Gestisci e monitora Trident

Trident

NetApp
April 08, 2026

Sommario

Gestisci e monitora Trident

Aggiorna Trident

Aggiorna Trident

A partire dalla versione 24.02, Trident segue una cadenza di rilascio di quattro mesi, rilasciando tre release principali ogni anno solare. Ogni nuova release si basa sulle precedenti e offre nuove funzionalità, miglioramenti delle prestazioni, correzioni di bug e miglioramenti. Ti invitiamo ad aggiornare almeno una volta all'anno per sfruttare le nuove funzionalità di Trident.

Considerazioni prima dell'aggiornamento

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

- Dovrebbe essere installata una sola istanza di Trident in tutti gli spazi dei nomi in un dato cluster Kubernetes.
- Trident 23.07 e versioni successive richiedono snapshot del volume v1 e non supportano più snapshot alpha o beta.
- Durante l'aggiornamento, è importante fornire `parameter.fsType` in `StorageClasses` utilizzati da Trident. È possibile eliminare e ricreare `StorageClasses` senza interrompere i volumi preesistenti.
  - Questo è un **requisito** per l'applicazione "contesti di sicurezza" per i volumi SAN.
  - La directory sample input contiene esempi, come `storage-class-basic.yaml.templ` e `storage-class-bronze-default.yaml`.
  - Per ulteriori informazioni, fare riferimento a "Problemi noti".

Passaggio 1: seleziona una versione

Le versioni Trident seguono una convenzione di naming basata sulla data `YY.MM`, dove "YY" sono le ultime due cifre dell'anno e "MM" è il mese. Le versioni Dot seguono una convenzione `YY.MM.X`, dove "X" è il livello di patch. Selezionerai la versione a cui eseguire l'aggiornamento in base alla versione da cui stai aggiornando.

- È possibile eseguire un aggiornamento diretto a qualsiasi release di destinazione che rientri in una finestra di quattro release della versione installata. Ad esempio, è possibile eseguire un aggiornamento diretto dalla 24.06 (o da qualsiasi 24.06 dot release) alla 25.06.
- Se stai eseguendo l'aggiornamento da una versione al di fuori della finestra di quattro release, esegui un aggiornamento in più fasi. Utilizza le istruzioni di aggiornamento per la versione precedente da cui stai eseguendo l'aggiornamento per passare alla versione più recente che rientra nella finestra di quattro release. Ad esempio, se stai utilizzando la versione 23.07 e desideri eseguire l'aggiornamento alla versione 25.06:
  a. Primo upgrade dal 23.07 al 24.06.
  b. Quindi eseguire l'aggiornamento da 24.06 a 25.06.

Quando si esegue l'aggiornamento utilizzando l'operatore Trident su OpenShift Container Platform, è necessario eseguire l'aggiornamento a Trident 21.01.1 o versioni successive. L'operatore Trident rilasciato con 21.01.0 contiene un problema noto che è stato risolto in 21.01.1. Per ulteriori dettagli, consultare il "dettagli del problema su GitHub".

Fase 2: determinare il metodo di installazione originale

Per determinare quale versione hai utilizzato per installare originariamente Trident:

1. Usa `kubectl get pods -n trident` per esaminare i pod.
   - Se non è presente alcun pod operatore, Trident è stato installato usando `tridentctl`.
   - Se è presente un pod operatore, Trident è stato installato utilizzando l'operatore Trident manualmente o tramite Helm.
2. Se è presente un operator pod, usa `kubectl describe torc` per determinare se Trident è stato installato usando Helm.
   - Se è presente un'etichetta Helm, Trident è stato installato utilizzando Helm.
   - Se non è presente alcuna etichetta Helm, Trident è stato installato manualmente utilizzando l'operatore Trident.

Fase 3: Seleziona un metodo di aggiornamento

In generale, dovresti eseguire l'aggiornamento utilizzando lo stesso metodo usato per l'installazione iniziale, tuttavia puoi "passare da un metodo di installazione all'altro". Ci sono due opzioni per aggiornare Trident.

