



# **Gestire Astra Trident**

## **Astra Trident**

NetApp  
November 14, 2025

# Sommario

Gestire Astra Trident .....	1
Aggiorna Astra Trident .....	1
Aggiorna Astra Trident .....	1
Eseguire l'upgrade con l'operatore .....	2
Upgrade con tridentctl .....	10
Disinstallare Astra Trident .....	14
Disinstallare utilizzando Helm .....	14
Disinstallare utilizzando l'operatore Trident .....	15
Disinstallare utilizzando tridentctl .....	16
Downgrade di Astra Trident .....	16
Quando eseguire il downgrade .....	16
Quando non eseguire il downgrade .....	16
Processo di downgrade quando Astra Trident viene installato utilizzando l'operatore .....	16
Processo di downgrade quando Astra Trident viene installato mediante tridentctl .....	18

# Gestire Astra Trident

## Aggiorna Astra Trident

### Aggiorna Astra Trident

Astra Trident segue una cadenza di rilascio trimestrale, fornendo quattro release principali ogni anno di calendario. Ogni nuova release si basa sulle release precedenti, fornendo nuove funzionalità e miglioramenti delle performance, oltre a correzioni di bug e miglioramenti. Ti consigliamo di effettuare l'upgrade almeno una volta all'anno per sfruttare le nuove funzionalità di Astra Trident.

### Selezionare una versione

Le versioni di Astra Trident seguono una data-based YY.MM Convenzione di naming, dove "YY" è l'ultima cifra dell'anno e "MM" è il mese. I rilasci di punti seguono un YY.MM.X convention, dove "X" è il livello di patch. Selezionare la versione a cui eseguire l'aggiornamento in base alla versione da cui si sta eseguendo l'aggiornamento.

- È possibile eseguire un aggiornamento diretto a qualsiasi release di destinazione che si trova all'interno di una finestra di quattro release della versione installata. Ad esempio, è possibile eseguire direttamente l'aggiornamento alla versione 23.01 da 22.01 (incluse le release DOT, ad esempio 22.01.1).
- Se si dispone di una release precedente, è necessario eseguire un aggiornamento in più fasi utilizzando la documentazione della release corrispondente per istruzioni specifiche. Ciò richiede l'aggiornamento alla versione più recente che si adatta alla finestra di quattro release. Ad esempio, se si utilizza 18.07 e si desidera eseguire l'aggiornamento alla versione 20.07, seguire il processo di aggiornamento in più fasi come indicato di seguito:
  - a. Primo aggiornamento da 18.07 a 19.07.
  - b. Eseguire l'aggiornamento da 19.07 a 20.07.



- Tutti gli aggiornamenti per le versioni 19.04 e precedenti richiedono la migrazione dei metadati di Astra Trident `etcd` Agli oggetti CRD. Verificare la documentazione della release per comprendere il funzionamento dell'aggiornamento.
- Durante l'aggiornamento, è importante fornire `parameter.fsType poll StorageClasses` Utilizzato da Astra Trident. È possibile eliminare e ricreare `StorageClasses` senza interrompere i volumi preesistenti. Si tratta di un **requisito** per l'applicazione di **security contesti** per i volumi SAN. La directory [sample input](#) contiene esempi, ad esempio `storage-class-basic.yaml.templ` e `storage-class-bronze-default.yaml`. Per ulteriori informazioni, vedere ["Problemi noti"](#).

### Selezionare un'opzione di aggiornamento

Ci sono due opzioni per aggiornare Astra Trident. In genere, si utilizzerà la stessa opzione utilizzata per l'installazione iniziale, tuttavia è possibile ["passare da un metodo di installazione all'altro"](#).

- ["Eseguire l'aggiornamento utilizzando l'operatore Trident"](#)  
\*



CSI Volume Snapshots è ora una funzionalità GA, a partire da Kubernetes 1.20. Quando si aggiorna Astra Trident, tutti i CRS e i CRD di snapshot alfa precedenti (classi di snapshot dei volumi, snapshot dei volumi e contenuti di snapshot dei volumi) devono essere rimossi prima di eseguire l'aggiornamento. Fare riferimento a ["questo blog"](#) Comprendere i passaggi necessari per la migrazione delle snapshot Alpha alle specifiche beta/GA.

## Modifiche all'operatore

La release 21.01 di Astra Trident introduce alcune importanti modifiche architetturali all'operatore, vale a dire quanto segue:

