Trident verwalten und überwachen

Trident

NetApp
July 01, 2026

Inhalt

Trident verwalten und überwachen

Trident aktualisieren

Trident aktualisieren

Ab der Version 24.02 folgt Trident einem viermonatigen Veröffentlichungszyklus und liefert drei Hauptversionen pro Kalenderjahr. Jede neue Version baut auf den vorherigen Versionen auf und bietet neue Funktionen, Leistungsverbesserungen, Fehlerbehebungen und Verbesserungen. Wir empfehlen Ihnen, mindestens einmal jährlich ein Upgrade durchzuführen, um die neuen Funktionen in Trident zu nutzen.

Überlegungen vor dem Upgrade

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

- Es sollte nur eine einzige Trident-Instanz über alle Namespaces hinweg in einem Kubernetes-Cluster installiert sein.
- Trident 23.07 und höher erfordert v1-Volume-Snapshots und unterstützt Alpha- oder Beta-Snapshots nicht mehr.
- Beim Upgrade ist es wichtig, dass Sie `parameter.fsType` in `StorageClasses` verwenden, die von Trident genutzt werden. Sie können `StorageClasses` löschen und neu erstellen, ohne bereits vorhandene Volumes zu beeinträchtigen.
  - Dies ist eine **Voraussetzung** für die Durchsetzung "Sicherheitskontexte" für SAN-Volumes.
  - Das sample input Verzeichnis enthält Beispiele, wie `storage-class-basic.yaml.templ` und `storage-class-bronze-default.yaml`.
  - Weitere Informationen finden sich unter "Bekannte Probleme".

Schritt 1: Wählen Sie eine Version

Trident-Versionen folgen einer datumsbasierten `YY.MM` Namenskonvention, wobei „YY“ die letzten beiden Ziffern des Jahres und „MM“ der Monat sind. Dot-Releases folgen einer `YY.MM.X` Konvention, wobei „X“ das Patch-Level ist. Sie wählen die Version aus, auf die Sie aktualisieren möchten, basierend auf der Version, 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 24.06 (oder jeder 24.06 dot release) auf 25.06 aktualisieren.
- Wenn Sie von einer Version außerhalb des Vier-Versions-Fensters aktualisieren, führen Sie ein mehrstufiges Upgrade durch. Verwenden Sie die Upgrade-Anweisungen für die "frühere Version" Version, von der Sie aktualisieren, um auf die neueste Version zu aktualisieren, die in das Vier-Versions-Fenster passt. Wenn Sie beispielsweise Version 23.07 verwenden und auf 25.06 aktualisieren möchten:
  a. Erstes Upgrade von 23.07 auf 24.06.
  b. Führen Sie dann ein Upgrade von 24.06 auf 25.06 durch.

Beim Upgrade mit dem Trident-Operator auf der OpenShift Container Platform sollten Sie auf Trident 21.01.1 oder höher aktualisieren. Der mit 21.01.0 veröffentlichte Trident-Operator enthielt ein bekanntes Problem, das in 21.01.1 behoben wurde. Weitere Einzelheiten finden Sie unter "Details zum Problem auf GitHub".

Schritt 2: Ermitteln Sie die ursprüngliche Installationsmethode

Um festzustellen, welche Version Sie ursprünglich zur Installation von Trident verwendet haben:

1. Verwenden Sie `kubectl get pods -n trident`, um die Pods zu untersuchen.
   - Wenn kein Operator-Pod vorhanden ist, wurde Trident mit `tridentctl` installiert.
   - Falls ein Operator-Pod vorhanden ist, wurde Trident entweder manuell oder mithilfe von Helm mit dem Trident-Operator installiert.
2. Wenn ein Operator-Pod 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

Im Allgemeinen sollte für das Upgrade dieselbe Methode verwendet werden wie für die Erstinstallation, jedoch kann "zwischen Installationsmethoden wechseln". Es gibt zwei Möglichkeiten, Trident zu aktualisieren.

- "Upgrade mit dem Trident operator"

Wir empfehlen Ihnen, "Den Workflow für das Operator-Upgrade verstehen" vor dem Upgrade mit dem Operator zu überprüfen.

