folgenden Beispiele verwenden, um Schlüssel für die Anmeldeinformationen jedes Cloud-Anbieters

zu 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=<amazon-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
```

Generisches 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=<ontap-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=<storagegrid-s3-trident-protect-src
-bucket-secret> \
-n trident-protect
```

Beispiele für die Erstellung von AppVault

Nachfolgend finden Sie Beispieldefinitionen für AppVault für jeden Anbieter.

AppVault CR-Beispiele

Anhand der folgenden CR-Beispiele können Sie AppVault-Objekte für jeden Cloud-Anbieter erstellen.

- Optional können Sie ein Kubernetes-Secret angeben, das benutzerdefinierte Passwörter für die Verschlüsselung der Restic- und Kopia-Repositorys enthält. SiehePasswörter für das Datenübertragungs-Repository für weitere Informationen.
- Bei Amazon S3 (AWS) AppVault-Objekten können Sie optional ein SessionToken angeben, was nützlich ist, wenn Sie Single Sign-On (SSO) zur Authentifizierung verwenden. Dieses Token wird erstellt, wenn Sie Schlüssel für den Anbieter generieren.AppVault-Schlüsselgenerierungsbeispiele für Cloud-Anbieter .
- Für S3 AppVault-Objekte können Sie optional eine Egress-Proxy-URL für ausgehenden S3-Datenverkehr angeben. `spec.providerConfig.S3.proxyURL` Schlüssel.

Google Cloud

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: gcp-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  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)

```yaml
---
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: amazon-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: AWS
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
    sessionToken:
      valueFromSecret:
        key: sessionToken
        name: s3-secret
```

Bei EKS-Umgebungen, die Pod Identity mit Kopia Data Mover verwenden, können Sie die `providerCredentials` Abschnitt und hinzufügen `useIAM: true` unter dem `s3` stattdessen Konfiguration.

Microsoft Azure

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: azure-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: Azure
  providerConfig:
    azure:
      accountName: account-name
      bucketName: trident-protect-src-bucket
  providerCredentials:
    accountKey:
      valueFromSecret:
        key: accountKey
        name: azure-trident-protect-src-bucket-secret
```

Generisches S3

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: generic-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: GenericS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
```

ONTAP S3

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: ontap-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: OntapS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
```

StorageGrid S3

```yaml
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: storagegrid-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: StorageGridS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
```

Beispiele zur AppVault-Erstellung mit der Trident Protect CLI

Sie können die folgenden CLI-Befehlsbeispiele verwenden, um AppVault CRs für jeden Anbieter zu erstellen.

- Optional können Sie ein Kubernetes-Secret angeben, das benutzerdefinierte Passwörter für die Verschlüsselung der Restic- und Kopia-Repositorys enthält. SiehePasswörter für das Datenübertragungs-Repository für weitere Informationen.
- Für S3 AppVault-Objekte können Sie optional eine Egress-Proxy-URL für ausgehenden S3-Datenverkehr angeben. `--proxy-url <ip_address:port>` Argument.

Google Cloud

```
tridentctl-protect create vault GCP <vault-name> \
--bucket <mybucket> \
--project <my-gcp-project> \
--secret <secret-name>/credentials \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

Amazon S3 (AWS)

```
tridentctl-protect create vault AWS <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

Microsoft Azure

```
tridentctl-protect create vault Azure <vault-name> \
--account <account-name> \
--bucket <bucket-name> \
--secret <secret-name> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

Generisches S3

```
tridentctl-protect create vault GenericS3 <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

ONTAP S3

```
tridentctl-protect create vault OntapS3 <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

StorageGrid S3

```
tridentctl-protect create vault StorageGridS3 <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

AppVault-Informationen anzeigen

Mit dem Trident Protect CLI-Plugin können Sie Informationen über AppVault-Objekte anzeigen, die Sie auf dem Cluster erstellt haben.

Schritte

1. Den Inhalt eines AppVault-Objekts anzeigen:

```
tridentctl-protect get appvaultcontent gcp-vault \
--show-resources all \
-n trident-protect
```

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. Optional kann der AppVaultPath für jede Ressource mit dem Flag angezeigt werden. `--show-paths` .

   Der Clustername in der ersten Spalte der Tabelle ist nur verfügbar, wenn bei der Trident Protect Helm-Installation ein Clustername angegeben wurde. Zum Beispiel: `--set clusterName=production1` .

AppVault entfernen

Sie können ein AppVault-Objekt jederzeit entfernen.

> Entfernen Sie nicht die `finalizers` Geben Sie den Schlüssel im AppVault CR ein, bevor Sie das AppVault-Objekt löschen. Wenn Sie dies tun, kann es zu Restdaten im AppVault-Bucket und zu verwaisten Ressourcen im Cluster kommen.

Bevor Sie beginnen

Stellen Sie sicher, dass Sie alle Snapshot- und Backup-CRs gelöscht haben, die von dem AppVault verwendet werden, den Sie löschen möchten.

Entfernen eines AppVaults mithilfe der Kubernetes-CLI

1. Entfernen Sie das AppVault-Objekt und ersetzen Sie es durch `appvault-name` mit dem 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 es durch `appvault-name` mit dem Namen des zu entfernenden AppVault-Objekts:

```
tridentctl-protect delete appvault <appvault-name> \
-n trident-protect
```

Definieren Sie eine Anwendung für die Verwaltung mit Trident Protect.

Sie können eine Anwendung, die Sie mit Trident Protect verwalten möchten, definieren, indem Sie eine Anwendungs-CR und eine zugehörige AppVault-CR erstellen.

Erstellen Sie eine AppVault CR

Sie müssen einen AppVault CR erstellen, der bei der Durchführung von Datenschutzoperationen an der Anwendung verwendet wird, und der AppVault CR muss sich auf dem Cluster befinden, auf dem Trident Protect installiert ist. Die AppVault-CR ist spezifisch für Ihre Umgebung; Beispiele für AppVault-CRs finden Sie unter"AppVault-Benutzerressourcen."

Definieren einer Anwendung

Sie müssen jede Anwendung definieren, die Sie mit Trident Protect verwalten möchten. Sie können eine Anwendung zur Verwaltung definieren, indem Sie entweder manuell eine Anwendungs-CR erstellen oder die Trident Protect CLI verwenden.

Fügen Sie eine Anwendung mithilfe eines CR hinzu

Schritte

1. Erstellen Sie die CR-Datei der Zielanwendung:
   a. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie (zum Beispiel). `maria-app.yaml`).
   b. Konfigurieren Sie die folgenden Attribute:
      - **metadata.name**: (*Erforderlich*) Der Name der benutzerdefinierten Anwendungsressource. Merken Sie sich den gewählten Namen, da andere für Schutzvorgänge benötigte CR-Dateien auf diesen Wert verweisen.
      - **spec.includedNamespaces**: (*Erforderlich*) Verwenden Sie den Namespace- und Label-Selektor, um die Namespaces und Ressourcen anzugeben, die die Anwendung verwendet. Der Anwendungs-Namespace muss Teil dieser Liste sein. Der Label-Selektor ist optional und kann verwendet werden, um Ressourcen innerhalb jedes angegebenen Namespace zu filtern.
      - **spec.includedClusterScopedResources**: (*Optional*) Verwenden Sie dieses Attribut, um Cluster-bezogene Ressourcen anzugeben, die in die Anwendungsdefinition aufgenommen werden sollen. Mithilfe dieses Attributs können Sie diese Ressourcen anhand ihrer Gruppe, Version, Art und Bezeichnungen auswählen.
        - **groupVersionKind**: (*Erforderlich*) Gibt die API-Gruppe, Version und Art der clusterweiten Ressource an.
        - **labelSelector**: (*Optional*) Filtert die clusterweiten Ressourcen anhand ihrer Labels.
      - **metadata.annotations.protect.trident.netapp.io/skip-vm-freeze**: (*Optional*) Diese Annotation ist nur für Anwendungen anwendbar, die von virtuellen Maschinen definiert werden, wie z. B. in KubeVirt-Umgebungen, wo Dateisystem-Freezes vor Snapshots auftreten. Legen Sie fest, ob diese Anwendung während einer Snapshot-Erstellung auf das Dateisystem schreiben darf. Wenn diese Option auf „true“ gesetzt ist, ignoriert die Anwendung die globale Einstellung und kann während eines Snapshots auf das Dateisystem schreiben. Wenn der Wert auf „false“ gesetzt ist, ignoriert die Anwendung die globale Einstellung und das Dateisystem wird während eines Snapshots eingefroren. Wird dies angegeben, die Anwendung aber keine virtuellen Maschinen in der Anwendungsdefinition aufweist, wird die Annotation ignoriert. Sofern nichts anderes angegeben ist, gilt die folgende Vorgehensweise:"globale Trident Schutzeinstellung zum Einfrieren" .

Falls Sie diese Annotation nachträglich anwenden müssen, nachdem eine Anwendung bereits erstellt wurde, können Sie folgenden Befehl verwenden:

```
kubectl annotate application -n <application CR namespace> <application CR
name> protect.trident.netapp.io/skip-vm-freeze="true"
```

+
Beispiel-YAML:

+

```
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  annotations:
    protect.trident.netapp.io/skip-vm-freeze: "false"
  name: my-app-name
  namespace: my-app-namespace
spec:
  includedNamespaces:
    - namespace: namespace-1
      labelSelector:
        matchLabels:
          app: example-app
    - namespace: namespace-2
      labelSelector:
        matchLabels:
          app: another-example-app
  includedClusterScopedResources:
    - groupVersionKind:
        group: rbac.authorization.k8s.io
        kind: ClusterRole
        version: v1
      labelSelector:
        matchLabels:
          mylabel: test
```

1. (*Optional*) Filter hinzufügen, die Ressourcen mit bestimmten Bezeichnungen ein- oder ausschließen:

   - **resourceFilter.resourceSelectionCriteria**: (Für die Filterung erforderlich) Verwenden Sie `Include` oder `Exclude` Eine in resourceMatchers definierte Ressource ein- oder ausschließen. Fügen Sie die folgenden resourceMatchers-Parameter hinzu, um die ein- oder auszuschließenden Ressourcen zu definieren:

     - **resourceFilter.resourceMatchers**: Ein Array von resourceMatcher-Objekten. Wenn Sie mehrere Elemente in diesem Array definieren, werden diese per OR-Verknüpfung abgeglichen, und die Felder innerhalb jedes Elements (Gruppe, Art, Version) werden per AND-Verknüpfung abgeglichen.
       - **resourceMatchers[].group**: (*Optional*) Gruppe der zu filternden Ressourcen.
       - **resourceMatchers[].kind**: (*Optional*) Art der zu filternden Ressource.
       - **resourceMatchers[].version**: (*Optional*) Version der zu filternden Ressource.

- **resourceMatchers[].names**: (*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-Selektorzeichenfolge im Kubernetes-Metadatenfeld „name“ der Ressource, wie in der Dokumentation definiert. "Kubernetes-Dokumentation" . Zum Beispiel: `"trident.netapp.io/os=linux"` .

> Wenn beide `resourceFilter` Und `labelSelector` werden verwendet `resourceFilter` zuerst läuft und dann `labelSelector` wird auf die resultierenden Ressourcen angewendet.

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"]
```

2. Nachdem Sie die Anwendungs-CR erstellt haben, die zu Ihrer Umgebung passt, wenden Sie die CR an. Beispiel:

```
kubectl apply -f maria-app.yaml
```

Schritte

1. Erstellen und wenden Sie die Anwendungsdefinition anhand eines der folgenden Beispiele an und ersetzen Sie die Werte in Klammern durch Informationen aus Ihrer Umgebung. Sie können Namespaces und Ressourcen in die Anwendungsdefinition einbinden, indem Sie kommagetrennte Listen mit den in den Beispielen gezeigten Argumenten verwenden.

   Optional können Sie beim Erstellen einer App eine Annotation verwenden, um anzugeben, ob die Anwendung während eines Snapshots auf das Dateisystem schreiben darf. Dies gilt nur für Anwendungen, die von virtuellen Maschinen definiert werden, wie beispielsweise in KubeVirt-Umgebungen, wo Dateisystem-Freezes vor Snapshots auftreten. Wenn Sie die Annotation auf `true`

Die Anwendung ignoriert die globale Einstellung und kann während eines Snapshots in das Dateisystem schreiben. Wenn Sie es einstellen auf `false` Die Anwendung ignoriert die globale Einstellung und das Dateisystem wird während eines Snapshots eingefroren. Wenn Sie die Annotation verwenden, die Anwendung aber keine virtuellen Maschinen in der Anwendungsdefinition aufweist, wird die Annotation ignoriert. Wenn Sie die Annotation nicht verwenden, folgt die Anwendung dem Standard."globale Trident Schutzeinstellung zum Einfrieren" .

Um die Annotation bei der Erstellung einer Anwendung über die CLI anzugeben, können Sie Folgendes verwenden: `--annotation` Flagge.

- Erstellen Sie die Anwendung und verwenden Sie die globale Einstellung für das Einfrieren des Dateisystems:

```
tridentctl-protect create application <my_new_app_cr_name>
--namespaces <namespaces_to_include> --csr
<cluster_scoped_resources_to_include> --namespace <my-app-
namespace>
```

- Erstellen Sie die Anwendung und konfigurieren Sie die lokalen Anwendungseinstellungen für das Dateisystem-Einfrierverhalten:

```
tridentctl-protect create application <my_new_app_cr_name>
--namespaces <namespaces_to_include> --csr
<cluster_scoped_resources_to_include> --namespace <my-app-
namespace> --annotation protect.trident.netapp.io/skip-vm-freeze
=<"true"|"false">
```

Sie können verwenden `--resource-filter-include` Und `--resource-filter-exclude` Kennzeichnungen zum Ein- oder Ausschließen von Ressourcen basierend auf `resourceSelectionCriteria` wie beispielsweise Gruppe, Art, Version, Bezeichnungen, Namen und Namensräume, wie im folgenden Beispiel gezeigt:

```
tridentctl-protect create application <my_new_app_cr_name>
--namespaces <namespaces_to_include> --csr
<cluster_scoped_resources_to_include> --namespace <my-app-namespace>
--resource-filter-include
'[{"Group":"apps","Kind":"Deployment","Version":"v1","Names":["my-
deployment"],"Namespaces":["my-
namespace"],"LabelSelectors":["app=my-app"]}]'
```

Schützen Sie Ihre Anwendungen mit Trident Protect.

Sie können alle von Trident Protect verwalteten Apps schützen, indem Sie mithilfe einer automatisierten Schutzrichtlinie oder ad hoc Snapshots und Backups erstellen.

Sie können Trident Protect so konfigurieren, dass Dateisysteme während Datensicherungsvorgängen eingefroren und wieder freigegeben werden. "Erfahren Sie mehr über die Konfiguration des Einfrierens von Dateisystemen mit Trident Protect.".

Erstellen Sie einen On-Demand-Snapshot

Sie können jederzeit einen On-Demand-Snapshot erstellen.

Clusterbezogene Ressourcen werden in eine Sicherung, einen Snapshot oder einen Klon aufgenommen, wenn sie in der Anwendungsdefinition explizit referenziert werden oder wenn sie Verweise auf einen der Anwendungs-Namespaces enthalten.

Erstellen Sie einen Snapshot mithilfe eines CR

Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie. `trident-protect-snapshot-cr.yaml` .
2. Konfigurieren Sie in der von Ihnen erstellten Datei die folgenden Attribute:
   - **metadata.name**: (*Erforderlich*) Der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und aussagekräftigen Namen für Ihre Umgebung.
   - **spec.applicationRef**: Der Kubernetes-Name der Anwendung, für die ein Snapshot erstellt werden soll.
   - **spec.appVaultRef**: (*Erforderlich*) Der Name des AppVaults, in dem die Snapshot-Inhalte (Metadaten) gespeichert werden sollen.
   - **spec.reclaimPolicy**: (*Optional*) Definiert, was mit dem AppArchive eines Snapshots geschieht, wenn der Snapshot CR gelöscht wird. Das bedeutet, dass selbst wenn auf eingestellt `Retain` Der Snapshot wird gelöscht. 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 `trident-protect-snapshot-cr.yaml` Datei mit den korrekten Werten, CR anwenden:

   ```
   kubectl apply -f trident-protect-snapshot-cr.yaml
   ```

Erstellen Sie einen Snapshot mithilfe der Befehlszeile.

Schritte

1. Erstellen Sie den Snapshot und ersetzen Sie die 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 eine bedarfsgesteuerte Datensicherung.

Sie können eine App jederzeit sichern.

> Clusterbezogene Ressourcen werden in eine Sicherung, einen Snapshot oder einen Klon aufgenommen, wenn sie in der Anwendungsdefinition explizit referenziert werden oder wenn sie Verweise auf einen der Anwendungs-Namespaces enthalten.

Bevor Sie beginnen

Stellen Sie sicher, dass die Gültigkeitsdauer des AWS-Sitzungstokens für alle länger laufenden S3-Backup-Vorgänge ausreichend ist. Wenn das Token während des Sicherungsvorgangs abläuft, kann der Vorgang fehlschlagen.

- Siehe die "AWS API-Dokumentation" Weitere Informationen zum Prüfen des Ablaufs des aktuellen Sitzungstokens finden Sie hier.
- Siehe die "AWS IAM-Dokumentation" Weitere Informationen zu Anmeldeinformationen für AWS-Ressourcen finden Sie hier.

Erstellen Sie eine Sicherung mithilfe eines CR

Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie. `trident-protect-backup-cr.yaml` .
2. Konfigurieren Sie in der von Ihnen erstellten Datei die folgenden Attribute:

   - **metadata.name**: (*Erforderlich*) Der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und aussagekräftigen Namen für Ihre Umgebung.
   - **spec.applicationRef**: (*Erforderlich*) Der Kubernetes-Name der zu sichernden Anwendung.
   - **spec.appVaultRef**: (*Erforderlich*) Der Name des AppVaults, in dem die Sicherungsinhalte gespeichert werden sollen.
   - **spec.dataMover**: (*Optional*) Eine Zeichenkette, die angibt, welches Sicherungstool für den Sicherungsvorgang verwendet werden soll. Mögliche Werte (Groß-/Kleinschreibung beachten):
     - `Restic`
     - `Kopia`(Standard)
   - **spec.reclaimPolicy**: (*Optional*) Definiert, was mit einem Backup geschieht, wenn es aus seinem Anspruch freigegeben wird. Mögliche Werte:
     - `Delete`
     - `Retain`(Standard)
   - **spec.snapshotRef**: (*Optional*): Name des Snapshots, der als Quelle für die Sicherung verwendet werden soll. Falls keine Daten angegeben werden, wird ein temporärer Snapshot erstellt und gesichert.
   - **metadata.annotations.protect.trident.netapp.io/full-backup** : (*Optional*) Mit dieser Annotation kann festgelegt werden, ob ein Backup nicht inkrementell sein soll. Standardmäßig sind alle Sicherungen inkrementell. Wenn diese Annotation jedoch auf „“ gesetzt ist, … `true` , die Datensicherung wird nicht mehr inkrementell. Sofern nicht anders angegeben, erfolgt die Datensicherung gemäß der Standardeinstellung für inkrementelle Datensicherung. Es empfiehlt sich, regelmäßig eine vollständige Datensicherung durchzuführen und zwischen den vollständigen Datensicherungen inkrementelle Datensicherungen vorzunehmen, um das mit der Wiederherstellung verbundene Risiko zu minimieren.

     Beispiel-YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Backup
metadata:
  namespace: my-app-namespace
  name: my-cr-name
  annotations:
    protect.trident.netapp.io/full-backup: "true"
spec:
  applicationRef: my-application
  appVaultRef: appvault-name
  dataMover: Kopia
```

3. Nachdem Sie die `trident-protect-backup-cr.yaml` Datei mit den korrekten Werten, CR anwenden:

```
kubectl apply -f trident-protect-backup-cr.yaml
```

Erstellen Sie ein Backup mithilfe der Befehlszeile.

Schritte

1. Erstellen Sie die Sicherungskopie und ersetzen Sie die 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> --data-mover
<Kopia_or_Restic> -n <application_namespace>
```

   Sie können optional die `--full-backup` Kennzeichen zur Angabe, ob eine Sicherung nicht inkrementell sein soll. Standardmäßig sind alle Sicherungen inkrementell. Wenn diese Option aktiviert ist, wird die Datensicherung nicht inkrementell durchgeführt. Es empfiehlt sich, regelmäßig eine vollständige Datensicherung durchzuführen und zwischen den vollständigen Datensicherungen inkrementelle Datensicherungen vorzunehmen, um das mit der Wiederherstellung verbundene Risiko zu minimieren.

Erstellen Sie einen Datenschutzplan

Eine Schutzrichtlinie schützt eine App, indem sie nach einem festgelegten Zeitplan Snapshots, Backups oder beides erstellt. Sie können stündlich, täglich, wöchentlich und monatlich Snapshots und Backups erstellen und die Anzahl der aufzubewahrenden Kopien angeben. Sie können eine nicht inkrementelle vollständige Sicherung planen, indem Sie die Annotation „full-backup-rule“ verwenden. Standardmäßig sind alle Sicherungen inkrementell. Durch regelmäßiges Durchführen einer vollständigen Sicherung und inkrementeller Sicherungen zwischendurch können Sie das mit Wiederherstellungen verbundene Risiko verringern.

- Zeitpläne für Snapshots können nur durch die folgende Einstellung erstellt werden: `backupRetention` auf Null und `snapshotRetention` auf einen Wert größer als Null. Einstellung `snapshotRetention` Der Wert Null bedeutet, dass bei geplanten Backups weiterhin Snapshots erstellt werden, diese jedoch nur temporär sind und unmittelbar nach Abschluss des Backups gelöscht werden.
- Clusterbezogene Ressourcen werden in eine Sicherung, einen Snapshot oder einen Klon aufgenommen, wenn sie in der Anwendungsdefinition explizit referenziert werden oder wenn sie Verweise auf einen der Anwendungs-Namespaces enthalten.

Erstellen Sie einen Zeitplan mithilfe eines CR

Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie. `trident-protect-schedule-cr.yaml` .
2. Konfigurieren Sie in der von Ihnen erstellten Datei die folgenden Attribute:
   - **metadata.name**: (*Erforderlich*) Der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und aussagekräftigen Namen für Ihre Umgebung.
   - **spec.dataMover**: (*Optional*) Eine Zeichenkette, die angibt, welches Sicherungstool für den Sicherungsvorgang verwendet werden soll. Mögliche Werte (Groß-/Kleinschreibung beachten):
     - `Restic`
     - `Kopia`(Standard)
   - **spec.applicationRef**: Der Kubernetes-Name der zu sichernden Anwendung.
   - **spec.appVaultRef**: (*Erforderlich*) Der Name des AppVaults, in dem die Sicherungsinhalte gespeichert werden sollen.
   - **spec.backupRetention**: Die Anzahl der aufzubewahrenden Backups. Null gibt an, dass keine Sicherungen erstellt werden sollen (nur Snapshots).
   - **spec.snapshotRetention**: Die Anzahl der aufzubewahrenden Snapshots. Null bedeutet, dass keine Snapshots erstellt werden sollen.
   - **spec.granularity**: Die Häufigkeit, mit der der Zeitplan ausgeführt werden soll. Mögliche Werte und zugehörige Pflichtfelder:
     - `Hourly`(erfordert, dass Sie angeben) `spec.minute`)
     - `Daily`(erfordert, dass Sie angeben) `spec.minute` Und `spec.hour`)
     - `Weekly`(erfordert, dass Sie angeben) `spec.minute, spec.hour` , Und `spec.dayOfWeek`)
     - `Monthly`(erfordert, dass Sie angeben) `spec.minute, spec.hour` , Und `spec.dayOfMonth`)
     - `Custom`
   - **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` . Der Wert muss als Zeichenfolge angegeben werden.
   - **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` . Der Wert muss als Zeichenfolge angegeben werden.
   - **spec.hour**: (*Optional*) Die Stunde des Tages (0 - 23), zu der der Zeitplan ausgeführt werden soll. Dieses Feld ist erforderlich, wenn die Granularität auf „“ eingestellt ist. `Daily` , `Weekly` , oder `Monthly` . Der Wert muss als Zeichenfolge angegeben werden.
   - **spec.minute**: (*Optional*) Die Minute der Stunde (0 – 59), zu der der Zeitplan ausgeführt werden soll. Dieses Feld ist erforderlich, wenn die Granularität auf „“ eingestellt ist. `Hourly` , `Daily` , `Weekly` , oder `Monthly` . Der Wert muss als Zeichenfolge angegeben werden.
   - **metadata.annotations.protect.trident.netapp.io/full-backup-rule**: (*Optional*) Diese Annotation dient zur Angabe der Regel für die Planung einer vollständigen Datensicherung. Sie können es

einstellen auf `always` für eine kontinuierliche vollständige Datensicherung oder zur individuellen Anpassung an Ihre Bedürfnisse. Wenn Sie beispielsweise die tägliche Granularität wählen, können Sie die Wochentage festlegen, an denen eine vollständige Datensicherung erfolgen soll.

Beispiel-YAML für Sicherungs- und Snapshot-Zeitplan:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  namespace: my-app-namespace
  name: my-cr-name
  annotations:
    protect.trident.netapp.io/full-backup-rule: "Monday,Thursday"
spec:
  dataMover: Kopia
  applicationRef: my-application
  appVaultRef: appvault-name
  backupRetention: "15"
  snapshotRetention: "15"
  granularity: Daily
  hour: "0"
  minute: "0"
```

Beispiel-YAML für einen Nur-Snapshot-Zeitplan:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  namespace: my-app-namespace
  name: my-snapshot-schedule
spec:
  applicationRef: my-application
  appVaultRef: appvault-name
  backupRetention: "0"
  snapshotRetention: "15"
  granularity: Daily
  hour: "2"
  minute: "0"
```

3. Nachdem Sie die `trident-protect-schedule-cr.yaml` Datei mit den korrekten Werten, CR anwenden:

```
kubectl apply -f trident-protect-schedule-cr.yaml
```

Erstellen Sie einen Zeitplan mithilfe der Befehlszeile.

