Riferimento

Astra Trident

NetApp
August 18, 2026

This PDF was generated from https://docs.netapp.com/it-it/trident-2304/trident-reference/ports.html on August 18, 2026. Always check docs.netapp.com for the latest.

Sommario

Riferimento . . . 1
- Porte Astra Trident . . . 1
  - Porte Astra Trident . . . 1
- API REST di Astra Trident . . . 1
  - Quando utilizzare l'API REST . . . 1
  - Utilizzo dell'API REST . . . 1
- Opzioni della riga di comando . . . 2
  - Registrazione . . . 2
  - Kubernetes . . . 2
  - Docker . . . 2
  - RIPOSO . . . 3
- Prodotti NetApp integrati con Kubernetes . . . 3
  - Astra . . . 3
  - ONTAP . . . 3
  - Cloud Volumes ONTAP . . . 3
  - Amazon FSX per NetApp ONTAP . . . 4
  - Software Element . . . 4
  - NetApp HCI . . . 4
  - Azure NetApp Files . . . 4
  - Cloud Volumes Service per Google Cloud . . . 4
- Kubernetes e Trident Objects . . . 4
  - In che modo gli oggetti interagiscono tra loro? . . . 4
  - Kubernetes `PersistentVolumeClaim` oggetti . . . 5
  - Kubernetes `PersistentVolume` oggetti . . . 7
  - Kubernetes `StorageClass` oggetti . . . 7
  - Kubernetes `VolumeSnapshotClass` oggetti . . . 11
  - Kubernetes `VolumeSnapshot` oggetti . . . 11
  - Kubernetes `VolumeSnapshotContent` oggetti . . . 12
  - Kubernetes `CustomResourceDefinition` oggetti . . . 12
  - Trident `StorageClass` oggetti . . . 13
  - Oggetti backend Trident . . . 13
  - Trident `StoragePool` oggetti . . . 13
  - Trident `Volume` oggetti . . . 13
  - Trident `Snapshot` oggetti . . . 15
  - Astra Trident `ResourceQuota` oggetto . . . 16
- comandi e opzioni tridentctl . . . 17
  - Comandi e opzioni disponibili . . . 17
  - `create` . . . 18
  - `delete` . . . 18
  - `get` . . . 18
  - `images` . . . 19

`import volume` 19
`install` 19
`logs` 20
`send` 21
`uninstall` 21
`update` 21
`upgrade` 21
`version` 22
Pod Security Standards (PSS) e Security Context Constraints (SCC) 22
Contesto di sicurezza Kubernetes obbligatorio e campi correlati 22
Standard di sicurezza Pod (PSS) 23
Policy di sicurezza Pod (PSP) 23
SCC (Security Context Constraints) 25

Riferimento

Porte Astra Trident

Scopri di più sulle porte utilizzate da Astra Trident per le comunicazioni.

Porte Astra Trident

Astra Trident comunica tramite le seguenti porte:

Porta	Scopo
8443	HTTPS backchannel
8001	Endpoint delle metriche Prometheus
8000	Server REST Trident
17546	Porta della sonda liveness/readiness utilizzata dai pod demonset di Trident

La porta della sonda liveness/Readiness può essere modificata durante l'installazione utilizzando `--probe-port` allarme. È importante assicurarsi che questa porta non venga utilizzata da un altro processo sui nodi di lavoro.

API REST di Astra Trident

Mentre "comandi e opzioni tridentctl" Sono il modo più semplice per interagire con l'API REST di Astra Trident, puoi utilizzare l'endpoint REST direttamente se preferisci.

Quando utilizzare l'API REST

REST API è utile per le installazioni avanzate che utilizzano Astra Trident come binario standalone nelle implementazioni non Kubernetes.

Per una maggiore sicurezza, Astra Trident `REST API` è limitato all'host locale per impostazione predefinita quando viene eseguito all'interno di un pod. Per modificare questo comportamento, devi impostare Astra Trident's. `-address` argomento nella configurazione del pod.

Utilizzo dell'API REST

L'API funziona come segue:

`GET`

- `GET <trident-address>/trident/v1/<object-type>`: Elenca tutti gli oggetti di quel tipo.
- `GET <trident-address>/trident/v1/<object-type>/<object-name>`: Ottiene i dettagli dell'oggetto denominato.

`POST`

`POST <trident-address>/trident/v1/<object-type>`: Crea un oggetto del tipo specificato.

- Richiede una configurazione JSON per la creazione dell'oggetto. Per la specifica di ciascun tipo di oggetto, vedere il `tridentctl` comandi e opzioni.
- Se l'oggetto esiste già, il comportamento varia: I backend aggiornano l'oggetto esistente, mentre tutti gli altri tipi di oggetto non riescono a eseguire l'operazione.

`DELETE`

`DELETE <trident-address>/trident/v1/<object-type>/<object-name>`: Elimina la risorsa indicata.

> I volumi associati ai backend o alle classi di storage continueranno ad esistere; questi devono essere cancellati separatamente. Per ulteriori informazioni, consulta il `tridentctl` comandi e opzioni.

Per esempi di come vengono chiamate queste API, passare il debug (`-d`). Per ulteriori informazioni, consulta il `tridentctl` comandi e opzioni.

Opzioni della riga di comando

Astra Trident espone diverse opzioni della riga di comando per Trident orchestrator. È possibile utilizzare queste opzioni per modificare la distribuzione.

Registrazione

- `-debug`: Attiva l'output di debug.
- `-loglevel <level>`: Consente di impostare il livello di registrazione (debug, info, warn, error, fatale). Il valore predefinito è INFO.

Kubernetes

- `-k8s_pod`: Utilizzare questa opzione o. `-k8s_api_server` Per abilitare il supporto Kubernetes. Questa impostazione fa in modo che Trident utilizzi le credenziali dell'account del servizio Kubernetes del pod che lo contiene per contattare il server API. Questo funziona solo quando Trident viene eseguito come pod in un cluster Kubernetes con account di servizio abilitati.
- `-k8s_api_server <insecure-address:insecure-port>`: Utilizzare questa opzione o. `-k8s_pod` Per abilitare il supporto Kubernetes. Quando specificato, Trident si connette al server API Kubernetes utilizzando l'indirizzo e la porta non sicuri forniti. Ciò consente a Trident di essere implementato all'esterno di un pod; tuttavia, supporta solo connessioni non sicure al server API. Per una connessione sicura, implementare Trident in un pod con `-k8s_pod` opzione.
- `-k8s_config_path <file>`Obbligatorio; specificare questo percorso per un file KubeConfig.

Docker

- `-volume_driver <name>`: Nome del driver utilizzato durante la registrazione del plugin Docker. L'impostazione predefinita è `netapp`.

- `-driver_port <port-number>`: Ascoltare su questa porta piuttosto che su un socket di dominio UNIX.
- `-config <file>`Obbligatorio; specificare questo percorso per un file di configurazione back-end.

RIPOSO

- `-address <ip-or-host>`: Specifica l'indirizzo su cui il server REST di Trident deve essere in ascolto. L'impostazione predefinita è localhost. Quando si ascolta su localhost e si esegue all'interno di un pod Kubernetes, l'interfaccia REST non è direttamente accessibile dall'esterno del pod. Utilizzare `-address ""` Per rendere l'interfaccia REST accessibile dall'indirizzo IP del pod.