- L'operatore è ora **con ambito cluster**. Le istanze precedenti dell'operatore Trident (versioni da 20.04 a 20.10) erano **namespace-scope**. Un operatore con ambito cluster è vantaggioso per i seguenti motivi:
  - Responsabilità delle risorse: L'operatore gestisce ora le risorse associate a un'installazione di Astra Trident a livello di cluster. Nell'ambito dell'installazione di Astra Trident, l'operatore crea e gestisce diverse risorse utilizzando `ownerReferences`. Manutenzione `ownerReferences` Su risorse con ambito cluster possono generare errori su alcuni distributori Kubernetes come OpenShift. Questo è mitigato da un operatore con ambito cluster. Per la riparazione automatica e l'applicazione di patch alle risorse Trident, questo è un requisito essenziale.
  - Pulizia durante la disinstallazione: Una rimozione completa di Astra Trident richiederebbe l'eliminazione di tutte le risorse associate. Un operatore con ambito spazio dei nomi potrebbe riscontrare problemi con la rimozione delle risorse con ambito del cluster (come `ClusterRole`, `ClusterRoleBinding` e `PodSecurityPolicy`) e portare a una pulizia incompleta. Un operatore con ambito cluster elimina questo problema. Gli utenti possono disinstallare completamente Astra Trident e installare di nuovo, se necessario.
- `TridentProvisioner` viene ora sostituito con `TridentOrchestrator` Come risorsa personalizzata utilizzata per installare e gestire Astra Trident. Inoltre, viene introdotto un nuovo campo in `TridentOrchestrator spec`. Gli utenti possono specificare che lo spazio dei nomi Trident deve essere installato/aggiornato utilizzando `spec.namespace` campo. Puoi dare un'occhiata a un esempio ["qui"](#).

## Eseguire l'upgrade con l'operatore

È possibile aggiornare facilmente un'installazione Astra Trident esistente utilizzando l'operatore.

### Prima di iniziare

Per eseguire l'aggiornamento utilizzando l'operatore, devono essere soddisfatte le seguenti condizioni:

- È necessario disporre di un'installazione Astra Trident basata su CSI. Tutte le release a partire dalla versione 19.07 sono basate su CSI. Per verificare, è possibile esaminare i pod nello spazio dei nomi Trident.
  - La denominazione dei pod nelle versioni precedenti alla 23.01 segue una `trident-csi-*` convenzione.
  - La naming dei pod nella versione 23.01 e successive utilizza: `trident-controller-<generated id>` per il controller pod; `trident-node-<operating system>-<generated id>` per i pod di nodi; `trident-operator-<generated id>` per il pod operatore.
- Se CSI Trident è stato disinstallato e i metadati dell'installazione persistono, è possibile eseguire l'aggiornamento utilizzando l'operatore.
- Deve esistere una sola installazione Astra Trident in tutti gli spazi dei nomi di un determinato cluster

Kubernetes.

- Si dovrebbe utilizzare un cluster Kubernetes in esecuzione "[Una versione di Kubernetes supportata](#)".
- Se sono presenti CRD Alpha Snapshot, rimuoverli con `tridentctl obliviate alpha-snapshot-crd`. In questo modo vengono eliminati i CRD per le specifiche di snapshot alfa. Per gli snapshot esistenti che devono essere cancellati/migrati, vedere "[questo blog](#)".



- Quando si aggiorna Trident utilizzando l'operatore su OpenShift Container Platform, è necessario eseguire l'aggiornamento a Trident 21.01.1 o versione successiva. L'operatore Trident rilasciato con 21.01.0 contiene un problema noto che è stato risolto nel 21.01.1. Per ulteriori informazioni, vedere "[Dettagli del problema su GitHub](#)".
- Non utilizzare l'operatore per aggiornare Trident se si utilizza un `etcd` Versione Trident basata su (19.04 o precedente).

## Aggiornare un'installazione di un operatore Trident con ambito cluster

Per aggiornare un'installazione di un operatore Trident con ambito cluster, procedere come segue. Tutte le versioni di Astra Trident 21.01 e successive utilizzano un operatore con ambito cluster.

### Fasi

1. Verificare la versione di Astra Trident:

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

2. Eliminare l'operatore Trident utilizzato per installare l'istanza corrente di Astra Trident. Ad esempio, se si esegue l'aggiornamento da 22.01, eseguire il seguente comando:

```
kubectl delete -f 22.01/trident-installer/deploy/bundle.yaml -n trident
```

3. Se l'installazione iniziale è stata personalizzata utilizzando `TridentOrchestrator` è possibile modificare `TridentOrchestrator` oggetto per modificare i parametri di installazione. Ciò potrebbe includere le modifiche apportate per specificare i registri di immagini Trident e CSI mirrorati per la modalità offline, abilitare i registri di debug o specificare i segreti di pull delle immagini.
4. Installare Astra Trident utilizzando il file YAML del bundle corretto per il proprio ambiente e la versione di Astra Trident. Ad esempio, se si sta installando Astra Trident 23.01 per Kubernetes 1.26, eseguire il seguente comando:

```
kubectl create -f 23.01.1/trident-installer/deploy/bundle_post_1_25.yaml  
-n trident
```

Trident fornisce un file bundle che può essere utilizzato per installare l'operatore e creare oggetti associati per la versione di Kubernetes.



- Per i cluster che eseguono Kubernetes 1.24 o versione precedente, utilizzare `"bundle_pre_1_25.yaml"`.
- Per i cluster che eseguono Kubernetes 1.25 o versioni successive, utilizzare `"bundle_post_1_25.yaml"`.

## Risultati

L'operatore Trident identificherà un'installazione Astra Trident esistente e la aggiornerà alla stessa versione dell'operatore.

## Aggiornare un'installazione dell'operatore con ambito namespace

Seguire questa procedura per eseguire l'aggiornamento da un'istanza di Astra Trident installata utilizzando l'operatore con ambito namespace (versioni da 20.07 a 20.10).