Schritte

1. Erstellen Sie den Schutzplan und ersetzen Sie die Werte in Klammern durch Informationen aus Ihrer Umgebung. Beispiel:

   Sie können verwenden `tridentctl-protect create schedule --help` Um detaillierte Hilfeinformationen zu diesem Befehl anzuzeigen.

   ```
   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>
   --full-backup-rule <string>
   ```

   Sie können die `--full-backup-rule` Flagge an `always` für eine kontinuierliche vollständige Datensicherung oder zur individuellen Anpassung an Ihre Bedürfnisse. Wenn Sie beispielsweise die tägliche Granularität wählen, können Sie die Wochentage festlegen, an denen die vollständige Datensicherung erfolgen soll. Verwenden Sie beispielsweise `--full-backup-rule "Monday,Thursday"` Um eine vollständige Datensicherung montags und donnerstags einzuplanen.

   Für Zeitpläne, die nur Momentaufnahmen enthalten, stellen Sie Folgendes ein: `--backup -retention 0` und geben Sie einen Wert größer als 0 an für `--snapshot-retention` .

Löschen eines Snapshots

Löschen Sie die geplanten oder bedarfsgesteuerten Snapshots, die Sie nicht mehr benötigen.

Schritte

1. Entfernen Sie die dem Snapshot zugeordnete Snapshot-CR:

   ```
   kubectl delete snapshot <snapshot_name> -n my-app-namespace
   ```

Sicherung löschen

Löschen Sie die geplanten oder bedarfsgesteuerten Backups, die Sie nicht mehr benötigen.

Stellen Sie sicher, dass die Rückforderungsrichtlinie auf Folgendes eingestellt ist: `Delete` Alle Sicherungsdaten aus dem Objektspeicher entfernen. Die Standardeinstellung der Richtlinie ist `Retain` um versehentlichen Datenverlust zu vermeiden. Wenn die Richtlinie nicht geändert wird `Delete` Die Sicherungsdaten bleiben im Objektspeicher erhalten und müssen manuell gelöscht werden.

Schritte

1. Entfernen Sie die zum Backup verknüpfte Sicherungs-CR:

```
kubectl delete backup <backup_name> -n my-app-namespace
```

Überprüfen Sie den Status eines Sicherungsvorgangs

Über die Befehlszeile können Sie den Status eines laufenden, abgeschlossenen oder fehlgeschlagenen Sicherungsvorgangs überprüfen.

Schritte

1. Verwenden Sie den folgenden Befehl, um den Status des Sicherungsvorgangs abzurufen. Ersetzen Sie dabei die Werte in Klammern durch Informationen aus Ihrer Umgebung:

```
kubectl get backup -n <namespace_name> <my_backup_cr_name> -o jsonpath
='{.status}'
```

Sichern und Wiederherstellen für Azure-NetApp-Dateien (ANF)-Vorgänge aktivieren

Wenn Sie Trident Protect installiert haben, können Sie die speichereffiziente Backup- und Wiederherstellungsfunktionalität für Speicher-Backends aktivieren, die die Speicherklasse azure-netapp-files verwenden und vor Trident 24.06 erstellt wurden. Diese Funktionalität funktioniert mit NFSv4-Volumes und verbraucht keinen zusätzlichen Speicherplatz aus dem Kapazitätspool.

Bevor Sie beginnen

Stellen Sie Folgendes sicher:

- Sie haben Trident Protect installiert.
- Sie haben eine Anwendung in Trident Protect definiert. Diese Anwendung bietet nur eingeschränkten Schutz, bis Sie dieses Verfahren abgeschlossen haben.
- Du hast `azure-netapp-files` Als Standard-Speicherklasse für Ihr Speicher-Backend ausgewählt.

Für Konfigurationsschritte erweitern

1. Führen Sie in Trident die folgenden Schritte aus, 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-files 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 zugehörige 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 reguläre Backup-Funktionalität, die während des Backup-Vorgangs vorübergehend Speicherplatz im Kapazitätspool belegt. Stellen Sie in diesem Fall sicher, dass im Kapazitätspool ausreichend Speicherplatz vorhanden ist, um ein temporäres Volume in der Größe des zu sichernden Volumes zu erstellen.

Ergebnis

Die Anwendung ist für die Datensicherung und -wiederherstellung mit Trident Protect bereit. Jede PVC kann auch von anderen Anwendungen für Backups und Wiederherstellungen verwendet werden.

Anwendungen wiederherstellen

Anwendungen mit Trident Protect wiederherstellen

Mit Trident Protect können Sie Ihre Anwendung aus einem Snapshot oder einer Sicherung wiederherstellen. Die Wiederherstellung aus einem vorhandenen Snapshot ist schneller, wenn die Anwendung im selben Cluster wiederhergestellt wird.

- Wenn Sie eine Anwendung wiederherstellen, werden alle für die Anwendung konfigurierten Ausführungs-Hooks mit der Anwendung wiederhergestellt. Wenn ein Hook für die Nachbearbeitung der Wiederherstellung vorhanden ist, wird dieser automatisch im Rahmen des Wiederherstellungsvorgangs ausgeführt.
- Die Wiederherstellung aus einer Sicherung in einen anderen Namespace oder in den ursprünglichen Namespace wird für Qtree-Volumes unterstützt. Die Wiederherstellung von einem Snapshot in einen anderen Namespace oder in den ursprünglichen Namespace wird für Qtree-Volumes jedoch nicht unterstützt.
- Sie können erweiterte Einstellungen verwenden, um Wiederherstellungsvorgänge anzupassen. Weitere Informationen finden Sie unter "Verwenden Sie die erweiterten Wiederherstellungseinstellungen von Trident Protect".

Wiederherstellung aus einem Backup in einen anderen Namensraum

Wenn Sie eine Sicherung mithilfe eines BackupRestore CR in einen anderen Namespace wiederherstellen, stellt Trident protect die Anwendung in einem neuen Namespace wieder her und erstellt einen Anwendungs-CR für die wiederhergestellte Anwendung. Um die wiederhergestellte Anwendung zu schützen, erstellen Sie bei Bedarf Backups oder Snapshots oder legen Sie einen Schutzzeitplan fest.

Das Wiederherstellen einer Sicherung in einem anderen Namensraum mit bereits vorhandenen Ressourcen ändert keine Ressourcen, die denselben Namen wie die Ressourcen in der Sicherung haben. Um alle Ressourcen im Backup wiederherzustellen, müssen Sie entweder den Ziel-Namespace löschen und neu erstellen oder das Backup in einem neuen Namespace wiederherstellen.

Bevor Sie beginnen

Stellen Sie sicher, dass die Gültigkeitsdauer des AWS-Sitzungstokens für alle länger andauernden S3-Wiederherstellungsvorgänge ausreichend ist. Wenn das Token während des Wiederherstellungsvorgangs abläuft, kann der Vorgang fehlschlagen.

- Siehe die "AWS API-Dokumentation" Weitere Informationen zum Prüfen des Ablaufs des aktuellen Sitzungstokens finden Sie hier.
- Siehe die "AWS IAM-Dokumentation" Weitere Informationen zu Anmeldeinformationen für AWS-Ressourcen finden Sie hier.

Wenn Sie Sicherungen mit Kopia als Datenverschieber wiederherstellen, können Sie optional Anmerkungen im CR angeben oder die CLI verwenden, um das Verhalten des von Kopia verwendeten temporären Speichers zu steuern. Siehe die "Kopia-Dokumentation" Weitere Informationen zu den konfigurierbaren Optionen finden Sie hier. Verwenden Sie die `tridentctl-protect create --help` Befehl für weitere Informationen zum Festlegen von Annotationen mit der Trident protect CLI.

Verwenden Sie einen CR

Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie. `trident-protect-backup-restore-cr.yaml` .
2. Konfigurieren Sie in der von Ihnen erstellten Datei die folgenden Attribute:
   - **metadata.name**: (*Erforderlich*) Der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und aussagekräftigen Namen für Ihre Umgebung.
   - **spec.appArchivePath**: Der Pfad innerhalb von AppVault, in dem die Sicherungsinhalte gespeichert sind. Sie können folgenden Befehl verwenden, um diesen Pfad zu finden:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

   - **spec.appVaultRef**: (*Erforderlich*) Der Name des AppVaults, in dem die Sicherungsinhalte gespeichert sind.
   - **spec.namespaceMapping**: Die Zuordnung des Quell-Namespace der Wiederherstellungsoperation zum Ziel-Namespace. Ersetzen `my-source-namespace` Und `my-destination-namespace` 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"}]
     ```

3. (*Optional*) Falls Sie nur bestimmte Ressourcen der Anwendung für die Wiederherstellung auswählen möchten, fügen Sie Filter hinzu, die Ressourcen mit bestimmten Bezeichnungen ein- oder ausschließen:

   > Trident Protect wählt einige Ressourcen automatisch aufgrund ihrer Beziehung zu den von Ihnen ausgewählten Ressourcen aus. Wenn Sie beispielsweise eine persistente Volume-Claim-Ressource auswählen und diese einen zugehörigen Pod hat, stellt Trident Protect auch den zugehörigen Pod wieder her.

   - **resourceFilter.resourceSelectionCriteria**: (Für die Filterung erforderlich) Verwenden Sie `Include` oder `Exclude` Eine in resourceMatchers definierte Ressource ein- oder ausschließen. Fügen Sie die folgenden resourceMatchers-Parameter hinzu, um die ein- oder auszuschließenden Ressourcen zu definieren:

- **resourceFilter.resourceMatchers**: Ein Array von resourceMatcher-Objekten. Wenn Sie mehrere Elemente in diesem Array definieren, werden diese per OR-Verknüpfung abgeglichen, und die Felder innerhalb jedes Elements (Gruppe, Art, Version) werden per AND-Verknüpfung abgeglichen.
  - **resourceMatchers[].group**: (*Optional*) Gruppe der zu filternden Ressourcen.
  - **resourceMatchers[].kind**: (*Optional*) Art der zu filternden Ressource.
  - **resourceMatchers[].version**: (*Optional*) Version der zu filternden Ressource.
  - **resourceMatchers[].names**: (*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-Selektorzeichenfolge im Kubernetes-Metadatenfeld „name“ der Ressource, wie in der Dokumentation 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 `trident-protect-backup-restore-cr.yaml` Datei mit den korrekten Werten, CR anwenden:

```
kubectl apply -f trident-protect-backup-restore-cr.yaml
```

Verwenden der CLI

Schritte

1. Stellen Sie die Sicherung in einem anderen Namespace wieder her und ersetzen Sie die Werte in Klammern durch Informationen aus Ihrer Umgebung. Der `namespace-mapping` Das Argument verwendet durch Doppelpunkte getrennte Namensräume, um Quell-Namensräume den korrekten

Ziel-Namensräumen im folgenden Format 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 aus einer Sicherung in den ursprünglichen Namensraum

Sie können eine Sicherung jederzeit im ursprünglichen Namensraum wiederherstellen.

Bevor Sie beginnen

Stellen Sie sicher, dass die Gültigkeitsdauer des AWS-Sitzungstokens für alle länger andauernden S3-Wiederherstellungsvorgänge ausreichend ist. Wenn das Token während des Wiederherstellungsvorgangs abläuft, kann der Vorgang fehlschlagen.

- Siehe die "AWS API-Dokumentation" Weitere Informationen zum Prüfen des Ablaufs des aktuellen Sitzungstokens finden Sie hier.
- Siehe die "AWS IAM-Dokumentation" Weitere Informationen zu Anmeldeinformationen für AWS-Ressourcen finden Sie hier.

Wenn Sie Sicherungen mit Kopia als Datenverschieber wiederherstellen, können Sie optional Anmerkungen im CR angeben oder die CLI verwenden, um das Verhalten des von Kopia verwendeten temporären Speichers zu steuern. Siehe die "Kopia-Dokumentation" Weitere Informationen zu den konfigurierbaren Optionen finden Sie hier. Verwenden Sie die `tridentctl-protect create --help` Befehl für weitere Informationen zum Festlegen von Annotationen mit der Trident protect CLI.

Verwenden Sie einen CR

Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie. `trident-protect-backup-ipr-cr.yaml` .
2. Konfigurieren Sie in der von Ihnen erstellten Datei die folgenden Attribute:

   - **metadata.name**: (*Erforderlich*) Der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und aussagekräftigen Namen für Ihre Umgebung.
   - **spec.appArchivePath**: Der Pfad innerhalb von AppVault, in dem die Sicherungsinhalte gespeichert sind. Sie können folgenden Befehl verwenden, um diesen Pfad zu finden:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o jsonpath='{.status.appArchivePath}'
     ```

   - **spec.appVaultRef**: (*Erforderlich*) Der Name des AppVaults, in dem die Sicherungsinhalte gespeichert sind.

     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*) Falls Sie nur bestimmte Ressourcen der Anwendung für die Wiederherstellung auswählen möchten, fügen Sie Filter hinzu, die Ressourcen mit bestimmten Bezeichnungen ein- oder ausschließen:

   > Trident Protect wählt einige Ressourcen automatisch aufgrund ihrer Beziehung zu den von Ihnen ausgewählten Ressourcen aus. Wenn Sie beispielsweise eine persistente Volume-Claim-Ressource auswählen und diese einen zugehörigen Pod hat, stellt Trident Protect auch den zugehörigen Pod wieder her.

   - **resourceFilter.resourceSelectionCriteria**: (Für die Filterung erforderlich) Verwenden Sie `Include` oder `Exclude` Eine in resourceMatchers definierte Ressource ein- oder ausschließen. Fügen Sie die folgenden resourceMatchers-Parameter hinzu, um die ein- oder auszuschließenden Ressourcen zu definieren:
     - **resourceFilter.resourceMatchers**: Ein Array von resourceMatcher-Objekten. Wenn Sie mehrere Elemente in diesem Array definieren, werden diese per OR-Verknüpfung abgeglichen, und die Felder innerhalb jedes Elements (Gruppe, Art, Version) werden per AND-Verknüpfung abgeglichen.

- **resourceMatchers[].group**: (*Optional*) Gruppe der zu filternden Ressourcen.
- **resourceMatchers[].kind**: (*Optional*) Art der zu filternden Ressource.
- **resourceMatchers[].version**: (*Optional*) Version der zu filternden Ressource.
- **resourceMatchers[].names**: (*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-Selektorzeichenfolge im Kubernetes-Metadatenfeld „name“ der Ressource, wie in der Dokumentation 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 `trident-protect-backup-ipr-cr.yaml` Datei mit den korrekten Werten, CR anwenden:

```
kubectl apply -f trident-protect-backup-ipr-cr.yaml
```

Verwenden der CLI

Schritte

1. Stellen Sie die Sicherung im ursprünglichen Namensraum wieder her und ersetzen Sie die Werte in Klammern durch Informationen aus Ihrer Umgebung. Der `backup` Das Argument verwendet einen Namespace und einen Backup-Namen im folgenden Format: `<namespace>/<name>` . Beispiel:

```
tridentctl-protect create backupinplacerestore <my_restore_name> \
--backup <namespace/backup_to_restore> \
-n <application_namespace>
```

Wiederherstellung aus einem Backup auf einem anderen Cluster

Sie können eine Sicherung auf einem anderen Cluster wiederherstellen, falls es ein Problem mit dem ursprünglichen Cluster gibt.

Wenn Sie Sicherungen mit Kopia als Datenverschieber wiederherstellen, können Sie optional Anmerkungen im CR angeben oder die CLI verwenden, um das Verhalten des von Kopia verwendeten temporären Speichers zu steuern. Siehe die "Kopia-Dokumentation" Weitere Informationen zu den konfigurierbaren Optionen finden Sie hier. Verwenden Sie die `tridentctl-protect create --help` Befehl für weitere Informationen zum Festlegen von Annotationen mit der Trident protect CLI.

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 der Quellcluster, in dem das Backup gespeichert ist.
- Stellen Sie sicher, dass Ihre lokale Umgebung beim Ausführen des AppVault CR eine Verbindung zum im AppVault CR definierten Objektspeicher-Bucket herstellen kann. `tridentctl-protect get appvaultcontent` Befehl. Wenn Netzwerkeinschränkungen den Zugriff verhindern, führen Sie stattdessen die Trident Protect-CLI innerhalb eines Pods auf dem Zielcluster aus.
- Stellen Sie sicher, dass die Gültigkeitsdauer des AWS-Sitzungstokens für alle länger andauernden Wiederherstellungsvorgänge ausreichend ist. Wenn das Token während des Wiederherstellungsvorgangs abläuft, kann der Vorgang fehlschlagen.
  - Siehe die "AWS API-Dokumentation" Weitere Informationen zum Prüfen des Ablaufs des aktuellen Sitzungstokens finden Sie hier.
  - Siehe die "AWS-Dokumentation" Weitere Informationen zu Anmeldeinformationen für AWS-Ressourcen finden Sie hier.

Schritte

1. Überprüfen Sie die Verfügbarkeit des AppVault CR auf dem Zielcluster mithilfe des Trident Protect CLI-Plugins:

   ```
   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 Sicherungsinhalte des verfügbaren AppVaults vom Zielcluster an:

```
tridentctl-protect get appvaultcontent <appvault_name> \
--show-resources backup \
--show-paths \
--context <destination_cluster_name>
```

Durch Ausführen dieses Befehls werden die verfügbaren Backups im AppVault angezeigt, einschließlich ihrer Ursprungscluster, zugehörigen 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. Die Anwendung mithilfe des AppVault-Namens und des Archivpfads im Zielcluster wiederherstellen:

Verwenden Sie einen CR

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie. `trident-protect-backup-restore-cr.yaml` .
2. Konfigurieren Sie in der von Ihnen erstellten Datei die folgenden Attribute:
   - **metadata.name**: (*Erforderlich*) Der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und aussagekräftigen Namen für Ihre Umgebung.
   - **spec.appVaultRef**: (*Erforderlich*) Der Name des AppVaults, in dem die Sicherungsinhalte gespeichert sind.
   - **spec.appArchivePath**: Der Pfad innerhalb von AppVault, in dem die Sicherungsinhalte gespeichert sind. Sie können folgenden Befehl verwenden, um diesen Pfad zu finden:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

     Falls BackupRestore CR nicht verfügbar ist, können Sie den in Schritt 2 genannten Befehl verwenden, um den Sicherungsinhalt anzuzeigen.

   - **spec.namespaceMapping**: Die Zuordnung des Quell-Namespace der Wiederherstellungsoperation zum Ziel-Namespace. Ersetzen `my-source-namespace` 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 `trident-protect-backup-restore-cr.yaml` Datei mit den korrekten Werten, CR anwenden:

```
kubectl apply -f trident-protect-backup-restore-cr.yaml
```

Verwenden der CLI

1. Verwenden Sie den folgenden Befehl, um die Anwendung wiederherzustellen, und ersetzen Sie dabei die Werte in Klammern durch Informationen aus Ihrer Umgebung. Das Argument namespace-

mapping verwendet durch Doppelpunkte getrennte Namensräume, um Quell-Namensräume den richtigen Ziel-Namensräumen im Format source1:dest1,source2:dest2 zuzuordnen. Beispiel:

```
tridentctl-protect create backuprestore <restore_name> \
--namespace-mapping <source_to_destination_namespace_mapping> \
--appvault <appvault_name> \
--path <backup_path> \
--context <destination_cluster_name> \
-n <application_namespace>
```

Wiederherstellung aus einem Snapshot in einen anderen Namespace

Sie können Daten aus einem Snapshot mithilfe einer benutzerdefinierten Ressourcendatei (CR-Datei) entweder in einem anderen Namensraum oder im ursprünglichen Quellnamensraum 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 einen Anwendungs-CR für die wiederhergestellte Anwendung. Um die wiederhergestellte Anwendung zu schützen, erstellen Sie bei Bedarf Backups oder Snapshots oder legen Sie einen Schutzzeitplan fest.

SnapshotRestore unterstützt die `spec.storageClassMapping` Attribut, jedoch nur dann, wenn die Quell- und Zielspeicherklassen das gleiche Speicher-Backend verwenden. Wenn Sie versuchen, einen Wiederherstellungszustand wiederherzustellen `StorageClass` Wenn ein anderes Speichersystem verwendet wird, schlägt die Wiederherstellung fehl.

Bevor Sie beginnen

Stellen Sie sicher, dass die Gültigkeitsdauer des AWS-Sitzungstokens für alle länger andauernden S3-Wiederherstellungsvorgänge ausreichend ist. Wenn das Token während des Wiederherstellungsvorgangs abläuft, kann der Vorgang fehlschlagen.

- Siehe die "AWS API-Dokumentation" Weitere Informationen zum Prüfen des Ablaufs des aktuellen Sitzungstokens finden Sie hier.
- Siehe die "AWS IAM-Dokumentation" Weitere Informationen zu Anmeldeinformationen für AWS-Ressourcen finden Sie hier.

Verwenden Sie einen CR

Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie. `trident-protect-snapshot-restore-cr.yaml` .
2. Konfigurieren Sie in der von Ihnen erstellten Datei die folgenden Attribute:
   - **metadata.name**: (*Erforderlich*) Der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und aussagekräftigen Namen für Ihre Umgebung.
   - **spec.appVaultRef**: (*Erforderlich*) Der Name des AppVaults, in dem die Snapshot-Inhalte gespeichert sind.
   - **spec.appArchivePath**: Der Pfad innerhalb von AppVault, in dem die Snapshot-Inhalte gespeichert sind. Sie können folgenden Befehl verwenden, um diesen Pfad zu finden:

     ```
     kubectl get snapshots <SNAPHOT_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

   - **spec.namespaceMapping**: Die Zuordnung des Quell-Namespace der Wiederherstellungsoperation zum Ziel-Namespace. Ersetzen `my-source-namespace` Und `my-destination-namespace` 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"}]
     ```

3. (*Optional*) Falls Sie nur bestimmte Ressourcen der Anwendung für die Wiederherstellung auswählen möchten, fügen Sie Filter hinzu, die Ressourcen mit bestimmten Bezeichnungen ein- oder ausschließen:

   > Trident Protect wählt einige Ressourcen automatisch aufgrund ihrer Beziehung zu den von Ihnen ausgewählten Ressourcen aus. Wenn Sie beispielsweise eine persistente Volume-Claim-Ressource auswählen und diese einen zugehörigen Pod hat, stellt Trident Protect auch den zugehörigen Pod wieder her.

   - **resourceFilter.resourceSelectionCriteria**: (Für die Filterung erforderlich) Verwenden Sie `Include` oder `Exclude` Eine in resourceMatchers definierte Ressource ein- oder ausschließen. Fügen Sie die folgenden resourceMatchers-Parameter hinzu, um die ein- oder auszuschließenden Ressourcen zu definieren:

- **resourceFilter.resourceMatchers**: Ein Array von resourceMatcher-Objekten. Wenn Sie mehrere Elemente in diesem Array definieren, werden diese per OR-Verknüpfung abgeglichen, und die Felder innerhalb jedes Elements (Gruppe, Art, Version) werden per AND-Verknüpfung abgeglichen.
  - **resourceMatchers[].group**: (*Optional*) Gruppe der zu filternden Ressourcen.
  - **resourceMatchers[].kind**: (*Optional*) Art der zu filternden Ressource.
  - **resourceMatchers[].version**: (*Optional*) Version der zu filternden Ressource.
  - **resourceMatchers[].names**: (*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-Selektorzeichenfolge im Kubernetes-Metadatenfeld „name“ der Ressource, wie in der Dokumentation 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 `trident-protect-snapshot-restore-cr.yaml` Datei mit den korrekten Werten, CR anwenden:

```
kubectl apply -f trident-protect-snapshot-restore-cr.yaml
```

Verwenden der CLI

Schritte

1. Stellen Sie den Snapshot in einem anderen Namespace wieder her und ersetzen Sie die Werte in Klammern durch Informationen aus Ihrer Umgebung.

- Der `snapshot` Das Argument verwendet einen Namespace und einen Snapshot-Namen im folgenden Format: `<namespace>/<name>` .
- Der `namespace-mapping` Das Argument verwendet durch Doppelpunkte getrennte Namensräume, um Quell-Namensräume den korrekten Ziel-Namensräumen im folgenden Format 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 aus einem Snapshot in den ursprünglichen Namensraum

Sie können jederzeit einen Snapshot im ursprünglichen Namensraum wiederherstellen.

Bevor Sie beginnen

Stellen Sie sicher, dass die Gültigkeitsdauer des AWS-Sitzungstokens für alle länger andauernden S3-Wiederherstellungsvorgänge ausreichend ist. Wenn das Token während des Wiederherstellungsvorgangs abläuft, kann der Vorgang fehlschlagen.

- Siehe die "AWS API-Dokumentation" Weitere Informationen zum Prüfen des Ablaufs des aktuellen Sitzungstokens finden Sie hier.
- Siehe die "AWS IAM-Dokumentation" Weitere Informationen zu Anmeldeinformationen für AWS-Ressourcen finden Sie hier.

Verwenden Sie einen CR

Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie. `trident-protect-snapshot-ipr-cr.yaml` .
2. Konfigurieren Sie in der von Ihnen erstellten Datei die folgenden Attribute:
   - **metadata.name**: (*Erforderlich*) Der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und aussagekräftigen Namen für Ihre Umgebung.
   - **spec.appVaultRef**: (*Erforderlich*) Der Name des AppVaults, in dem die Snapshot-Inhalte gespeichert sind.
   - **spec.appArchivePath**: Der Pfad innerhalb von AppVault, in dem die Snapshot-Inhalte gespeichert sind. Sie können 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*) Falls Sie nur bestimmte Ressourcen der Anwendung für die Wiederherstellung auswählen möchten, fügen Sie Filter hinzu, die Ressourcen mit bestimmten Bezeichnungen ein- oder ausschließen:

   > Trident Protect wählt einige Ressourcen automatisch aufgrund ihrer Beziehung zu den von Ihnen ausgewählten Ressourcen aus. Wenn Sie beispielsweise eine persistente Volume-Claim-Ressource auswählen und diese einen zugehörigen Pod hat, stellt Trident Protect auch den zugehörigen Pod wieder her.

   - **resourceFilter.resourceSelectionCriteria**: (Für die Filterung erforderlich) Verwenden Sie `Include` oder `Exclude` Eine in resourceMatchers definierte Ressource ein- oder ausschließen. Fügen Sie die folgenden resourceMatchers-Parameter hinzu, um die ein- oder auszuschließenden Ressourcen zu definieren:
     - **resourceFilter.resourceMatchers**: Ein Array von resourceMatcher-Objekten. Wenn Sie mehrere Elemente in diesem Array definieren, werden diese per OR-Verknüpfung abgeglichen, und die Felder innerhalb jedes Elements (Gruppe, Art, Version) werden per AND-Verknüpfung abgeglichen.
       - **resourceMatchers[].group**: (*Optional*) Gruppe der zu filternden Ressourcen.
       - **resourceMatchers[].kind**: (*Optional*) Art der zu filternden Ressource.

- **resourceMatchers[].version**: (*Optional*) Version der zu filternden Ressource.
- **resourceMatchers[].names**: (*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-Selektorzeichenfolge im Kubernetes-Metadatenfeld „name“ der Ressource, wie in der Dokumentation 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 `trident-protect-snapshot-ipr-cr.yaml` Datei mit den korrekten Werten, CR anwenden:

```
kubectl apply -f trident-protect-snapshot-ipr-cr.yaml
```

Verwenden der CLI

Schritte

1. Stellen Sie den Snapshot im ursprünglichen Namespace wieder her und ersetzen Sie die 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 läuft, abgeschlossen ist oder fehlgeschlagen ist.

Schritte

1. Verwenden Sie den folgenden Befehl, um den Status des Wiederherstellungsvorgangs abzurufen. Ersetzen Sie dabei die Werte in Klammern durch Informationen aus Ihrer Umgebung:

```
kubectl get backuprestore -n <namespace_name> <my_restore_cr_name> -o
jsonpath='{.status}'
```

Verwenden Sie die erweiterten Wiederherstellungseinstellungen von Trident Protect

Sie können Wiederherstellungsvorgänge mithilfe erweiterter Einstellungen wie Anmerkungen, Namespace-Einstellungen und Speicheroptionen an Ihre spezifischen Anforderungen anpassen.

Namespace-Annotationen und -Bezeichnungen während Wiederherstellungs- und Failover-Operationen

Bei Wiederherstellungs- und Failover-Vorgängen werden die Bezeichnungen und Annotationen im Ziel-Namensraum so angepasst, dass sie mit den Bezeichnungen und Annotationen im Quell-Namensraum übereinstimmen. Es werden Bezeichnungen oder Annotationen aus dem Quell-Namensraum hinzugefügt, die im Ziel-Namensraum nicht existieren, und alle bereits vorhandenen Bezeichnungen oder Annotationen werden überschrieben, um dem Wert aus dem Quell-Namensraum zu entsprechen. Labels oder Annotationen, die nur im Ziel-Namensraum existieren, bleiben unverändert.

> Wenn Sie Red Hat OpenShift verwenden, ist es wichtig, die entscheidende Rolle von Namespace-Annotationen in OpenShift-Umgebungen zu beachten. Namespace-Anmerkungen stellen sicher, dass wiederhergestellte Pods die entsprechenden Berechtigungen und Sicherheitskonfigurationen einhalten, die durch die OpenShift-Sicherheitskontextbeschränkungen (SCCs) definiert sind, und ohne Berechtigungsprobleme auf Volumes zugreifen können. Weitere Informationen finden Sie unter "Dokumentation zu den Sicherheitskontextbeschränkungen von OpenShift" .

Sie können verhindern, dass bestimmte Annotationen im Ziel-Namespace überschrieben werden, indem Sie die Kubernetes-Umgebungsvariable festlegen. `RESTORE_SKIP_NAMESPACE_ANNOTATIONS` bevor Sie die Wiederherstellungs- oder Failover-Operation durchführen. Beispiel:

```
helm upgrade trident-protect --set
restoreSkipNamespaceAnnotations=<annotation_key_to_skip_1>,<annotation_key
_to_skip_2> --reuse-values
```

Bei der Durchführung einer Wiederherstellungs- oder Failover-Operation werden alle in angegebenen Namespace-Annotationen und -Labels berücksichtigt. `restoreSkipNamespaceAnnotations` Und `restoreSkipNamespaceLabels` sind von der Wiederherstellungs- oder Failover-Operation ausgeschlossen. Stellen Sie sicher, dass diese Einstellungen während der ersten Helm-Installation konfiguriert werden. Weitere Informationen finden Sie unter "Konfigurieren Sie AutoSupport und Namespace-Filteroptionen".

Wenn Sie die Quellanwendung mit Helm installiert haben `--create-namespace` Flagge, besondere Behandlung wird der `name` Legende mit Beschriftung. Während des Wiederherstellungs- oder Failover-Prozesses kopiert Trident Protect diese Bezeichnung in den Ziel-Namespace, aktualisiert aber den Wert auf den Wert des Ziel-Namespace, wenn der Wert aus der Quelle mit dem Quell-Namespace übereinstimmt. Stimmt dieser Wert nicht mit dem Quell-Namespace überein, wird er unverändert in den Ziel-Namespace kopiert.

Beispiel

Das folgende Beispiel zeigt einen Quell- und einen Ziel-Namensraum, die jeweils unterschiedliche Annotationen und Bezeichnungen aufweisen. Sie können den Zustand des Ziel-Namensraums vor und nach der Operation einsehen und erkennen, wie die Annotationen und Labels im Ziel-Namensraum kombiniert oder überschrieben werden.

Vor dem Wiederherstellungs- oder Failover-Vorgang

Die folgende Tabelle veranschaulicht den Zustand der Beispiel-Quell- und Ziel-Namespaces vor der Wiederherstellungs- oder Failover-Operation:

Namensraum	Anmerkungen	Labels
Namespace ns-1 (Quelle)	• annotation.one/key: "updatedvalue" • annotation.two/key: "true"	• Umgebung=Produktion • Compliance=HIPAA • Name=ns-1
Namespace ns-2 (Ziel)	• annotation.one/key: "true" • annotation.three/key: "false"	• Rolle=Datenbank

Nach dem Wiederherstellungsvorgang

Die folgende Tabelle veranschaulicht den Zustand des Beispiel-Ziel-Namespace nach der Wiederherstellungs- oder Failover-Operation. Einige Schlüssel wurden hinzugefügt, einige wurden überschrieben, und die `name` Die Bezeichnung wurde aktualisiert, um dem Ziel-Namespace zu entsprechen:

Namensraum	Anmerkungen	Labels
Namespace ns-2 (Ziel)	• annotation.one/key: "updatedvalue" • annotation.two/key: "true" • annotation.three/key: "false"	• Name=ns-2 • Compliance=HIPAA • Umgebung=Produktion • Rolle=Datenbank

Unterstützte Felder

In diesem Abschnitt werden zusätzliche Felder beschrieben, die für Wiederherstellungsvorgänge verfügbar sind.

Speicherklassenzuordnung

Der `spec.storageClassMapping` Das Attribut definiert eine Zuordnung von einer in der Quellanwendung vorhandenen Speicherklasse zu einer neuen Speicherklasse im Zielcluster. Sie können dies verwenden, wenn Sie Anwendungen zwischen Clustern mit unterschiedlichen Speicherklassen migrieren oder wenn Sie das Speicher-Backend für BackupRestore-Vorgänge ändern.

Beispiel:

```
storageClassMapping:
  - destination: "destinationStorageClass1"
    source: "sourceStorageClass1"
  - destination: "destinationStorageClass2"
    source: "sourceStorageClass2"
```

Unterstützte Annotationen

Dieser Abschnitt listet die unterstützten Annotationen zur Konfiguration verschiedener Verhaltensweisen im System auf. Wenn eine Annotation nicht explizit vom Benutzer festgelegt wird, verwendet das System den Standardwert.

Anmerkung	Typ	Beschreibung	Standardwert
protect.trident.netapp.io/data-mover-timeout-sec	Schnur	Die maximal zulässige Zeit (in Sekunden) für einen Stillstand des Datenübertragungsvorgangs.	"300"
protect.trident.netapp.io/kopia-content-cache-size-limit-mb	Schnur	Die maximale Größenbeschränkung (in Megabyte) für den Kopia-Inhaltscache.	"1000"

Anwendungen mit NetApp SnapMirror und Trident Protect replizieren

Mit Trident Protect können Sie die asynchronen Replikationsfunktionen der NetApp SnapMirror -Technologie nutzen, um Daten und Anwendungsänderungen von einem Speichersystem auf ein anderes zu replizieren, entweder innerhalb desselben Clusters oder zwischen verschiedenen Clustern.

Namespace-Annotationen und -Bezeichnungen während Wiederherstellungs- und Failover-Operationen

Bei Wiederherstellungs- und Failover-Vorgängen werden die Bezeichnungen und Annotationen im Ziel-Namensraum so angepasst, dass sie mit den Bezeichnungen und Annotationen im Quell-Namensraum übereinstimmen. Es werden Bezeichnungen oder Annotationen aus dem Quell-Namensraum hinzugefügt, die im Ziel-Namensraum nicht existieren, und alle bereits vorhandenen Bezeichnungen oder Annotationen werden

überschrieben, um dem Wert aus dem Quell-Namensraum zu entsprechen. Labels oder Annotationen, die nur im Ziel-Namensraum existieren, bleiben unverändert.

> Wenn Sie Red Hat OpenShift verwenden, ist es wichtig, die entscheidende Rolle von Namespace-Annotationen in OpenShift-Umgebungen zu beachten. Namespace-Anmerkungen stellen sicher, dass wiederhergestellte Pods die entsprechenden Berechtigungen und Sicherheitskonfigurationen einhalten, die durch die OpenShift-Sicherheitskontextbeschränkungen (SCCs) definiert sind, und ohne Berechtigungsprobleme auf Volumes zugreifen können. Weitere Informationen finden Sie unter "Dokumentation zu den Sicherheitskontextbeschränkungen von OpenShift" .

Sie können verhindern, dass bestimmte Annotationen im Ziel-Namespace überschrieben werden, indem Sie die Kubernetes-Umgebungsvariable festlegen. `RESTORE_SKIP_NAMESPACE_ANNOTATIONS` bevor Sie die Wiederherstellungs- oder Failover-Operation durchführen. Beispiel:

```
helm upgrade trident-protect --set
restoreSkipNamespaceAnnotations=<annotation_key_to_skip_1>,<annotation_key
_to_skip_2> --reuse-values
```

> Bei der Durchführung einer Wiederherstellungs- oder Failover-Operation werden alle in angegebenen Namespace-Annotationen und -Labels berücksichtigt. `restoreSkipNamespaceAnnotations` Und `restoreSkipNamespaceLabels` sind von der Wiederherstellungs- oder Failover-Operation ausgeschlossen. Stellen Sie sicher, dass diese Einstellungen während der ersten Helm-Installation konfiguriert werden. Weitere Informationen finden Sie unter "Konfigurieren Sie AutoSupport und Namespace-Filteroptionen".

Wenn Sie die Quellanwendung mit Helm installiert haben `--create-namespace` Flagge, besondere Behandlung wird der `name` Legende mit Beschriftung. Während des Wiederherstellungs- oder Failover-Prozesses kopiert Trident Protect diese Bezeichnung in den Ziel-Namespace, aktualisiert aber den Wert auf den Wert des Ziel-Namespace, wenn der Wert aus der Quelle mit dem Quell-Namespace übereinstimmt. Stimmt dieser Wert nicht mit dem Quell-Namespace überein, wird er unverändert in den Ziel-Namespace kopiert.

Beispiel

Das folgende Beispiel zeigt einen Quell- und einen Ziel-Namensraum, die jeweils unterschiedliche Annotationen und Bezeichnungen aufweisen. Sie können den Zustand des Ziel-Namensraums vor und nach der Operation einsehen und erkennen, wie die Annotationen und Labels im Ziel-Namensraum kombiniert oder überschrieben werden.

Vor dem Wiederherstellungs- oder Failover-Vorgang

Die folgende Tabelle veranschaulicht den Zustand der Beispiel-Quell- und Ziel-Namespaces vor der Wiederherstellungs- oder Failover-Operation:

Namensraum	Anmerkungen	Labels
Namespace ns-1 (Quelle)	• annotation.one/key: "updatedvalue" • annotation.two/key: "true"	• Umgebung=Produktion • Compliance=HIPAA • Name=ns-1
Namespace ns-2 (Ziel)	• annotation.one/key: "true" • annotation.three/key: "false"	• Rolle=Datenbank

Nach dem Wiederherstellungsvorgang

Die folgende Tabelle veranschaulicht den Zustand des Beispiel-Ziel-Namespace nach der Wiederherstellungs- oder Failover-Operation. Einige Schlüssel wurden hinzugefügt, einige wurden überschrieben, und die `name` Die Bezeichnung wurde aktualisiert, um dem Ziel-Namespace zu entsprechen:

Namensraum	Anmerkungen	Labels
Namespace ns-2 (Ziel)	• annotation.one/key: "updatedvalue" • annotation.two/key: "true" • annotation.three/key: "false"	• Name=ns-2 • Compliance=HIPAA • Umgebung=Produktion • Rolle=Datenbank

Sie können Trident Protect so konfigurieren, dass Dateisysteme während Datensicherungsvorgängen eingefroren und wieder freigegeben werden. "Erfahren Sie mehr über die Konfiguration des Einfrierens von Dateisystemen mit Trident Protect.".

Ausführungs-Hooks während Failover- und Rückwärtsoperationen

Bei der Verwendung einer AppMirror-Beziehung zum Schutz Ihrer Anwendung gibt es spezifische Verhaltensweisen im Zusammenhang mit Ausführungs-Hooks, die Sie während Failover- und Rückwärtsoperationen beachten sollten.

- Beim Failover werden die Ausführungs-Hooks automatisch vom Quellcluster in den Zielcluster kopiert. Sie müssen sie nicht manuell neu erstellen. Nach einem Failover sind Ausführungs-Hooks in der Anwendung vorhanden und werden bei allen relevanten Aktionen ausgeführt.
- Bei einer umgekehrten Synchronisierung oder einer umgekehrten Resynchronisierung werden alle vorhandenen Ausführungs-Hooks der Anwendung entfernt. Wenn die Quellanwendung zur Zielanwendung wird, sind diese Ausführungs-Hooks ungültig und werden gelöscht, um ihre Ausführung zu verhindern.

Weitere Informationen zu Ausführungshooks finden Sie unter"Trident Protect-Ausführungshooks verwalten" .

Eine Replikationsbeziehung einrichten

Die Einrichtung einer Replikationsbeziehung umfasst Folgendes:

- Auswahl der Häufigkeit, mit der Trident Protect einen App-Snapshot erstellen soll (einschließlich der Kubernetes-Ressourcen der App sowie der Volume-Snapshots für jedes Volume der App).
- Auswahl des Replikationszeitplans (einschließlich Kubernetes-Ressourcen sowie persistenter Volume-

Daten)

- Einstellen des Zeitpunkts für die Aufnahme des Schnappschusses

Schritte

1. Erstellen Sie auf dem Quellcluster einen AppVault für die Quellanwendung. Je nach Speicheranbieter können Sie ein Beispiel in folgendem Format anpassen:"AppVault-Benutzerressourcen" um sich Ihrer Umgebung anzupassen:

Erstellen Sie einen AppVault mithilfe einer CR

a. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie (zum Beispiel). `trident-protect-appvault-primary-source.yaml`).

b. Konfigurieren Sie die folgenden Attribute:

- **metadata.name**: (*Erforderlich*) Der Name der AppVault-Benutzerressource. Notieren Sie sich den gewählten Namen, da andere für eine Replikationsbeziehung benötigte CR-Dateien auf diesen Wert verweisen.
- **spec.providerConfig**: (*Erforderlich*) Speichert die Konfiguration, die für den Zugriff auf den AppVault mit dem angegebenen Provider erforderlich ist. Wählen Sie einen Bucket-Namen und alle weiteren erforderlichen Details für Ihren Anbieter. Notieren Sie sich die von Ihnen gewählten Werte, da andere für eine Replikationsbeziehung benötigte CR-Dateien auf diese Werte verweisen. Siehe"AppVault-Benutzerressourcen" Beispiele für AppVault CRs bei anderen Anbietern finden Sie hier.
- **spec.providerCredentials**: (*Erforderlich*) Speichert Verweise auf alle Anmeldeinformationen, die für den Zugriff auf den AppVault mit dem angegebenen Provider erforderlich sind.
  - **spec.providerCredentials.valueFromSecret**: (*Erforderlich*) Gibt an, dass der Wert der Anmeldeinformationen aus einem Geheimnis stammen soll.
    - **Schlüssel**: (*Erforderlich*) Der gültige Schlüssel des Geheimnisses, aus dem ausgewählt werden soll.
    - **Name**: (*Erforderlich*) Name des Geheimnisses, das den Wert für dieses Feld enthält. Muss sich im selben Namensraum befinden.
  - **spec.providerCredentials.secretAccessKey**: (*Erforderlich*) Der Zugriffsschlüssel, der für den Zugriff auf den Provider verwendet wird. Der **Name** sollte mit **spec.providerCredentials.valueFromSecret.name** übereinstimmen.
- **spec.providerType**: (*Erforderlich*) Legt fest, was die Datensicherung bereitstellt; zum Beispiel NetApp ONTAP S3, generisches S3, Google Cloud oder Microsoft Azure. Mögliche Werte:
  - AWS
  - azurblau
  - gcp
  - generisches-s3
  - ontap-s3
  - Speichergitter-S3

c. Nachdem Sie die `trident-protect-appvault-primary-source.yaml` Datei mit den korrekten Werten, CR anwenden:

```
kubectl apply -f trident-protect-appvault-primary-source.yaml -n
trident-protect
```

Erstellen Sie einen AppVault mithilfe der CLI.

a. Erstellen Sie den AppVault und ersetzen Sie die Werte in Klammern durch Informationen aus Ihrer Umgebung:

```
tridentctl-protect create vault Azure <vault-name> --account
<account-name> --bucket <bucket-name> --secret <secret-name> -n
trident-protect
```

2. Erstellen Sie auf dem Quellcluster die Quellanwendung CR:

Erstellen Sie die Quellanwendung mithilfe eines CR

a. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie (zum Beispiel). `trident-protect-app-source.yaml`).

b. Konfigurieren Sie die folgenden Attribute:

- **metadata.name**: (*Erforderlich*) Der Name der benutzerdefinierten Anwendungsressource. Notieren Sie sich den gewählten Namen, da andere für eine Replikationsbeziehung benötigte CR-Dateien auf diesen Wert verweisen.
- **spec.includedNamespaces**: (*Erforderlich*) Ein Array von Namespaces und zugehörigen Bezeichnungen. Verwenden Sie Namespace-Namen und grenzen Sie optional den Geltungsbereich der Namespaces mit Bezeichnungen ein, um Ressourcen anzugeben, die in den hier aufgeführten Namespaces vorhanden sind. Der Anwendungs-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 `trident-protect-app-source.yaml` Datei mit den korrekten Werten, CR anwenden:

```
kubectl apply -f trident-protect-app-source.yaml -n my-app-
namespace
```

Erstellen Sie die Quellanwendung mithilfe der Befehlszeilenschnittstelle.

a. Erstellen Sie die Quellanwendung. Beispiel:

```
tridentctl-protect create app <my-app-name> --namespaces
<namespaces-to-be-included> -n <my-app-namespace>
```

3. Optional kann auf dem Quellcluster ein Snapshot der Quellanwendung erstellt werden. Dieser Snapshot dient als Grundlage für die Anwendung auf dem Zielcluster. Wenn Sie diesen Schritt überspringen, müssen Sie warten, bis der nächste geplante Snapshot erstellt wird, damit Sie über einen aktuellen Snapshot verfügen. Informationen zum Erstellen eines On-Demand-Snapshots finden Sie unter "Erstellen Sie einen

On-Demand-Snapshot"Die

4. Erstellen Sie auf dem Quellcluster den Replikationsplan CR:

> Zusätzlich zum unten angegebenen Zeitplan wird empfohlen, einen separaten täglichen Snapshot-Zeitplan mit einer Aufbewahrungsdauer von 7 Tagen zu erstellen, um einen gemeinsamen Snapshot zwischen verbundenen ONTAP Clustern zu gewährleisten. Dadurch wird sichergestellt, dass Snapshots bis zu 7 Tage lang verfügbar sind, die Aufbewahrungsdauer kann jedoch an die Bedürfnisse des Benutzers angepasst werden.
>
> Im Falle eines Failovers kann das System diese Snapshots bis zu 7 Tage lang für Rückoperationen nutzen. Dieser Ansatz beschleunigt und optimiert den umgekehrten Prozess, da nur die seit dem letzten Snapshot vorgenommenen Änderungen übertragen werden, nicht aber alle Daten.
>
> Wenn ein bestehender Zeitplan für die Anwendung bereits die gewünschten Aufbewahrungsanforderungen erfüllt, sind keine zusätzlichen Zeitpläne erforderlich.

Erstellen Sie den Replikationszeitplan mithilfe einer CR.

a. Erstellen Sie einen Replikationszeitplan für die Quellanwendung:

i. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie (zum Beispiel). `trident-protect-schedule.yaml`).

ii. Konfigurieren Sie die folgenden Attribute:

- **metadata.name**: (*Erforderlich*) Der Name der benutzerdefinierten Zeitplanressource.
- **spec.appVaultRef**: (*Erforderlich*) Dieser Wert muss mit dem Feld metadata.name des AppVaults für die Quellanwendung übereinstimmen.
- **spec.applicationRef**: (*Erforderlich*) Dieser Wert muss mit dem Feld metadata.name der Quellanwendung CR übereinstimmen.
- **spec.backupRetention**: (*Erforderlich*) Dieses Feld ist erforderlich und muss den Wert 0 haben.
- **spec.enabled**: Muss auf true gesetzt sein.
- **spec.granularity**: Muss auf gesetzt werden `Custom` .
- **spec.recurrenceRule**: Definiert 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
  namespace: my-app-namespace
spec:
  appVaultRef: my-appvault-name
  applicationRef: my-app-name
  backupRetention: "0"
  enabled: true
  granularity: Custom
  recurrenceRule: |-
    DTSTART:20220101T000200Z
    RRULE:FREQ=MINUTELY;INTERVAL=5
  snapshotRetention: "2"
```

i. Nachdem Sie die `trident-protect-schedule.yaml` Datei mit den korrekten Werten, CR anwenden:

```
kubectl apply -f trident-protect-schedule.yaml -n my-app-
namespace
```

Erstellen Sie den Replikationszeitplan mithilfe der Befehlszeilenschnittstelle.

a. Erstellen Sie den Replikationszeitplan und ersetzen Sie die Werte in Klammern durch Informationen aus Ihrer Umgebung:

```
tridentctl-protect create schedule --name appmirror-schedule
--app <my_app_name> --appvault <my_app_vault> --granularity
Custom --recurrence-rule <rule> --snapshot-retention
<snapshot_retention_count> -n <my_app_namespace>
```

Beispiel:

```
tridentctl-protect create schedule --name appmirror-schedule
--app <my_app_name> --appvault <my_app_vault> --granularity
Custom --recurrence-rule  "DTSTART:20220101T000200Z
\nRRULE:FREQ=MINUTELY;INTERVAL=5" --snapshot-retention 2 -n
<my_app_namespace>
```

5. Erstellen Sie auf dem Zielcluster eine AppVault-CR für die Quellanwendung, die mit der auf dem Quellcluster angewendeten AppVault-CR identisch ist, und benennen Sie sie (zum Beispiel). `trident-protect-appvault-primary-destination.yaml`).
6. Wenden Sie die CR an:

```
kubectl apply -f trident-protect-appvault-primary-destination.yaml -n
trident-protect
```

7. Erstellen Sie einen AppVault CR für die Zielanwendung auf dem Zielcluster. Je nach Speicheranbieter können Sie ein Beispiel in folgendem Format anpassen:"AppVault-Benutzerressourcen" um sich Ihrer Umgebung anzupassen:
   a. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie (zum Beispiel). `trident-protect-appvault-secondary-destination.yaml`).
   b. Konfigurieren Sie die folgenden Attribute:
      - **metadata.name**: (*Erforderlich*) Der Name der AppVault-Benutzerressource. Notieren Sie sich den gewählten Namen, da andere für eine Replikationsbeziehung benötigte CR-Dateien auf diesen Wert verweisen.
      - **spec.providerConfig**: (*Erforderlich*) Speichert die Konfiguration, die für den Zugriff auf den AppVault mit dem angegebenen Provider erforderlich ist. Wählen Sie eine `bucketName` und alle weiteren für Ihren Anbieter notwendigen Angaben. Notieren Sie sich die von Ihnen gewählten Werte, da andere für eine Replikationsbeziehung benötigte CR-Dateien auf diese Werte verweisen.

Siehe"AppVault-Benutzerressourcen" Beispiele für AppVault CRs bei anderen Anbietern finden Sie hier.