*

Upgrade mit dem Operator

Den Workflow für das Operator-Upgrade verstehen

Bevor Sie den Trident-Operator verwenden, um Trident zu aktualisieren, sollten Sie die Hintergrundprozesse verstehen, die während des Upgrades ablaufen. Dies umfasst Änderungen am Trident-Controller, Controller-Pod und Node-Pods sowie am Node-DaemonSet, die Rolling Updates ermöglichen.

Trident Operator-Upgrade-Handhabung

Einer der vielen "Vorteile der Verwendung des Trident Operators" Möglichkeiten, Trident zu installieren und zu aktualisieren, ist die automatische Verwaltung von Trident- und Kubernetes-Objekten, ohne bestehende eingebundene Volumes zu beeinträchtigen. Auf diese Weise kann Trident Upgrades ohne Ausfallzeiten oder "*laufende Aktualisierungen*" unterstützen. Insbesondere kommuniziert der Trident Operator mit dem Kubernetes-Cluster, um:

- Löschen und erstellen Sie die Trident Controller-Bereitstellung und den Node DaemonSet neu.

- Ersetzen Sie den Trident Controller Pod und die Trident Node Pods durch neue Versionen.
  - Wenn ein Knoten nicht aktualisiert wird, verhindert dies nicht, dass die übrigen Knoten aktualisiert werden.
  - Nur Knoten mit einem laufenden Trident Node Pod können Volumes einbinden.

Weitere Informationen zur Trident-Architektur auf dem Kubernetes-Cluster finden Sie unter "Trident-Architektur".

Operator-Upgrade-Workflow

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 nie mehr als ein Trident Node Pod, unabhängig von der Version, auf einem Knoten befindet.

Aktualisieren Sie eine Trident-Installation mit Trident operator oder Helm

Sie können Trident mithilfe des Trident-Operators entweder manuell oder mit Helm aktualisieren. Sie können von einer Trident-Operator-Installation auf eine andere Trident-Operator-Installation upgraden oder von einer `tridentctl` Installation auf eine Trident-Operator-Version upgraden. Überprüfen Sie "Wählen Sie eine Upgrade-Methode" vor dem Upgrade einer Trident-Operator-Installation.

Eine manuelle Installation aktualisieren

Sie können von einer Trident-Operatorinstallation mit Clusterumfang auf eine andere Trident-Operatorinstallation mit Clusterumfang aktualisieren. Alle Trident Versionen verwenden einen Operator mit Clusterumfang.

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

Über diese Aufgabe

Trident stellt eine Bundle-Datei bereit, die Sie verwenden können, um den Operator zu installieren und zugehörige Objekte für Ihre Kubernetes-Version zu erstellen.

- Für Cluster, auf denen Kubernetes 1.25 oder später läuft, verwenden Sie "bundle_post_1_25.yaml".

Bevor Sie beginnen

Stellen Sie sicher, dass Sie einen Kubernetes-Cluster mit "eine unterstützte Kubernetes-Version" verwenden.

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 Imagepfaden für die Version, auf die Sie aktualisieren (z. B. 25.06), sowie dem korrekten Secret.
3. Löschen Sie den Trident-Operator, der zur Installation der aktuellen Trident Instanz verwendet wurde. Wenn Sie beispielsweise von 25.02 aktualisieren, führen Sie den folgenden Befehl aus:

```
kubectl delete -f 25.02.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

4. Wenn Sie Ihre Erstinstallation mithilfe von `TridentOrchestrator` Attributen angepasst haben, können Sie das `TridentOrchestrator` Objekt bearbeiten, um die Installationsparameter zu ändern. Dies kann beispielsweise Änderungen umfassen, die vorgenommen wurden, 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 passenden Bundle-YAML-Datei für Ihre Umgebung, wobei *<bundle.yaml>* `bundle_pre_1_25.yaml` oder `bundle_post_1_25.yaml` entsprechend Ihrer Kubernetes-Version ist. 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 Trident-Torc, um das Image 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, müssen Sie values.yaml aktualisieren, um `excludePodSecurityPolicy` auf `true` zu setzen oder `--set excludePodSecurityPolicy=true` zum `helm upgrade` Befehl hinzuzufügen, bevor Sie das Cluster aktualisieren können.

Wenn Sie Ihren Kubernetes-Cluster bereits von Version 1.24 auf 1.25 aktualisiert haben, ohne den Trident helm zu aktualisieren, schlägt das helm-Upgrade fehl. Damit das helm-Upgrade durchgeführt werden kann, führen Sie diese Schritte als Voraussetzungen aus:

1. Installieren Sie das helm-mapkubeapis-Plugin von https://github.com/helm/helm-mapkubeapis.
2. Führen Sie einen Testlauf für die Trident-Release im Namespace durch, in dem Trident installiert ist. Dies listet die Ressourcen auf, die bereinigt werden.

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

3. Führen Sie einen vollständigen Lauf mit helm durch, um die Bereinigung vorzunehmen.

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

Schritte

1. Wenn Sie "Trident mit Helm installiert" haben, können Sie `helm upgrade trident netapp-trident/trident-operator --version 100.2602.0` verwenden, um das Upgrade in einem Schritt durchzuführen. Wenn Sie das Helm-Repository nicht hinzugefügt haben oder es nicht für das Upgrade verwenden können:
   a. Laden Sie die neueste Trident Version von "der Abschnitt *Assets* auf GitHub" herunter.
   b. Verwenden Sie den `helm upgrade` Befehl, wobei `trident-operator-26.02.0.tgz` die Version angibt, auf die Sie aktualisieren möchten.