### Fasi

1. Verificare lo stato dell'installazione Trident esistente. Per eseguire questa operazione, selezionare il valore **Status** di `TridentProvisioner`. Lo stato deve essere `Installed`.

```
kubectl describe tprov trident -n trident | grep Message: -A 3
Message:  Trident installed
Status:   Installed
Version:  v20.10.1
```



Se viene visualizzato lo stato `Updating`, assicurarsi di risolverlo prima di procedere. Per un elenco dei possibili valori di stato, vedere ["qui"](#).

2. Creare il `TridentOrchestrator` CRD utilizzando il manifesto fornito con il programma di installazione di Trident.

```
# Download the release required [23.01.1]
mkdir 23.01.1
cd 23.01.1
wget
https://github.com/NetApp/trident/releases/download/v23.01.1/trident-
installer-23.01.1.tar.gz
tar -xf trident-installer-23.01.1.tar.gz
cd trident-installer
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
```

3. Eliminare l'operatore con ambito dello spazio dei nomi utilizzando il relativo manifesto. Per completare questo passaggio, è necessario il file YAML bundle utilizzato per implementare l'operatore con ambito dello spazio dei nomi da <https://github.com/NetApp/trident/tree/stable/vXX.XX>

/deploy/*BUNDLE.YAML* dove *vXX.XX* è il numero di versione e *BUNDLE.YAML* È il nome del file YAML del bundle.



È necessario apportare le modifiche necessarie ai parametri di installazione di Trident (ad esempio, modificando i valori per `tridentImage`, `autosupportImage`, `repository` di immagini privato e fornitura `imagePullSecrets`) dopo aver eliminato l'operatore con ambito dello spazio dei nomi e prima di installare l'operatore con ambito del cluster. Per un elenco completo dei parametri che è possibile aggiornare, fare riferimento a ["opzioni di configurazione"](#).

```
#Ensure you are in the right directory
pwd
/root/20.10.1/trident-installer

#Delete the namespace-scoped operator
kubectl delete -f deploy/<BUNDLE.YAML> -n trident
serviceaccount "trident-operator" deleted
clusterrole.rbac.authorization.k8s.io "trident-operator" deleted
clusterrolebinding.rbac.authorization.k8s.io "trident-operator" deleted
deployment.apps "trident-operator" deleted
podsecuritypolicy.policy "tridentoperatorpods" deleted

#Confirm the Trident operator was removed
kubectl get all -n trident
NAME                                READY   STATUS    RESTARTS   AGE
pod/trident-csi-68d979fb85-dsrmn    6/6    Running   12         99d
pod/trident-csi-8jfhf                2/2    Running   6          105d
pod/trident-csi-jtnjz                2/2    Running   6          105d
pod/trident-csi-lcxvh                2/2    Running   8          105d

NAME                                TYPE          CLUSTER-IP      EXTERNAL-IP      PORT(S)
AGE
service/trident-csi                 ClusterIP     10.108.174.125  <none>
34571/TCP,9220/TCP                 105d

NAME                                DESIRED   CURRENT   READY   UP-TO-DATE   AGE
AVAILABLE   NODE SELECTOR
daemonset.apps/trident-csi          3         3         3       3             3
kubernetes.io/arch=amd64,kubernetes.io/os=linux  105d

NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/trident-csi         1/1     1             1           105d

NAME                                DESIRED   CURRENT   READY   AGE
replicaset.apps/trident-csi-68d979fb85  1         1         1       105d
```

A questo punto, il `trident-operator-xxxxxxxx-xxxxx` pod eliminato.

4. (Facoltativo) se è necessario modificare i parametri di installazione, aggiornare `TridentProvisioner` spec. Tali modifiche potrebbero essere apportate, ad esempio, alla modifica del Registro di sistema dell'immagine privata per estrarre le immagini container, abilitare i registri di debug o specificare i segreti di pull delle immagini.

```
kubectl patch tprov <trident-provisioner-name> -n <trident-namespace>
--type=merge -p '{"spec":{"debug":true}}'
```

5. Installare l'operatore Trident.



L'installazione dell'operatore con ambito cluster avvia la migrazione di `TridentProvisioner` oggetti a `TridentOrchestrator` oggetti, elimina `TridentProvisioner` oggetti e il `tridentprovisioner` CRD e aggiorna Astra Trident alla versione dell'operatore cluster-scoped in uso. Nell'esempio seguente, Trident viene aggiornato alla versione 23.01.1.



L'aggiornamento di Astra Trident con l'operatore Trident comporta la migrazione di `tridentProvisioner` a un `tridentOrchestrator` oggetto con lo stesso nome. Questo viene gestito automaticamente dall'operatore. Nell'aggiornamento verrà installato anche Astra Trident nello stesso namespace di prima.

```

#Ensure you are in the correct directory
pwd
/root/23.01.1/trident-installer

#Install the cluster-scoped operator in the **same namespace**
kubectl create -f deploy/<BUNDLE.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

#All tridentProvisioners will be removed, including the CRD itself
kubectl get tprov -n trident
Error from server (NotFound): Unable to list "trident.netapp.io/v1,
Resource=tridentprovisioners": the server could not find the requested
resource (get tridentprovisioners.trident.netapp.io)

#tridentProvisioners are replaced by tridentOrchestrator
kubectl get torc
NAME          AGE
trident       13s

#Examine Trident pods in the namespace
kubectl get pods -n trident
NAME                                                    READY   STATUS    RESTARTS
AGE
trident-controller-79df798bdc-m79dc                    6/6     Running   0
1m41s
trident-node-linux-xrst8                               2/2     Running   0
1m41s
trident-operator-5574dbbc68-nthjv                      1/1     Running   0
1m52s

#Confirm Trident has been updated to the desired version
kubectl describe torc trident | grep Message -A 3
Message:          Trident installed
Namespace:       trident
Status:          Installed
Version:         v23.01.1

```



Il `trident-controller` e i nomi dei pod riflettono la convenzione di naming introdotta nel 23.01.