> L'interfaccia REST di Trident può essere configurata per l'ascolto e la distribuzione solo su 127.0.0.1 (per IPv4) o [::1] (per IPv6).

- `-port <port-number>`: Specifica la porta su cui il server REST di Trident deve essere in ascolto. Il valore predefinito è 8000.
- `-rest`: Attiva l'interfaccia REST. L'impostazione predefinita è true.

Prodotti NetApp integrati con Kubernetes

Il portfolio di prodotti storage NetApp si integra con molti aspetti diversi di un cluster Kubernetes, fornendo funzionalità avanzate di gestione dei dati che migliorano funzionalità, funzionalità, performance e disponibilità dell'implementazione di Kubernetes.

Astra

"Astra" Semplifica la gestione, la protezione e lo spostamento dei carichi di lavoro containerizzati ricchi di dati eseguiti su Kubernetes all'interno e tra cloud pubblici e on-premise. Astra fornisce e fornisce storage container persistente utilizzando Trident del comprovato e esteso portfolio di storage NetApp nel cloud pubblico e on-premise. Offre inoltre una serie completa di funzionalità avanzate di gestione dei dati applicative, come snapshot, backup e ripristino, log di attività e cloning attivo per la protezione dei dati, disaster recovery/data, audit dei dati e casi di utilizzo della migrazione per i carichi di lavoro Kubernetes.

ONTAP

ONTAP è il sistema operativo per lo storage unificato multiprotocollo di NetApp che offre funzionalità avanzate di gestione dei dati per qualsiasi applicazione. I sistemi ONTAP sono dotati di configurazioni all-flash, ibride o all-HDD e offrono diversi modelli di implementazione, tra cui hardware progettato (FAS e AFF), white-box (ONTAP Select) e solo cloud (Cloud Volumes ONTAP).

> Trident supporta tutti i modelli di implementazione ONTAP sopra menzionati.

Cloud Volumes ONTAP

Cloud Volumes ONTAP è un'appliance di storage esclusivamente software che esegue il software di gestione ONTAP nel cloud. Puoi usare Cloud Volumes ONTAP per carichi di lavoro di produzione, ripristino di emergenza, DevOps, file share e gestione del database. Estende lo storage enterprise al cloud offrendo efficienza di storage, alta disponibilità, replica dei dati, tiering dei dati e coerenza delle applicazioni.

Amazon FSX per NetApp ONTAP

"Amazon FSX per NetApp ONTAP" È un servizio AWS completamente gestito che consente ai clienti di lanciare ed eseguire file system basati sul sistema operativo per lo storage ONTAP di NetApp. FSX per ONTAP consente ai clienti di sfruttare le funzionalità, le performance e le funzionalità amministrative di NetApp con cui hanno familiarità, sfruttando al contempo la semplicità, l'agilità, la sicurezza e la scalabilità dell'archiviazione dei dati su AWS. FSX per ONTAP supporta molte delle funzionalità del file system e delle API di amministrazione di ONTAP.

Software Element

"Elemento" consente all'amministratore dello storage di consolidare i carichi di lavoro garantendo le performance e consentendo un footprint dello storage semplificato e ottimizzato. Insieme a un'API per consentire l'automazione di tutti gli aspetti della gestione dello storage, Element consente agli amministratori dello storage di fare di più con meno sforzo.

NetApp HCI

"NetApp HCI" semplifica la gestione e la scalabilità del data center automatizzando le attività di routine e consentendo agli amministratori dell'infrastruttura di concentrarsi su funzioni più importanti.

NetApp HCI è completamente supportato da Trident. Trident è in grado di eseguire il provisioning e la gestione dei dispositivi di storage per le applicazioni containerizzate direttamente sulla piattaforma di storage NetApp HCI sottostante.

Azure NetApp Files

"Azure NetApp Files" è un servizio di condivisione file Azure enterprise, alimentato da NetApp. Puoi eseguire i tuoi carichi di lavoro basati su file più impegnativi in Azure in modo nativo, con la performance e la ricca gestione dei dati che ti aspetti da NetApp.

Cloud Volumes Service per Google Cloud

"NetApp Cloud Volumes Service per Google Cloud" è un file service cloud native che fornisce volumi NAS su NFS e SMB con performance all-flash. Questo servizio ti permette di eseguire qualsiasi workload, incluse le applicazioni legacy, nel cloud GCP. Fornisce un servizio gestito che offre performance elevate e costanti, clonazione istantanea, protezione dei dati e accesso sicuro alle istanze di Google Compute Engine (GCE).

Kubernetes e Trident Objects

È possibile interagire con Kubernetes e Trident utilizzando API REST leggendo e scrivendo oggetti di risorse. Esistono diversi oggetti di risorse che determinano la relazione tra Kubernetes e Trident, Trident e storage, Kubernetes e storage. Alcuni di questi oggetti vengono gestiti tramite Kubernetes, mentre altri vengono gestiti tramite Trident.

In che modo gli oggetti interagiscono tra loro?

Forse il modo più semplice per comprendere gli oggetti, il loro scopo e il modo in cui interagiscono è seguire una singola richiesta di storage da parte di un utente Kubernetes:

1. Un utente crea un `PersistentVolumeClaim` richiesta di un nuovo `PersistentVolume` Di una dimensione particolare da un Kubernetes `StorageClass` precedentemente configurato dall'amministratore.
2. Kubernetes `StorageClass` Identifica Trident come provider e include parametri che indicano a Trident come eseguire il provisioning di un volume per la classe richiesta.
3. Trident si guarda da solo `StorageClass` con lo stesso nome che identifica la corrispondenza `Backends` e. `StoragePools` che può utilizzare per eseguire il provisioning dei volumi per la classe.
4. Trident esegue il provisioning dello storage su un backend corrispondente e crea due oggetti: A. `PersistentVolume` In Kubernetes che indica a Kubernetes come trovare, montare e trattare il volume e un volume in Trident che mantiene la relazione tra `PersistentVolume` e lo storage effettivo.
5. Kubernetes lega il `PersistentVolumeClaim` al nuovo `PersistentVolume`. Pod che includono `PersistentVolumeClaim` Montare il PersistentVolume su qualsiasi host su cui viene eseguito.
6. Un utente crea un `VolumeSnapshot` Di un PVC esistente, utilizzando un `VolumeSnapshotClass` Questo indica Trident.
7. Trident identifica il volume associato al PVC e crea un'istantanea del volume sul backend. Inoltre, crea un `VolumeSnapshotContent` Che indica a Kubernetes come identificare lo snapshot.
8. Un utente può creare un `PersistentVolumeClaim` utilizzo di `VolumeSnapshot` come fonte.
9. Trident identifica lo snapshot richiesto ed esegue la stessa serie di passaggi necessari per la creazione di `PersistentVolume` e a. `Volume`.

Per ulteriori informazioni sugli oggetti Kubernetes, si consiglia di leggere il "Volumi persistenti" Della documentazione Kubernetes.

Kubernetes `PersistentVolumeClaim` oggetti

Un Kubernetes `PersistentVolumeClaim` Object è una richiesta di storage effettuata da un utente del cluster Kubernetes.