- **spec.providerCredentials**: (*Erforderlich*) Speichert Verweise auf alle Anmeldeinformationen, die für den Zugriff auf den AppVault mit dem angegebenen Provider erforderlich sind.
  - **spec.providerCredentials.valueFromSecret**: (*Erforderlich*) Gibt an, dass der Wert der Anmeldeinformationen aus einem Geheimnis stammen soll.
    - **Schlüssel**: (*Erforderlich*) Der gültige Schlüssel des Geheimnisses, aus dem ausgewählt werden soll.
    - **Name**: (*Erforderlich*) Name des Geheimnisses, das den Wert für dieses Feld enthält. Muss sich im selben Namensraum befinden.
  - **spec.providerCredentials.secretAccessKey**: (*Erforderlich*) Der Zugriffsschlüssel, der für den Zugriff auf den Provider verwendet wird. Der **Name** sollte mit **spec.providerCredentials.valueFromSecret.name** übereinstimmen.
- **spec.providerType**: (*Erforderlich*) Legt fest, was die Datensicherung bereitstellt; zum Beispiel NetApp ONTAP S3, generisches S3, Google Cloud oder Microsoft Azure. Mögliche Werte:
  - AWS
  - azurblau
  - gcp
  - generisches-s3
  - ontap-s3
  - Speichergitter-S3

c. Nachdem Sie die `trident-protect-appvault-secondary-destination.yaml` Datei mit den korrekten Werten, CR anwenden:

```
kubectl apply -f trident-protect-appvault-secondary-destination.yaml
-n trident-protect
```

8. Erstellen Sie auf dem Zielcluster eine AppMirrorRelationship-CR-Datei:

Erstellen Sie eine AppMirrorRelationship mithilfe eines CR

a. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie (zum Beispiel). `trident-protect-relationship.yaml`).

b. Konfigurieren Sie die folgenden Attribute:

- **metadata.name:** (Erforderlich) Der Name der benutzerdefinierten Ressource AppMirrorRelationship.
- **spec.destinationAppVaultRef**: (*Erforderlich*) Dieser Wert muss mit dem Namen des AppVaults für die Zielanwendung auf dem Zielcluster übereinstimmen.
- **spec.namespaceMapping**: (*Erforderlich*) Die Ziel- und Quell-Namespaces müssen mit dem in der jeweiligen Anwendungs-CR definierten Anwendungs-Namespace übereinstimmen.
- **spec.sourceAppVaultRef**: (*Erforderlich*) Dieser Wert muss mit dem Namen des AppVaults für die Quellanwendung übereinstimmen.
- **spec.sourceApplicationName**: (*Erforderlich*) Dieser Wert muss mit dem Namen der Quellanwendung übereinstimmen, die Sie im Quellanwendungs-CR definiert haben.
- **spec.sourceApplicationUID**: (Erforderlich) Dieser Wert muss mit der UID der Quellanwendung übereinstimmen, die Sie im Quellanwendungs-CR definiert haben.
- **spec.storageClassName**: (*Optional*) Wählen Sie den Namen einer gültigen Speicherklasse im Cluster. Die Speicherklasse muss mit einer ONTAP Speicher-VM verknüpft sein, die mit der Quellumgebung per Peering verbunden ist. Wird keine Speicherklasse angegeben, wird standardmäßig die im Cluster vorhandene Standard-Speicherklasse verwendet.
- **spec.recurrenceRule**: Definiert 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-vsim-2
```

c. Nachdem Sie die `trident-protect-relationship.yaml` Datei mit den korrekten Werten, CR anwenden:

```
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 die Werte in Klammern durch Informationen aus Ihrer Umgebung:

```
tridentctl-protect create appmirrorrelationship
<name_of_appmirorrelationship> --destination-app-vault
<my_vault_name> --source-app-vault <my_vault_name> --recurrence
-rule <rule> --namespace-mapping <ns_mapping> --source-app-id
<source_app_UID> --source-app <my_source_app_name> --storage
-class <storage_class_name> -n <application_namespace>
```

Beispiel:

```
tridentctl-protect create appmirrorrelationship my-amr
--destination-app-vault appvault2 --source-app-vault appvault1
--recurrence-rule
"DTSTART:20220101T000200Z\nRRULE:FREQ=MINUTELY;INTERVAL=5"
--source-app my-app --namespace-mapping "my-source-ns1:my-dest-
ns1,my-source-ns2:my-dest-ns2" --source-app-id 373f24c1-5769-
404c-93c3-5538af6ccc36 --storage-class my-storage-class -n my-
dest-ns1
```

9. (*Optional*) Überprüfen Sie auf dem Zielcluster den Status und die Stellung der Replikationsbeziehung:

```
kubectl get amr -n my-app-namespace <relationship name> -o=jsonpath
='{.status}' | jq
```

Failover zum Zielcluster

Mit Trident Protect können Sie replizierte Anwendungen auf einen Zielcluster verschieben. Dieses Verfahren beendet die Replikationsbeziehung und schaltet die Anwendung auf dem Zielcluster online. Trident Protect stoppt die Anwendung auf dem Quellcluster nicht, wenn sie betriebsbereit war.

Schritte

1. Bearbeiten Sie auf dem Zielcluster die AppMirrorRelationship-CR-Datei (zum Beispiel, `trident-protect-relationship.yaml`) und ändern Sie den Wert von **spec.desiredState** in `Promoted` .
2. Speichern Sie die CR-Datei.
3. Wenden Sie die CR an:

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-namespace
```

4. (*Optional*) Erstellen Sie alle benötigten Schutzzeitpläne für die ausgefallene Anwendung.
5. (*Optional*) Überprüfen Sie den Status und die Stellung der Replikationsbeziehung:

```
kubectl get amr -n my-app-namespace <relationship name> -o=jsonpath
='{.status}' | jq
```

Eine fehlgeschlagene Replikationsbeziehung erneut synchronisieren

Der Resynchronisierungsvorgang stellt die Replikationsbeziehung wieder her. Nach der Durchführung einer Resynchronisierung wird die ursprüngliche Quellanwendung zur laufenden Anwendung, und alle Änderungen, die an der laufenden Anwendung auf dem Zielcluster vorgenommen wurden, werden verworfen.

Der Prozess stoppt die Anwendung auf dem Zielcluster, bevor die Replikation wiederhergestellt wird.

Alle Daten, die während des Failovers in die Zielanwendung geschrieben werden, gehen verloren.

Schritte

1. Optional: Erstellen Sie auf dem Quellcluster einen Snapshot der Quellanwendung. Dadurch wird sichergestellt, dass die neuesten Änderungen aus dem Quellcluster erfasst werden.
2. Bearbeiten Sie auf dem Zielcluster die AppMirrorRelationship-CR-Datei (zum Beispiel, `trident-protect-relationship.yaml`) und ändern Sie den Wert von spec.desiredState auf `Established` .
3. Speichern Sie die CR-Datei.
4. Wenden Sie die CR an:

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-namespace
```

5. Falls Sie auf dem Zielcluster Schutzpläne erstellt haben, um die ausgefallene Anwendung zu schützen, entfernen Sie diese. Alle verbleibenden Zeitpläne führen zu Fehlern bei der Erstellung von Volume-Snapshots.

Umgekehrte Resynchronisierung einer fehlgeschlagenen Replikationsbeziehung

Wenn Sie eine Replikationsbeziehung, die aufgrund eines Fehlers ausgefallen ist, rückgängig machen, wird die Zielanwendung zur Quellanwendung und die Quelle zum Ziel. Änderungen, die während des Failovers an der Zielanwendung vorgenommen werden, bleiben erhalten.

Schritte

1. Löschen Sie auf dem ursprünglichen Zielcluster die AppMirrorRelationship CR. Dadurch wird das Ziel zum Ausgangspunkt. Falls auf dem neuen Zielcluster noch Schutzpläne vorhanden sind, entfernen Sie diese.
2. Richten Sie eine Replikationsbeziehung ein, indem Sie die CR-Dateien, die Sie ursprünglich zum Einrichten der Beziehung verwendet haben, auf die anderen Cluster anwenden.
3. Stellen Sie sicher, dass das neue Ziel (ursprünglicher Quellcluster) mit beiden AppVault CRs konfiguriert ist.
4. Richten Sie eine Replikationsbeziehung auf dem gegenüberliegenden Cluster ein und konfigurieren Sie die Werte für die umgekehrte Richtung.

Replikationsrichtung der Anwendung umkehren

Wenn Sie die Replikationsrichtung umkehren, verschiebt Trident Protect die Anwendung zum Ziel-Speicher-Backend, während die Replikation zurück zum ursprünglichen Quell-Speicher-Backend fortgesetzt wird. Trident Protect stoppt die Quellanwendung und repliziert die Daten zum Ziel, bevor ein Failover zur Zielanwendung erfolgt.

In dieser Situation werden Quelle und Ziel vertauscht.

Schritte

1. Erstellen Sie auf dem Quellcluster einen Shutdown-Snapshot:

Erstellen Sie einen Shutdown-Snapshot mithilfe eines CR

a. Deaktivieren Sie die Zeitpläne der Schutzrichtlinien für die Quellanwendung.

b. Erstellen Sie eine ShutdownSnapshot-CR-Datei:

   i. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie (zum Beispiel). `trident-protect-shutdownsnapshot.yaml`).

   ii. Konfigurieren Sie die folgenden Attribute:

   - **metadata.name**: (*Erforderlich*) Der Name der benutzerdefinierten Ressource.
   - **spec.AppVaultRef**: (*Erforderlich*) Dieser Wert muss mit dem Feld metadata.name des AppVaults für die Quellanwendung übereinstimmen.
   - **spec.ApplicationRef**: (*Erforderlich*) Dieser Wert muss mit dem Feld metadata.name der Quell-CR-Datei der Anwendung 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 `trident-protect-shutdownsnapshot.yaml` Datei mit den korrekten Werten, CR anwenden:

```
kubectl apply -f trident-protect-shutdownsnapshot.yaml -n my-app-
namespace
```

Erstellen Sie einen Shutdown-Snapshot mithilfe der CLI.

a. Erstellen Sie den Shutdown-Snapshot und ersetzen Sie die 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. Rufen Sie auf dem Quellcluster nach Abschluss des Shutdown-Snapshots den Status des Shutdown-Snapshots ab:

```
kubectl get shutdownsnapshot -n my-app-namespace
<shutdown_snapshot_name> -o yaml
```

3. Ermitteln Sie auf dem Quellcluster 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 ein Failover vom neuen Zielcluster zum neuen Quellcluster mit folgender Änderung durch:

> Fügen Sie im zweiten Schritt des Failover-Verfahrens Folgendes hinzu: `spec.promotedSnapshot` Setzen Sie den Wert des Feldes in der AppMirrorRelationship CR-Datei auf den Basisnamen, den Sie in Schritt 3 oben aufgezeichnet haben.

5. Führen Sie die Schritte zur umgekehrten Resynchronisierung durch inUmgekehrte Resynchronisierung einer fehlgeschlagenen Replikationsbeziehung .
6. Aktivieren Sie die Schutzzeitpläne auf dem neuen Quellcluster.

Ergebnis

Die folgenden Aktionen erfolgen aufgrund der umgekehrten Replikation:

- Es wird eine Momentaufnahme der Kubernetes-Ressourcen der ursprünglichen Quellanwendung erstellt.
- Die Pods der ursprünglichen Quell-App werden ordnungsgemäß gestoppt, indem die Kubernetes-Ressourcen der App gelöscht werden (PVCs und PVs bleiben erhalten).
- Nach dem Herunterfahren der Pods werden Snapshots der App-Volumes erstellt und repliziert.
- Die SnapMirror -Beziehungen wurden aufgehoben, wodurch die Zielvolumes zum Lesen/Schreiben bereit sind.
- Die Kubernetes-Ressourcen der App werden aus dem Snapshot vor dem Herunterfahren wiederhergestellt, wobei die Volume-Daten verwendet werden, die nach dem Herunterfahren der ursprünglichen Quell-App repliziert wurden.
- Die Replikation wird in umgekehrter Richtung wiederhergestellt.

Failback-Anwendungen auf den ursprünglichen Quellcluster

Mit Trident Protect können Sie nach einem Failover-Vorgang ein „Failback“ erreichen, indem Sie die folgende Abfolge von Operationen verwenden. In diesem Workflow zur Wiederherstellung der ursprünglichen Replikationsrichtung repliziert (resynchronisiert) Trident Protect alle Anwendungsänderungen zurück in die ursprüngliche Quellanwendung, bevor die Replikationsrichtung umgekehrt wird.

Dieser Prozess beginnt mit einer Beziehung, die einen Failover auf ein Ziel abgeschlossen hat, und umfasst die folgenden Schritte:

- Beginnen Sie mit einem Ausfallzustand.
- Die Replikationsbeziehung wird umgekehrt neu synchronisiert.

> Führen Sie keine normale Resynchronisierungsoperation durch, da dadurch die während des Failover-Vorgangs auf den Zielcluster geschriebenen Daten verworfen werden.

- Die Replikationsrichtung umkehren.

Schritte

1. Führe dieUmgekehrte Resynchronisierung einer fehlgeschlagenen Replikationsbeziehung Schritte.
2. Führe dieReplikationsrichtung der Anwendung umkehren Schritte.

Eine Replikationsbeziehung löschen

Sie können eine Replikationsbeziehung jederzeit löschen. Wenn Sie die Replikationsbeziehung der Anwendung löschen, entstehen zwei separate Anwendungen ohne jegliche Beziehung zueinander.

Schritte

1. Löschen Sie im aktuellen Zielcluster die AppMirrorRelationship-CR:

```
kubectl delete -f trident-protect-relationship.yaml -n my-app-namespace
```

Anwendungen mit Trident Protect migrieren

Sie können Ihre Anwendungen zwischen Clustern oder in verschiedene Speicherklassen migrieren, indem Sie Sicherungsdaten wiederherstellen.

> Bei der Migration einer Anwendung werden alle für die Anwendung konfigurierten Ausführungs-Hooks mit der Anwendung migriert. Wenn ein Hook für die Nachbearbeitung der Wiederherstellung vorhanden ist, wird dieser automatisch im Rahmen des Wiederherstellungsvorgangs ausgeführt.

Sicherungs- und Wiederherstellungsvorgänge

Um Sicherungs- und Wiederherstellungsvorgänge für die folgenden Szenarien durchzuführen, können Sie bestimmte Sicherungs- und Wiederherstellungsaufgaben automatisieren.

Klonen auf denselben Cluster

Um eine Anwendung auf denselben Cluster zu klonen, erstellen Sie einen Snapshot oder eine Sicherung und stellen Sie die Daten auf demselben Cluster wieder her.

Schritte

1. Führen Sie einen der folgenden Schritte aus:
   a. "Erstellen eines Snapshots".
   b. "Erstellen Sie ein Backup".
2. Führen Sie auf demselben Cluster einen der folgenden Schritte aus, je nachdem, ob Sie einen Snapshot oder eine Sicherung erstellt haben:

a. "Stellen Sie Ihre Daten aus dem Snapshot wieder her.".

b. "Stellen Sie Ihre Daten aus der Sicherung wieder her.".

Klonen auf anderen Cluster

Um eine Anwendung auf einen anderen Cluster zu klonen (clusterübergreifendes Klonen), erstellen Sie eine Sicherung auf dem Quellcluster und stellen Sie diese Sicherung anschließend auf dem anderen Cluster wieder her. Stellen Sie sicher, dass Trident Protect auf dem Zielcluster installiert ist.

Sie können eine Anwendung zwischen verschiedenen Clustern replizieren, indem Sie"SnapMirror -Replikation" .

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 der Sicherung wieder her." .

Anwendungen von einer Speicherklasse in eine andere Speicherklasse migrieren

Sie können Anwendungen von einer Speicherklasse in eine andere Speicherklasse migrieren, indem Sie eine Sicherung in der Zielspeicherklasse wiederherstellen.

Zum 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 einer CR wieder her.

Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie. `trident-protect-snapshot-restore-cr.yaml` .
2. Konfigurieren Sie in der von Ihnen erstellten Datei die folgenden Attribute:
   - **metadata.name**: (*Erforderlich*) Der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und aussagekräftigen Namen für Ihre Umgebung.
   - **spec.appArchivePath**: Der Pfad innerhalb von AppVault, in dem die Snapshot-Inhalte gespeichert sind. Sie können folgenden Befehl verwenden, um diesen Pfad zu finden:

     ```
     kubectl get snapshots <my-snapshot-name> -n trident-protect -o jsonpath='{.status.appArchivePath}'
     ```

   - **spec.appVaultRef**: (*Erforderlich*) Der Name des AppVaults, in dem die Snapshot-Inhalte gespeichert sind.
   - **spec.namespaceMapping**: Die Zuordnung des Quell-Namespace der Wiederherstellungsoperation zum Ziel-Namespace. Ersetzen `my-source-namespace` 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. Optional können Sie, falls Sie nur bestimmte Ressourcen der Anwendung für die Wiederherstellung auswählen möchten, Filter hinzufügen, die Ressourcen mit bestimmten Bezeichnungen ein- oder ausschließen:
   - **resourceFilter.resourceSelectionCriteria**: (Für die Filterung erforderlich) Verwenden Sie `include or exclude` Eine in resourceMatchers definierte Ressource ein- oder ausschließen. Fügen Sie die folgenden resourceMatchers-Parameter hinzu, um die ein- oder auszuschließenden Ressourcen zu definieren:
     - **resourceFilter.resourceMatchers**: Ein Array von resourceMatcher-Objekten. Wenn Sie mehrere Elemente in diesem Array definieren, werden diese per OR-Verknüpfung abgeglichen, und die Felder innerhalb jedes Elements (Gruppe, Art, Version) werden per AND-Verknüpfung abgeglichen.
       - **resourceMatchers[].group**: (*Optional*) Gruppe der zu filternden Ressourcen.

- **resourceMatchers[].kind**: (*Optional*) Art der zu filternden Ressource.
- **resourceMatchers[].version**: (*Optional*) Version der zu filternden Ressource.
- **resourceMatchers[].names**: (*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-Selektorzeichenfolge im Kubernetes-Metadatenfeld „name“ der Ressource, wie in der Dokumentation 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 `trident-protect-snapshot-restore-cr.yaml` Datei mit den korrekten Werten, CR anwenden:

```
kubectl apply -f trident-protect-snapshot-restore-cr.yaml
```

Stellen Sie den Snapshot mithilfe der Befehlszeile wieder her.

Schritte

1. Stellen Sie den Snapshot in einem anderen Namespace wieder her und ersetzen Sie die Werte in Klammern durch Informationen aus Ihrer Umgebung.

   - Der `snapshot` Das Argument verwendet einen Namespace und einen Snapshot-Namen im folgenden Format: `<namespace>/<name>` .
   - Der `namespace-mapping` Das Argument verwendet durch Doppelpunkte getrennte Namensräume, um Quell-Namensräume den korrekten Ziel-Namensräumen im folgenden Format 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>
```

Trident Protect-Ausführungshooks verwalten

Ein Ausführungshook ist eine benutzerdefinierte Aktion, die Sie so konfigurieren können, dass sie in Verbindung mit einer Datensicherungsoperation einer verwalteten App ausgeführt wird. Wenn Sie beispielsweise über eine Datenbank-App verfügen, können Sie mithilfe eines Ausführungs-Hooks alle Datenbanktransaktionen vor einem Snapshot anhalten und die Transaktionen nach Abschluss des Snapshots fortsetzen. Dadurch werden anwendungskonsistente Snapshots gewährleistet.

Arten von Ausführungs-Hooks

Trident Protect unterstützt die folgenden Arten von Ausführungs-Hooks, je nachdem, wann sie ausgeführt werden können:

- Vorab-Schnappschuss
- Nach dem Snapshot
- Vorsicherung
- Nach der Sicherung
- Nach der Wiederherstellung
- Nach dem Failover

Reihenfolge der Ausführung

Wenn ein Datenschutzvorgang ausgeführt wird, finden Ausführungs-Hook-Ereignisse in der folgenden Reihenfolge statt:

1. Alle anwendbaren benutzerdefinierten Ausführungs-Hooks vor der Operation werden auf den entsprechenden Containern ausgeführt. Sie können so viele benutzerdefinierte Hooks vor der Operation erstellen und ausführen, wie Sie benötigen, aber die Reihenfolge der Ausführung dieser Hooks vor der Operation ist weder garantiert noch konfigurierbar.
2. Gegebenenfalls kommt es zum Einfrieren des Dateisystems. "Erfahren Sie mehr über die Konfiguration des Einfrierens von Dateisystemen mit Trident Protect.".
3. Der Datenschutzvorgang wird durchgeführt.
4. Eingefrorene Dateisysteme werden gegebenenfalls wieder freigegeben.
5. Alle anwendbaren benutzerdefinierten Ausführungs-Hooks nach der Operation werden auf den entsprechenden Containern ausgeführt. Sie können so viele benutzerdefinierte Post-Operation-Hooks erstellen und ausführen, wie Sie benötigen, aber die Reihenfolge der Ausführung dieser Hooks nach der Operation ist weder garantiert noch konfigurierbar.

Wenn Sie mehrere Ausführungs-Hooks desselben Typs erstellen (z. B. Pre-Snapshot), ist die Ausführungsreihenfolge dieser Hooks nicht garantiert. Die Reihenfolge der Ausführung von Hooks unterschiedlichen Typs ist jedoch gewährleistet. Im Folgenden sehen Sie beispielsweise die Ausführungsreihenfolge einer Konfiguration, die alle verschiedenen Hook-Typen enthält:

1. Vor dem Snapshot ausgeführte Hooks
2. Nach dem Snapshot ausgeführte Hooks
3. Vor der Sicherung ausgeführte Hooks
4. Nach der Sicherung ausgeführte Hooks

Das vorhergehende Befehlsbeispiel gilt nur, wenn Sie eine Sicherung durchführen, die keinen vorhandenen Snapshot verwendet.

Sie sollten Ihre Ausführungs-Hook-Skripte immer testen, bevor Sie sie in einer Produktionsumgebung aktivieren. Mit dem Befehl „kubectl exec“ können Sie die Skripte bequem testen. Nachdem Sie die Ausführungs-Hooks in einer Produktionsumgebung aktiviert haben, testen Sie die resultierenden Snapshots und Backups, um sicherzustellen, dass sie konsistent sind. Sie können dies tun, indem Sie die App in einen temporären Namespace klonen, den Snapshot oder das Backup wiederherstellen und dann die App testen.

Wenn ein Pre-Snapshot-Ausführungs-Hook Kubernetes-Ressourcen hinzufügt, ändert oder entfernt, werden diese Änderungen in den Snapshot oder die Sicherung und in alle nachfolgenden Wiederherstellungsvorgänge einbezogen.

Wichtige Hinweise zu benutzerdefinierten Ausführungs-Hooks

Berücksichtigen Sie Folgendes, wenn Sie Ausführungs-Hooks für Ihre Apps planen.

- Ein Ausführungs-Hook muss ein Skript verwenden, um Aktionen auszuführen. Viele Ausführungs-Hooks können auf dasselbe Skript verweisen.
- Trident Protect verlangt, dass die Skripte, die von den Ausführungs-Hooks verwendet werden, im Format ausführbarer Shell-Skripte geschrieben sind.
- Die Skriptgröße ist auf 96 KB begrenzt.
- Trident Protect verwendet Execution Hook-Einstellungen und alle übereinstimmenden Kriterien, um zu bestimmen, welche Hooks für einen Snapshot-, Backup- oder Wiederherstellungsvorgang anwendbar sind.

Da Ausführungs-Hooks häufig die Funktionalität der Anwendung, für die sie ausgeführt werden, einschränken oder vollständig deaktivieren, sollten Sie immer versuchen, die Ausführungszeit Ihrer benutzerdefinierten Ausführungs-Hooks zu minimieren. Wenn Sie einen Sicherungs- oder Snapshot-Vorgang mit zugehörigen Ausführungs-Hooks starten, ihn dann aber abbrechen, können die Hooks weiterhin ausgeführt werden, wenn der Sicherungs- oder Snapshot-Vorgang bereits begonnen hat. Dies bedeutet, dass die in einem Ausführungs-Hook nach der Sicherung verwendete Logik nicht davon ausgehen kann, dass die Sicherung abgeschlossen wurde.

Ausführungs-Hook-Filter

Wenn Sie einen Ausführungs-Hook für eine Anwendung hinzufügen oder bearbeiten, können Sie dem Ausführungs-Hook Filter hinzufügen, um zu verwalten, mit welchen Containern der Hook übereinstimmt. Filter sind nützlich für Anwendungen, die auf allen Containern dasselbe Container-Image verwenden, aber jedes Image möglicherweise für einen anderen Zweck verwenden (z. B. Elasticsearch). Mithilfe von Filtern können

Sie Szenarien erstellen, in denen Ausführungs-Hooks auf einigen, aber nicht unbedingt allen identischen Containern ausgeführt werden. Wenn Sie mehrere Filter für einen einzelnen Ausführungs-Hook erstellen, werden diese mit einem logischen UND-Operator kombiniert. Sie können bis zu 10 aktive Filter pro Ausführungs-Hook haben.

Jeder Filter, den Sie einem Ausführungs-Hook hinzufügen, verwendet einen regulären Ausdruck, um Container in Ihrem Cluster abzugleichen. Wenn ein Hook mit einem Container übereinstimmt, führt der Hook das zugehörige Skript auf diesem Container aus. Reguläre Ausdrücke für Filter verwenden die Syntax „Regulärer Ausdruck 2“ (RE2), die das Erstellen eines Filters, der Container aus der Liste der Übereinstimmungen ausschließt, nicht unterstützt. Informationen zur Syntax, die Trident Protect für reguläre Ausdrücke in Ausführungshook-Filtern unterstützt, finden Sie unter "Unterstützung der Syntax „Regulärer Ausdruck 2“ (RE2)" .

> Wenn Sie einem Ausführungs-Hook, der nach einem Wiederherstellungs- oder Klonvorgang ausgeführt wird, einen Namespace-Filter hinzufügen und sich die Wiederherstellungs- oder Klonquelle und das Ziel in unterschiedlichen Namespaces befinden, wird der Namespace-Filter nur auf den Ziel-Namespace angewendet.

Beispiele für Ausführungs-Hooks

Besuchen Sie die "NetApp Verda GitHub-Projekt" um echte Ausführungs-Hooks für beliebte Apps wie Apache Cassandra und Elasticsearch herunterzuladen. Sie können sich auch Beispiele ansehen und Ideen für die Strukturierung Ihrer eigenen benutzerdefinierten Ausführungs-Hooks holen.

Erstelle einen Ausführungs-Hook

Sie können mit Trident Protect einen benutzerdefinierten Ausführungs-Hook für eine App erstellen. Sie benötigen die Berechtigungen Eigentümer, Administrator oder Mitglied, um Ausführungs-Hooks zu erstellen.

Verwenden Sie einen CR

Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie. `trident-protect-hook.yaml` .
2. Konfigurieren Sie die folgenden Attribute so, dass sie Ihrer Trident Protect-Umgebung und Clusterkonfiguration entsprechen:
   - **metadata.name**: (*Erforderlich*) Der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und aussagekräftigen Namen für Ihre Umgebung.
   - **spec.applicationRef**: (*Erforderlich*) Der Kubernetes-Name der Anwendung, für die der Ausführungshook ausgeführt werden soll.
   - **spec.stage**: (*Erforderlich*) Eine Zeichenkette, die angibt, in welcher Phase der Aktion der Ausführungs-Hook ausgeführt werden soll. Mögliche Werte:
     - Vor
     - Post
   - **spec.action**: (*Erforderlich*) Eine Zeichenkette, die angibt, welche Aktion der Ausführungshook ausführen wird, vorausgesetzt, alle angegebenen Ausführungshook-Filter stimmen überein. Mögliche Werte:
     - Schnappschuss
     - Sicherung
     - Wiederherstellen
     - Ausfallsicherung
   - **spec.enabled**: (*Optional*) Gibt an, ob dieser Ausführungs-Hook aktiviert oder deaktiviert ist. Wenn kein Wert angegeben ist, ist der Standardwert „true“.
   - **spec.hookSource**: (*Erforderlich*) Eine Zeichenkette, die das Base64-kodierte Hook-Skript enthält.
   - **spec.timeout**: (*Optional*) Eine Zahl, die angibt, wie lange in Minuten der Ausführungs-Hook ausgeführt werden darf. Der Mindestwert beträgt 1 Minute, der Standardwert beträgt 25 Minuten, sofern nichts anderes angegeben ist.
   - **spec.arguments**: (*Optional*) Eine YAML-Liste von Argumenten, die Sie für den Ausführungs-Hook angeben können.
   - **spec.matchingCriteria**: (*Optional*) Eine optionale Liste von Kriterien-Schlüssel-Wert-Paaren, wobei jedes Paar einen Ausführungs-Hook-Filter bildet. Sie können pro Ausführungs-Hook bis zu 10 Filter hinzufügen.
   - **spec.matchingCriteria.type**: (*Optional*) Eine Zeichenkette, die den Typ des Ausführungs-Hook-Filters identifiziert. Mögliche Werte:
     - ContainerImage
     - ContainerName
     - PodName
     - PodLabel
     - NamespaceName
   - **spec.matchingCriteria.value**: (*Optional*) Eine Zeichenkette oder ein regulärer Ausdruck, der den Wert des Ausführungs-Hook-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: IyEvYmluL2Jhc2gKZWNobyAiZXhhbXBsZSBzY3JpcHQiCg==
  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 korrekten Werten gefüllt haben, wenden Sie die CR an:

```
kubectl apply -f trident-protect-hook.yaml
```

Verwenden der CLI

Schritte

1. Erstellen Sie den Ausführungs-Hook und ersetzen Sie die Werte in Klammern durch Informationen aus Ihrer Umgebung. Beispiel:

```
tridentctl-protect create exechook <my_exec_hook_name> --action
<action_type> --app <app_to_use_hook> --stage <pre_or_post_stage>
--source-file <script-file> -n <application_namespace>
```

Einen Ausführungs-Hook manuell ausführen

Sie können einen Ausführungshook manuell ausführen, beispielsweise zu Testzwecken oder wenn Sie den Hook nach einem Fehler manuell erneut ausführen müssen. Sie benötigen Eigentümer-, Administrator- oder Mitgliedsberechtigungen, um Ausführungshooks manuell auszuführen.

Das manuelle Ausführen eines Ausführungshooks besteht aus zwei grundlegenden Schritten:

1. Erstellen Sie eine Ressourcensicherung, die Ressourcen sammelt und eine Sicherungskopie davon erstellt, um festzulegen, wo der Hook ausgeführt wird.
2. Führe den Ausführungshook für das Backup aus.

Schritt 1: Erstellen Sie eine Ressourcensicherung.

Verwenden Sie einen CR

Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie. `trident-protect-resource-backup.yaml` .
2. Konfigurieren Sie die folgenden Attribute so, dass sie Ihrer Trident Protect-Umgebung und Clusterkonfiguration entsprechen:
   - **metadata.name**: (*Erforderlich*) Der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und aussagekräftigen Namen für Ihre Umgebung.
   - **spec.applicationRef**: (*Erforderlich*) Der Kubernetes-Name der Anwendung, für die das Ressourcen-Backup erstellt werden soll.
   - **spec.appVaultRef**: (*Erforderlich*) Der Name des AppVaults, in dem die Sicherungsinhalte gespeichert sind.
   - **spec.appArchivePath**: Der Pfad innerhalb von AppVault, in dem die Sicherungsinhalte gespeichert sind. Sie können folgenden Befehl verwenden, um diesen Pfad zu finden:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

     Beispiel-YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: ResourceBackup
metadata:
  name: example-resource-backup
spec:
  applicationRef: my-app-name
  appVaultRef: my-appvault-name
  appArchivePath: example-resource-backup
```

3. Nachdem Sie die CR-Datei mit den korrekten Werten gefüllt haben, wenden Sie die CR an:

```
kubectl apply -f trident-protect-resource-backup.yaml
```

Verwenden der CLI

Schritte

1. Erstellen Sie die Sicherungskopie und ersetzen Sie die Werte in Klammern durch Informationen aus Ihrer Umgebung. Beispiel:

```
tridentctl protect create resourcebackup <my_backup_name> --app
<my_app_name> --appvault <my_appvault_name> -n
<my_app_namespace> --app-archive-path <app_archive_path>
```

2. Den Status der Datensicherung anzeigen. Sie können diesen Beispielbefehl so oft verwenden, bis der Vorgang abgeschlossen ist:

```
tridentctl protect get resourcebackup -n <my_app_namespace>
<my_backup_name>
```

3. Überprüfen Sie, ob die Datensicherung erfolgreich war:

```
kubectl describe resourcebackup <my_backup_name>
```

Schritt 2: Den Ausführungshook ausführen

Verwenden Sie einen CR

Schritte

1. Erstellen Sie die benutzerdefinierte Ressourcendatei (CR-Datei) und benennen Sie sie. `trident-protect-hook-run.yaml` .
2. Konfigurieren Sie die folgenden Attribute so, dass sie Ihrer Trident Protect-Umgebung und Clusterkonfiguration entsprechen:
   - **metadata.name**: (*Erforderlich*) Der Name dieser benutzerdefinierten Ressource; wählen Sie einen eindeutigen und aussagekräftigen Namen für Ihre Umgebung.
   - **spec.applicationRef**: (*Erforderlich*) Stellen Sie sicher, dass dieser Wert mit dem Anwendungsnamen aus dem ResourceBackup CR übereinstimmt, den Sie in Schritt 1 erstellt haben.
   - **spec.appVaultRef**: (*Erforderlich*) Stellen Sie sicher, dass dieser Wert mit dem appVaultRef aus dem ResourceBackup CR übereinstimmt, den Sie in Schritt 1 erstellt haben.
   - **spec.appArchivePath**: Stellen Sie sicher, dass dieser Wert mit dem appArchivePath des ResourceBackup CR übereinstimmt, den Sie in Schritt 1 erstellt haben.

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

   - **spec.action**: (*Erforderlich*) Eine Zeichenkette, die angibt, welche Aktion der Ausführungshook ausführen wird, vorausgesetzt, alle angegebenen Ausführungshook-Filter stimmen überein. Mögliche Werte:
     - Schnappschuss
     - Sicherung
     - Wiederherstellen
     - Ausfallsicherung
   - **spec.stage**: (*Erforderlich*) Eine Zeichenkette, die angibt, in welcher Phase der Aktion der Ausführungs-Hook ausgeführt werden soll. Dieser Hakenlauf wird in keiner anderen Phase Haken ausführen. Mögliche Werte:
     - Vor
     - Post

     Beispiel-YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: ExecHooksRun
metadata:
  name: example-hook-run
spec:
  applicationRef: my-app-name
  appVaultRef: my-appvault-name
  appArchivePath: example-resource-backup
  stage: Post
  action: Failover
```

3. Nachdem Sie die CR-Datei mit den korrekten Werten gefüllt haben, wenden Sie die CR an:

```
kubectl apply -f trident-protect-hook-run.yaml
```

Verwenden der CLI

Schritte

1. Erstellen Sie die manuelle Ausführungs-Hook-Anforderung:

```
tridentctl protect create exechooksrun <my_exec_hook_run_name>
-n <my_app_namespace> --action snapshot --stage <pre_or_post>
--app <my_app_name> --appvault <my_appvault_name> --path
<my_backup_name>
```

2. Überprüfen Sie den Status des Ausführungs-Hooks. Sie können diesen Befehl so oft ausführen, bis der Vorgang abgeschlossen ist:

```
tridentctl protect get exechooksrun -n <my_app_namespace>
<my_exec_hook_run_name>
```

3. Beschreiben Sie das exechooksrun-Objekt, um die endgültigen Details und den Status anzuzeigen:

```
kubectl -n <my_app_namespace> describe exechooksrun
<my_exec_hook_run_name>
```

Trident Protect deinstallieren

Möglicherweise müssen Sie Trident Protect-Komponenten entfernen, wenn Sie von einer Testversion auf eine Vollversion des Produkts aktualisieren.

Um Trident Protect zu entfernen, führen Sie die folgenden Schritte aus.

Schritte

1. Entfernen Sie die Trident geschützten CR-Dateien:

   Dieser Schritt ist für Version 25.06 und spätere Versionen nicht mehr erforderlich.

   ```
   helm uninstall -n trident-protect trident-protect-crds
   ```

2. Entfernen Sie den Trident Schutz:

   ```
   helm uninstall -n trident-protect trident-protect
   ```

3. Entfernen Sie den Trident Schutz-Namespace:

   ```
   kubectl delete ns trident-protect
   ```

Trident und Trident schützen Blogs

Hier finden Sie einige großartige Blogs zu NetApp Trident und Trident Protect:

Trident -Blogs

- 9. Mai 2025:"Automatische Trident Backend-Konfiguration für FSx für ONTAP mit dem Amazon EKS-Add-on"
- 19. August 2025:"Verbesserung der Datenkonsistenz: Volume Group Snapshots in der OpenShift-Virtualisierung mit Trident"
- 15. April 2025:"NetApp Trident mit Google Cloud NetApp Volumes für das SMB-Protokoll"
- 14. April 2025:"Nutzung des Fiber Channel Protocol mit Trident 25.02 für persistente Speicherung auf Kubernetes"
- 14. April 2025:"Die Leistungsfähigkeit von NetApp ASA r2-Systemen für Kubernetes-Blockspeicher erschließen"
- 31. März 2025:"Vereinfachung der Trident -Installation auf Red Hat OpenShift mit dem neuen Certified Operator"
- 27. März 2025:"Bereitstellung von Trident für KMU mit Google Cloud NetApp Volumes"
- 5. März 2025:"Nahtlose iSCSI-Speicherintegration freischalten: Ein Leitfaden zu FSxN auf ROSA-Clustern für AWS"
- 27. Februar 2025:"Bereitstellen einer Cloud-Identität mit Trident, GKE und Google Cloud NetApp Volumes"
- 12. Dezember 2024:"Einführung der Fibre-Channel-Unterstützung in Trident"
- 26. November 2024:"Trident 25.01: Verbesserung des Kubernetes-Speichererlebnisses mit neuen Funktionen und Verbesserungen"
- 11. November 2024:"NetApp Trident mit Google Cloud NetApp Volumes"
- 29. Oktober 2024:"Amazon FSx for NetApp ONTAP mit Red Hat OpenShift Service auf AWS (ROSA) unter Verwendung von Trident"
- 29. Oktober 2024:"Live-Migration von VMs mit OpenShift-Virtualisierung auf ROSA und Amazon FSx for NetApp ONTAP"
- 8. Juli 2024:"Verwenden von NVMe/TCP zur Nutzung von ONTAP -Speicher für Ihre modernen containerisierten Apps auf Amazon EKS"
- 1. Juli 2024:"Nahtloser Kubernetes-Speicher mit Google Cloud NetApp Volumes Flex und Astra Trident"
- 11. Juni 2024:"ONTAP als Backend-Speicher für die integrierte Image-Registrierung in OpenShift"

Trident schützt Blogs

- 16. Mai 2025:"Automatisieren des Registry-Failovers für die Notfallwiederherstellung mit Trident Protect Post-Restore-Hooks"
- 16. Mai 2025:"Notfallwiederherstellung der OpenShift-Virtualisierung mit NetApp Trident Protect"
- 13. Mai 2025:"Speicherklassenmigration mit Trident Protect Backup Restore"
- 9. Mai 2025:"Skalieren Sie Kubernetes-Anwendungen mit Trident Protect Post-Restore-Hooks neu"
- 3. April 2025:"Trident Protect Power Up: Kubernetes-Replikation für Schutz und Notfallwiederherstellung"

- 13. März 2025:"Absturzsichere Sicherungs- und Wiederherstellungsvorgänge für OpenShift Virtualisierungs-VMs"
- 11. März 2025:"Erweiterung von GitOps-Mustern auf den Anwendungsdatenschutz mit NetApp Trident"
- 3. März 2025:"Trident 25.02: Ein optimiertes Red Hat OpenShift-Erlebnis mit spannenden neuen Funktionen"
- 15. Januar 2025:"Einführung von Trident Protect – rollenbasierte Zugriffskontrolle"
- 11. November 2024: "Wir stellen tridentctl protect vor: die leistungsstarke CLI für Trident protect"
- 11. November 2024:"Kubernetes-gesteuertes Datenmanagement: Die neue Ära mit Trident Protect"

Wissen und Unterstützung

Häufig gestellte Fragen

Hier finden Sie Antworten auf häufig gestellte Fragen zur Installation, Konfiguration, Aktualisierung und Fehlerbehebung von Trident.

Allgemeine Fragen

Wie häufig erscheint Trident ?

Seit der Veröffentlichung am 24.02. erscheint Trident alle vier Monate: im Februar, Juni und Oktober.

Unterstützt Trident alle Funktionen, die in einer bestimmten Kubernetes-Version veröffentlicht werden?

Trident unterstützt normalerweise keine Alpha-Funktionen in Kubernetes. Trident könnte Beta-Funktionen in den beiden Trident -Releases unterstützen, die auf das Kubernetes-Beta-Release folgen.

Ist Trident für seine Funktionsfähigkeit von anderen NetApp -Produkten abhängig?

Trident ist von keiner anderen NetApp -Software abhängig und funktioniert als eigenständige Anwendung. Sie sollten jedoch ein NetApp Backend-Speichergerät verwenden.

Wie erhalte ich die vollständigen Konfigurationsdetails von Trident ?

Verwenden Sie die `tridentctl get` Befehl, um weitere Informationen zu Ihrer Trident -Konfiguration zu erhalten.

Kann ich Kennzahlen darüber erhalten, wie Speicher von Trident bereitgestellt wird?

Ja. Prometheus-Endpunkte, die verwendet werden können, um Informationen über den Betrieb von Trident zu sammeln, wie z. B. die Anzahl der verwalteten Backends, die Anzahl der bereitgestellten Volumes, die verbrauchten Bytes usw. Sie können auch"Cloud Insights" zur Überwachung und Analyse.

Verändert sich die Benutzererfahrung bei der Verwendung von Trident als CSI-Provisioner?

Nein. Hinsichtlich Benutzererfahrung und Funktionalitäten gibt es keine Änderungen. Der verwendete Provisionierungsname lautet: `csi.trident.netapp.io` . Diese Installationsmethode für Trident wird empfohlen, wenn Sie alle neuen Funktionen der aktuellen und zukünftigen Versionen nutzen möchten.

Installieren und verwenden Sie Trident auf einem Kubernetes-Cluster

Unterstützt Trident eine Offline-Installation aus einer privaten Registry?

Ja, Trident kann offline installiert werden. Siehe"Erfahren Sie mehr über die Trident Installation." .

Kann ich Trident per Fernzugriff installieren?

Ja. Trident 18.10 und höher unterstützen die Ferninstallation von jedem beliebigen Rechner aus, der über die entsprechende Funktion verfügt. `kubectl` Zugriff auf den Cluster. Nach `kubectl` Der Zugriff wird verifiziert (z. B. durch Initiieren eines `kubectl get nodes` Führen Sie einen Befehl vom Remote-Rechner aus, um dies zu überprüfen, und 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 Hochverfügbarkeit. Sie sollten die Anzahl der Replikate im Deployment nicht erhöhen. Wenn der Knoten, auf dem Trident installiert ist, ausfällt oder der Pod aus anderen Gründen nicht erreichbar ist, stellt Kubernetes den Pod automatisch auf einem fehlerfreien Knoten in Ihrem Cluster neu bereit. Trident ist nur eine Steuerungsebene, daher sind aktuell installierte Pods nicht betroffen, wenn Trident erneut eingesetzt wird.

Benötigt Trident Zugriff auf den kube-system-Namespace?

Trident liest vom Kubernetes API-Server, um festzustellen, wann Anwendungen neue PVCs anfordern, daher benötigt es Zugriff auf kube-system.

Welche Rollen und Privilegien werden von Trident genutzt?

Das Trident -Installationsprogramm erstellt eine Kubernetes-ClusterRole, die spezifischen Zugriff auf die PersistentVolume-, PersistentVolumeClaim-, StorageClass- und Secret-Ressourcen des Kubernetes-Clusters hat. Siehe"tridentctl-Installation anpassen" .

Kann ich die exakten Manifestdateien, die Trident für die Installation verwendet, lokal generieren?

Bei Bedarf können Sie die Manifestdateien, die Trident für die Installation verwendet, lokal generieren und modifizieren. Siehe"tridentctl-Installation anpassen" .

Kann ich dieselbe ONTAP Backend-SVM für zwei separate Trident Instanzen in zwei separaten Kubernetes-Clustern verwenden?

Obwohl es nicht empfehlenswert ist, können Sie für zwei Trident -Instanzen dieselbe Backend-SVM verwenden. Geben Sie während der Installation für jede Instanz einen eindeutigen Volumennamen an und/oder geben Sie einen eindeutigen `StoragePrefix` Parameter im `setup/backend.json` Datei. Dies dient dazu, sicherzustellen, dass nicht für beide Instanzen dasselbe FlexVol volume verwendet wird.

Ist es möglich, Trident unter ContainerLinux (ehemals CoreOS) zu installieren?

Trident ist einfach ein Kubernetes-Pod und kann überall dort installiert werden, wo Kubernetes läuft.

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 sowohl den Azure NetApp Files -Dienst in Azure als auch den Cloud Volumes Service in GCP.

Fehlerbehebung und Support

Unterstützt NetApp Trident?

Obwohl Trident Open Source ist und kostenlos zur Verfügung gestellt wird, bietet NetApp vollen Support, sofern Ihr NetApp Backend unterstützt wird.

Wie kann ich ein Support-Ticket erstellen?

Um einen Supportfall zu eröffnen, gehen Sie wie folgt vor:

1. Wenden Sie sich an Ihren Support-Account-Manager und lassen Sie sich bei der Erstellung eines Tickets helfen.
2. Stellen Sie einen Supportfall, indem Sie sich an uns wenden. "NetApp Support" .

Wie generiere ich ein Support-Log-Bundle?

Sie können ein Support-Bundle erstellen, indem Sie Folgendes ausführen `tridentctl logs -a` . Zusätzlich zu den im Bundle erfassten Protokollen sollte das Kubelet-Protokoll erfasst werden, um die Mount-Probleme auf der Kubernetes-Seite zu diagnostizieren. Die Vorgehensweise zum Abrufen des Kubelet-Logs variiert je nach Installationsart von Kubernetes.

Was muss ich tun, wenn ich einen Antrag für eine neue Funktion stellen möchte?

Erstellen Sie ein Problem am "Trident GitHub" und erwähnen Sie **RFE** im Betreff und in der Beschreibung des Problems.

Wo kann ich einen Defekt melden?

Erstellen Sie ein Problem am "Trident GitHub" . Bitte fügen Sie alle notwendigen Informationen und Protokolle zum Problem bei.

Was passiert, wenn ich eine kurze Frage zu Trident habe, die ich klären möchte? Gibt es eine Community oder ein Forum?

Bei Fragen, Problemen oder Anliegen kontaktieren Sie uns bitte über unser Trident."Discord-Kanal" oder GitHub.

Das Passwort meines Speichersystems wurde geändert und Trident funktioniert nicht mehr. Wie kann ich es wiederherstellen?

Aktualisieren Sie das Backend-Passwort mit `tridentctl update backend myBackend -f </path/to_new_backend.json> -n trident` . Ersetzen `myBackend` im Beispiel mit Ihrem Backend-Namen, `` `/path/to_new_backend.json `` mit dem Pfad zum richtigen `backend.json` Datei.

Trident kann meinen Kubernetes-Knoten nicht finden. Wie kann ich das beheben?

Es gibt zwei wahrscheinliche Szenarien, warum Trident keinen Kubernetes-Knoten finden kann. Es kann an einem Netzwerkproblem innerhalb von Kubernetes oder an einem DNS-Problem liegen. Der auf jedem Kubernetes-Knoten laufende Trident -Node-Daemonset muss mit dem Trident -Controller kommunizieren können, um den Knoten bei Trident zu registrieren. Wenn Netzwerkänderungen nach der Installation von Trident vorgenommen wurden, tritt dieses Problem nur bei neuen Kubernetes-Knoten auf, die dem Cluster hinzugefügt werden.

Wenn die Trident -Kapsel zerstört wird, gehen dann die Daten verloren?

Die Daten gehen nicht verloren, wenn die Trident Kapsel zerstört wird. Die Metadaten von Trident werden in CRD-Objekten gespeichert. Alle von Trident bereitgestellten PVs funktionieren normal.

Upgrade Trident

Kann ich direkt von einer älteren Version auf eine neuere Version aktualisieren (und dabei einige Versionen überspringen)?

NetApp unterstützt das Upgrade von Trident von einer Hauptversion auf die nächstfolgende Hauptversion. Sie können von Version 18.xx auf 19.xx, von 19.xx auf 20.xx usw. aktualisieren. Sie sollten das Upgrade vor der Produktionsbereitstellung in einer Testumgebung prüfen.

Ist es möglich, Trident auf eine frühere Version herunterzustufen?

Wenn Sie eine Lösung für nach einem Upgrade aufgetretene Fehler, Abhängigkeitsprobleme oder ein fehlgeschlagenes oder unvollständiges Upgrade benötigen, sollten Sie"Trident deinstallieren" und installieren Sie die frühere Version anhand der spezifischen Anweisungen für diese Version neu. Dies ist die einzige empfohlene Methode, um auf eine frühere Version zurückzukehren.

Backends und Volumes verwalten

Muss ich sowohl Management- als auch DataLIFs in einer ONTAP Backend-Definitionsdatei definieren?

Die Management-LIF ist obligatorisch. DataLIF variiert:

- ONTAP SAN: Für iSCSI nicht angeben. Trident Anwendungen"ONTAP Selective LUN Map" um die iSCI LIFs zu ermitteln, die zum Aufbau einer Multipath-Sitzung benötigt werden. Es wird eine Warnung generiert, wenn `dataLIF` ist explizit definiert. Siehe "ONTAP SAN-Konfigurationsoptionen und Beispiele" für Details.
- ONTAP NAS: NetApp empfiehlt die Angabe `dataLIF` . Falls keine Daten angegeben werden, ruft Trident die dataLIFs vom SVM ab. Sie können einen vollqualifizierten Domänennamen (FQDN) angeben, der für die NFS-Mount-Operationen verwendet werden soll. Dadurch können Sie ein Round-Robin-DNS erstellen, um die Last auf mehrere DataLIFs zu verteilen. Siehe"ONTAP NAS-Konfigurationsoptionen und Beispiele" für Details

Kann Trident CHAP für ONTAP -Backends konfigurieren?

Ja. Trident unterstützt bidirektionales CHAP für ONTAP Backends. Dies erfordert eine Einstellung `useCHAP=true` in Ihrer Backend-Konfiguration.

Wie verwalte ich Exportrichtlinien mit Trident?

Ab Version 20.04 kann Trident Exportrichtlinien dynamisch erstellen und verwalten. Dies ermöglicht es dem Speicheradministrator, einen oder mehrere CIDR-Blöcke in seiner Backend-Konfiguration anzugeben, und Trident kann dann Knoten-IPs, die in diese Bereiche fallen, zu einer von ihm erstellten Exportrichtlinie hinzufügen. Auf diese Weise verwaltet Trident automatisch das Hinzufügen und Löschen von Regeln für Knoten mit IPs innerhalb der angegebenen CIDRs.

Können IPv6-Adressen für Management- und DataLIFs verwendet werden?

Trident unterstützt die Definition von IPv6-Adressen für:

- `managementLIF`Und `dataLIF` für ONTAP NAS-Backends.
- `managementLIF`für ONTAP SAN-Backends. Sie können nicht angeben `dataLIF` auf einem ONTAP SAN-Backend.

Trident muss mithilfe der Flagge installiert werden. `--use-ipv6` (für `tridentctl` Installation), `IPv6` (für Trident -Bediener) oder `tridentTPv6` (für die Helm-Installation) damit es über IPv6 funktioniert.

Ist es möglich, das Management-LIF im Backend zu aktualisieren?

Ja, es ist möglich, das Backend-Management-LIF mithilfe des `tridentctl update backend` Befehl.

Ist es möglich, die DataLIF-Datei im Backend zu aktualisieren?

Sie können die DataLIF aktualisieren am `ontap-nas` Und `ontap-nas-economy` nur.

Kann ich in Trident für Kubernetes mehrere Backends erstellen?

Trident kann viele Backends gleichzeitig unterstützen, entweder mit demselben Treiber oder mit verschiedenen Treibern.

Wie speichert Trident die Backend-Zugangsdaten?

Trident speichert die Backend-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, `storagePools` Und `additionalStoragePools` Mithilfe von Parametern wird eine bestimmte Gruppe von Pools ausgewählt.

Wie kann ich sicherstellen, dass Trident nicht von einem bestimmten Backend bereitgestellt wird?

Der `excludeStoragePools` Der Parameter dient dazu, die Menge der Pools zu filtern, die Trident für die Bereitstellung verwendet, und entfernt alle Pools, die übereinstimmen.

Wenn mehrere Backends desselben Typs vorhanden sind, wie wählt Trident das zu verwendende Backend aus?

Wenn mehrere Backends desselben Typs konfiguriert sind, wählt Trident das geeignete Backend anhand der vorhandenen Parameter aus. `StorageClass` Und `PersistentVolumeClaim` . Wenn beispielsweise mehrere ONTAP-NAS-Treiber-Backends vorhanden sind, versucht Trident , die Parameter in der `StorageClass` Und `PersistentVolumeClaim` kombiniert und ein Backend entwickelt, das die aufgeführten Anforderungen erfüllen kann `StorageClass` Und `PersistentVolumeClaim` . Wenn mehrere Backends zur Anfrage passen, wählt Trident nach dem Zufallsprinzip eines davon aus.

Unterstützt Trident bidirektionales CHAP mit Element/ SolidFire?

Ja.

Wie stellt Trident Qtrees auf einem ONTAP -Volume bereit? Wie viele Qtrees können auf einem einzelnen Volume bereitgestellt werden?

Der `ontap-nas-economy` Der Treiber erstellt bis zu 200 Qtrees im selben FlexVol volume (konfigurierbar zwischen 50 und 300), 100.000 Qtrees pro Clusterknoten und 2,4 Millionen pro Cluster. Wenn Sie ein neues Konto eingeben `PersistentVolumeClaim` Der Treiber, der vom Economy-Treiber bedient wird, prüft, ob bereits ein FlexVol volume existiert, das den neuen Qtree bedienen kann. Falls kein FlexVol volume existiert, das den Qtree bedienen kann, wird ein neues FlexVol volume erstellt.

Wie kann ich Unix-Berechtigungen für auf einem ONTAP NAS bereitgestellte Volumes festlegen?

Sie können Unix-Berechtigungen für das von Trident bereitgestellte Volume festlegen, indem Sie einen Parameter in der Backend-Definitionsdatei setzen.

Wie kann ich beim Bereitstellen eines Volumes einen expliziten Satz von ONTAP NFS-Mount-Optionen konfigurieren?

Standardmäßig setzt Trident bei Kubernetes keine Mount-Optionen auf einen Wert. Um die Mount-Optionen in der Kubernetes-Speicherklasse festzulegen, folgen Sie dem angegebenen Beispiel."hier," .

Wie kann ich die bereitgestellten Volumes einer bestimmten Exportrichtlinie zuweisen?

Um den entsprechenden Hosts Zugriff auf ein Volume zu ermöglichen, verwenden Sie die `exportPolicy` Parameter, der in der Backend-Definitionsdatei konfiguriert ist.

Wie richte ich die Volumenverschlüsselung über Trident mit ONTAP ein?

Sie können die Verschlüsselung des von Trident bereitgestellten Volumes mithilfe des Verschlüsselungsparameters in der Backend-Definitionsdatei aktivieren. Weitere Informationen finden Sie unter:"Wie Trident mit NVE und NAE zusammenarbeitet"

Wie lässt sich QoS für ONTAP am besten über Trident implementieren?

Verwenden `StorageClasses` QoS für ONTAP implementieren.

Wie kann ich Thin oder Thick Provisioning über Trident festlegen?

Die ONTAP -Treiber unterstützen sowohl Thin Provisioning als auch Thick Provisioning. Die ONTAP -Treiber verwenden standardmäßig Thin Provisioning. Wenn Thick Provisioning gewünscht ist, sollten Sie entweder die Backend-Definitionsdatei oder die `StorageClass` . Wenn beide konfiguriert sind, `StorageClass` hat Vorrang. Konfigurieren Sie Folgendes für ONTAP:

1. An `StorageClass` , setze die `provisioningType` als dick bezeichnen.
2. Aktivieren Sie in der Backend-Definitionsdatei die Unterstützung für dicke Volumen durch die folgende Einstellung: `backend spaceReserve parameter` als Volumen.

Wie kann ich sicherstellen, dass die verwendeten Volumes nicht gelöscht werden, selbst wenn ich versehentlich die PVC lösche?

Der PVC-Schutz ist ab Kubernetes-Version 1.10 automatisch aktiviert.

Kann ich NFS-PVCs anbauen, die von Trident hergestellt wurden?

Ja. Sie können ein von Trident erstelltes PVC erweitern. Beachten Sie, dass die automatische Volumenvergrößerung eine ONTAP Funktion ist, die für Trident nicht anwendbar ist.

Kann ich ein Volume importieren, während es sich im SnapMirror Datenschutzmodus (DP) oder im Offline-Modus befindet?

Der Volume-Import schlägt fehl, wenn sich das externe Volume im DP-Modus befindet oder offline ist. Sie erhalten 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 das Ressourcenkontingent in einen NetApp -Cluster übersetzt?

Die Speicherressourcenkontingente von Kubernetes sollten funktionieren, solange der NetApp -Speicher über Kapazität verfügt. Wenn der NetApp -Speicher die Kubernetes-Quota-Einstellungen aufgrund mangelnder Kapazität nicht einhalten kann, versucht Trident die Bereitstellung, schlägt aber fehl.

Kann ich mit Trident Volumen-Snapshots erstellen?

Ja. Trident unterstützt das Erstellen von On-Demand-Volume-Snapshots und Persistent Volumes aus Snapshots. Um PVs aus Snapshots zu erstellen, stellen Sie sicher, dass die `VolumeSnapshotDataSource` Die Feature-Gate-Funktion wurde aktiviert.

Welche Treiber unterstützen Trident -Volume-Snapshots?

Ab heute ist die Unterstützung für bedarfsgesteuerte Snapshots für unsere `ontap-nas` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy` , `solidfire-san` , `gcp-cvs` , Und `azure-netapp-files` Backend-Treiber.

Wie erstelle ich mit ONTAP eine Snapshot-Sicherung eines von Trident bereitgestellten Volumes?

Dies ist verfügbar auf `ontap-nas` , `ontap-san` , Und `ontap-nas-flexgroup` Fahrer. Sie können auch eine angeben `snapshotPolicy` für die `ontap-san-economy` Treiber auf FlexVol -Niveau.

Dies ist auch verfügbar auf der `ontap-nas-economy` Treiber, jedoch auf der FlexVol volume Lautstärkeebene und nicht auf der Qtree-Ebene. Um die Möglichkeit zum Erstellen von Snapshots von mit Trident bereitgestellten Volumes zu aktivieren, legen Sie die entsprechende Backend-Parameteroption fest. `snapshotPolicy` auf die gewünschte Snapshot-Richtlinie, wie sie im ONTAP -Backend definiert ist. Vom Speicherkontroller erstellte Snapshots sind Trident nicht bekannt.

Kann ich einen Snapshot-Reservierungsprozentsatz für ein über Trident bereitgestelltes Volume festlegen?

Ja, Sie können einen bestimmten Prozentsatz des Speicherplatzes für die Speicherung der Snapshot-Kopien über Trident reservieren, indem Sie die entsprechende Einstellung vornehmen. `snapshotReserve` Attribut in der Backend-Definitionsdatei. Wenn Sie konfiguriert haben `snapshotPolicy` Und `snapshotReserve` In der Backend-Definitionsdatei wird der Snapshot-Reservierungsprozentsatz entsprechend festgelegt. `snapshotReserve` Der im Backend-File angegebene Prozentsatz. Wenn die `snapshotReserve` Der Prozentsatz wird nicht angegeben, ONTAP verwendet standardmäßig einen Snapshot-Reservierungsprozentsatz von 5. Wenn die `snapshotPolicy` Wenn die Option auf „keine“ gesetzt ist, ist der Snapshot-Reserveprozentsatz auf 0 gesetzt.

Kann ich direkt auf das Volume-Snapshot-Verzeichnis zugreifen und Dateien kopieren?

Ja, Sie können auf das Snapshot-Verzeichnis auf dem von Trident bereitgestellten Volume zugreifen, indem Sie die entsprechende Einstellung vornehmen. `snapshotDir` Parameter in der Backend-Definitionsdatei.

Kann ich SnapMirror für Volumes über Trident einrichten?

Aktuell muss SnapMirror extern über die ONTAP CLI oder den OnCommand System Manager konfiguriert werden.

Wie kann ich persistente Volumes auf einen bestimmten ONTAP Snapshot wiederherstellen?

Um ein Volume auf einen ONTAP Snapshot wiederherzustellen, führen Sie die folgenden Schritte aus:

1. Der Anwendungspod, der das persistente Volume verwendet, soll in den Ruhezustand versetzt werden.
2. Stellen Sie den erforderlichen Snapshot über die ONTAP CLI oder den OnCommand System Manager wieder her.
3. Starten Sie den Anwendungspod neu.

Kann Trident Volumes auf SVMs bereitstellen, die mit einem Load-Sharing Mirror konfiguriert sind?

Für Root-Volumes von SVMs, die Daten über NFS bereitstellen, können Load-Sharing-Mirrors erstellt werden. ONTAP aktualisiert automatisch Load-Sharing-Mirror für Volumes, die von Trident erstellt wurden. Dies kann zu Verzögerungen bei der Montage der Volumes führen. Wenn mit Trident mehrere Volumes erstellt werden, ist die Bereitstellung eines Volumes davon abhängig, dass ONTAP den Load-Sharing-Mirror aktualisiert.

Wie kann ich die Speicherklassennutzung für jeden Kunden/Mandanten separat aufschlüsseln?

Kubernetes erlaubt keine Speicherklassen in Namespaces. Allerdings können Sie mit Kubernetes die Nutzung einer bestimmten Speicherklasse pro Namespace begrenzen, indem Sie Storage Resource Quotas verwenden, die pro Namespace gelten. Um einem bestimmten Namespace den Zugriff auf einen bestimmten Speicher zu verweigern, setzen Sie das Ressourcenkontingent für diese Speicherklasse auf 0.

Fehlerbehebung

Nutzen Sie die hier bereitgestellten Hinweise zur Fehlerbehebung bei Problemen, die während der Installation und Verwendung von Trident auftreten könnten.

Um Hilfe zu Trident zu erhalten, erstellen Sie ein Support-Bundle mit `tridentctl logs -a -n trident` und senden Sie es an den NetApp Support.

Allgemeine Fehlerbehebung

- Wenn die Trident -Kapsel nicht ordnungsgemäß aufsteigt (zum Beispiel, wenn die Trident Kapsel im… feststeckt…) `ContainerCreating` Phase mit weniger als zwei bereiten Containern), laufend `kubectl -n trident describe deployment trident` Und `kubectl -n trident describe pod trident-**` können weitere Erkenntnisse liefern. Abrufen von Kubelet-Logs (zum Beispiel über `journalctl -xeu kubelet`) kann auch hilfreich sein.
- Falls die Trident -Protokolle nicht genügend Informationen enthalten, können Sie versuchen, den Debug-Modus für Trident zu aktivieren, indem Sie den folgenden Befehl übergeben: `-d` Den Installationsparameter entsprechend Ihrer Installationsoption kennzeichnen.

  Bestätigen Sie anschließend, dass der Debug-Modus aktiviert ist. `./tridentctl logs -n trident` und auf der Suche nach `level=debug msg` im Protokoll.

Installiert mit Bediener