## Aggiornare un'installazione basata su Helm

Per aggiornare un'installazione basata su Helm, procedere come segue.



Quando si aggiorna un cluster Kubernetes dalla versione 1.24 alla 1.25 o successiva su cui è installato Astra Trident, è necessario aggiornare `values.yaml` per impostarlo `excludePodSecurityPolicy` a `true` oppure aggiungi `--set excludePodSecurityPolicy=true` al `helm upgrade` prima di aggiornare il cluster.

### Fasi

1. Scarica l'ultima release di Astra Trident.
2. Utilizzare `helm upgrade` comando dove `trident-operator-23.01.1.tgz` indica la versione alla quale si desidera eseguire l'aggiornamento.

```
helm upgrade <name> trident-operator-23.01.1.tgz
```

Se si impostano opzioni non predefinite durante l'installazione iniziale (ad esempio, se si specificano registri privati mirrorati per le immagini Trident e CSI), utilizzare `--set` per assicurarsi che tali opzioni siano incluse nel comando `upgrade`, altrimenti i valori torneranno ai valori predefiniti.



Ad esempio, per modificare il valore predefinito di `tridentDebug`, eseguire il seguente comando:

```
helm upgrade <name> trident-operator-23.01.1-custom.tgz --set tridentDebug=true
```

3. Eseguire `helm list` per verificare che la versione del grafico e dell'applicazione sia stata aggiornata. Eseguire `tridentctl logs` per esaminare eventuali messaggi di debug.

### Risultati

L'operatore Trident identificherà un'installazione Astra Trident esistente e la aggiornerà alla stessa versione dell'operatore.

### Aggiornamento da un'installazione non eseguita dall'operatore

È possibile eseguire l'aggiornamento all'ultima versione dell'operatore Trident da un `tridentctl` installazione.

### Fasi

1. Scarica l'ultima release di Astra Trident.

```
# Download the release required [23.01.1]
mkdir 23.01.1
cd 23.01.1
wget
https://github.com/NetApp/trident/releases/download/v22.01.1/trident-
installer-23.01.1.tar.gz
tar -xf trident-installer-23.01.1.tar.gz
cd trident-installer
```

## 2. Creare il tridentorchestrator CRD dal manifesto.

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

## 3. Implementare l'operatore.

```
#Install the cluster-scoped operator in the same namespace
kubectl create -f deploy/<BUNDLE.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. Creare un TridentOrchestrator CR per l'installazione di Astra Trident.

```

#Create a tridentOrchestrator to initiate a Trident install
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

#Confirm Trident was upgraded to the desired version
kubectl describe torc trident | grep Message -A 3
Message:                             Trident installed
Namespace:                           trident
Status:                               Installed
Version:                              v23.01.1

```

## Risultati

I backend e i PVC esistenti sono automaticamente disponibili.

## Upgrade con tridentctl

È possibile aggiornare facilmente un'installazione Astra Trident utilizzando `tridentctl`.

### Considerazioni prima dell'aggiornamento

Quando si esegue l'aggiornamento all'ultima release di Astra Trident, considerare quanto segue:

- A partire da Trident 20.01, solo la versione beta di ["snapshot dei volumi"](#) è supportato. Gli amministratori di Kubernetes devono fare attenzione a eseguire il backup o la conversione degli oggetti snapshot alfa in versione beta in modo sicuro per conservare le snapshot alfa legacy.
- La versione beta delle snapshot dei volumi introduce un set modificato di CRD e un controller di snapshot, entrambi da configurare prima di installare Astra Trident. ["Questo blog"](#) vengono illustrate le fasi della migrazione delle snapshot dei volumi alpha al formato beta.
- La disinstallazione e la reinstallazione di Astra Trident funge da aggiornamento. Quando si disinstalla Trident, i PVC (Persistent Volume Claim) e PV (Persistent Volume) utilizzati dall'implementazione di Astra Trident non vengono cancellati. I PVS già forniti resteranno disponibili mentre Astra Trident è offline e Astra Trident effettuerà il provisioning dei volumi per i PVC creati nel frattempo una volta tornati online.



Durante l'aggiornamento di Astra Trident, non interrompere il processo di aggiornamento. Assicurarsi che il programma di installazione venga completato.

## Passi successivi dopo l'aggiornamento

Per utilizzare l'insieme completo di funzionalità disponibili nelle versioni più recenti di Trident (ad esempio, le snapshot dei volumi on-Demand), è possibile aggiornare i volumi utilizzando `tridentctl upgrade` comando.

Se sono presenti volumi legacy, è necessario aggiornarli da un tipo NFS/iSCSI al tipo CSI per poter utilizzare il set completo di nuove funzionalità di Astra Trident. Un PV legacy che è stato fornito da Trident supporta il set tradizionale di funzionalità.

Quando si decide di aggiornare i volumi al tipo CSI, considerare quanto segue:

- Potrebbe non essere necessario aggiornare tutti i volumi. I volumi creati in precedenza continueranno ad essere accessibili e funzioneranno normalmente.
- Un PV può essere montato come parte di un'implementazione/StatefulSet durante l'aggiornamento. Non è necessario mettere fuori servizio il deployment/StatefulSet.
- Non è possibile collegare un PV a un pod standalone durante l'aggiornamento. Chiudere il pod prima di aggiornare il volume.
- È possibile aggiornare solo un volume associato a un PVC. I volumi che non sono associati a PVC devono essere rimossi e importati prima dell'aggiornamento.

## Esempio di aggiornamento del volume

Ecco un esempio che mostra come viene eseguito un aggiornamento di un volume.

1. Eseguire `kubectl get pv` Per elencare il PVS.