Oltre alla specifica standard, Trident consente agli utenti di specificare le seguenti annotazioni specifiche del volume se desiderano sovrascrivere i valori predefiniti impostati nella configurazione di backend:

Annotazione	Opzione volume	Driver supportati
trident.netapp.io/fileSystem	Filesystem	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	protocollo	qualsiasi
trident.netapp.io/exportPolicy	ExportPolicy	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup

Annotazione	Opzione volume	Driver supportati
trident.netapp.io/snapshotPolicy	SnapshotPolicy	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san
trident.netapp.io/snapshotReserve	SnapshotReserve	ontap-nas, ontap-nas-flexgroup, ontap-san, gcp-cvs
trident.netapp.io/snapshotDirectory	SnapshotDirectory	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup
trident.netapp.io/unixPermissions	UnixPermissions	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup
trident.netapp.io/blockSize	Dimensione blocco	solidfire-san

Se il PV creato dispone di `Delete` Recuperare la policy, Trident elimina sia il PV che il volume di backup quando il PV viene rilasciato (ovvero quando l'utente elimina il PVC). In caso di errore dell'azione di eliminazione, Trident contrassegna il PV come tale e riprova periodicamente l'operazione fino a quando non viene eseguita correttamente o finché il PV non viene cancellato manualmente. Se il PV utilizza `Retain` Policy, Trident lo ignora e presuppone che l'amministratore lo pulisca da Kubernetes e dal backend, consentendo il backup o l'ispezione del volume prima della sua rimozione. L'eliminazione del PV non comporta l'eliminazione del volume di backup da parte di Trident. È necessario rimuoverlo utilizzando l'API REST (`tridentctl`).

Trident supporta la creazione di snapshot dei volumi utilizzando la specifica CSI: È possibile creare un'istantanea del volume e utilizzarla come origine dati per clonare i PVC esistenti. In questo modo, le copie point-in-time di PVS possono essere esposte a Kubernetes sotto forma di snapshot. Le istantanee possono quindi essere utilizzate per creare un nuovo PVS. Dai un'occhiata a. `On-Demand Volume Snapshots` per vedere come funziona.

Trident fornisce anche `cloneFromPVC` e. `splitOnClone` annotazioni per la creazione di cloni. È possibile utilizzare queste annotazioni per clonare un PVC senza utilizzare l'implementazione CSI (su Kubernetes 1.13 e versioni precedenti) o se la release di Kubernetes non supporta le snapshot dei volumi beta (Kubernetes 1.16 e versioni precedenti). Tenere presente che Trident 19.10 supporta il workflow CSI per la clonazione da PVC.

È possibile utilizzare `cloneFromPVC` e. `splitOnClone` Annotazioni con CSI Trident e con il frontend tradizionale non CSI.

Ecco un esempio: Se un utente ha già un PVC chiamato `mysql`, L'utente può creare un nuovo PVC chiamato `mysqlclone` utilizzando l'annotazione, ad esempio `trident.netapp.io/cloneFromPVC: mysql`. Con questo set di annotazioni, Trident clona il volume corrispondente al PVC mysql, invece di eseguire il provisioning di un volume da zero.

Considerare i seguenti punti:

- Si consiglia di clonare un volume inattivo.
- Un PVC e il relativo clone devono trovarsi nello stesso spazio dei nomi Kubernetes e avere la stessa classe di storage.

- Con `ontap-nas` e. `ontap-san` Driver, potrebbe essere consigliabile impostare l'annotazione PVC `trident.netapp.io/splitOnClone` in combinazione con `trident.netapp.io/cloneFromPVC`. Con `trident.netapp.io/splitOnClone` impostare su `true`, Trident suddivide il volume clonato dal volume padre e, di conseguenza, disaccadeva completamente il ciclo di vita del volume clonato dal volume padre a scapito di una certa efficienza dello storage. Non impostato `trident.netapp.io/splitOnClone` o impostarlo su `false` si ottiene un consumo di spazio ridotto sul backend a scapito della creazione di dipendenze tra i volumi padre e clone, in modo che il volume padre non possa essere cancellato a meno che il clone non venga cancellato per primo. Uno scenario in cui la suddivisione del clone ha senso è la clonazione di un volume di database vuoto in cui si prevede che il volume e il relativo clone divergano notevolmente e non traggano beneficio dall'efficienza dello storage offerta da ONTAP.

Il `sample-input` La directory contiene esempi di definizioni PVC da utilizzare con Trident. Vedere oggetti Trident Volume per una descrizione completa dei parametri e delle impostazioni associate ai volumi Trident.

Kubernetes `PersistentVolume` oggetti

Un Kubernetes `PersistentVolume` Object rappresenta un elemento di storage che viene reso disponibile per il cluster Kubernetes. Ha un ciclo di vita indipendente dal pod che lo utilizza.

Trident crea `PersistentVolume` E li registra automaticamente con il cluster Kubernetes in base ai volumi forniti. Non ci si aspetta di gestirli da soli.

Quando si crea un PVC che si riferisce a un Trident-based `StorageClass`, Trident esegue il provisioning di un nuovo volume utilizzando la classe di storage corrispondente e registra un nuovo PV per quel volume. Nella configurazione del volume sottoposto a provisioning e del PV corrispondente, Trident segue le seguenti regole:

- Trident genera un nome PV per Kubernetes e un nome interno utilizzato per il provisioning dello storage. In entrambi i casi, garantisce che i nomi siano univoci nel loro scopo.
- La dimensione del volume corrisponde alla dimensione richiesta nel PVC il più possibile, anche se potrebbe essere arrotondata alla quantità allocabile più vicina, a seconda della piattaforma.

Kubernetes `StorageClass` oggetti

Kubernetes `StorageClass` gli oggetti sono specificati in base al nome `PersistentVolumeClaims` per eseguire il provisioning dello storage con un set di proprietà. La stessa classe di storage identifica il provider da utilizzare e definisce il set di proprietà in termini che il provider riconosce.

Si tratta di uno dei due oggetti di base che devono essere creati e gestiti dall'amministratore. L'altro è l'oggetto backend Trident.

Un Kubernetes `StorageClass` L'oggetto che utilizza Trident è simile al seguente:

```
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
```

Questi parametri sono specifici di Trident e indicano a Trident come eseguire il provisioning dei volumi per la classe.

I parametri della classe di storage sono:

Attributo	Tipo	Obbligatorio	Descrizione
attributi	map[string]string	no	Vedere la sezione attributi riportata di seguito
StoragePools	map[string]StringList	no	Mappatura dei nomi backend negli elenchi di pool di storage all'interno di
AddtionalStoragePools	map[string]StringList	no	Mappa dei nomi backend a elenchi di pool di storage all'interno di
EsclusiveStoragePools	map[string]StringList	no	Mappa dei nomi backend in elenchi di pool di storage all'interno di

Gli attributi di storage e i loro possibili valori possono essere classificati in attributi di selezione del pool di storage e attributi Kubernetes.

Attributi di selezione del pool di storage

Questi parametri determinano quali pool di storage gestiti da Trident devono essere utilizzati per eseguire il provisioning di volumi di un determinato tipo.