```
kubectl patch torc trident -n <namespace> --type=merge -p
'{"spec":{"debug":true}}'
```

Dadurch werden alle Trident Pods neu gestartet, was einige Sekunden dauern kann. Sie können dies überprüfen, indem Sie die Spalte „AGE“ in der Ausgabe betrachten. `kubectl get pod -n trident` .

Für Trident 20.07 und 20.10 verwenden `tprov` anstelle `torc` .

Mit Helm installiert

```
helm upgrade <name> trident-operator-21.07.1-custom.tgz --set
tridentDebug=true`
```

Installiert mit tridentctl

```
./tridentctl uninstall -n trident
./tridentctl install -d -n trident
```

- Sie können auch Debug-Protokolle für jedes Backend erhalten, indem Sie Folgendes einbinden: `debugTraceFlags` in Ihrer Backend-Definition. Zum Beispiel: `debugTraceFlags: {"api":true, "method":true,}` um API-Aufrufe und Methodendurchläufe in den Trident -Protokollen zu erhalten. Bestehende Backends können haben `debugTraceFlags` konfiguriert mit einem `tridentctl backend update` .
- Bei der Verwendung von Red Hat Enterprise Linux CoreOS (RHCOS) ist Folgendes sicherzustellen: `iscsid` ist auf den Worker-Knoten aktiviert und wird standardmäßig gestartet. Dies kann mithilfe von OpenShift MachineConfigs oder durch Modifizierung der Ignition-Vorlagen erfolgen.
- Ein häufiges Problem, das bei der Verwendung von Trident mit "Azure NetApp Files" Dies ist der Fall, wenn die Mandanten- und Clientgeheimnisse aus einer App-Registrierung mit unzureichenden Berechtigungen stammen. Eine vollständige Liste der Trident Anforderungen finden Sie unter"Azure NetApp Files" Konfiguration.
- Falls es Probleme bei der Montage einer Photovoltaikanlage an einem Container gibt, stellen Sie sicher, dass `rpcbind` ist installiert und läuft. Verwenden Sie den erforderlichen Paketmanager für das Host-Betriebssystem und prüfen Sie, ob `rpcbind` läuft. Sie können den Status des `rpcbind` Dienst durch Ausführen eines `systemctl status rpcbind` oder etwas Gleichwertiges.
- Wenn ein Trident -Backend meldet, dass es sich im `failed` Obwohl der Zustand zuvor funktioniert hat, wird er wahrscheinlich durch eine Änderung der mit dem Backend verbundenen SVM/Admin-Zugangsdaten verursacht. Aktualisierung der Backend-Informationen mithilfe von `tridentctl update backend` Oder das Problem lässt sich durch kurzes Aufprallen des Trident -Pods beheben.
- Falls bei der Installation von Trident mit Docker als Container-Laufzeitumgebung Berechtigungsprobleme auftreten, versuchen Sie die Installation von Trident mit der `--in cluster=false` Flagge. Dadurch wird kein Installations-Pod verwendet und Berechtigungsprobleme, die aufgrund von … auftreten können, werden vermieden. `trident-installer` Benutzer.
- Verwenden Sie die `uninstall parameter <Uninstalling Trident>` zum Aufräumen nach einem fehlgeschlagenen Lauf. Standardmäßig entfernt das Skript die von Trident erstellten CRDs nicht, sodass

eine Deinstallation und Neuinstallation auch während einer laufenden Bereitstellung sicher ist.

- Wenn Sie auf eine frühere Version von Trident downgraden möchten, führen Sie zuerst den folgenden Befehl aus: `tridentctl uninstall` Befehl zum Entfernen des Trident. Laden Sie die gewünschte Datei herunter "Trident -Version" und installieren Sie mit dem `tridentctl install` Befehl.
- Nach erfolgreicher Installation, falls ein PVC-Rohr im System feststeckt `Pending` Phase, laufend `kubectl describe pvc` kann zusätzliche Informationen darüber liefern, warum Trident es versäumt hat, einen PV für diesen PVC bereitzustellen.

Fehlgeschlagener Trident Einsatz mit dem Bediener

Wenn Sie Trident mithilfe des Operators bereitstellen, wird der Status von `TridentOrchestrator` Änderungen von `Installing` Zu `Installed` . Wenn Sie die `Failed` Wenn der Status nicht gegeben ist 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
```

Die Analyse der Protokolle des Trident-Operator-Containers kann Aufschluss darüber geben, wo das Problem liegt. Ein solches Problem könnte beispielsweise die Unfähigkeit sein, die benötigten Container-Images aus Upstream-Registries in einer Airgap-Umgebung abzurufen.

Um zu verstehen, warum die Installation von Trident fehlgeschlagen ist, sollten Sie sich Folgendes ansehen: `TridentOrchestrator` Status.

```
kubectl 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 weist darauf hin, dass bereits ein Eintrag existiert. `TridentOrchestrator` Das wurde zur Installation von Trident verwendet. Da jeder Kubernetes-Cluster nur eine Instanz von Trident enthalten kann, stellt der Betreiber sicher, dass zu jedem Zeitpunkt nur eine aktive Instanz existiert. `TridentOrchestrator` dass es erzeugen kann.

Darüber hinaus kann die Beobachtung des Zustands der Trident -Pods oft Aufschluss darüber geben, ob etwas nicht stimmt.

```
kubectl get pods -n trident

NAME                                READY   STATUS             RESTARTS
AGE
trident-csi-4p5kq                   1/2     ImagePullBackOff   0
5m18s
trident-csi-6f45bfd8b6-vfrkw        4/5     ImagePullBackOff   0
5m19s
trident-csi-9q5xc                   1/2     ImagePullBackOff   0
5m18s
trident-csi-9v95z                   1/2     ImagePullBackOff   0
5m18s
trident-operator-766f7b8658-ldzsv   1/1     Running            0
8m17s
```

Man kann deutlich sehen, dass die Pods nicht vollständig initialisiert werden können, weil ein oder mehrere Container-Images nicht abgerufen wurden.

Um das Problem zu beheben, sollten Sie die folgende Datei bearbeiten: `TridentOrchestrator` CR. Alternativ können Sie löschen `TridentOrchestrator` und erstellen Sie eine neue mit der geänderten und genauen Definition.

Fehlgeschlagene Trident Bereitstellung mit `tridentctl`

Um herauszufinden, was schiefgelaufen ist, könnten Sie das Installationsprogramm erneut ausführen.`-d` Argument, das den Debug-Modus aktiviert und Ihnen hilft, das Problem zu verstehen:

```
./tridentctl install -n trident -d
```

Nachdem das Problem behoben wurde, können Sie die Installation wie folgt bereinigen und anschließend das Programm ausführen. `tridentctl install` Befehl erneut:

```
./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.
```

Trident und CRDs vollständig entfernen

Sie können Trident und alle erstellten CRDs sowie die zugehörigen benutzerdefinierten Ressourcen vollständig entfernen.

Dies kann nicht rückgängig gemacht werden. Tun Sie dies nur, wenn Sie eine komplett neue Installation von Trident wünschen. Informationen zur Deinstallation von Trident ohne Entfernung der CRDs finden Sie unter"Trident deinstallieren" .

Trident -Betreiber

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>tridentctl</code>

Um CRDs nach der Deinstallation von Trident vollständig zu entfernen, verwenden Sie `tridentctl`

```
tridentctl obliviate crd
```

Fehler beim Entfernen von NVMe-Knoten aus dem Staging-Bereich bei Verwendung von RWX-Raw-Block-Namespaces in Kubernetes 1.26

Wenn Sie Kubernetes 1.26 verwenden, kann das Unstaging von Knoten fehlschlagen, wenn NVMe/TCP mit RWX-Raw-Block-Namespaces verwendet wird. Die folgenden Szenarien bieten eine Umgehungsmöglichkeit für den Fehler. Alternativ können Sie Kubernetes auf Version 1.27 aktualisieren.

Namespace und Pod gelöscht

Stellen Sie sich ein Szenario vor, in dem ein von Trident verwalteter Namespace (persistentes NVMe-Volume) an einen Pod angehängt ist. Wenn Sie den Namespace direkt vom ONTAP -Backend löschen, bleibt der Unstaging-Prozess hängen, nachdem Sie versucht haben, den Pod zu löschen. Dieses Szenario hat keine Auswirkungen auf den Kubernetes-Cluster oder andere Funktionen.

Problemumgehung

Das persistente Volume (das diesem Namespace entspricht) wird vom jeweiligen Knoten ausgehängt und gelöscht.

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.
.Problemumgehung
Aktivieren Sie dataLIFS, um die volle Funktionalität wiederherzustellen.
```

Gelöschte Namensraumzuordnung

```
 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.
.Problemumgehung
Füge die `hostNQN` zurück zum Subsystem.
```

NFSv4.2-Clients melden nach dem Upgrade von ONTAP „ungültiges Argument“, wenn erwartet wird, dass „v4.2-xattrs“ aktiviert ist

Nach dem Upgrade von ONTAP melden NFSv4.2-Clients möglicherweise Fehler aufgrund „ungültiger Argumente“, wenn sie versuchen, NFSv4.2-Exporte zu mounten. Dieses Problem tritt auf, wenn die `v4.2-xattrs` Die Option ist auf der SVM nicht aktiviert. .Workaround Aktivieren Sie die `v4.2-xattrs` Option auf dem SVM oder Upgrade auf ONTAP 9.12.1 oder höher, wo diese Option standardmäßig aktiviert ist.

Support

NetApp bietet Trident auf vielfältige Weise Unterstützung. Umfangreiche kostenlose Selbsthilfeoptionen stehen rund um die Uhr zur Verfügung, wie zum Beispiel Artikel in der Wissensdatenbank (KB) und ein Discord-Kanal.

Trident Unterstützung

Trident bietet je nach Version drei Support-Stufen an. Siehe"NetApp Softwareversionsunterstützung für Definitionen" .

Volle Unterstützung

Trident bietet zwölf Monate lang vollen Support ab dem Veröffentlichungsdatum.

Eingeschränkte Unterstützung

Trident bietet nur eingeschränkten Support für die Monate 13 bis 24 ab dem Veröffentlichungsdatum.

Selbstversorgung

Die Trident Dokumentation ist für die Monate 25 bis 36 ab dem Veröffentlichungsdatum verfügbar.

Version	Volle Unterstützung	Eingeschränkte Unterstützung	Selbstversorgung

"25,06"	Juni 2026	Juni 2027	Juni 2028
"25,02"	Februar 2026	Februar 2027	Februar 2028
"24,10"	Oktober 2025	Oktober 2026	Oktober 2027
"24,06"	Juni 2025	Juni 2026	Juni 2027
"24,02"	Februar 2025	Februar 2026	Februar 2027
"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

Selbstversorgung

Eine umfassende Liste von Artikeln zur Fehlerbehebung finden Sie unter: "NetApp Wissensdatenbank (Anmeldung erforderlich)" .

Unterstützung durch die Gemeinschaft

Auf unserer Plattform gibt es eine lebendige öffentliche Community von Container-Nutzern (einschließlich Trident -Entwicklern)."Discord-Kanal" . Dies ist ein hervorragender Ort, um allgemeine Fragen zum Projekt zu stellen und verwandte Themen mit Gleichgesinnten zu diskutieren.

Technischer Support von NetApp

Um Hilfe zu Trident zu erhalten, erstellen Sie ein Support-Bundle mit `tridentctl logs -a -n trident` und senden Sie es an `NetApp Support <Getting Help>`.

Weitere Informationen

- "Trident -Ressourcen"
- "Kubernetes Hub"

Referenz

Trident Häfen

Erfahren Sie mehr über die Anschlüsse, die Trident zur Kommunikation nutzt.

Trident Häfen

Trident verwendet die folgenden Ports für die Kommunikation innerhalb von Kubernetes:

Hafen	Zweck
8443	Backchannel HTTPS
8001	Prometheus-Metriken-Endpunkt
8000	Trident REST-Server
17546	Liveness-/Readiness-Probe-Port, der von Trident -DaemonSet-Pods verwendet wird

Der Anschluss für die Lebendigkeits-/Bereitschaftsprüfung kann während der Installation mithilfe des `--probe-port` Flagge. Es ist wichtig sicherzustellen, dass dieser Port nicht von einem anderen Prozess auf den Worker-Knoten verwendet wird.

Trident REST-API

Während"tridentctl-Befehle und Optionen" Dies ist die einfachste Möglichkeit, mit der Trident REST API zu interagieren; Sie können aber auch direkt den REST-Endpunkt verwenden, wenn Sie dies bevorzugen.

Wann sollte die REST-API verwendet werden?

Die REST-API ist nützlich für fortgeschrittene Installationen, die Trident als eigenständige Binärdatei in Nicht-Kubernetes-Bereitstellungen verwenden.

Für mehr Sicherheit, das Trident `REST API` Beim Ausführen innerhalb eines Pods ist die Ausführung standardmäßig auf localhost beschränkt. Um dieses Verhalten zu ändern, müssen Sie die Einstellungen von Trident anpassen. `-address` Argument in seiner Pod-Konfiguration.

Verwendung der REST-API

Beispiele für den Aufruf dieser APIs erhalten Sie durch Übergeben des Debug-Logs.(`-d`) Flagge. Weitere Informationen finden Sie unter"Trident mit tridentctl verwalten" .

Die API funktioniert wie folgt:

ERHALTEN

`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 angegebenen Objekts ab.

POST

`POST <trident-address>/trident/v1/<object-type>`

Erzeugt ein Objekt des angegebenen Typs.