```
kubectl get pv
NAME                                CAPACITY  ACCESS MODES  RECLAIM POLICY
STATUS  CLAIM                                STORAGECLASS  REASON  AGE
default-pvc-1-a8475                 1073741824  RWO           Delete
Bound  default/pvc-1                       standard      19h
default-pvc-2-a8486                 1073741824  RWO           Delete
Bound  default/pvc-2                       standard      19h
default-pvc-3-a849e                 1073741824  RWO           Delete
Bound  default/pvc-3                       standard      19h
default-pvc-4-a84de                 1073741824  RWO           Delete
Bound  default/pvc-4                       standard      19h
trident                             2Gi        RWO           Retain
Bound  trident/trident                      19h
```

Attualmente sono disponibili quattro PVS creati da Trident 20.07, utilizzando `netapp.io/trident` provisioner.

2. Eseguire `kubectl describe pv` Per ottenere i dettagli del PV.

```
kubectl describe pv default-pvc-2-a8486

Name:                default-pvc-2-a8486
Labels:              <none>
Annotations:         pv.kubernetes.io/provisioned-by: netapp.io/trident
                    volume.beta.kubernetes.io/storage-class: standard
Finalizers:          [kubernetes.io/pv-protection]
StorageClass:        standard
Status:              Bound
Claim:               default/pvc-2
Reclaim Policy:      Delete
Access Modes:        RWO
VolumeMode:          Filesystem
Capacity:            1073741824
Node Affinity:       <none>
Message:
Source:
  Type:              NFS (an NFS mount that lasts the lifetime of a pod)
  Server:            10.xx.xx.xx
  Path:              /trid_1907_alpha_default_pvc_2_a8486
  ReadOnly:          false
```

Il PV è stato creato utilizzando `netapp.io/trident` provisioner ed è del tipo NFS. Per supportare tutte le nuove funzioni fornite da Astra Trident, questo PV deve essere aggiornato al tipo CSI.

3. Eseguire `tridentctl upgrade volume <name-of-trident-volume>` Comando per aggiornare un volume Astra Trident legacy alla specifica CSI.

```

./tridentctl get volumes -n trident
+-----+-----+-----+-----+
+-----+-----+-----+-----+
|          NAME          |  SIZE  | STORAGE CLASS | PROTOCOL |
BACKEND UUID            | STATE  | MANAGED      |          |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
| default-pvc-2-a8486 | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true         |          |
| default-pvc-3-a849e | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true         |          |
| default-pvc-1-a8475 | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true         |          |
| default-pvc-4-a84de | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true         |          |
+-----+-----+-----+-----+
+-----+-----+-----+-----+

./tridentctl upgrade volume default-pvc-2-a8486 -n trident
+-----+-----+-----+-----+
+-----+-----+-----+-----+
|          NAME          |  SIZE  | STORAGE CLASS | PROTOCOL |
BACKEND UUID            | STATE  | MANAGED      |          |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
| default-pvc-2-a8486 | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true         |          |
+-----+-----+-----+-----+
+-----+-----+-----+-----+

```

4. Eseguire un `kubectl describe pv` Per verificare che il volume sia un volume CSI.

```

kubect1 describe pv default-pvc-2-a8486
Name:                default-pvc-2-a8486
Labels:              <none>
Annotations:         pv.kubernetes.io/provisioned-by: csi.trident.netapp.io
                    volume.beta.kubernetes.io/storage-class: standard
Finalizers:          [kubernetes.io/pv-protection]
StorageClass:        standard
Status:              Bound
Claim:               default/pvc-2
Reclaim Policy:      Delete
Access Modes:        RWO
VolumeMode:          Filesystem
Capacity:            1073741824
Node Affinity:       <none>
Message:
Source:
  Type:               CSI (a Container Storage Interface (CSI) volume
source)
  Driver:             csi.trident.netapp.io
  VolumeHandle:       default-pvc-2-a8486
  ReadOnly:           false
  VolumeAttributes:   backendUUID=c5a6f6a4-b052-423b-80d4-
8fb491a14a22

internalName=trid_1907_alpha_default_pvc_2_a8486
                    name=default-pvc-2-a8486
                    protocol=file
Events:              <none>

```

In questo modo, è possibile aggiornare i volumi di tipo NFS/iSCSI creati da Astra Trident al tipo CSI, in base al volume.

## Disinstallare Astra Trident

A seconda della modalità di installazione di Astra Trident, esistono diverse opzioni per disinstallarlo.

### Disinstallare utilizzando Helm

Se Astra Trident è stato installato utilizzando Helm, è possibile disinstallarlo utilizzando `helm uninstall`.