```
helm upgrade <name> trident-operator-26.02.0.tgz
```

> Wenn Sie bei der Erstinstallation benutzerdefinierte Optionen festlegen (z. B. die Angabe privater, gespiegelter Registries für Trident und CSI-Images), hängen Sie den `helm upgrade` Befehl mit `--set` an, um sicherzustellen, dass diese Optionen in den Upgrade-Befehl aufgenommen werden, andernfalls werden die Werte auf die Standardwerte zurückgesetzt.

2. Führen Sie `helm list` aus, um zu überprüfen, ob sowohl die Chart- als auch die App-Version aktualisiert wurden. Führen Sie `tridentctl logs` aus, um eventuelle Debug-Meldungen einzusehen.

Upgrade von einer `tridentctl` Installation auf Trident operator

Sie können auf die neueste Version des Trident-Operators von einer `tridentctl`Installation aus aktualisieren. Die vorhandenen Backends und PVCs stehen automatisch zur Verfügung.

> Vor dem Wechsel zwischen den Installationsmethoden überprüfen Sie "Wechsel zwischen Installationsmethoden".

Schritte

1. Laden Sie die neueste Trident-Version herunter.

```
# Download the release required [26.02.0]
mkdir 26.02.0
cd 26.02.0
wget
https://github.com/NetApp/trident/releases/download/v26.02.0/trident-
installer-26.02.0.tar.gz
tar -xf trident-installer-26.02.0.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-scoped 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 eine `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 beabsichtigte Version aktualisiert wurde.

```
kubectl describe torc trident | grep Message -A 3

Message:                Trident installed
Namespace:              trident
Status:                 Installed
Version:                v26.02.0
```

Upgrade mit tridentctl

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

Über diese Aufgabe

Die Deinstallation und Neuinstallation von Trident entspricht einem Upgrade. Wenn Sie Trident deinstallieren, werden die von der Trident-Bereitstellung verwendeten Persistent Volume Claim (PVC) und Persistent Volume (PV) nicht gelöscht. PVs, die bereits bereitgestellt wurden, bleiben verfügbar, während Trident offline ist, und Trident stellt Volumes für alle PVCs bereit, die in der Zwischenzeit erstellt werden, nachdem es wieder online ist.

Bevor Sie beginnen

Überprüfen Sie "Wählen Sie eine Upgrade-Methode" vor dem Upgrade mit `tridentctl`.

Schritte

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

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

2. Installieren Sie Trident erneut. Siehe "Installieren Sie Trident mit tridentctl".

> Unterbrechen Sie den Aktualisierungsvorgang nicht. Stellen Sie sicher, dass das Installationsprogramm vollständig durchläuft.

Trident mit tridentctl verwalten

Das "Trident Installer-Bundle" beinhaltet das `tridentctl` Befehlszeilenprogramm, um einfachen Zugriff auf Trident zu ermöglichen. 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 Flags

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

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

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

Befehle

`create`
Fügen Sie eine Ressource zu Trident hinzu.

`delete`
Entfernen Sie eine oder mehrere Ressourcen aus Trident.

`get`
Holen Sie eine oder mehrere Ressourcen von Trident.

`help`
Hilfe zu jedem Befehl.

`images`
Drucken Sie eine Tabelle der Container-Images, die Trident benötigt.