- Für die Erstellung des Objekts ist eine JSON-Konfiguration erforderlich. Die Spezifikation der einzelnen Objekttypen finden Sie unter"Trident mit tridentctl verwalten" .
- Existiert das Objekt bereits, verhält es sich unterschiedlich: Die Backends aktualisieren das bestehende Objekt, während bei allen anderen Objekttypen der Vorgang fehlschlägt.

LÖSCHEN

`DELETE <trident-address>/trident/v1/<object-type>/<object-name>`

Löscht die angegebene Ressource.

> Mit Backends oder Speicherklassen verknüpfte Volumes bleiben bestehen; diese müssen separat gelöscht werden. Weitere Informationen finden Sie unter"Trident mit tridentctl verwalten" .

Befehlszeilenoptionen

Trident stellt mehrere Befehlszeilenoptionen für den Trident -Orchestrator bereit. Sie können diese Optionen nutzen, um Ihre Bereitstellung anzupassen.

Protokollierung

`-debug`

Aktiviert die Debug-Ausgabe.

`-loglevel <level>`

Legt den Protokollierungsgrad fest (debug, info, warn, error, fatal). Standardmäßig wird die Info-Seite angezeigt.

Kubernetes

`-k8s_pod`

Nutzen Sie diese Option oder `-k8s_api_server` um die Kubernetes-Unterstützung zu aktivieren. Durch diese Einstellung verwendet Trident die Anmeldeinformationen des Kubernetes-Dienstkontos des übergeordneten Pods, um den API-Server zu kontaktieren. Dies funktioniert nur, wenn Trident als Pod in einem Kubernetes-Cluster mit aktivierten Service Accounts ausgeführt wird.

-k8s_api_server <insecure-address:insecure-port>

Nutzen Sie diese Option oder `-k8s_pod` um die Kubernetes-Unterstützung zu aktivieren. Wenn angegeben, stellt Trident über die bereitgestellte unsichere Adresse und den Port eine Verbindung zum Kubernetes-API-Server her. Dies ermöglicht den Einsatz von Trident außerhalb eines Pods; allerdings werden dabei nur unsichere Verbindungen zum API-Server unterstützt. Um eine sichere Verbindung herzustellen, stellen Sie Trident in einem Pod mit dem bereit. `-k8s_pod` Option.

Docker

-volume_driver <name>

Name des Treibers, der bei der Registrierung des Docker-Plugins verwendet wird. Standardmäßig `netapp` .

-driver_port <port-number>

Lauschen Sie an diesem Port anstatt an einem UNIX-Domain-Socket.

-config <file>

Erforderlich; Sie müssen diesen Pfad zu einer Backend-Konfigurationsdatei angeben.

AUSRUHEN

-address <ip-or-host>

Gibt die Adresse an, an der der REST-Server von Trident lauschen soll. Standardmäßig wird localhost verwendet. Beim Lauschen auf localhost und Ausführen innerhalb eines Kubernetes-Pods ist die REST-Schnittstelle von außerhalb des Pods nicht direkt zugänglich. Verwenden `-address ""` um die REST-Schnittstelle von der Pod-IP-Adresse aus zugänglich zu machen.

Die Trident REST-Schnittstelle kann so konfiguriert werden, dass sie nur unter 127.0.0.1 (für IPv4) oder [::1] (für IPv6) lauscht und Anfragen beantwortet.

-port <port-number>

Gibt den Port an, an dem der REST-Server von Trident lauschen soll. Standardwert ist 8000.

-rest

Aktiviert die REST-Schnittstelle. Standardmäßig auf „true“ gesetzt.

Kubernetes- und Trident Objekte

Sie können mit Kubernetes und Trident über REST-APIs interagieren, indem Sie Ressourcenobjekte lesen und schreiben. Es gibt mehrere Ressourcenobjekte, die die Beziehung zwischen Kubernetes und Trident, Trident und Speicher sowie Kubernetes und Speicher festlegen. Einige dieser Objekte werden über Kubernetes verwaltet, die anderen über Trident.

Wie interagieren die Objekte miteinander?

Am einfachsten lassen sich die Objekte, ihr Zweck und ihre Interaktion verstehen, indem man eine einzelne Speicheranfrage eines Kubernetes-Benutzers verfolgt:

1. Ein Benutzer erstellt einen `PersistentVolumeClaim` eine neue anfordern `PersistentVolume` einer bestimmten Größe aus einem Kubernetes `StorageClass` das zuvor vom Administrator konfiguriert wurde.
2. Kubernetes `StorageClass` identifiziert Trident als seinen Provisioner und enthält Parameter, die Trident mitteilen, wie ein Volume für die angeforderte Klasse bereitgestellt werden soll.
3. Trident betrachtet sich selbst `StorageClass` mit demselben Namen, der die Übereinstimmung identifiziert `Backends` Und `StoragePools` dass es zur Bereitstellung von Datenträgern für die Klasse verwendet werden kann.
4. Trident stellt Speicherplatz auf einem passenden Backend bereit und erstellt zwei Objekte: ein `PersistentVolume` in Kubernetes, das Kubernetes mitteilt, wie das Volume gefunden, eingebunden und behandelt wird, und ein Volume in Trident , das die Beziehung zwischen dem Volume beibehält. `PersistentVolume` und der eigentliche Speicher.
5. Kubernetes bindet die `PersistentVolumeClaim` zum Neuen `PersistentVolume` . Kapseln, die Folgendes beinhalten `PersistentVolumeClaim` Mounten Sie dieses PersistentVolume auf jedem Host, auf dem es ausgeführt wird.
6. Ein Benutzer erstellt einen `VolumeSnapshot` eines vorhandenen PVC, unter Verwendung eines `VolumeSnapshotClass` Das deutet auf Trident hin.
7. Trident identifiziert das mit dem PVC verknüpfte Volume und erstellt im Backend einen Snapshot des Volumes. Es erzeugt auch ein `VolumeSnapshotContent` Das weist Kubernetes an, wie der Snapshot identifiziert werden kann.
8. Ein Benutzer kann einen erstellen `PersistentVolumeClaim` mit `VolumeSnapshot` als Quelle.
9. Trident identifiziert den erforderlichen Snapshot und führt dieselben Schritte aus, die auch beim Erstellen eines Snapshots erforderlich sind. `PersistentVolume` und ein `Volume` .

Für weiterführende Informationen zu Kubernetes-Objekten empfehlen wir Ihnen dringend, Folgendes zu lesen: "Persistente Datenträger" Abschnitt der Kubernetes-Dokumentation.

Kubernetes `PersistentVolumeClaim` Objekte

Ein Kubernetes `PersistentVolumeClaim` Ein Objekt ist eine Speicheranforderung, die von einem Kubernetes-Cluster-Benutzer gestellt wird.

Zusätzlich zur Standardspezifikation ermöglicht Trident den Benutzern, die folgenden volumespezifischen Annotationen anzugeben, wenn sie die in der Backend-Konfiguration festgelegten Standardeinstellungen überschreiben möchten:

Anmerkung	Lautstärkeoption	Unterstützte Treiber
trident.netapp.io/fileSystem	Dateisystem	ontap-san, solidfire-san,ontap-san-economy
trident.netapp.io/cloneFromPVC	cloneSourceVolume	ontap-nas, ontap-san, solidfire-san, azure-netapp-files, gcp-cvs, ontap-san-economy
trident.netapp.io/splitOnClone	splitOnClone	ontap-nas, ontap-san
trident.netapp.io/protocol	Protokoll	beliebig

Anmerkung	Lautstärkeoption	Unterstützte Treiber
trident.netapp.io/exportPolicy	Exportrichtlinie	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup
trident.netapp.io/snapshotPolicy	Snapshot-Richtlinie	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san
trident.netapp.io/snapshotReserve	Snapshot-Reserve	ontap-nas, ontap-nas-flexgroup, ontap-san, gcp-cvs
trident.netapp.io/snapshotDirectory	Snapshot-Verzeichnis	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup
trident.netapp.io/unixPermissions	unixPermissions	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup
trident.netapp.io/blockSize	Blockgröße	solidfire-san

Wenn die erstellte PV die `Delete` Gemäß der Rückforderungsrichtlinie löscht Trident sowohl das PV als auch das zugehörige Datenvolumen, wenn das PV freigegeben wird (d. h. wenn der Benutzer das PVC löscht). Sollte der Löschvorgang fehlschlagen, kennzeichnet Trident den PV entsprechend und wiederholt den Vorgang regelmäßig, bis er erfolgreich ist oder der PV manuell gelöscht wird. Wenn die PV die `Retain` Trident ignoriert diese Richtlinie und geht davon aus, dass der Administrator das Volume aus Kubernetes und dem Backend entfernt, sodass es vor seiner Löschung gesichert oder überprüft werden kann. Beachten Sie, dass das Löschen des PV nicht dazu führt, dass Trident das zugehörige Datenträgervolumen löscht. Sie sollten es mithilfe der REST-API entfernen.(`tridentctl`).

Trident unterstützt die Erstellung von Volume Snapshots mithilfe der CSI-Spezifikation: Sie können einen Volume Snapshot erstellen und ihn als Datenquelle verwenden, um vorhandene PVCs zu klonen. Auf diese Weise können zeitpunktbezogene Kopien von PVs in Form von Snapshots für Kubernetes bereitgestellt werden. Die Snapshots können dann verwendet werden, um neue PVs zu erstellen. Schau dir das mal an `On-Demand Volume Snapshots` um zu sehen, wie das funktionieren würde.

Trident bietet außerdem die `cloneFromPVC` Und `splitOnClone` Anmerkungen zum Erstellen von Klonen. Mithilfe dieser Annotationen können Sie ein PVC klonen, ohne die CSI-Implementierung verwenden zu müssen.

Hier ein Beispiel: Wenn ein Benutzer bereits eine PVC namens besitzt `mysql` Der Benutzer kann eine neue PVC erstellen, die `mysqlclone` durch Verwendung der Annotation, wie zum Beispiel `trident.netapp.io/cloneFromPVC: mysql` . Mit dieser Annotation klont Trident das Volume, das dem MySQL-PVC entspricht, anstatt ein Volume von Grund auf neu zu erstellen.

Beachten Sie folgende Punkte:

- NetApp empfiehlt das Klonen eines ungenutzten Volumes.
- Ein PVC und sein Klon sollten sich im selben Kubernetes-Namespace befinden und dieselbe Storage-Klasse haben.
- Mit der `ontap-nas` Und `ontap-san` Für Fahrer könnte es wünschenswert sein, die PVC-Anmerkung festzulegen. `trident.netapp.io/splitOnClone` in Verbindung mit `trident.netapp.io/cloneFromPVC` . Mit `trident.netapp.io/splitOnClone` eingestellt auf `true` Trident trennt das geklonte Volume vom übergeordneten Volume und entkoppelt so den Lebenszyklus des geklonten Volumes vollständig von seinem übergeordneten Volume, allerdings auf Kosten einer gewissen Speichereffizienz. Keine Einstellung `trident.netapp.io/splitOnClone` oder es auf `false` Dies führt zu einem geringeren Speicherplatzverbrauch im Backend, allerdings auf Kosten

der Schaffung von Abhängigkeiten zwischen dem übergeordneten Volume und dem Klon-Volume, sodass das übergeordnete Volume erst gelöscht werden kann, wenn das Klon-Volume zuvor gelöscht wurde. Ein Szenario, in dem das Aufteilen des Klons sinnvoll ist, ist das Klonen eines leeren Datenbank-Volumes, bei dem zu erwarten ist, dass sich das Volume und sein Klon stark unterscheiden und nicht von den Speichereffizienzen von ONTAP profitieren.

Der `sample-input` Das Verzeichnis enthält Beispiele für PVC-Definitionen zur Verwendung mit Trident. Siehe Eine vollständige Beschreibung der Parameter und Einstellungen im Zusammenhang mit Trident -Volumes finden Sie hier.

Kubernetes `PersistentVolume` Objekte

Ein Kubernetes `PersistentVolume` Das Objekt repräsentiert einen Speicherbereich, der dem Kubernetes-Cluster zur Verfügung gestellt wird. Es hat einen Lebenszyklus, der unabhängig von der Kapsel ist, die es verwendet.

Trident erschafft `PersistentVolume` Objekte werden automatisch im Kubernetes-Cluster registriert, basierend auf den von ihm bereitgestellten Volumes. Es wird nicht von Ihnen erwartet, dass Sie diese selbst verwalten.

Wenn Sie ein PVC erstellen, das sich auf ein Trident-basiertes System bezieht `StorageClass` Trident provisioniert ein neues Volume unter Verwendung der entsprechenden Speicherklasse und registriert ein neues PV für dieses Volume. Bei der Konfiguration des bereitgestellten Volumes und des zugehörigen PV befolgt Trident die folgenden Regeln:

- Trident generiert einen PV-Namen für Kubernetes und einen internen Namen, der zur Bereitstellung des Speichers verwendet wird. In beiden Fällen wird dadurch sichergestellt, dass die Namen in ihrem Geltungsbereich einzigartig sind.
- Das Volumen entspricht so genau wie möglich der gewünschten Größe im PVC, wird aber je nach Plattform gegebenenfalls auf die nächstliegende zuordenbare Menge aufgerundet.

Kubernetes `StorageClass` Objekte

Kubernetes `StorageClass` Objekte werden anhand ihres Namens angegeben in `PersistentVolumeClaims` Speicherplatz mit einer Reihe von Eigenschaften bereitstellen. Die Speicherklasse selbst identifiziert den zu verwendenden Provisionierer und definiert diesen Satz von Eigenschaften in einer für den Provisionierer verständlichen Weise.

Es handelt sich um eines von zwei grundlegenden Objekten, die vom Administrator erstellt und verwaltet werden müssen. Das andere ist das Trident Backend-Objekt.

Ein Kubernetes `StorageClass` Ein 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 geben Trident an, wie Volumes für die Klasse bereitgestellt werden sollen.

Die Parameter der Speicherklasse sind:

Attribut	Typ	Erforderlich	Beschreibung
Attribute	map[string]string	NEIN	Siehe den Abschnitt „Attribute“ unten.
Speicherbecken	map[string]StringList	NEIN	Zuordnung von Backend-Namen zu Listen von Speicherpools innerhalb
zusätzliche Speicherpools	map[string]StringList	NEIN	Zuordnung von Backend-Namen zu Listen von Speicherpools innerhalb
Speicherpools ausschließen	map[string]StringList	NEIN	Zuordnung von Backend-Namen zu Listen von Speicherpools innerhalb

Speicherattribute und ihre möglichen Werte lassen sich in Speicherpool-Auswahlattribute und Kubernetes-Attribute unterteilen.

Auswahlattribute für Speicherpools

Diese Parameter legen fest, welche von Trident verwalteten Speicherpools zur Bereitstellung von Volumes eines bestimmten Typs verwendet werden sollen.

Attribut	Typ	Werte	Angebot	Anfrage	Unterstützt von
media[1]	Schnur	HDD, Hybrid, SSD	Der Pool enthält Medien dieses Typs; hybrid bedeutet beides	Medientyp angegeben	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san
Bereitstellungstyp	Schnur	dünn, dick	Pool unterstützt diese Bereitstellungsmethode	Bereitstellungsmethode angegeben	dick: alles vom Fass; dünn: alles vom Fass & Solidfire-San

Attribut	Typ	Werte	Angebot	Anfrage	Unterstützt von
Backend-Typ	Schnur	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san, gcp-cvs, azure-netapp-files, ontap-san-economy	Pool gehört zu dieser Art von Backend.	Backend spezifiziert	Alle Fahrer
Momentaufnahmen	bool	wahr, falsch	Pool unterstützt Volumes mit Snapshots	Volume mit aktivierten Snapshots	ontap-nas, ontap-san, solidfire-san, gcp-cvs
Klone	bool	wahr, falsch	Pool unterstützt das Klonen von Volumes	Volume mit aktivierten Klonen	ontap-nas, ontap-san, solidfire-san, gcp-cvs
Verschlüsselung	bool	wahr, falsch	Pool unterstützt verschlüsselte Volumes	Volume mit aktivierter Verschlüsselung	ontap-nas, ontap-nas-economy, ontap-nas-flexgroups, ontap-san
IOPS	int	positive ganze Zahl	Pool ist in der Lage, IOPS in diesem Bereich zu garantieren.	Volumen garantiert diese IOPS	solidfire-san

[1]: Wird von ONTAP Select -Systemen nicht unterstützt.

In den meisten Fällen haben die angeforderten Werte direkten Einfluss auf die Bereitstellung; beispielsweise führt die Anforderung von Thick Provisioning zu einem Thick-Provisioning-Volume. Allerdings verwendet ein Element-Speicherpool seine angebotenen minimalen und maximalen IOPS-Werte zur Festlegung der QoS-Werte anstelle der angeforderten Werte. In diesem Fall wird der angeforderte Wert nur zur Auswahl des Speicherpools verwendet.

Im Idealfall können Sie verwenden `attributes` allein die Eigenschaften des Speichers zu modellieren, die Sie benötigen, um die Bedürfnisse einer bestimmten Klasse zu befriedigen. Trident erkennt und wählt automatisch Speicherpools aus, die *allen* der Anforderung entsprechen. `attributes` die Sie angeben.

Sollten Sie feststellen, dass Sie nicht in der Lage sind, zu nutzen `attributes` Um die richtigen Pools für eine Klasse automatisch auszuwählen, können Sie Folgendes verwenden: `storagePools` Und `additionalStoragePools` Parameter, um die Pools weiter zu verfeinern oder sogar eine bestimmte Gruppe von Pools auszuwählen.

Sie können die `storagePools` Parameter zur weiteren Einschränkung der Menge der Pools, die mit einem bestimmten Parameter übereinstimmen. `attributes` . Mit anderen Worten, Trident nutzt die Schnittmenge der durch die `attributes` Und `storagePools` Parameter für die Bereitstellung. Sie können entweder den einen Parameter einzeln oder beide zusammen verwenden.

Sie können die `additionalStoragePools` Parameter zur Erweiterung der von Trident für die Bereitstellung verwendeten Pools, unabhängig von den vom `attributes` Und `storagePools` Parameter.

Sie können die `excludeStoragePools` Parameter zum Filtern der Pools, die Trident für die Bereitstellung verwendet. Durch die Verwendung dieses Parameters werden alle passenden Pools entfernt.

Im `storagePools` Und `additionalStoragePools` Parameter, jeder Eintrag hat die Form `<backend>:<storagePoolList>` , Wo `<storagePoolList>` ist eine durch Kommas getrennte Liste von Speicherpools für das angegebene Backend. Zum Beispiel ein Wert für `additionalStoragePools` könnte aussehen `ontapnas_192.168.1.100:aggr1,aggr2;solidfire_192.168.1.101:bronze` . Diese Listen akzeptieren reguläre Ausdrücke sowohl für die Backend- als auch für die Listenwerte. Sie können verwenden `tridentctl get backend` um die Liste der Backends und ihrer Pools zu erhalten.

Kubernetes-Attribute

Diese Attribute haben keinen Einfluss auf die Auswahl von Speicherpools/Backends durch Trident während der dynamischen Bereitstellung. Stattdessen liefern diese Attribute lediglich Parameter, die von Kubernetes Persistent Volumes unterstützt werden. Die Worker-Knoten sind für Dateisystem-Erstellungsoperationen zuständig und benötigen möglicherweise Dateisystem-Dienstprogramme wie xfsprogs.

Attribut	Typ	Werte	Beschreibung	Relevante Treiber	Kubernetes-Version
fsType	Schnur	ext4, ext3, xfs	Der Dateisystemtyp für Blockvolumes	solidfire-san, ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economy	Alle
Volumenerweiterung zulassen	boolescher Wert	wahr, falsch	Unterstützung für die Vergrößerung der PVC-Größe aktivieren oder deaktivieren	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economy, solidfire-san, gcp-cvs, azure-netapp-files	1.11+
volumeBindingMode	Schnur	Sofort, Warten auf den ersten Kunden	Wählen Sie den Zeitpunkt für die Volumenbindung und die dynamische Bereitstellung aus.	Alle	1,19 - 1,26

- Der `fsType` Der Parameter dient zur Steuerung des gewünschten Dateisystemtyps für SAN-LUNs. Darüber hinaus nutzt Kubernetes auch die Anwesenheit von `fsType` in einer Speicherklasse, um anzuzeigen, dass ein Dateisystem existiert. Die Volumenbesitzverhältnisse können über die `fsGroup` Sicherheitskontext eines Pods nur wenn `fsType` ist festgelegt. Siehe"Kubernetes: Konfigurieren eines Sicherheitskontexts für einen Pod oder Container" Für eine Übersicht über die Festlegung der Volumenzugehörigkeit mithilfe des `fsGroup` Kontext. Kubernetes wird die `fsGroup` Wert nur dann, wenn:

  - `fsType`wird in der Speicherklasse festgelegt.
  - Der PVC-Zugriffsmodus ist RWO.

  Bei NFS-Speichertreibern ist ein Dateisystem bereits als Teil des NFS-Exports vorhanden. Um zu verwenden `fsGroup` Die Speicherklasse muss noch eine angeben `fsType` Sie können es so einstellen: `nfs` oder ein beliebiger Wert ungleich null.

- Siehe"Volumen erweitern" Weitere Einzelheiten zur Volumenerweiterung finden Sie hier.
- Das Trident -Installationspaket enthält mehrere Beispieldefinitionen für Speicherklassen zur Verwendung mit Trident `.sample-input/storage-class-*.yaml` . Das Löschen einer Kubernetes-Speicherklasse führt dazu, dass auch die entsprechende Trident -Speicherklasse gelöscht wird.

Kubernetes `VolumeSnapshotClass` Objekte

Kubernetes `VolumeSnapshotClass` Objekte sind analog zu `StorageClasses` . Sie helfen dabei, 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.

A `VolumeSnapshotClass` Um Snapshots zu erstellen, muss ein Administrator dies definieren. 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` legt fest, dass Kubernetes Anfragen für Volume-Snapshots des `csi-snapclass` Die Klassen werden von Trident verwaltet. Der `deletionPolicy` legt fest, welche Aktion ausgeführt werden soll, wenn ein Snapshot gelöscht werden muss. Wann `deletionPolicy` ist eingestellt auf `Delete` Beim Löschen eines Snapshots werden sowohl die Volume-Snapshot-Objekte als auch der zugrunde liegende Snapshot auf dem Speichercluster entfernt. Alternativ können Sie es auch so einstellen: `Retain` bedeutet, dass `VolumeSnapshotContent` und die physische Momentaufnahme wird beibehalten.

Kubernetes `VolumeSnapshot` Objekte

Ein Kubernetes `VolumeSnapshot` Bei „object“ handelt es sich um eine Anfrage zur Erstellung eines

Snapshots eines Volumes. So wie ein PVC eine Anfrage eines Benutzers nach einem Volume darstellt, ist ein Volume-Snapshot eine Anfrage eines Benutzers zur Erstellung eines Snapshots eines bestehenden PVC.

Wenn eine Anfrage für einen Volume-Snapshot eingeht, verwaltet Trident automatisch die Erstellung des Snapshots für das Volume im Backend und stellt den Snapshot durch die Erstellung einer eindeutigen ID bereit.
`VolumeSnapshotContent` Objekt. Sie können Snapshots aus bestehenden PVCs erstellen und diese Snapshots als Datenquelle beim Erstellen neuer PVCs verwenden.

> Der Lebenszyklus eines VolumeSnapshots ist unabhängig vom Quell-PVC: Ein Snapshot bleibt auch dann erhalten, wenn der Quell-PVC gelöscht wird. Beim Löschen eines PVCs mit zugehörigen Snapshots markiert Trident das zugehörige Datenträgervolume im Status **Wird gelöscht**, entfernt es aber nicht vollständig. Das Volume wird entfernt, sobald alle zugehörigen Snapshots gelöscht sind.

Kubernetes `VolumeSnapshotContent` Objekte

Ein Kubernetes `VolumeSnapshotContent` Das Objekt stellt eine Momentaufnahme eines bereits bereitgestellten Volumes dar. Es ist analog zu einem `PersistentVolume` und kennzeichnet einen bereitgestellten Snapshot auf dem Speichercluster. Ähnlich `PersistentVolumeClaim` Und `PersistentVolume` Objekte, wenn ein Snapshot erstellt wird, `VolumeSnapshotContent` Das Objekt verwaltet eine Eins-zu-Eins-Zuordnung zum `VolumeSnapshot` Objekt, das die Erstellung des Snapshots angefordert hatte.

Der `VolumeSnapshotContent` Das Objekt enthält Details, die den Snapshot eindeutig identifizieren, wie zum Beispiel die `snapshotHandle` . Das `snapshotHandle` ist eine einzigartige Kombination aus dem Namen der PV und dem Namen der `VolumeSnapshotContent` Objekt.

Wenn eine Snapshot-Anfrage eingeht, erstellt Trident den Snapshot im Backend. Nachdem der Snapshot erstellt wurde, konfiguriert Trident einen `VolumeSnapshotContent` Das Objekt wird erstellt und somit der Kubernetes-API zugänglich gemacht.

> Normalerweise müssen Sie die `VolumeSnapshotContent` Objekt. Eine Ausnahme hiervon besteht, wenn Sie möchten"einen Volume-Snapshot importieren" außerhalb von Trident erstellt.

Kubernetes `VolumeGroupSnapshotClass` Objekte

Kubernetes `VolumeGroupSnapshotClass` Objekte sind analog zu `VolumeSnapshotClass` . Sie helfen dabei, mehrere Speicherklassen zu definieren und werden von Volume-Gruppen-Snapshots referenziert, um den Snapshot der erforderlichen Snapshot-Klasse zuzuordnen. Jeder Volume-Group-Snapshot ist einer einzelnen Volume-Group-Snapshot-Klasse zugeordnet.