```
#List the Helm release corresponding to the Astra 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
```

## Disinstallare utilizzando l'operatore Trident

Se Astra Trident è stato installato utilizzando l'operatore, è possibile disinstallarlo eseguendo una delle seguenti operazioni:

- **Modifica `TridentOrchestrator`** Per impostare il flag di disinstallazione: è possibile modificare `TridentOrchestrator` e impostare `spec.uninstall=true`. Modificare il `TridentOrchestrator` CR e impostare `uninstall` contrassegnare come mostrato di seguito:

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

Quando il `uninstall` flag è impostato su `true`, L'operatore Trident disinstalla Trident, ma non rimuove il `TridentOrchestrator` stesso. Se si desidera installare di nuovo Trident, è necessario ripulire `TridentOrchestrator` e crearne uno nuovo.

- **Elimina `TridentOrchestrator`**: rimuovendo il `TridentOrchestrator` CR utilizzato per implementare Astra Trident, si richiede all'operatore di disinstallare Trident. L'operatore elabora la rimozione di `TridentOrchestrator` E procede alla rimozione dell'implementazione e del demonset di Astra Trident, eliminando i pod Trident creati come parte dell'installazione. Per rimuovere completamente Astra Trident (inclusi i CRD creati) e pulire efficacemente l'ardesia pulita, è possibile modificare `TridentOrchestrator` per superare il `wipeout` opzione. Vedere il seguente esempio:

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

In questo modo Astra Trident viene disinstallato completamente e vengono cancellati tutti i metadati relativi ai backend e ai volumi gestiti. Le installazioni successive vengono trattate come installazioni nuove.



Considerare la cancellazione dei CRD solo quando si esegue una disinstallazione completa. Questa operazione non può essere annullata. **Non cancellare i CRD a meno che non si desideri ricominciare e creare una nuova installazione di Astra Trident.**

## Disinstallare utilizzando `tridentctl`

Eseguire `uninstall` ingresso comando `tridentctl` Come segue per rimuovere tutte le risorse associate ad Astra Trident, ad eccezione dei CRD e degli oggetti correlati, semplificando l'esecuzione del programma di installazione per l'aggiornamento a una versione più recente.

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

Per eseguire una rimozione completa di Astra Trident, rimuovere i finalizzatori dei CRD creati da Astra Trident ed eliminare i CRD.

## Downgrade di Astra Trident

Scopri i passaggi necessari per il downgrade a una versione precedente di Astra Trident.

### Quando eseguire il downgrade

È possibile valutare la possibilità di eseguire il downgrade per diversi motivi, ad esempio:

- Pianificazione di emergenza
- Correzione immediata dei bug osservati in seguito a un aggiornamento
- Problemi di dipendenza, aggiornamenti non riusciti e incompleti

Quando si esegue il passaggio a una release di Astra Trident che utilizza i CRD, si consiglia di considerare un downgrade. Poiché Astra Trident utilizza i CRD per mantenere lo stato, tutte le entità di storage create (backend, classi di storage, PV e snapshot di volumi) dispongono di oggetti CRD associati invece che di dati scritti in `trident` PV (utilizzato dalla versione installata in precedenza di Astra Trident). I PVS, i backend e le classi di storage appena creati vengono mantenuti come oggetti CRD.

Provare a eseguire il downgrade solo per una versione di Astra Trident eseguita utilizzando CRD (19.07 e versioni successive). Ciò garantisce che le operazioni eseguite sulla release corrente di Astra Trident siano visibili dopo il downgrade.

### Quando non eseguire il downgrade

Non eseguire il downgrade a una release di Trident che utilizza `etcd` per mantenere lo stato (19.04 e versioni precedenti). Tutte le operazioni eseguite con la release corrente di Astra Trident non vengono riflesse dopo il downgrade. I PVS appena creati non sono utilizzabili quando si torna a una versione precedente. Le modifiche apportate a oggetti come backend, PVS, classi di storage e snapshot di volumi (`create/aggiornate/eliminate`) non sono visibili ad Astra Trident quando si torna a una versione precedente. Il ritorno a una versione precedente non interrompe l'accesso ai PVS già creati utilizzando la versione precedente, a meno che non siano stati aggiornati.

## Processo di downgrade quando Astra Trident viene installato utilizzando l'operatore

Per le installazioni eseguite con Trident Operator, il processo di downgrade è diverso e non richiede l'utilizzo di `tridentctl`.

Per le installazioni eseguite utilizzando l'operatore Trident, Astra Trident può essere retrocesso a una delle

seguenti opzioni:

- Versione installata utilizzando l'operatore namespace-scoped (20.07 - 20.10).
- Versione installata utilizzando l'operatore con ambito cluster (21.01 e versioni successive).

### Eseguire il downgrade all'operatore con ambito cluster

Per eseguire il downgrade di Astra Trident a una release che utilizza l'operatore cluster-scoped, attenersi alla procedura indicata di seguito.

#### Fasi

1. **"Disinstallare Astra Trident". Non eliminare i CRD a meno che non si desideri rimuovere completamente un'installazione esistente.**
2. L'operatore Trident può essere cancellato utilizzando il manifesto dell'operatore associato alla versione di Trident. Ad esempio, <https://github.com/NetApp/trident/tree/stable/vXX.XX/deploy/bundle.yaml> dove *vXX.XX* è il numero di versione (ad esempio *v22.10*) e *bundle.yaml* È il nome del file YAML del bundle.
3. Continuare a eseguire il downgrade installando la versione desiderata di Astra Trident. Seguire la documentazione relativa alla release desiderata.

### Eseguire il downgrade all'operatore con ambito spazio dei nomi

Questa sezione riassume i passaggi necessari per il downgrade a una release Astra Trident compresa tra 20.07 e 20.10, che verrà installata utilizzando l'operatore con ambito namespace.

#### Fasi

1. **"Disinstallare Astra Trident". Non estrarre i CRD a meno che non si desideri rimuovere completamente un'installazione esistente.** assicurarsi di `tridentorchestrator` viene cancellato.