`import`
Importieren Sie eine vorhandene Ressource in Trident.

`install`
Installieren Sie Trident.

`logs`
Drucken Sie die Protokolle von Trident.

`send`
Senden Sie eine Ressource von Trident.

`uninstall`
Deinstallieren Sie Trident.

`update`
Modifizieren Sie eine Ressource in Trident.

`update backend state`
Backend-Operationen vorübergehend aussetzen.

`upgrade`
Aktualisieren Sie eine Ressource in Trident.

`version`
Geben Sie die Version von Trident aus.

Globale Flags

`-d, --debug`
Debug-Ausgabe.

`-h, --help`
Hilfe für `tridentctl`.

`-k, --kubeconfig string`
Geben Sie den `KUBECONFIG` Pfad an, um Befehle lokal oder von einem Kubernetes-Cluster zu einem anderen auszuführen.

> Alternativ können Sie die `KUBECONFIG` Variable so exportieren, dass sie auf einen bestimmten Kubernetes-Cluster verweist und `tridentctl` Befehle an diesen Cluster ausführen.

`-n, --namespace string`
Namespace der Trident-Bereitstellung.

`-o, --output string`
Ausgabeformat. Eines der folgenden: json|yaml|name|wide|ps (Standard).

`-s, --server string`
Adresse/Port der Trident REST-Schnittstelle.

> Trident REST interface 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 den `create` Befehl, um eine Ressource zu Trident hinzuzufügen.

```
tridentctl create [option]
```

Optionen

`backend`: Fügen Sie ein Backend zu Trident hinzu.

löschen

Verwenden Sie den `delete` Befehl, um eine oder mehrere Ressourcen aus Trident zu entfernen.

```
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`: Löschen Sie ein oder mehrere Speichervolumes aus Trident.

erhalten

Verwenden Sie den `get` Befehl, um eine oder mehrere Ressourcen von Trident abzurufen.

```
tridentctl get [option]
```

Optionen

`backend`: Ein oder mehrere Speicher-Backends von Trident.
`snapshot`: Ein oder mehrere Snapshots von Trident.
`storageclass`: Eine oder mehrere Speicherklassen von Trident.
`volume`: Ein oder mehrere Volumes von Trident.

Flags

`-h`, `--help`: Hilfe für Volumes.
`--parentOfSubordinate string`: Abfrage auf untergeordnetes Quellvolume beschränken.
`--subordinateOf string`: Abfrage auf untergeordnete Volumes beschränken.

Bilder

Verwenden Sie `images`-Flags, um eine Tabelle der von Trident benötigten Container-Images auszugeben.

```
tridentctl images [flags]
```

Flags

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

Importvolumen

Verwenden Sie den `import volume` Befehl, um ein vorhandenes Volume in Trident zu importieren.

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

Aliase

`volume`, `v`

Flags

`-f`, `--filename string`: Pfad zur YAML- oder JSON-PVC-Datei.
`-h`, `--help`: Hilfe für Volume.
`--no-manage`: Nur PV/PVC erstellen. Lebenszyklusverwaltung des Volumes wird nicht vorausgesetzt.

installieren

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