Attributo	Tipo	Valori	Offerta	Richiesta	Supportato da
supporti[1]	stringa	hdd, ibrido, ssd	Il pool contiene supporti di questo tipo; ibridi significa entrambi	Tipo di supporto specificato	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san

Attributo	Tipo	Valori	Offerta	Richiesta	Supportato da
ProvisioningType	stringa	sottile, spesso	Il pool supporta questo metodo di provisioning	Metodo di provisioning specificato	thick: all ONTAP; thin: all ONTAP e solidfire-san
BackendType	stringa	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san, gcp-cvs, azure-netapp-files, ontap-san-economy	Il pool appartiene a questo tipo di backend	Backend specificato	Tutti i driver
snapshot	bool	vero, falso	Il pool supporta volumi con snapshot	Volume con snapshot attivate	ontap-nas, ontap-san, solidfire-san, gcp-cvs
cloni	bool	vero, falso	Il pool supporta la clonazione dei volumi	Volume con cloni attivati	ontap-nas, ontap-san, solidfire-san, gcp-cvs
crittografia	bool	vero, falso	Il pool supporta volumi crittografati	Volume con crittografia attivata	ontap-nas, ontap-nas-economy, ontap-nas-flexgroups, ontap-san
IOPS	int	intero positivo	Il pool è in grado di garantire IOPS in questa gamma	Volume garantito per questi IOPS	solidfire-san

[1]: Non supportato dai sistemi ONTAP Select

Nella maggior parte dei casi, i valori richiesti influiscono direttamente sul provisioning; ad esempio, la richiesta di thick provisioning comporta un volume con provisioning spesso. Tuttavia, un pool di storage di elementi utilizza i valori IOPS minimi e massimi offerti per impostare i valori QoS, piuttosto che il valore richiesto. In questo caso, il valore richiesto viene utilizzato solo per selezionare il pool di storage.

Idealmente, è possibile utilizzare `attributes` da soli per modellare le qualità dello storage necessarie per soddisfare le esigenze di una particolare classe. Trident rileva e seleziona automaticamente i pool di storage che corrispondono a *tutti* di `attributes` specificato dall'utente.

Se non si riesce a utilizzare `attributes` per selezionare automaticamente i pool giusti per una classe, è possibile utilizzare `storagePools` e. `additionalStoragePools` parametri per perfezionare ulteriormente i pool o anche per selezionare un set specifico di pool.

È possibile utilizzare `storagePools` parametro per limitare ulteriormente il set di pool che corrispondono a qualsiasi specificato `attributes`. In altre parole, Trident utilizza l'intersezione di pool identificati da `attributes` e. `storagePools` parametri per il provisioning. È possibile utilizzare uno dei due parametri da

solo o entrambi insieme.

È possibile utilizzare `additionalStoragePools` Parametro per estendere l'insieme di pool che Trident utilizza per il provisioning, indipendentemente dai pool selezionati da `attributes` e. `storagePools` parametri.

È possibile utilizzare `excludeStoragePools` Parametro per filtrare il set di pool che Trident utilizza per il provisioning. L'utilizzo di questo parametro consente di rimuovere i pool corrispondenti.

In `storagePools` e. `additionalStoragePools` parametri, ogni voce assume la forma `<backend>:<storagePoolList>`, dove `<storagePoolList>` è un elenco separato da virgole di pool di storage per il backend specificato. Ad esempio, un valore per `additionalStoragePools` potrebbe sembrare `ontapnas_192.168.1.100:aggr1,aggr2;solidfire_192.168.1.101:bronze`. Questi elenchi accettano valori regex sia per i valori di backend che per quelli di elenco. È possibile utilizzare `tridentctl get backend` per ottenere l'elenco dei backend e dei relativi pool.

Attributi Kubernetes

Questi attributi non hanno alcun impatto sulla selezione dei pool/backend di storage da parte di Trident durante il provisioning dinamico. Invece, questi attributi forniscono semplicemente parametri supportati dai volumi persistenti Kubernetes. I nodi di lavoro sono responsabili delle operazioni di creazione del file system e potrebbero richiedere utility del file system, come xfsprogs.

Attributo	Tipo	Valori	Descrizione	Driver pertinenti	Kubernetes Versione
Fstype	stringa	ext4, ext3, xfs, ecc.	Il tipo di file system per il blocco volumi	solidfire-san, ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economy	Tutto
AllowVolumeExpansion	booleano	vero, falso	Abilitare o disabilitare il supporto per aumentare le dimensioni del PVC	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economy, solidfire-san, gcp-cvs, azure-netapp-files	1.11+
VolumeBindingMode	stringa	Immediato, WaitForFirstConsumer	Scegliere quando si verifica il binding del volume e il provisioning dinamico	Tutto	1.19 - 1.26

- Il `fsType` Il parametro viene utilizzato per controllare il tipo di file system desiderato per LE LUN SAN. Inoltre, Kubernetes utilizza anche la presenza di `fsType` in una classe di storage per indicare l'esistenza di un file system. La proprietà del volume può essere controllata tramite `fsGroup` contesto di sicurezza di un pod solo se `fsType` è impostato. Vedere "Kubernetes: Consente di configurare un contesto di protezione per un Pod o un container" per una panoramica sull'impostazione della proprietà del volume mediante `fsGroup` contesto. Kubernetes applicherà il `fsGroup` valore solo se:

  - `fsType` viene impostato nella classe di storage.
  - La modalità di accesso PVC è RWO.

  Per i driver di storage NFS, esiste già un filesystem come parte dell'esportazione NFS. Per l'utilizzo `fsGroup` la classe di storage deve ancora specificare un `fsType`. È possibile impostarlo su `nfs` o qualsiasi valore non nullo.

- Vedere "Espandere i volumi" per ulteriori dettagli sull'espansione dei volumi.
- Il bundle del programma di installazione Trident fornisce diverse definizioni di classi di storage di esempio da utilizzare con Trident in `sample-input/storage-class-*.yaml`. L'eliminazione di una classe di storage Kubernetes comporta l'eliminazione anche della classe di storage Trident corrispondente.

Kubernetes `VolumeSnapshotClass` oggetti

Kubernetes `VolumeSnapshotClass` gli oggetti sono analoghi a. `StorageClasses`. Consentono di definire più classi di storage e vengono utilizzate dagli snapshot dei volumi per associare lo snapshot alla classe di snapshot richiesta. Ogni snapshot di volume è associato a una singola classe di snapshot di volume.

R `VolumeSnapshotClass` deve essere definito da un amministratore per creare snapshot. Viene creata una classe di snapshot del volume con la seguente definizione:

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: csi-snapclass
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

Il `driver` Specifica a Kubernetes che richiede snapshot di volume di `csi-snapclass` Le classi sono gestite da Trident. Il `deletionPolicy` specifica l'azione da eseguire quando è necessario eliminare uno snapshot. Quando `deletionPolicy` è impostato su `Delete`, gli oggetti snapshot del volume e lo snapshot sottostante nel cluster di storage vengono rimossi quando viene eliminata una snapshot. In alternativa, impostarla su `Retain` significa che `VolumeSnapshotContent` e lo snapshot fisico viene conservato.

Kubernetes `VolumeSnapshot` oggetti

Un Kubernetes `VolumeSnapshot` object è una richiesta per creare uno snapshot di un volume. Proprio come un PVC rappresenta una richiesta fatta da un utente per un volume, uno snapshot di volume è una richiesta fatta da un utente per creare uno snapshot di un PVC esistente.

Quando arriva una richiesta di snapshot di un volume, Trident gestisce automaticamente la creazione dello snapshot per il volume sul back-end ed espone lo snapshot creando un unico `VolumeSnapshotContent` oggetto. È possibile creare snapshot da PVC esistenti e utilizzarle come DataSource durante la creazione di nuovi PVC.