- "Aggiorna utilizzando l'operatore Trident"

Si consiglia di esaminare "Comprendere il workflow di upgrade dell'operatore" prima di eseguire l'aggiornamento con l'operatore.

*

Aggiorna con l'operatore

Comprendere il workflow di upgrade dell'operatore

Prima di utilizzare l'operatore Trident per aggiornare Trident, è necessario comprendere i processi in background che si verificano durante l'aggiornamento. Ciò include le modifiche al Trident controller, al controller Pod e ai node Pods, e al node DaemonSet che abilitano gli aggiornamenti continui.

Gestione dell'aggiornamento dell'operatore Trident

Una delle tante "vantaggi dell'utilizzo dell'operatore Trident" modalità per installare e aggiornare Trident è la gestione automatica degli oggetti Trident e Kubernetes senza interrompere i volumi montati esistenti. In questo modo, Trident può supportare gli aggiornamenti senza tempi di inattività, ovvero "*aggiornamenti rolling*" offline. In particolare, l'operatore Trident comunica con il cluster Kubernetes per:

- Eliminare e ricreare la distribuzione del Trident Controller e il nodo DaemonSet.

- Sostituisci i Trident Controller Pod e i Trident Node Pod con nuove versioni.
  - Se un nodo non viene aggiornato, non impedisce che i nodi rimanenti vengano aggiornati.
  - Solo i nodi con un Trident Node Pod in esecuzione possono montare volumi.

Per ulteriori informazioni sull'architettura Trident sul cluster Kubernetes, fare riferimento a "Architettura di Trident".

Flusso di lavoro di aggiornamento dell'operatore

Quando si avvia un aggiornamento utilizzando l'operatore Trident:

1. L'operatore **Trident**:
   a. Rileva la versione attualmente installata di Trident (versione *n*).
   b. Aggiorna tutti gli oggetti Kubernetes, inclusi CRDs, RBAC e Trident SVC.
   c. Elimina la distribuzione del Trident Controller per la versione *n*.
   d. Crea la distribuzione del Trident Controller per la versione *n+1*.
2. **Kubernetes** crea il Trident Controller Pod per *n+1*.
3. L'operatore **Trident**:
   a. Elimina il DaemonSet del nodo Trident per *n*. L'operatore non attende la terminazione del Node Pod.
   b. Crea il Trident Node Daemonset per *n+1*.
4. **Kubernetes** crea Trident Node Pod sui nodi che non eseguono Trident Node Pod *n*. Questo garantisce che non ci sia mai più di un Trident Node Pod, di qualsiasi versione, su un nodo.

Aggiorna un'installazione Trident utilizzando l'operatore Trident o Helm

Puoi aggiornare Trident utilizzando l'operatore Trident manualmente o tramite Helm. Puoi aggiornare da un'installazione dell'operatore Trident a un'altra installazione dell'operatore Trident oppure aggiornare da un'installazione `tridentctl` a una versione dell'operatore Trident. Esamina "Seleziona un metodo di aggiornamento" prima di aggiornare un'installazione dell'operatore Trident.

Aggiorna un'installazione manuale

È possibile effettuare l'aggiornamento da un'installazione dell'operatore Trident con ambito cluster a un'altra installazione dell'operatore Trident con ambito cluster. Tutte le versioni di Trident utilizzano un operatore con ambito cluster.

Per eseguire l'aggiornamento da Trident installato utilizzando l'operatore namespace-scoped (versioni 20.07 fino a 20.10), utilizzare le istruzioni di aggiornamento per la versione di Trident installata.

Informazioni su questa attività

Trident fornisce un file bundle che puoi utilizzare per installare l'operatore e creare oggetti associati per la tua versione di Kubernetes.

- Per i cluster che eseguono Kubernetes 1.24, usa "bundle_pre_1_25.yaml".

- Per i cluster che eseguono Kubernetes 1.25 o versioni successive, usa "bundle_post_1_25.yaml".