```
tridentctl install [flags]
```

Flags

`--autosupport-image string`: Das Container-Image für Autosupport-Telemetrie (Standardwert „netapp/trident autosupport:<current-version>“).
`--autosupport-proxy string`: Die Adresse/der Port eines Proxys zum Senden von Autosupport-Telemetrie.
`--enable-node-prep`: Versuch, erforderliche Pakete auf den Nodes zu installieren.
`--generate-custom-yaml`: YAML-Dateien generieren, ohne etwas zu installieren.
`-h`, `--help`: Hilfe für die Installation.
`--http-request-timeout`: Die HTTP-Anfrage-Zeitüberschreitung für die REST-API des Trident-Controllers überschreiben (Standardwert 1m30s).
`--image-registry string`: Die Adresse/der Port einer internen Image-Registry.
`--k8s-timeout duration`: Die Zeitüberschreitung für alle Kubernetes-Operationen (Standardwert 3m0s).
`--kubelet-dir string`: Der Host-Speicherort des internen Status von kubelet (Standardwert „/var/lib/kubelet“).
`--log-format string`: Das Trident-Protokollierungsformat (text, json) (Standardwert „text“).
`--node-prep`: Ermöglicht Trident, die Nodes des Kubernetes-Clusters für die Verwaltung von Volumes mit dem angegebenen Datenspeicherprotokoll vorzubereiten. **Aktuell ist `iscsi` der einzige unterstützte Wert. Ab OpenShift 4.19 ist die minimale unterstützte Trident-Version für dieses Feature 25.06.1.**
`--pv string`: Der Name des von Trident verwendeten Legacy-PV, stellen Sie sicher, dass dieses nicht existiert (Standardwert „trident“).
`--pvc string`: Der Name des von Trident verwendeten Legacy-PVC, stellen Sie sicher, dass dieses nicht existiert (Standardwert „trident“).
`--silence-autosupport`: Keine Autosupport-Bundles automatisch an NetApp senden (Standardwert true).
`--silent`: Die meiste Ausgabe während der Installation deaktivieren.
`--trident-image string`: Das zu installierende Trident-Image.
`--k8s-api-qps`: Das Queries-per-Second-(QPS)-Limit für Kubernetes-API-Anfragen (Standardwert 100; optional).
`--use-custom-yaml`: Vorhandene YAML-Dateien im Setup-Verzeichnis verwenden.
`--use-ipv6`: IPv6 für die Kommunikation von Trident verwenden.

Protokolle

Verwenden Sie `logs`-Flags, um die Protokolle von Trident auszugeben.

```
tridentctl logs [flags]
```

Flags

`-a`, `--archive`: Erstellt ein Support-Archiv mit allen Protokollen, sofern nicht anders angegeben.
`-h`, `--help`: Hilfe für Protokolle.
`-l`, `--log string`: Anzuzeigendes Trident-Protokoll. Einer von trident|auto|trident-operator|all (Standard "auto").
`--node string`: Der Kubernetes Node-Name, von dem die Node-Pod-Protokolle gesammelt werden sollen.
`-p`, `--previous`: Ruft die Protokolle der vorherigen Containerinstanz ab, falls vorhanden.
`--sidecars`: Ruft die Protokolle der Sidecar-Container ab.

senden

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

```
tridentctl send [option]
```

Optionen

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

deinstallieren

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

```
tridentctl uninstall [flags]
```

Flags

`-h, --help`: Hilfe zur Deinstallation.
`--silent`: Deaktiviert die meisten Ausgaben während der Deinstallation.

aktualisieren

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

```
tridentctl update [option]
```

Optionen

`backend`: Aktualisieren Sie ein Backend in Trident.

Backend-Status aktualisieren

Verwenden Sie den `update backend state` Befehl, um Backend-Operationen anzuhalten oder fortzusetzen.

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

Zu berücksichtigende Punkte

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

  Flags

  `-h, --help`: Hilfe zum Backend-Status.
  `--user-state`: Auf `suspended` setzen, um Backend-Operationen anzuhalten. Auf `normal` setzen, um Backend-Operationen fortzusetzen. Wenn auf `suspended` gesetzt:

- `AddVolume` und `Import Volume` sind pausiert.

- `CloneVolume`, `ResizeVolume`, `PublishVolume`, `UnPublishVolume`, `CreateSnapshot`, `GetSnapshot`, `RestoreSnapshot`, `DeleteSnapshot`, `RemoveVolume`, `GetVolumeExternal`, `ReconcileNodeAccess` bleiben verfügbar.

Sie können den Backend-Status auch über das `userState` Feld in der Backend-Konfigurationsdatei `TridentBackendConfig` oder `backend.json` aktualisieren. 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` mit der `backend.json` Datei zu aktualisieren:

1. Bearbeiten Sie die `backend.json` Datei, um das `userState` Feld mit dem Wert 'suspended' einzufügen.
2. Aktualisieren Sie das Backend mit dem `tridentctl update backend` Befehl und dem 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 nach ihrer Anwendung mit dem `kubectl edit <tbc-name> -n <namespace>` Befehl bearbeiten. Das folgende Beispiel aktualisiert den Backend-Status auf „Angehalten“ mit der `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 Sie `version`Flags, um die Version von `tridentctl` und den laufenden Trident Service auszugeben.