> Il ciclo di vita di una VolumeSnapshot è indipendente dal PVC di origine: Una snapshot persiste anche dopo la cancellazione del PVC di origine. Quando si elimina un PVC con snapshot associate, Trident contrassegna il volume di backup per questo PVC in uno stato di **eliminazione**, ma non lo rimuove completamente. Il volume viene rimosso quando vengono eliminate tutte le snapshot associate.

Kubernetes `VolumeSnapshotContent` oggetti

Un Kubernetes `VolumeSnapshotContent` object rappresenta uno snapshot preso da un volume già sottoposto a provisioning. È analogo a a. `PersistentVolume` e indica uno snapshot con provisioning sul cluster di storage. Simile a. `PersistentVolumeClaim` e. `PersistentVolume` oggetti, quando viene creata una snapshot, il `VolumeSnapshotContent` l'oggetto mantiene un mapping uno a uno a `VolumeSnapshot` oggetto, che aveva richiesto la creazione dello snapshot.

> Trident crea `VolumeSnapshotContent` E li registra automaticamente con il cluster Kubernetes in base ai volumi forniti. Non ci si aspetta di gestirli da soli.

Il `VolumeSnapshotContent` oggetto contiene dettagli che identificano in modo univoco lo snapshot, ad esempio `snapshotHandle`. Questo `snapshotHandle` È una combinazione univoca del nome del PV e del nome del `VolumeSnapshotContent` oggetto.

Quando arriva una richiesta di snapshot, Trident crea lo snapshot sul back-end. Una volta creata la snapshot, Trident configura una `VolumeSnapshotContent` E quindi espone lo snapshot all'API Kubernetes.

Kubernetes `CustomResourceDefinition` oggetti

Kubernetes Custom Resources sono endpoint dell'API Kubernetes definiti dall'amministratore e utilizzati per raggruppare oggetti simili. Kubernetes supporta la creazione di risorse personalizzate per l'archiviazione di un insieme di oggetti. È possibile ottenere queste definizioni delle risorse eseguendo `kubectl get crds`.

Le definizioni delle risorse personalizzate (CRD) e i relativi metadati degli oggetti associati vengono memorizzati da Kubernetes nel relativo archivio di metadati. Ciò elimina la necessità di un punto vendita separato per Trident.

A partire dalla versione 19.07, Trident utilizza una serie di `CustomResourceDefinition` Oggetti per preservare l'identità degli oggetti Trident, come backend Trident, classi di storage Trident e volumi Trident. Questi oggetti sono gestiti da Trident. Inoltre, il framework di snapshot dei volumi CSI introduce alcuni CRD necessari per definire le snapshot dei volumi.

I CRD sono un costrutto Kubernetes. Gli oggetti delle risorse sopra definite vengono creati da Trident. Come semplice esempio, quando viene creato un backend utilizzando `tridentctl`, un corrispondente `tridentbackends` L'oggetto CRD viene creato per l'utilizzo da parte di Kubernetes.

Ecco alcuni punti da tenere a mente sui CRD di Trident:

- Una volta installato Trident, viene creato un set di CRD che possono essere utilizzati come qualsiasi altro tipo di risorsa.

- Quando si esegue l'aggiornamento da una versione precedente di Trident (quella utilizzata `etcd` Per mantenere lo stato), il programma di installazione di Trident esegue la migrazione dei dati da `etcd` Archiviazione dei dati Key-Value e creazione degli oggetti CRD corrispondenti.
- Quando si disinstalla Trident utilizzando `tridentctl uninstall` Comando, i pod Trident vengono cancellati ma i CRD creati non vengono ripuliti. Vedere "Disinstallare Trident" Per capire come Trident può essere completamente rimosso e riconfigurato da zero.

Trident `StorageClass` oggetti

Trident crea classi di storage corrispondenti per Kubernetes `StorageClass` oggetti che specificano `csi.trident.netapp.io/netapp.io/trident` nel campo dei provider. Il nome della classe di storage corrisponde a quello di Kubernetes `StorageClass` oggetto che rappresenta.

> Con Kubernetes, questi oggetti vengono creati automaticamente quando un Kubernetes `StorageClass` Che utilizza Trident come provisioner è registrato.

Le classi di storage comprendono un insieme di requisiti per i volumi. Trident abbina questi requisiti agli attributi presenti in ciascun pool di storage; se corrispondono, tale pool di storage è una destinazione valida per il provisioning dei volumi che utilizzano tale classe di storage.

È possibile creare configurazioni delle classi di storage per definire direttamente le classi di storage utilizzando l'API REST. Tuttavia, per le implementazioni di Kubernetes, ci aspettiamo che vengano create al momento della registrazione dei nuovi Kubernetes `StorageClass` oggetti.

Oggetti backend Trident

I backend rappresentano i provider di storage in cima ai quali Trident esegue il provisioning dei volumi; una singola istanza Trident può gestire qualsiasi numero di backend.

> Si tratta di uno dei due tipi di oggetti creati e gestiti dall'utente. L'altro è Kubernetes `StorageClass` oggetto.

Per ulteriori informazioni sulla creazione di questi oggetti, vedere "configurazione dei backend".

Trident `StoragePool` oggetti

I pool di storage rappresentano le diverse posizioni disponibili per il provisioning su ciascun backend. Per ONTAP, questi corrispondono agli aggregati nelle SVM. Per NetApp HCI/SolidFire, queste corrispondono alle bande QoS specificate dall'amministratore. Per Cloud Volumes Service, questi corrispondono alle regioni dei provider di cloud. Ogni pool di storage dispone di un insieme di attributi di storage distinti, che definiscono le caratteristiche di performance e di protezione dei dati.

A differenza degli altri oggetti qui presenti, i candidati del pool di storage vengono sempre rilevati e gestiti automaticamente.

Trident `Volume` oggetti

I volumi sono l'unità di provisioning di base, che comprende endpoint back-end, come condivisioni NFS e LUN iSCSI. In Kubernetes, questi corrispondono direttamente a. `PersistentVolumes`. Quando si crea un volume, assicurarsi che disponga di una classe di storage, che determini la destinazione del provisioning di quel volume, insieme a una dimensione.

In Kubernetes, questi oggetti vengono gestiti automaticamente. È possibile visualizzarli per visualizzare il provisioning di Trident.

Quando si elimina un PV con snapshot associati, il volume Trident corrispondente viene aggiornato allo stato **Deleting**. Per eliminare il volume Trident, è necessario rimuovere le snapshot del volume.

Una configurazione del volume definisce le proprietà che un volume sottoposto a provisioning deve avere.