Prima di iniziare

Assicurati di utilizzare un cluster Kubernetes che esegue "una versione supportata di Kubernetes".

Passaggi

1. Verifica la tua versione di Trident:

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

2. Aggiorna `operator.yaml`, `tridentorchestrator_cr.yaml` e `post_1_25_bundle.yaml` con il registry e i percorsi delle immagini per la versione a cui stai eseguendo l'aggiornamento (ad esempio 25.06) e il secret corretto.
3. Elimina l'operatore Trident che è stato utilizzato per installare l'istanza corrente di Trident. Ad esempio, se stai eseguendo l'aggiornamento dalla versione 25.02, esegui il seguente comando:

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

4. Se hai personalizzato l'installazione iniziale utilizzando `TridentOrchestrator` attributi, puoi modificare l'oggetto `TridentOrchestrator` per modificare i parametri di installazione. Questo potrebbe includere modifiche apportate per specificare registri di immagini Trident e CSI con mirroring per la modalità offline, abilitare i log di debug o specificare i segreti di pull delle immagini.
5. Installa Trident utilizzando il file YAML del bundle corretto per il tuo ambiente, dove *<bundle.yaml>* è `bundle_pre_1_25.yaml` o `bundle_post_1_25.yaml` in base alla tua versione di Kubernetes. Ad esempio, se stai installando Trident 25.06.0, esegui il seguente comando:

```
kubectl create -f 25.06.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

6. Modifica il trident torc per includere l'immagine 25.06.0.

Aggiornare un'installazione di Helm

È possibile aggiornare un'installazione di Trident Helm.

> Quando si aggiorna un cluster Kubernetes dalla versione 1.24 alla 1.25 o successiva che ha Trident installato, è necessario aggiornare values.yaml per impostare `excludePodSecurityPolicy` su `true` o aggiungere `--set excludePodSecurityPolicy=true` al comando `helm upgrade` prima di poter aggiornare il cluster.

Se hai già aggiornato il tuo cluster Kubernetes dalla versione 1.24 alla 1.25 senza aggiornare il Trident helm, l'upgrade di helm fallisce. Perché l'upgrade di helm vada a buon fine, esegui questi passaggi come prerequisiti:

1. Installa il plugin helm-mapkubeapis da https://github.com/helm/helm-mapkubeapis.
2. Eseguire una simulazione per la release di Trident nello spazio dei nomi in cui Trident è installato. Questo elenca le risorse che verranno ripulite.

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

3. Eseguire una corsa completa con helm per effettuare la pulizia.

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

Passaggi

1. Se "installato Trident usando Helm", puoi usare `helm upgrade trident netapp-trident/trident-operator --version 100.2506.0` per aggiornare in un unico passaggio. Se non hai aggiunto il Helm repo o non puoi usarlo per aggiornare:
   a. Scarica l'ultima release di Trident da "la sezione *Assets* su GitHub".
   b. Usa il `helm upgrade` comando dove `trident-operator-25.10.0.tgz` riflette la versione a cui vuoi eseguire l'aggiornamento.

```
helm upgrade <name> trident-operator-25.10.0.tgz
```

> Se imposti opzioni personalizzate durante l'installazione iniziale (ad esempio specificando registri privati o mirror per le immagini di Trident e CSI), aggiungi il comando `helm upgrade` usando `--set` per assicurarti che tali opzioni siano incluse nel comando di aggiornamento, altrimenti i valori verranno reimpostati su quelli predefiniti.

2. Esegui `helm list` per verificare che la versione del chart e dell'app siano state entrambe aggiornate. Esegui `tridentctl logs` per rivedere eventuali messaggi di debug.

Aggiorna da un'installazione `tridentctl` a Trident operator

Puoi eseguire l'aggiornamento all'ultima release dell'operatore Trident da un' `tridentctl`installazione. I backend e i PVC esistenti saranno automaticamente disponibili.

> Prima di passare da un metodo di installazione all'altro, rivedere "Spostarsi tra i metodi di installazione".

Passaggi

1. Scarica l'ultima release di Trident.