```
tridentctl version [flags]
```

Flags

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

Plugin-Unterstützung

Tridentctl unterstützt Plugins ähnlich 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, durchsucht tridentctl nur den angegebenen Ordner.

Trident überwachen

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 Vorgänge ablaufen und ob Trident wie erwartet mit den Backends kommunizieren kann.
- Untersuchen Sie die Backend-Nutzungsinformationen und verstehen Sie, wie viele Volumes auf einem Backend bereitgestellt werden und wie viel Speicherplatz verbraucht wird, und so weiter.
- Pflegen Sie eine Zuordnung der Anzahl der auf den verfügbaren Backends bereitgestellten Volumes.
- Performance verfolgen. Sie können sich ansehen, wie lange Trident benötigt, um mit den Backends zu kommunizieren und Operationen auszuführen.

> Standardmäßig werden die Metriken von Trident am Zielport `8001` am `/metrics` Endpoint bereitgestellt. Diese Metriken sind **standardmäßig aktiviert**, wenn Trident installiert ist. Sie können Trident so konfigurieren, dass die Metriken auch über HTTPS an Port `8444` abgerufen werden.

Was Sie benötigen

- Ein Kubernetes-Cluster mit installiertem Trident.
- Eine Prometheus-Instanz. Dies kann eine "containerisierte Prometheus-Bereitstellung" sein oder Sie können wählen, Prometheus als "native Anwendung" auszuführen.

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 und so weiter zu erhalten. Siehe "Prometheus Operator-Dokumentation".

Schritt 2: Erstellen Sie einen Prometheus ServiceMonitor

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

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

Diese ServiceMonitor-Definition ruft die von dem `trident-csi` Service zurückgegebenen Metriken ab und sucht gezielt nach dem `metrics` Endpoint des Service. Dadurch ist Prometheus nun so konfiguriert, dass es die Metriken von Trident versteht.

Zusätzlich zu den direkt von Trident verfügbaren Metriken stellt Kubelet viele `kubelet_volume_*` Metriken über seinen eigenen Metrik-Endpunkt bereit. Kubelet kann Informationen über die Volumes, die angehängt sind, sowie über Pods und andere interne Operationen, die es verarbeitet, bereitstellen. Siehe "hier,".

Trident-Metriken über HTTPS abrufen

Um Trident-Metriken über HTTPS (Port 8444) zu empfangen, müssen Sie die ServiceMonitor-Definition um die TLS-Konfiguration ergänzen. Außerdem müssen Sie das `trident-csi` Secret aus dem `trident` Namespace in den Namespace kopieren, in dem Prometheus ausgeführt wird. Dies können Sie mit folgendem Befehl tun:

```
kubectl get secret trident-csi -n trident -o yaml | sed 's/namespace:
trident/namespace: monitoring/' | kubectl apply -f -
```

Ein Beispiel ServiceMonitor für HTTPS-Metriken 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:
    - interval: 15s
      path: /metrics
      port: https-metrics
      scheme: https
      tlsConfig:
        ca:
          secret:
            key: caCert
            name: trident-csi
        cert:
          secret:
            key: clientCert
            name: trident-csi
        keySecret:
          key: clientKey
          name: trident-csi
        serverName: trident-csi
```

Trident unterstützt HTTPS-Metriken in allen Installationsmethoden: tridentctl, Helm chart und Operator:

- Wenn Sie den `tridentctl install` Befehl verwenden, können Sie das `--https-metrics` Flag übergeben, um HTTPS-Metriken zu aktivieren.
- Wenn Sie das Helm-Chart verwenden, können Sie den `httpsMetrics` Parameter einstellen, um HTTPS-Metriken zu aktivieren.
- Wenn Sie YAML-Dateien verwenden, können Sie das `--https_metrics` Flag zum `trident-main` Container in der `trident-deployment.yaml` Datei hinzufügen.

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 Trident-Gesundheitsinformationen

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

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

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

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

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

```
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

Dies ist nur aktiviert, wenn auch kubelet-Metriken erfasst werden.

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