A `VolumeGroupSnapshotClass` Um eine Gruppe von Snapshots zu erstellen, muss dies von einem Administrator definiert werden. Eine Snapshot-Klasse für eine Volumengruppe wird mit folgender Definition erstellt:

```
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshotClass
metadata:
  name: csi-group-snap-class
  annotations:
    kubernetes.io/description: "Trident group snapshot class"
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

Der `driver` legt fest, dass Kubernetes Anfragen für Volume-Gruppen-Snapshots der `csi-group-snap-class` Die Klassen werden von Trident verwaltet. Der `deletionPolicy` legt die Aktion fest, die ausgeführt werden soll, wenn ein Gruppen-Snapshot gelöscht werden muss. Wann `deletionPolicy` ist eingestellt auf `Delete` Beim Löschen eines Snapshots werden sowohl die Volume-Group-Snapshot-Objekte als auch der zugrunde liegende Snapshot auf dem Speichercluster entfernt. Alternativ können Sie es auch so einstellen: `Retain` bedeutet, dass `VolumeGroupSnapshotContent` und die physische Momentaufnahme wird beibehalten.

Kubernetes `VolumeGroupSnapshot` Objekte

Ein Kubernetes `VolumeGroupSnapshot` Das Objekt ist eine Anfrage zur Erstellung eines Snapshots mehrerer Volumes. So wie eine PVC eine Anfrage eines Benutzers nach einem Volume darstellt, ist ein Volume-Gruppen-Snapshot eine Anfrage eines Benutzers zur Erstellung eines Snapshots einer bestehenden PVC.

Wenn eine Anfrage für einen Volume-Gruppen-Snapshot eingeht, verwaltet Trident automatisch die Erstellung des Gruppen-Snapshots für die Volumes im Backend und stellt den Snapshot durch die Erstellung einer eindeutigen Kennung bereit. `VolumeGroupSnapshotContent` Objekt. Sie können Snapshots aus bestehenden PVCs erstellen und diese Snapshots als Datenquelle beim Erstellen neuer PVCs verwenden.

> ⓘ Der Lebenszyklus eines VolumeGroupSnapshot ist unabhängig vom Quell-PVC: Ein Snapshot bleibt auch dann erhalten, wenn der Quell-PVC gelöscht wird. Beim Löschen eines PVCs mit zugehörigen Snapshots markiert Trident das zugehörige Datenträgervolume im Status **Wird gelöscht**, entfernt es aber nicht vollständig. Der Snapshot der Volumengruppe wird entfernt, wenn alle zugehörigen Snapshots gelöscht werden.

Kubernetes `VolumeGroupSnapshotContent` Objekte

Ein Kubernetes `VolumeGroupSnapshotContent` Das Objekt stellt einen Gruppen-Snapshot dar, der von einem bereits bereitgestellten Volume erstellt wurde. Es ist analog zu einem `PersistentVolume` und kennzeichnet einen bereitgestellten Snapshot auf dem Speichercluster. Ähnlich `PersistentVolumeClaim` Und `PersistentVolume` Objekte, wenn ein Snapshot erstellt wird, `VolumeSnapshotContent` Das Objekt verwaltet eine Eins-zu-Eins-Zuordnung zum `VolumeSnapshot` Objekt, das die Erstellung des Snapshots angefordert hatte.

Der `VolumeGroupSnapshotContent` Das Objekt enthält Details, die die Snapshot-Gruppe identifizieren, wie zum Beispiel die `volumeGroupSnapshotHandle` und und einzelne VolumeSnapshotHandles, die auf dem Speichersystem vorhanden sind.

Wenn eine Snapshot-Anforderung eingeht, erstellt Trident im Backend den Volume-Group-Snapshot. Nachdem

der Snapshot der Volumengruppe erstellt wurde, konfiguriert Trident einen `VolumeGroupSnapshotContent` Das Objekt wird erstellt und somit der Kubernetes-API zugänglich gemacht.

Kubernetes `CustomResourceDefinition` Objekte

Kubernetes Custom Resources sind Endpunkte in der Kubernetes-API, die vom Administrator definiert werden und dazu dienen, ähnliche Objekte zu gruppieren. Kubernetes unterstützt die Erstellung benutzerdefinierter Ressourcen zum Speichern einer Sammlung von Objekten. Sie können diese Ressourcendefinitionen durch Ausführen von `kubectl get crds` .

Custom Resource Definitions (CRDs) und die zugehörigen Objektmetadaten werden von Kubernetes in seinem Metadatenspeicher abgelegt. Dadurch entfällt die Notwendigkeit eines separaten Ladengeschäfts für Trident.

Trident Anwendungen `CustomResourceDefinition` Objekte, um die Identität von Trident -Objekten wie Trident -Backends, Trident -Speicherklassen und Trident Volumes zu erhalten. Diese Objekte werden von Trident verwaltet. Darüber hinaus führt das CSI-Volume-Snapshot-Framework einige CRDs ein, die zur Definition von Volume-Snapshots erforderlich sind.

CRDs sind ein Konstrukt von Kubernetes. Objekte der oben definierten Ressourcen werden von Trident erstellt. Ein einfaches Beispiel: Wenn ein Backend erstellt wird mit `tridentctl` ein entsprechendes `tridentbackends` Ein CRD-Objekt wird zur Verwendung durch Kubernetes erstellt.

Hier einige Punkte, die Sie bei Tridents CRDs beachten sollten:

- Bei der Installation von Trident wird ein Satz von CRDs erstellt, die wie jeder andere Ressourcentyp verwendet werden können.
- Bei der Deinstallation von Trident mithilfe der `tridentctl uninstall` Der Befehl führt dazu, dass Trident -Pods gelöscht werden, die erstellten CRDs jedoch nicht bereinigt werden. Siehe"Trident deinstallieren" zu verstehen, wie Trident komplett entfernt und von Grund auf neu konfiguriert werden kann.

Trident `StorageClass` Objekte

Trident erstellt passende Speicherklassen für Kubernetes. `StorageClass` Objekte, die spezifizieren `csi.trident.netapp.io` in ihrem Bereitstellungsfeld. Der Name der Speicherklasse stimmt mit dem von Kubernetes überein. `StorageClass` Das Objekt, das es repräsentiert.

> Mit Kubernetes werden diese Objekte automatisch erstellt, wenn ein Kubernetes-Update durchgeführt wird. `StorageClass` Das System, das Trident als Provisionierer verwendet, ist registriert.

Speicherklassen umfassen eine Reihe von Anforderungen an Datenträger. Trident gleicht diese Anforderungen mit den Attributen jedes Speicherpools ab; wenn sie übereinstimmen, ist dieser Speicherpool ein gültiges Ziel für die Bereitstellung von Volumes mit dieser Speicherklasse.

Sie können Speicherklassenkonfigurationen erstellen, um Speicherklassen direkt über die REST-API zu definieren. Bei Kubernetes-Bereitstellungen gehen wir jedoch davon aus, dass sie bei der Registrierung neuer Kubernetes-Instanzen erstellt werden. `StorageClass` Objekte.

Trident Backend-Objekte

Backends stellen die Speicheranbieter dar, auf denen Trident Volumes bereitstellt; eine einzelne Trident

-Instanz kann beliebig viele Backends verwalten.

Dies ist einer der beiden Objekttypen, die Sie selbst erstellen und verwalten. Das andere ist Kubernetes. `StorageClass` Objekt.

Weitere Informationen zum Erstellen dieser Objekte finden Sie unter"Backends konfigurieren" .

Trident `StoragePool` Objekte

Speicherpools stellen die verschiedenen Speicherorte dar, die auf jedem Backend für die Bereitstellung zur Verfügung stehen. Für ONTAP entsprechen diese Aggregaten in SVMs. Bei NetApp HCI/ SolidFire entsprechen diese den vom Administrator festgelegten QoS-Bändern. Für den Cloud Volumes Service entsprechen diese den Regionen der Cloud-Anbieter. Jeder Speicherpool verfügt über eine Reihe unterschiedlicher Speicherattribute, die seine Leistungsmerkmale und Datenschutzeigenschaften definieren.

Im Gegensatz zu den anderen Objekten hier werden Speicherpoolkandidaten immer automatisch erkannt und verwaltet.

Trident `Volume` Objekte

Volumes sind die grundlegende Bereitstellungseinheit und umfassen Backend-Endpunkte wie NFS-Freigaben sowie iSCSI- und FC-LUNs. In Kubernetes entsprechen diese direkt `PersistentVolumes` . Wenn Sie ein Volume erstellen, stellen Sie sicher, dass es über eine Speicherklasse verfügt, die festlegt, wo dieses Volume bereitgestellt werden kann, sowie über eine Größe.

- In Kubernetes werden diese Objekte automatisch verwaltet. Sie können diese einsehen, um zu sehen, was Trident bereitgestellt hat.
- Beim Löschen eines PV mit zugehörigen Snapshots wird das entsprechende Trident Volume auf den Status **Wird gelöscht** aktualisiert. Damit das Trident -Volume gelöscht werden kann, müssen die Snapshots des Volumes entfernt werden.

Eine Volumenkonfiguration definiert die Eigenschaften, die ein bereitgestelltes Volumen haben soll.

Attribut	Typ	Erforderlich	Beschreibung
Version	Schnur	NEIN	Version der Trident API ("1")
Name	Schnur	Ja	Name des zu erstellenden Volumes
Speicherklasse	Schnur	Ja	Speicherklasse, die beim Bereitstellen des Volumes verwendet werden soll
Größe	Schnur	Ja	Größe des bereitzustellenden Datenvolumens in Bytes
Protokoll	Schnur	NEIN	Zu verwendender Protokolltyp: „Datei“ oder „Block“

Attribut	Typ	Erforderlich	Beschreibung
interner Name	Schnur	NEIN	Name des Objekts im Speichersystem; generiert von Trident
cloneSourceVolume	Schnur	NEIN	ontap (nas, san) & solidfire-*: Name des Volumes, von dem geklont werden soll
splitOnClone	Schnur	NEIN	ontap (nas, san): Trennt den Klon von seinem Elternknoten.
Snapshot-Richtlinie	Schnur	NEIN	ontap-*: Zu verwendende Snapshot-Richtlinie
Snapshot-Reserve	Schnur	NEIN	ontap-*: Prozentsatz des für Snapshots reservierten Volumens
Exportrichtlinie	Schnur	NEIN	ontap-nas*: Exportrichtlinie verwenden
Snapshot-Verzeichnis	bool	NEIN	ontap-nas*: Gibt an, ob das Snapshot-Verzeichnis sichtbar ist.
unixPermissions	Schnur	NEIN	ontap-nas*: Initial UNIX-Berechtigungen
Blockgröße	Schnur	NEIN	solidfire-*: Block-/Sektorgröße
Dateisystem	Schnur	NEIN	Dateisystemtyp

Trident erzeugt `internalName` beim Erstellen des Volumens. Dies besteht aus zwei Schritten. Zunächst wird das Speicherpräfix vorangestellt (entweder das Standardpräfix). `trident` oder dem Präfix in der Backend-Konfiguration) zum Volume-Namen, was zu einem Namen der Form führt `<prefix>-<volume-name>` . Anschließend wird der Name bereinigt, indem im Backend nicht zulässige Zeichen ersetzt werden. Bei ONTAP Backends werden Bindestriche durch Unterstriche ersetzt (dadurch wird der interne Name zu `<prefix>_<volume-name>`). Bei Element-Backends werden Unterstriche durch Bindestriche ersetzt.

Sie können Volume-Konfigurationen verwenden, um Volumes direkt über die REST-API bereitzustellen, aber bei Kubernetes-Bereitstellungen gehen wir davon aus, dass die meisten Benutzer die Standard-Kubernetes-Konfiguration verwenden. `PersistentVolumeClaim` Verfahren. Trident erstellt dieses Volume-Objekt automatisch im Rahmen des Bereitstellungsprozesses.

Trident `Snapshot` Objekte

Snapshots sind Momentaufnahmen von Datenträgern, die zur Bereitstellung neuer Datenträger oder zur Wiederherstellung des vorherigen Zustands verwendet werden können. In Kubernetes entsprechen diese direkt `VolumeSnapshotContent` Objekte. Jeder Snapshot ist mit einem Volume verknüpft, das die Datenquelle für den Snapshot darstellt.

Jede `Snapshot` Das Objekt umfasst die unten aufgeführten Eigenschaften:

Attribut	Typ	Erforderlich	Beschreibung
Version	Zeichenfolge	Ja	Version der Trident API ("1")
Name	Zeichenfolge	Ja	Name des Trident -Snapshot-Objekts
interner Name	Zeichenfolge	Ja	Name des Trident -Snapshot-Objekts auf dem Speichersystem
volumeName	Zeichenfolge	Ja	Name des persistenten Volumes, für das der Snapshot erstellt wurde
volumeInternalName	Zeichenfolge	Ja	Name des zugehörigen Trident Volume-Objekts im Speichersystem

In Kubernetes werden diese Objekte automatisch verwaltet. Sie können diese einsehen, um zu sehen, was Trident bereitgestellt hat.

Wenn ein Kubernetes `VolumeSnapshot` Wenn eine Objektanforderung erstellt wird, funktioniert Trident , indem ein Snapshot-Objekt auf dem zugrunde liegenden Speichersystem erstellt wird. Der `internalName` Dieses Snapshot-Objekt wird durch die Kombination des Präfixes generiert. `snapshot-` mit dem `UID` der `VolumeSnapshot` Objekt (zum Beispiel, `snapshot-e8d8a0ca-9826-11e9-9807-525400f3f660`). `volumeName` Und `volumeInternalName` werden durch Abrufen der Details des zugrunde liegenden Volumens befüllt.

Trident `ResourceQuota` Objekt

Der Trident Dämonensatz verzehrt einen `system-node-critical` Prioritätsklasse – die höchste in Kubernetes verfügbare Prioritätsklasse – um sicherzustellen, dass Trident Volumes während des ordnungsgemäßen Herunterfahrens von Knoten identifizieren und bereinigen kann und dass Trident -Daemonset-Pods in Clustern mit hohem Ressourcendruck Workloads mit niedrigerer Priorität unterbrechen können.

Um dies zu erreichen, setzt Trident ein `ResourceQuota` Objekt, um sicherzustellen, dass eine Prioritätsklasse "system-node-critical" auf dem Trident -Daemonset erfüllt ist. Vor der Bereitstellung und der Erstellung des Daemonsets sucht Trident nach den `ResourceQuota` Objekt und, falls nicht gefunden, wendet es an.

Wenn Sie mehr Kontrolle über das Standardressourcenkontingent und die Prioritätsklasse benötigen, können Sie eine generieren. `custom.yaml` oder konfigurieren Sie die `ResourceQuota` Objekt, das das Helm-Chart verwendet.

Nachfolgend ein Beispiel für ein `ResourceQuota`-Objekt, das dem Trident -Daemonset Priorität einräumt.

```
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 Ressourcenkontingenten finden Sie unter"Kubernetes: Ressourcenkontingente" .

Aufräumen `ResourceQuota` wenn die Installation fehlschlägt

In dem seltenen Fall, dass die Installation nach der `ResourceQuota` Objekt wurde erstellt, erster Versuch"Deinstallation" und dann neu installieren.

Sollte das nicht funktionieren, entfernen Sie die `ResourceQuota` Objekt.

Entfernen `ResourceQuota`

Wenn Sie Ihre Ressourcenzuweisung lieber selbst steuern möchten, können Sie den Trident entfernen. `ResourceQuota` Objekt mithilfe des Befehls:

```
kubectl delete quota trident-csi -n trident
```

Pod-Sicherheitsstandards (PSS) und Sicherheitskontextbeschränkungen (SCC)

Die Kubernetes Pod Security Standards (PSS) und Pod Security Policies (PSP) definieren Berechtigungsstufen und schränken das Verhalten von Pods ein. OpenShift Security Context Constraints (SCC) definieren analog dazu Pod-Beschränkungen, die spezifisch für die OpenShift Kubernetes Engine sind. Um diese Anpassung zu ermöglichen, aktiviert Trident während der Installation bestimmte Berechtigungen. In den folgenden Abschnitten werden die von Trident festgelegten Berechtigungen detailliert beschrieben.

PSS ersetzt Pod Security Policies (PSP). PSP wurde in Kubernetes v1.21 als veraltet markiert und wird in v1.25 entfernt. Weitere Informationen finden Sie unter"Kubernetes: Sicherheit" .

Erforderlicher Kubernetes-Sicherheitskontext und zugehörige Felder

Erlaubnis	Beschreibung
Privilegiert	CSI erfordert bidirektionale Mount-Punkte, was bedeutet, dass auf dem Trident -Node-Pod ein privilegierter Container ausgeführt werden muss. Weitere Informationen finden Sie unter"Kubernetes: Mount-Propagation" .
Host-Netzwerk	Erforderlich für den iSCSI-Daemon. `iscsiadm` Verwaltet iSCSI-Mounts und nutzt das Host-Netzwerk zur Kommunikation mit dem iSCSI-Daemon.
Host-IPC	NFS nutzt Interprozesskommunikation (IPC) zur Kommunikation mit dem NFSD.
Host-PID	Erforderlich zum Starten `rpc-statd` für NFS. Trident fragt Hostprozesse ab, um festzustellen, ob `rpc-statd` wird vor dem Einbinden von NFS-Volumes ausgeführt.
Funktionen	Der `SYS_ADMIN` Diese Funktionalität wird als Teil der Standardfunktionen für privilegierte Container bereitgestellt. Docker legt beispielsweise diese Berechtigungen für privilegierte Container fest: `CapPrm: 0000003fffffffff` `CapEff: 0000003fffffffff`
Seccomp	Das Seccomp-Profil ist in privilegierten Containern immer "Unconfided"; daher kann es in Trident nicht aktiviert werden.
SELinux	Auf OpenShift werden privilegierte Container in der `spc_t` ("Super Privileged Container")-Domäne, und unprivilegierte Container werden in der `container_t` Domain. An `containerd` , mit `container-selinux` installiert, alle Container werden in der `spc_t` Domain, wodurch SELinux effektiv deaktiviert wird. Daher fügt Trident nichts hinzu `seLinuxOptions` zu Containern.
DAC	Privilegierte Container müssen als Root ausgeführt werden. Nicht privilegierte Container werden als Root ausgeführt, um auf die von CSI benötigten Unix-Sockets zuzugreifen.

Pod-Sicherheitsstandards (PSS)

Etikett	Beschreibung	Standard
`pod-security.kubernetes.io/enforce pod-security.kubernetes.io/enforce-version`	Ermöglicht es dem Trident Controller und den Knoten, in den Installations-Namespace aufgenommen zu werden. Ändern Sie die Namespace-Bezeichnung nicht.	`enforce: privileged enforce-version: <version of the current cluster or highest version of PSS tested.>`

Das Ändern der Namespace-Labels kann dazu führen, dass Pods nicht eingeplant werden, eine Fehlermeldung wie „Fehler beim Erstellen: …“ oder „Warnung: trident-csi-…“ erscheint. Wenn dies der Fall ist, prüfen Sie, ob die Namespace-Bezeichnung für `privileged` wurde geändert. Installieren Sie Trident in diesem Fall neu.

Pod-Sicherheitsrichtlinien (PSP)

Feld	Beschreibung	Standard
`allowPrivilegeEscalation`	Privilegierte Container müssen eine Rechteausweitung ermöglichen.	`true`
`allowedCSIDrivers`	Trident verwendet keine Inline-CSI-Ephemeralvolumina.	Leer
`allowedCapabilities`	Nicht privilegierte Trident Container benötigen keine weiteren Berechtigungen als die Standardberechtigungen, während privilegierten Containern alle möglichen Berechtigungen gewährt werden.	Leer
`allowedFlexVolumes`	Trident verwendet keine"FlexVolume-Treiber" Daher sind sie nicht in der Liste der zulässigen Bände enthalten.	Leer
`allowedHostPaths`	Der Trident -Node-Pod mountet das Root-Dateisystem des Nodes, daher bringt das Festlegen dieser Liste keinen Vorteil.	Leer
`allowedProcMountTypes`	Trident verwendet keine `ProcMountTypes` .	Leer
`allowedUnsafeSysctls`	Trident benötigt keine unsicheren… `sysctls` .	Leer
`defaultAddCapabilities`	Für privilegierte Container müssen keine zusätzlichen Funktionen hinzugefügt werden.	Leer
`defaultAllowPrivilegeEscalation`	Die Gewährung von Privilegienerweiterungen wird in jedem Trident -Pod gehandhabt.	`false`
`forbiddenSysctls`	NEIN `sysctls` sind erlaubt.	Leer

Feld	Beschreibung	Standard
`fsGroup`	Trident Container werden als Root ausgeführt.	`RunAsAny`
`hostIPC`	Das Einbinden von NFS-Volumes erfordert die Kommunikation zwischen Host und IPC. `nfsd`	`true`
`hostNetwork`	iscsiadm benötigt das Host-Netzwerk zur Kommunikation mit dem iSCSI-Daemon.	`true`
`hostPID`	Die Host-PID wird benötigt, um zu prüfen, ob `rpc-statd` läuft auf dem Knoten.	`true`
`hostPorts`	Trident verwendet keine Host-Ports.	Leer
`privileged`	Trident Node-Pods müssen einen privilegierten Container ausführen, um Volumes einzubinden.	`true`
`readOnlyRootFilesystem`	Trident Node-Pods müssen in das Node-Dateisystem schreiben.	`false`
`requiredDropCapabilities`	Trident -Node-Pods führen einen privilegierten Container aus und können keine Capabilities verlieren.	`none`
`runAsGroup`	Trident Container werden als Root ausgeführt.	`RunAsAny`
`runAsUser`	Trident Container werden als Root ausgeführt.	`runAsAny`
`runtimeClass`	Trident verwendet nicht `RuntimeClasses` .	Leer
`seLinux`	Trident lässt sich nicht einstellen `seLinuxOptions` weil es derzeit Unterschiede in der Art und Weise gibt, wie Container-Laufzeitumgebungen und Kubernetes-Distributionen SELinux handhaben.	Leer
`supplementalGroups`	Trident Container werden als Root ausgeführt.	`RunAsAny`
`volumes`	Trident Pods benötigen diese Volumen-Plugins.	`hostPath, projected, emptyDir`

Sicherheitskontextbeschränkungen (SCC)

Labels	Beschreibung	Standard
`allowHostDirVolumePlugin`	Trident Node-Pods mounten das Root-Dateisystem des Nodes.	`true`
`allowHostIPC`	Das Einbinden von NFS-Volumes erfordert die Kommunikation zwischen Host und IPC. `nfsd`.	`true`
`allowHostNetwork`	iscsiadm benötigt das Host-Netzwerk zur Kommunikation mit dem iSCSI-Daemon.	`true`
`allowHostPID`	Die Host-PID wird benötigt, um zu prüfen, ob `rpc-statd` läuft auf dem Knoten.	`true`
`allowHostPorts`	Trident verwendet keine Host-Ports.	`false`
`allowPrivilegeEscalation`	Privilegierte Container müssen eine Rechteausweitung ermöglichen.	`true`
`allowPrivilegedContainer`	Trident Node-Pods müssen einen privilegierten Container ausführen, um Volumes einzubinden.	`true`
`allowedUnsafeSysctls`	Trident benötigt keine unsicheren… `sysctls`.	`none`
`allowedCapabilities`	Nicht privilegierte Trident Container benötigen keine weiteren Berechtigungen als die Standardberechtigungen, während privilegierten Containern alle möglichen Berechtigungen gewährt werden.	Leer
`defaultAddCapabilities`	Für privilegierte Container müssen keine zusätzlichen Funktionen hinzugefügt werden.	Leer
`fsGroup`	Trident Container werden als Root ausgeführt.	`RunAsAny`
`groups`	Diese SCC ist spezifisch für Trident und an ihren Benutzer gebunden.	Leer
`readOnlyRootFilesystem`	Trident Node-Pods müssen in das Node-Dateisystem schreiben.	`false`
`requiredDropCapabilities`	Trident -Node-Pods führen einen privilegierten Container aus und können keine Capabilities verlieren.	`none`
`runAsUser`	Trident Container werden als Root ausgeführt.	`RunAsAny`

Labels	Beschreibung	Standard
`seLinuxContext`	Trident lässt sich nicht einstellen `seLinuxOptions` weil es derzeit Unterschiede in der Art und Weise gibt, wie Container-Laufzeitumgebungen und Kubernetes-Distributionen SELinux handhaben.	Leer
`seccompProfiles`	Privilegierte Container laufen immer "Unconfided".	Leer
`supplementalGroups`	Trident Container werden als Root ausgeführt.	`RunAsAny`
`users`	Es wird ein Eintrag bereitgestellt, um diese SCC an den Trident Benutzer im Trident -Namensraum zu binden.	n/a
`volumes`	Trident Pods benötigen diese Volumen-Plugins.	`hostPath, downwardAPI, projected, emptyDir`

Rechtliche Hinweise

Rechtliche Hinweise bieten Zugriff auf Urheberrechtserklärungen, Marken, Patente und mehr.

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 ihrer jeweiligen Eigentümer sein.

"https://www.netapp.com/company/legal/trademarks/"

Patente

Eine aktuelle Liste der Patente im Besitz von NetApp finden Sie unter:

https://www.netapp.com/pdf.html?item=/media/11887-patentspage.pdf

Datenschutzrichtlinie

"https://www.netapp.com/company/legal/privacy-policy/"

Open Source

Die Urheberrechte und Lizenzen von Drittanbietern, die in der NetApp -Software für Trident verwendet werden, können Sie in der Hinweisdatei für jede Version einsehen unter: 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 Urheberrechtsinhabers in keiner Form und durch keine Mittel – weder grafische noch elektronische oder mechanische, einschließlich Fotokopieren, Aufnehmen oder Speichern in einem elektronischen Abrufsystem – auch nicht in Teilen, vervielfältigt werden.

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

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

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

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

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

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

Markeninformationen

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