Attributo	Tipo	Obbligatorio	Descrizione
versione	stringa	no	Versione dell'API Trident ("1")
nome	stringa	sì	Nome del volume da creare
StorageClass	stringa	sì	Classe di storage da utilizzare durante il provisioning del volume
dimensione	stringa	sì	Dimensione del volume per il provisioning in byte
protocollo	stringa	no	Tipo di protocollo da utilizzare; "file" o "blocco"
InternalName (Nome interno)	stringa	no	Nome dell'oggetto sul sistema di storage; generato da Trident
CloneSourceVolume	stringa	no	ONTAP (nas, san) e SolidFire-*: Nome del volume da cui clonare
SplitOnClone	stringa	no	ONTAP (nas, san): Suddividere il clone dal suo padre
SnapshotPolicy	stringa	no	ONTAP-*: Policy di snapshot da utilizzare
SnapshotReserve	stringa	no	ONTAP-*: Percentuale di volume riservato agli snapshot
ExportPolicy	stringa	no	ontap-nas*: Policy di esportazione da utilizzare
SnapshotDirectory	bool	no	ontap-nas*: Indica se la directory di snapshot è visibile
UnixPermissions	stringa	no	ontap-nas*: Autorizzazioni UNIX iniziali
Dimensione blocco	stringa	no	SolidFire-*: Dimensione blocco/settore

Attributo	Tipo	Obbligatorio	Descrizione
Filesystem	stringa	no	Tipo di file system

Trident genera `internalName` durante la creazione del volume. Si tratta di due fasi. Prima di tutto, prepende il prefisso di storage (predefinito) `trident` o il prefisso nella configurazione back-end) al nome del volume, con conseguente nome del modulo `<prefix>-<volume-name>`. Quindi, procede alla cancellazione del nome, sostituendo i caratteri non consentiti nel backend. Per i backend ONTAP, sostituisce i trattini con i caratteri di sottolineatura (quindi, il nome interno diventa `<prefix>_<volume-name>`). Per i backend degli elementi, sostituisce i caratteri di sottolineatura con trattini.

È possibile utilizzare le configurazioni dei volumi per eseguire il provisioning diretto dei volumi utilizzando l'API REST, ma nelle implementazioni di Kubernetes ci aspettiamo che la maggior parte degli utenti utilizzi il Kubernetes standard `PersistentVolumeClaim` metodo. Trident crea automaticamente questo oggetto volume come parte del provisioning
processo.

Trident `Snapshot` oggetti

Gli snapshot sono una copia point-in-time dei volumi, che può essere utilizzata per eseguire il provisioning di nuovi volumi o lo stato di ripristino. In Kubernetes, questi corrispondono direttamente a. `VolumeSnapshotContent` oggetti. Ogni snapshot è associato a un volume, che è l'origine dei dati per lo snapshot.

Ciascuno `Snapshot` l'oggetto include le proprietà elencate di seguito:

Attributo	Tipo	Obbligatorio	Descrizione
versione	Stringa	Sì	Versione dell'API Trident ("1")
nome	Stringa	Sì	Nome dell'oggetto snapshot Trident
InternalName (Nome interno)	Stringa	Sì	Nome dell'oggetto snapshot Trident sul sistema di storage
VolumeName	Stringa	Sì	Nome del volume persistente per il quale viene creato lo snapshot
VolumeInternalName	Stringa	Sì	Nome dell'oggetto volume Trident associato nel sistema di storage

In Kubernetes, questi oggetti vengono gestiti automaticamente. È possibile visualizzarli per visualizzare il provisioning di Trident.

Quando un Kubernetes `VolumeSnapshot` Viene creata la richiesta di oggetti, Trident lavora creando un oggetto snapshot sul sistema di storage di backup. Il `internalName` di questo oggetto snapshot viene generato combinando il prefisso `snapshot-` con `UID` di `VolumeSnapshot` oggetto (ad esempio, `snapshot-e8d8a0ca-9826-11e9-9807-525400f3f660`). `volumeName` e. `volumeInternalName` vengono popolati ottenendo i dettagli del backup

volume.

Astra Trident `ResourceQuota` oggetto

Il deambonset di Trident consuma un `system-node-critical` Classe di priorità - la classe di priorità più alta disponibile in Kubernetes - per garantire che Astra Trident sia in grado di identificare e pulire i volumi durante lo shutdown dei nodi aggraziati e consentire ai pod demonset di Trident di prevenire i carichi di lavoro con una priorità inferiore nei cluster in cui vi è un'elevata pressione sulle risorse.

Per ottenere questo risultato, Astra Trident impiega un `ResourceQuota` Scopo di garantire che una classe di priorità "system-node-critical" sul demonset Trident sia soddisfatta. Prima dell'implementazione e della creazione di demonset, Astra Trident cerca il `ResourceQuota` e, se non rilevato, lo applica.

Se è necessario un maggiore controllo sulla quota di risorse e sulla classe di priorità predefinite, è possibile generare un `custom.yaml` in alternativa, configurare `ResourceQuota` Oggetto che utilizza il grafico Helm.

Di seguito viene riportato un esempio di oggetto `ResourceQuota`che dà priorità al demonset Trident.

```
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"]
```

Per ulteriori informazioni sulle quote delle risorse, vedere "Kubernetes: Quote delle risorse".

Pulizia `ResourceQuota` se l'installazione non riesce

Nei rari casi in cui l'installazione non riesce dopo `ResourceQuota` l'oggetto viene creato, primo tentativo "disinstallazione in corso" quindi reinstallare.

In caso contrario, rimuovere manualmente `ResourceQuota` oggetto.

Rimuovere `ResourceQuota`

Se preferisci controllare la tua allocazione delle risorse, puoi rimuovere Astra Trident `ResourceQuota` oggetto mediante il comando:

```
kubectl delete quota trident-csi -n trident
```

comandi e opzioni tridentctl

Il "Pacchetto di installazione Trident" include un'utility della riga di comando, `tridentctl`, Che fornisce un accesso semplice ad Astra Trident. Kubernetes gli utenti con privilegi sufficienti possono utilizzarlo per installare Astra Trident e per interagire direttamente con esso per gestire lo spazio dei nomi che contiene il pod Astra Trident.

Comandi e opzioni disponibili

Per informazioni sull'utilizzo, eseguire `tridentctl --help`.

I comandi e le opzioni globali disponibili sono:

```
Usage:
  tridentctl [command]
```

Comandi disponibili:

- `create`: Aggiungere una risorsa ad Astra Trident.
- `delete`: Rimuovere una o più risorse da Astra Trident.
- `get`: Ottieni una o più risorse da Astra Trident.
- `help`: Guida su qualsiasi comando.
- `images`: Stampare una tabella delle immagini container di Astra Trident.
- `import`: Importare una risorsa esistente in Astra Trident.
- `install`: Installare Astra Trident.
- `logs`: Stampare i log da Astra Trident.
- `send`: Inviare una risorsa da Astra Trident.
- `uninstall`: Disinstallare Astra Trident.
- `update`: Modificare una risorsa in Astra Trident.
- `upgrade`: Aggiornare una risorsa in Astra Trident.
- `version`: Stampa la versione di Astra Trident.

Allarmi:

- `-d, --debug`: Output di debug.
- `-h, --help`: Guida per `tridentctl`.
- `-n, --namespace string`: Namespace dell'implementazione di Astra Trident.
- `-o, --output string`: Formato di output. Uno tra json|yaml|name|wide|ps (impostazione predefinita).
- `-s, --server string`: Indirizzo/porta dell'interfaccia REST di Astra Trident.

L'interfaccia REST di Trident può essere configurata per l'ascolto e la distribuzione solo su 127.0.0.1 (per IPv4) o [::1] (per IPv6).

L'interfaccia REST di Trident può essere configurata per l'ascolto e la distribuzione solo su 127.0.0.1 (per IPv4) o [::1] (per IPv6).

`create`

È possibile utilizzare il `create` Comando per aggiungere una risorsa ad Astra Trident.

```
Usage:
  tridentctl create [option]
```

Opzione disponibile:
`backend`: Aggiungi un backend ad Astra Trident.