```
# Download the release required [25.10.0]
mkdir 25.10.0
cd 25.10.0
wget
https://github.com/NetApp/trident/releases/download/v25.10.0/trident-
installer-25.10.0.tar.gz
tar -xf trident-installer-25.10.0.tar.gz
cd trident-installer
```

2. Crea il `tridentorchestrator` CRD dal manifesto.

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

3. Distribuisci l'operatore con ambito cluster nello stesso namespace.

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

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

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

4. Crea una `TridentOrchestrator` CR per installare Trident.

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

kubectl create -f deploy/crds/tridentorchestrator_cr.yaml

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

5. Conferma che Trident è stato aggiornato alla versione prevista.

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

Message:                Trident installed
Namespace:              trident
Status:                 Installed
Version:                v25.10.0
```

Aggiorna con tridentctl

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

Informazioni su questa attività

La disinstallazione e la reinstallazione di Trident agiscono come un aggiornamento. Quando si disinstalla Trident, il Persistent Volume Claim (PVC) e il Persistent Volume (PV) utilizzati dalla distribuzione di Trident non vengono eliminati. I PV che sono già stati forniti rimarranno disponibili mentre Trident è offline e Trident fornirà volumi per qualsiasi PVC creato nel frattempo dopo che è tornato online.

Prima di iniziare

Rivedi "Seleziona un metodo di aggiornamento" prima di aggiornare usando `tridentctl`.

Passaggi

1. Esegui il comando di disinstallazione in `tridentctl` per rimuovere tutte le risorse associate a Trident, ad eccezione dei CRD e degli oggetti correlati.

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

2. Reinstallare Trident. Fare riferimento a "Installare Trident usando tridentctl".

> Non interrompere il processo di aggiornamento. Assicurati che il programma di installazione venga eseguito fino al completamento.

Gestisci Trident usando tridentctl

Il "Bundle di installazione Trident" include l'utilità a riga di comando `tridentctl` per fornire un accesso semplice a Trident. Gli utenti di Kubernetes con privilegi sufficienti possono usarla per installare Trident o gestire lo spazio dei nomi che contiene il pod Trident.

Comandi e flag globali

È possibile eseguire `tridentctl help` per ottenere un elenco dei comandi disponibili per `tridentctl` o aggiungere il `--help` flag a qualsiasi comando per ottenere un elenco di opzioni e flag per quel comando specifico.

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

L'utilità Trident `tridentctl` supporta i seguenti comandi e flag globali.

Comandi

`create`
Aggiungi una risorsa a Trident.

`delete`
Rimuovi una o più risorse da Trident.

`get`
Ottieni una o più risorse da Trident.

`help`
Guida su qualsiasi comando.

`images`
Stampa una tabella delle immagini dei container di cui Trident ha bisogno.

`import`
Importa una risorsa esistente in Trident.

`install`
Installa Trident.

`logs`
Stampa i log da Trident.

`send`
Invia una risorsa da Trident.

`uninstall`
Disinstalla Trident.

`update`
Modifica una risorsa in Trident.

`update backend state`
Sospendere temporaneamente le operazioni di backend.

`upgrade`
Aggiorna una risorsa in Trident.

`version`
Stampa la versione di Trident.

Bandiere globali

-d, --debug
Output di debug.

-h, --help
Aiuto per `tridentctl`.

-k, --kubeconfig string
Specificare il `KUBECONFIG` percorso per eseguire i comandi localmente o da un cluster Kubernetes a un altro.

In alternativa, puoi esportare la `KUBECONFIG` variabile per puntare a uno specifico cluster Kubernetes e inviare `tridentctl` comandi a quel cluster.

-n, --namespace string
Namespace della distribuzione Trident.

-o, --output string
Formato di output. Uno tra json|yaml|name|wide|ps (predefinito).

-s, --server string
Indirizzo/porta dell'interfaccia REST di Trident.