```
#Check to see if there are any tridentorchestrators present
kubectl get torc
NAME          AGE
trident      20h

#Looks like there is a tridentorchestrator that needs deleting
kubectl delete torc trident
tridentorchestrator.trident.netapp.io "trident" deleted
```

2. L'operatore Trident può essere cancellato utilizzando il manifesto dell'operatore associato alla versione di Trident. Ad esempio, <https://github.com/NetApp/trident/tree/stable/vXX.XX/deploy/bundle.yaml> dove *vXX.XX* è il numero di versione (ad esempio *v22.10*) e *bundle.yaml* È il nome del file YAML del bundle.
3. Eliminare `tridentorchestrator` CRD.

```
#Check to see if ``tridentorchestrators.trident.netapp.io`` CRD is present and delete it.
```

```
kubectl get crd tridentorchestrators.trident.netapp.io
```

```
NAME                                CREATED AT
tridentorchestrators.trident.netapp.io 2021-01-21T21:11:37Z
```

```
kubectl delete crd tridentorchestrators.trident.netapp.io
```

```
customresourcedefinition.apiextensions.k8s.io
"tridentorchestrators.trident.netapp.io" deleted
```

Astra Trident è stato disinstallato.

4. Continuare con il downgrade installando la versione desiderata. Seguire la documentazione relativa alla release desiderata.

### Eseguire il downgrade utilizzando Helm

Per eseguire il downgrade, utilizzare `helm rollback trident [revision #]` comando. Vedere il seguente esempio:

```
helm rollback trident [revision #]
```

### Processo di downgrade quando Astra Trident viene installato mediante `tridentctl`

Se Astra Trident è stato installato utilizzando `tridentctl`, il processo di downgrade prevede i seguenti passaggi. Questa sequenza illustra il processo di downgrade per passare da Astra Trident 21.07 a 20.07.



Prima di iniziare il downgrade, è necessario creare un'istantanea del cluster Kubernetes `etcd`. Ciò consente di eseguire il backup dello stato corrente dei CRD di Astra Trident.

### Fasi

1. Assicurarsi che Trident sia installato utilizzando `tridentctl`. Se non si è sicuri di come sia installato Astra Trident, eseguire questo semplice test:
  - a. Elencare i pod presenti nello spazio dei nomi Trident.
  - b. Identificare la versione di Astra Trident in esecuzione nel cluster. È possibile utilizzare entrambi `tridentctl` Oppure guarda l'immagine utilizzata nei pod Trident.
  - c. **Se non viene visualizzato** `A. tridentOrchestrator`, (o) `a. tridentprovisioner`, (o) un pod denominato `trident-operator-xxxxxxxxxx-xxxxx`, Astra Trident **è installato** con `tridentctl`.
2. Disinstallare Astra Trident con l'esistente `tridentctl` binario. In questo caso, verrà disinstallato con il file binario 21.07.

```

tridentctl version -n trident
+-----+-----+
| SERVER VERSION | CLIENT VERSION |
+-----+-----+
| 21.07.0       | 21.07.0       |
+-----+-----+

tridentctl uninstall -n trident
INFO Deleted Trident deployment.
INFO Deleted Trident daemonset.
INFO Deleted Trident service.
INFO Deleted Trident secret.
INFO Deleted Trident cluster role binding.
INFO Deleted Trident cluster role.
INFO Deleted Trident service account.
INFO Deleted Trident pod security policy.
podSecurityPolicy=tridentpods
INFO The uninstaller did not delete Trident's namespace in case it is
going to be reused.
INFO Trident uninstallation succeeded.

```

- Una volta completata questa operazione, ottenere il binario Trident per la versione desiderata (in questo esempio, 20.07) e utilizzarlo per installare Astra Trident. È possibile generare YAML personalizzati per a. ["installazione personalizzata"](#) se necessario.