`delete`

È possibile eseguire `delete` Comando per rimuovere una o più risorse da Astra Trident.

```
Usage:
  tridentctl delete [option]
```

Opzioni disponibili:

- `backend`: Eliminare uno o più backend di storage da Astra Trident.
- `snapshot`: Consente di eliminare una o più snapshot di volumi da Astra Trident.
- `storageclass`: Eliminare una o più classi di storage da Astra Trident.
- `volume`: Eliminare uno o più volumi di storage da Astra Trident.

`get`

È possibile eseguire `get` Comando per ottenere una o più risorse da Astra Trident.

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

Opzioni disponibili:

- `backend`: Ottieni uno o più backend di storage da Astra Trident.
- `snapshot`: Ottenere una o più istantanee da Astra Trident.
- `storageclass`: Ottieni una o più classi di storage da Astra Trident.

- `volume`: Procurarsi uno o più volumi da Astra Trident.

`volume` allarmi:
* `-h, --help`: Guida per i volumi.
* `--parentOfSubordinate string`: Limita query al volume di origine subordinato.
* `--subordinateOf string`: Limita la query alle subordinate del volume.

images

È possibile eseguire `images` Flag per stampare una tabella delle immagini container richieste da Astra Trident.

```
Usage:
  tridentctl images [flags]
```

Allarmi:
* `-h, --help`: Help for images.
* -V, --k8s-version string`: Versione semantica del cluster Kubernetes.

import volume

È possibile eseguire `import volume` Comando per importare un volume esistente in Astra Trident.

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

Alias:
`volume, v`

Allarmi:

- `-f, --filename string`: Percorso al file PVC YAML o JSON.
- `-h, --help`: Guida per il volume.
- `--no-manage`: Crea solo PV/PVC. Non presupporre la gestione del ciclo di vita dei volumi.

install

È possibile eseguire `install` Flag per installare Astra Trident.

```
Usage:
  tridentctl install [flags]
```

Allarmi:

- `--autosupport-image string`: L'immagine container per il sistema di Telemetria AutoSupport (impostazione predefinita "netapp/Trident autosupport:20.07.0").

- ``--autosupport-proxy string`Indirizzo/porta di un proxy per l'invio di telemetria AutoSupport.
- `--csi`: Installare CSI Trident (override solo per Kubernetes 1.13, richiede feature gate).
- `--enable-node-prep`: Tentare di installare i pacchetti richiesti sui nodi.
- `--generate-custom-yaml`: Generare file YAML senza installare nulla.
- `-h, --help`: Guida all'installazione.
- `--http-request-timeout`: Consente di sovrascrivere il timeout della richiesta HTTP per l'API REST del controller Trident (valore predefinito 1 m30s).
- `--image-registry string`: L'indirizzo/porta di un registro di immagini interno.
- `--k8s-timeout duration`: Il timeout per tutte le operazioni Kubernetes (valore predefinito: 3 m0s).
- `--kubelet-dir string`: La posizione host dello stato interno di kubelet (default "/var/lib/kubelet").
- `--log-format string`: Il formato di registrazione Astra Trident (text, json) (default "text").
- `--pv string`: Il nome del PV legacy utilizzato da Astra Trident, garantisce che non esista (il "tridente" predefinito).
- `--pvc string`: Il nome del PVC legacy utilizzato da Astra Trident, garantisce che non esista (il "tridente" predefinito).
- `--silence-autosupport`: Non inviare pacchetti AutoSupport a NetApp automaticamente (valore predefinito vero).
- `--silent`: Disattivare l'output MOST durante l'installazione.
- `--trident-image string`: L'immagine Astra Trident da installare.
- `--use-custom-yaml`: Utilizzare tutti i file YAML esistenti nella directory di installazione.
- `--use-ipv6`: Utilizza IPv6 per la comunicazione di Astra Trident.

logs

È possibile eseguire `logs` Flag per stampare i log da Astra Trident.

```
Usage:
  tridentctl logs [flags]
```

Allarmi:

- `-a, --archive`: Creare un archivio di supporto con tutti i log, se non diversamente specificato.
- `-h, --help`: Guida per i log.
- `-l, --log string`: Registro Astra Trident da visualizzare. Uno tra trident|auto|trident-operator|all (impostazione predefinita "auto").
- `--node string`: Il nome del nodo Kubernetes da cui raccogliere i log dei pod dei nodi.
- `-p, --previous`: Ottenere i log per l'istanza di container precedente, se esistente.
- `--sidecars`: Ottenere i log per i contenitori del sidecar.

send

È possibile eseguire `send` Comando per inviare una risorsa da Astra Trident.

```
Usage:
  tridentctl send [option]
```

Opzione disponibile:
`autosupport`: Inviare un archivio AutoSupport a NetApp.

uninstall

È possibile eseguire `uninstall` Flag per disinstallare Astra Trident.

```
Usage:
  tridentctl uninstall [flags]
```

Allarmi:
* `-h, --help`: Guida per la disinstallazione.
* `--silent`: Disattivare la maggior parte dell'output durante la disinstallazione.

update

È possibile eseguire `update` Comandi per modificare una risorsa in Astra Trident.

```
Usage:
  tridentctl update [option]
```

Opzioni disponibili:
`backend`: Aggiornare un backend in Astra Trident.

upgrade

È possibile eseguire `upgrade` Comandi per aggiornare una risorsa in Astra Trident.

```
Usage:
tridentctl upgrade [option]
```

Opzione disponibile:
`volume`: Aggiornare uno o più volumi persistenti da NFS/iSCSI a CSI.

`version`

È possibile eseguire `version` contrassegni per stampare la versione di `tridentctl` E il servizio Running Trident.

```
Usage:
  tridentctl version [flags]
```

Allarmi:
* `--client`: Solo versione client (non è richiesto alcun server).
* `-h, --help`: Guida per la versione.

Pod Security Standards (PSS) e Security Context Constraints (SCC)

Kubernetes Pod Security Standards (PSS) e Pod Security Policy (PSP) definiscono i livelli di autorizzazione e limitano il comportamento dei pod. OpenShift Security Context Constraints (SCC) definisce analogamente la restrizione pod specifica per OpenShift Kubernetes Engine. Per fornire questa personalizzazione, Astra Trident abilita alcune autorizzazioni durante l'installazione. Nelle sezioni seguenti sono descritte in dettaglio le autorizzazioni impostate da Astra Trident.

PSS sostituisce Pod Security Policies (PSP). PSP è stato deprecato in Kubernetes v1.21 e verrà rimosso nella versione 1.25. Per ulteriori informazioni, vedere "Kubernetes: Sicurezza".

Contesto di sicurezza Kubernetes obbligatorio e campi correlati

Permesso	Descrizione
Privilegiato	CSI richiede che i punti di montaggio siano bidirezionali, il che significa che il pod di nodi Trident deve eseguire un container privilegiato. Per ulteriori informazioni, vedere "Kubernetes: Propagazione del mount".
Rete host	Necessario per il daemon iSCSI. `iscsiadm` Gestisce i montaggi iSCSI e utilizza la rete host per comunicare con il daemon iSCSI.
Host IPC (IPC host)	NFS utilizza la comunicazione interprocesso (IPC) per comunicare con NFSD.
PID host	Necessario per iniziare `rpc-statd` Per NFS. Astra Trident interroga i processi host per determinare se `rpc-statd` È in esecuzione prima di montare volumi NFS.