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

Opzioni e flag dei comandi

crea

Utilizza il `create` comando per aggiungere una risorsa a Trident.

```
tridentctl create [option]
```

Opzioni

`backend`: Aggiungi un backend a Trident.

eliminare

Utilizzare il comando `delete` per rimuovere una o più risorse da Trident.

```
tridentctl delete [option]
```

Opzioni

`backend`: Elimina uno o più backend di storage da Trident.
`snapshot`: Elimina uno o più snapshot di volume da Trident.
`storageclass`: Elimina una o più storage class da Trident.

`volume`: Elimina uno o più storage volume da Trident.

get

Utilizza il `get` comando per ottenere una o più risorse da Trident.

```
tridentctl get [option]
```

Opzioni

`backend`: Ottieni uno o più backend di storage da Trident.
`snapshot`: Ottieni uno o più snapshot da Trident.
`storageclass`: Ottieni una o più storage class da Trident.
`volume`: Ottieni uno o più volumi da Trident.

Flag

`-h`, `--help`: Guida per i volumi.
`--parentOfSubordinate string`: Limita la query al volume sorgente subordinato.
`--subordinateOf string`: Limita la query ai subordinati del volume.

immagini

Utilizza `images` i flag per stampare una tabella delle immagini dei container di cui Trident ha bisogno.

```
tridentctl images [flags]
```

Flag

`-h`, `--help`: Aiuto per le immagini.
`-v`, `--k8s-version string`: Versione semantica del cluster Kubernetes.

importa volume

Utilizzare il comando `import volume` per importare un volume esistente in Trident.

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

Alias

`volume`, `v`

Flag

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

installare

Utilizzare i `install` flag per installare Trident.

```
tridentctl install [flags]
```

Flag

`--autosupport-image string`: L'immagine del container per Autosupport Telemetry (predefinito "netapp/trident autosupport:<current-version>").
`--autosupport-proxy string`: L'indirizzo/porta di un proxy per l'invio di Autosupport Telemetry.
`--enable-node-prep`: Tenta di installare i pacchetti richiesti sui nodi.
`--generate-custom-yaml`: Genera file YAML senza installare nulla.
`-h, --help`: Guida per install.
`--http-request-timeout`: Ignora il timeout della richiesta HTTP per la REST API del controller Trident (predefinito 1m30s).
`--image-registry string`: L'indirizzo/porta di un registro di immagini interno.
`--k8s-timeout duration`: Il timeout per tutte le operazioni Kubernetes (predefinito 3m0s).
`--kubelet-dir string`: La posizione host dello stato interno di kubelet (predefinito "/var/lib/kubelet").
`--log-format string`: Il formato di logging di Trident (text, json) (predefinito "text").
`--node-prep`: Consente a Trident di preparare i nodi del cluster Kubernetes per gestire i volumi utilizzando il protocollo storage specificato. **Attualmente, `iscsi` è l'unico valore supportato. A partire da OpenShift 4.19, la versione minima di Trident supportata per questa funzionalità è 25.06.1.**
`--pv string`: Il nome del PV legacy utilizzato da Trident, assicurati che non esista (predefinito "trident").
`--pvc string`: Il nome del PVC legacy utilizzato da Trident, assicurati che non esista (predefinito "trident").
`--silence-autosupport`: Non inviare automaticamente i bundle di autosupport a NetApp (predefinito true).
`--silent`: Disabilita la maggior parte dell'output durante l'installazione.
`--trident-image string`: L'immagine Trident da installare.
`--k8s-api-qps`: Il limite di query per secondo (QPS) per le richieste API di Kubernetes (predefinito 100; opzionale).
`--use-custom-yaml`: Utilizza eventuali file YAML esistenti nella directory di setup.
`--use-ipv6`: Utilizza IPv6 per la comunicazione di Trident.

registri

Utilizzare `logs` flag per stampare i log da Trident.

```
tridentctl logs [flags]
```

Flag

`-a, --archive`: Crea un archivio di supporto con tutti i log salvo diversa indicazione.
`-h, --help`: Guida per i log.
`-l, --log string`: Trident log da visualizzare. Uno tra trident|auto|trident-operator|all (predefinito "auto").
`--node string`: Il nome del nodo Kubernetes da cui raccogliere i log dei pod del nodo.
`-p, --previous`: Ottieni i log per l'istanza precedente del container, se esistente.
`--sidecars`: Ottieni i log per i container sidecar.