```

cd 20.07/trident-installer/
./tridentctl install -n trident-ns
INFO Created installer service account.
serviceaccount=trident-installer
INFO Created installer cluster role.                clusterrole=trident-
installer
INFO Created installer cluster role binding.
clusterrolebinding=trident-installer
INFO Created installer configmap.                  configmap=trident-
installer
...
...
INFO Deleted installer cluster role binding.
INFO Deleted installer cluster role.
INFO Deleted installer service account.

```

Il processo di downgrade è completo.

## Informazioni sul copyright

Copyright © 2025 NetApp, Inc. Tutti i diritti riservati. Stampato negli Stati Uniti d'America. Nessuna porzione di questo documento soggetta a copyright può essere riprodotta in qualsiasi formato o mezzo (grafico, elettronico o meccanico, inclusi fotocopie, registrazione, nastri o storage in un sistema elettronico) senza previo consenso scritto da parte del detentore del copyright.

Il software derivato dal materiale sottoposto a copyright di NetApp è soggetto alla seguente licenza e dichiarazione di non responsabilità:

IL PRESENTE SOFTWARE VIENE FORNITO DA NETAPP "COSÌ COM'È" E SENZA QUALSIVOGLIA TIPO DI GARANZIA IMPLICITA O ESPRESSA FRA CUI, A TITOLO ESEMPLIFICATIVO E NON ESAUSTIVO, GARANZIE IMPLICITE DI COMMERCIALIZZABILITÀ E IDONEITÀ PER UNO SCOPO SPECIFICO, CHE VENGONO DECLINATE DAL PRESENTE DOCUMENTO. NETAPP NON VERRÀ CONSIDERATA RESPONSABILE IN ALCUN CASO PER QUALSIVOGLIA DANNO DIRETTO, INDIRETTO, ACCIDENTALE, SPECIALE, ESEMPLARE E CONSEGUENZIALE (COMPRESI, A TITOLO ESEMPLIFICATIVO E NON ESAUSTIVO, PROCUREMENT O SOSTITUZIONE DI MERCI O SERVIZI, IMPOSSIBILITÀ DI UTILIZZO O PERDITA DI DATI O PROFITTI OPPURE INTERRUZIONE DELL'ATTIVITÀ AZIENDALE) CAUSATO IN QUALSIVOGLIA MODO O IN RELAZIONE A QUALUNQUE TEORIA DI RESPONSABILITÀ, SIA ESSA CONTRATTUALE, RIGOROSA O DOVUTA A INSOLVENZA (COMPRESA LA NEGLIGENZA O ALTRO) INSORTA IN QUALSIASI MODO ATTRAVERSO L'UTILIZZO DEL PRESENTE SOFTWARE ANCHE IN PRESENZA DI UN PREAVVISO CIRCA L'EVENTUALITÀ DI QUESTO TIPO DI DANNI.

NetApp si riserva il diritto di modificare in qualsiasi momento qualunque prodotto descritto nel presente documento senza fornire alcun preavviso. NetApp non si assume alcuna responsabilità circa l'utilizzo dei prodotti o materiali descritti nel presente documento, con l'eccezione di quanto concordato espressamente e per iscritto da NetApp. L'utilizzo o l'acquisto del presente prodotto non comporta il rilascio di una licenza nell'ambito di un qualche diritto di brevetto, marchio commerciale o altro diritto di proprietà intellettuale di NetApp.

Il prodotto descritto in questa guida può essere protetto da uno o più brevetti degli Stati Uniti, esteri o in attesa di approvazione.

LEGENDA PER I DIRITTI SOTTOPOSTI A LIMITAZIONE: l'utilizzo, la duplicazione o la divulgazione da parte degli enti governativi sono soggetti alle limitazioni indicate nel sottoparagrafo (b)(3) della clausola Rights in Technical Data and Computer Software del DFARS 252.227-7013 (FEB 2014) e FAR 52.227-19 (DIC 2007).

I dati contenuti nel presente documento riguardano un articolo commerciale (secondo la definizione data in FAR 2.101) e sono di proprietà di NetApp, Inc. Tutti i dati tecnici e il software NetApp forniti secondo i termini del presente Contratto sono articoli aventi natura commerciale, sviluppati con finanziamenti esclusivamente privati. Il governo statunitense ha una licenza irrevocabile limitata, non esclusiva, non trasferibile, non cedibile, mondiale, per l'utilizzo dei Dati esclusivamente in connessione con e a supporto di un contratto governativo statunitense in base al quale i Dati sono distribuiti. Con la sola esclusione di quanto indicato nel presente documento, i Dati non possono essere utilizzati, divulgati, riprodotti, modificati, visualizzati o mostrati senza la previa approvazione scritta di NetApp, Inc. I diritti di licenza del governo degli Stati Uniti per il Dipartimento della Difesa sono limitati ai diritti identificati nella clausola DFARS 252.227-7015(b) (FEB 2014).

## Informazioni sul marchio commerciale

NETAPP, il logo NETAPP e i marchi elencati alla pagina <http://www.netapp.com/TM> sono marchi di NetApp, Inc. Gli altri nomi di aziende e prodotti potrebbero essere marchi dei rispettivi proprietari.