Permesso	Descrizione

Funzionalità	Il `SYS_ADMIN` la funzionalità viene fornita come parte delle funzionalità predefinite per i container con privilegi. Ad esempio, Docker imposta queste funzionalità per i container con privilegi:
	`CapPrm: 0000003fffffffff`
	`CapEff: 0000003fffffffff`

Seccomp	Il profilo Seccomp è sempre "non confinato" in container con privilegi; pertanto, non può essere abilitato in Astra Trident.
SELinux	In OpenShift, i container con privilegi vengono eseguiti in `spc_t` ("Super Privileged container") e i container senza privilegi vengono eseguiti in `container_t` dominio. Acceso `containerd`, con `container-selinux` installati, tutti i container vengono eseguiti in `spc_t` Domain, che disattiva effettivamente SELinux. Pertanto, Astra Trident non aggiunge `seLinuxOptions` ai container.
DAC	I container con privilegi devono essere eseguiti come root. I container non privilegiati vengono eseguiti come root per accedere ai socket unix richiesti da CSI.

Standard di sicurezza Pod (PSS)

Etichetta	Descrizione	Predefinito

`pod-security.kubernetes.io/enforce`
 	
`pod-security.kubernetes.io/enforce-version`
	Consente di ammettere il controller Trident e i nodi nello spazio dei nomi install.
	
	Non modificare l'etichetta dello spazio dei nomi.
		`enforce: privileged`
		
		`enforce-version: <version of the current cluster or highest version of PSS tested.>`

La modifica delle etichette dello spazio dei nomi può causare la mancata pianificazione dei pod, un "errore di creazione: …" Oppure "Warning: trident-csi-…". In tal caso, controllare se l'etichetta dello spazio dei nomi di `privileged` è stato modificato. In tal caso, reinstallare Trident.

Policy di sicurezza Pod (PSP)

Campo	Descrizione	Predefinito
`allowPrivilegeEscalation`	I container con privilegi devono consentire l'escalation dei privilegi.	`true`
`allowedCSIDrivers`	Trident non utilizza volumi effimeri CSI inline.	Vuoto

Campo	Descrizione	Predefinito
`allowedCapabilities`	I container Trident non con privilegi non richiedono più funzionalità rispetto al set predefinito e ai container con privilegi vengono concesse tutte le funzionalità possibili.	Vuoto
`allowedFlexVolumes`	Trident non utilizza un "Driver FlexVolume", quindi non sono inclusi nell'elenco dei volumi consentiti.	Vuoto
`allowedHostPaths`	Il pod di nodi Trident monta il filesystem root del nodo, quindi non c'è alcun beneficio nell'impostazione di questo elenco.	Vuoto
`allowedProcMountTypes`	Trident non ne utilizza alcuno `ProcMountTypes`.	Vuoto
`allowedUnsafeSysctls`	Trident non richiede alcuna operazione non sicura `sysctls`.	Vuoto
`defaultAddCapabilities`	Non è necessario aggiungere funzionalità ai container con privilegi.	Vuoto
`defaultAllowPrivilegeEscalation`	L'escalation dei privilegi viene gestita in ogni pod Trident.	`false`
`forbiddenSysctls`	No `sysctls` sono consentiti.	Vuoto
`fsGroup`	I container Trident vengono eseguiti come root.	`RunAsAny`
`hostIPC`	Il montaggio dei volumi NFS richiede l'IPC host per comunicare con `nfsd`	`true`
`hostNetwork`	Iscsiadm richiede che la rete host comunichi con il daemon iSCSI.	`true`
`hostPID`	Per verificare se è necessario utilizzare il PID host `rpc-statd` è in esecuzione sul nodo.	`true`
`hostPorts`	Trident non utilizza porte host.	Vuoto
`privileged`	I pod di nodi Trident devono eseguire un container privilegiato per poter montare i volumi.	`true`
`readOnlyRootFilesystem`	I pod di nodi Trident devono scrivere nel file system del nodo.	`false`
`requiredDropCapabilities`	I pod di nodi Trident eseguono un container privilegiato e non possono rilasciare funzionalità.	`none`

Campo	Descrizione	Predefinito
`runAsGroup`	I container Trident vengono eseguiti come root.	`RunAsAny`
`runAsUser`	I container Trident vengono eseguiti come root.	`runAsAny`
`runtimeClass`	Trident non utilizza `RuntimeClasses`.	Vuoto
`seLinux`	Trident non viene impostato `seLinuxOptions` Perché ci sono attualmente differenze nel modo in cui i runtime dei container e le distribuzioni Kubernetes gestiscono SELinux.	Vuoto
`supplementalGroups`	I container Trident vengono eseguiti come root.	`RunAsAny`
`volumes`	I pod Trident richiedono questi plug-in di volume.	`hostPath, projected, emptyDir`

SCC (Security Context Constraints)

Etichette	Descrizione	Predefinito
`allowHostDirVolumePlugin`	I pod di nodi Trident montano il filesystem root del nodo.	`true`
`allowHostIPC`	Il montaggio dei volumi NFS richiede l'IPC host per comunicare con `nfsd`.	`true`
`allowHostNetwork`	Iscsiadm richiede che la rete host comunichi con il daemon iSCSI.	`true`
`allowHostPID`	Per verificare se è necessario utilizzare il PID host `rpc-statd` è in esecuzione sul nodo.	`true`
`allowHostPorts`	Trident non utilizza porte host.	`false`
`allowPrivilegeEscalation`	I container con privilegi devono consentire l'escalation dei privilegi.	`true`
`allowPrivilegedContainer`	I pod di nodi Trident devono eseguire un container privilegiato per poter montare i volumi.	`true`
`allowedUnsafeSysctls`	Trident non richiede alcuna operazione non sicura `sysctls`.	`none`

Etichette	Descrizione	Predefinito
`allowedCapabilities`	I container Trident non con privilegi non richiedono più funzionalità rispetto al set predefinito e ai container con privilegi vengono concesse tutte le funzionalità possibili.	Vuoto
`defaultAddCapabilities`	Non è necessario aggiungere funzionalità ai container con privilegi.	Vuoto
`fsGroup`	I container Trident vengono eseguiti come root.	`RunAsAny`
`groups`	Questo SCC è specifico di Trident ed è vincolato al proprio utente.	Vuoto
`readOnlyRootFilesystem`	I pod di nodi Trident devono scrivere nel file system del nodo.	`false`
`requiredDropCapabilities`	I pod di nodi Trident eseguono un container privilegiato e non possono rilasciare funzionalità.	`none`
`runAsUser`	I container Trident vengono eseguiti come root.	`RunAsAny`
`seLinuxContext`	Trident non viene impostato `seLinuxOptions` Perché ci sono attualmente differenze nel modo in cui i runtime dei container e le distribuzioni Kubernetes gestiscono SELinux.	Vuoto
`seccompProfiles`	I container privilegiati vengono sempre eseguiti "senza confinare".	Vuoto
`supplementalGroups`	I container Trident vengono eseguiti come root.	`RunAsAny`
`users`	Viene fornita una voce per associare SCC all'utente Trident nello spazio dei nomi Trident.	n/a.
`volumes`	I pod Trident richiedono questi plug-in di volume.	`hostPath, downwardAPI, projected, emptyDir`

Informazioni sul copyright

Copyright © 2026 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 COMMERCIABILITÀ 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 CONSEQUENZIALE (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.