```
kubelet_volume_stats_used_bytes / kubelet_volume_stats_capacity_bytes *
100
```

Erfahren Sie mehr über Trident AutoSupport-Telemetrie

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 das `--silence-autosupport` Flag während der Trident-Installation.
- Trident kann Container-Logs auch auf Anfrage an den NetApp Support senden `tridentctl send autosupport`. Sie müssen Trident auslösen, damit es seine Logs hochlädt. Bevor Sie Logs einreichen, sollten Sie NetApps "Datenschutzrichtlinie" akzeptieren.
- Sofern nicht anders angegeben, ruft Trident die Protokolle der letzten 24 Stunden ab.
- Sie können den Aufbewahrungszeitraum für Protokolle mit dem `--since` Flag festlegen. Zum Beispiel: `tridentctl send autosupport --since=1h`. Diese Informationen werden erfasst und über einen `trident-autosupport` Container gesendet, der zusammen mit Trident installiert wird. Sie können das Container-Image unter "Trident AutoSupport" beziehen.
- Trident AutoSupport sammelt oder überträgt keine personenbezogenen Daten (PII) oder persönlichen Informationen. Es enthält eine "EULA" , die nicht für das Trident-Container-Image selbst gilt. Sie können mehr über das Engagement von NetApp 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-Nachrichten werden an NetApps AutoSupport-Endpunkt gesendet. Wenn Sie eine private Registry zum Speichern von Container-Images verwenden, können Sie das `--image-registry`-Flag verwenden.
- Sie können Proxy-URLs auch konfigurieren, indem Sie die Installations-YAML-Dateien generieren. Dies kann erfolgen, indem Sie `tridentctl install --generate-custom-yaml` verwenden, um die

YAML-Dateien zu erstellen, und das `--proxy-url` Argument für den `trident-autosupport` Container in `trident-deployment.yaml` hinzufügen.

Trident-Metriken deaktivieren

Um die Meldung von Metriken **zu deaktivieren**, sollten Sie benutzerdefinierte YAML-Dateien (unter Verwendung des `--generate-custom-yaml` Flags) generieren und diese bearbeiten, um das `--metrics` Flag aus der Ausführung für den `trident-main` Container zu entfernen.

Trident deinstallieren

Sie sollten die gleiche Methode zum Deinstallieren von Trident verwenden, die Sie auch zum Installieren von Trident verwendet haben.

Über diese Aufgabe

- Falls nach einem Upgrade Fehler, Abhängigkeitsprobleme oder ein fehlgeschlagenes bzw. unvollständiges Upgrade auftreten, 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 *downzugraden*.
- Für ein einfaches Upgrade und eine Neuinstallation entfernt das Deinstallieren von Trident weder die CRDs noch die von Trident erstellten zugehörigen Objekte. Wenn Sie Trident und alle seine Daten vollständig entfernen müssen, lesen Sie "Trident und CRDs vollständig entfernen".

Bevor Sie beginnen

Wenn Sie Kubernetes-Cluster außer Betrieb nehmen, müssen Sie alle Anwendungen löschen, die von Trident erstellte Volumes verwenden, bevor Sie Trident deinstallieren. Dadurch wird sichergestellt, dass PVCs auf den Kubernetes-Knoten entfernt werden, bevor sie gelöscht werden.

Ermitteln Sie die ursprüngliche Installationsmethode

Sie sollten die gleiche Methode zur Deinstallation von Trident verwenden, die Sie auch zur Installation verwendet haben. Überprüfen Sie vor der Deinstallation, welche Version Sie ursprünglich zur Installation von Trident verwendet haben.

1. Verwenden Sie `kubectl get pods -n trident`, um die Pods zu untersuchen.
   - Wenn kein Operator-Pod vorhanden ist, wurde Trident mit `tridentctl` installiert.
   - Falls ein Operator-Pod vorhanden ist, wurde Trident entweder manuell oder mithilfe von Helm mit dem Trident-Operator installiert.
2. Wenn ein Operator-Pod 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.

Entfernen einer Trident-Operatorinstallation

Sie können eine Trident-Operator-Installation manuell oder mit Helm deinstallieren.

Manuelle Installation deinstallieren

Wenn Sie Trident mithilfe des Operators installiert haben, können Sie es auf eine der folgenden Arten deinstallieren:

1. **Edit `TridentOrchestrator` CR und Deinstallationsflag setzen**:

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

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

2. **Löschen `TridentOrchestrator`**: Durch das Entfernen der `TridentOrchestrator` CR, die zur Bereitstellung von Trident verwendet wurde, weisen Sie den Operator an, Trident zu deinstallieren. Der Operator verarbeitet die Entfernung von `TridentOrchestrator` und fährt fort, die Trident-Bereitstellung und das DaemonSet zu entfernen, wobei die im Rahmen der Installation erstellten Trident-Pods gelöscht werden.

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

Helm-Installation deinstallieren

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

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

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

Deinstallieren einer `tridentctl`Installation

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

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