Invia

Utilizzare il comando `send` per inviare una risorsa da Trident.

```
tridentctl send [option]
```

Opzioni

`autosupport`: Invia un archivio Autosupport a NetApp.

disinstallare

Utilizzare i `uninstall` flag per disinstallare Trident.

```
tridentctl uninstall [flags]
```

Flag

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

aggiornamento

Utilizza il `update` comando per modificare una risorsa in Trident.

```
tridentctl update [option]
```

Opzioni

`backend`: Aggiorna un backend in Trident.

aggiorna lo stato del backend

Utilizza il comando `update backend state` per sospendere o riprendere le operazioni di backend.

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

Punti da considerare

- Se un backend viene creato utilizzando un TridentBackendConfig (tbc), il backend non può essere aggiornato utilizzando un `backend.json` file.
- Se `userState` è stato impostato in un tbc, non può essere modificato utilizzando il comando `tridentctl update backend state <backend-name> --user-state suspended/normal`.
- Per ripristinare la possibilità di impostare il `userState` tramite tridentctl dopo che è stato impostato tramite tbc, il campo `userState` deve essere rimosso dal tbc. Questo può essere fatto usando il comando `kubectl edit tbc`. Dopo che il campo `userState` è stato rimosso, è possibile usare il comando `tridentctl update backend state` per modificare il `userState` di un backend.
- Utilizza il `tridentctl update backend state` per modificare il `userState`. Puoi anche aggiornare il `userState` utilizzando `TridentBackendConfig` o `backend.json` file; ciò innesca una reinizializzazione completa del backend e può richiedere molto tempo.

  #### Flag

  `-h, --help`: Aiuto per lo stato del backend.
  `--user-state`: Impostare su `suspended` per mettere in pausa le operazioni del backend. Impostare su `normal` per riprendere le operazioni del backend. Quando impostato su `suspended`:

- `AddVolume` and `Import Volume` sono in pausa.
- `CloneVolume`, `ResizeVolume`, `PublishVolume`, `UnPublishVolume`, `CreateSnapshot`, `GetSnapshot`, `RestoreSnapshot`, `DeleteSnapshot`, `RemoveVolume`, `GetVolumeExternal`, `ReconcileNodeAccess` rimangono disponibili.

È anche possibile aggiornare lo stato del backend utilizzando `userState` il campo nel file di configurazione del backend `TridentBackendConfig` o `backend.json`. Per ulteriori informazioni, fare riferimento a

"Opzioni per la gestione dei backend" e "Esegui la gestione del backend con kubectl".

Esempio:

JSON

Segui questi passaggi per aggiornare il `userState` utilizzando il `backend.json` file:

1. Modifica il `backend.json` file per includere il `userState` campo con il valore impostato su 'suspended'.
2. Aggiorna il backend utilizzando il `tridentctl update backend` comando e il percorso del file `backend.json` aggiornato.

   Esempio: `tridentctl update backend -f /<path to backend JSON file>/backend.json -n trident`

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

YAML

È possibile modificare il tbc dopo averlo applicato utilizzando il comando `kubectl edit <tbc-name> -n <namespace>`. Il seguente esempio aggiorna lo stato del backend in sospensione utilizzando l'opzione `userState: suspended`:

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

versione

Utilizzare `version` i flag per stampare la versione di `tridentctl` e il servizio Trident in esecuzione.

```
tridentctl version [flags]
```

Flag

`--client`: Solo versione client (nessun server richiesto).
`-h, --help`: Guida per la versione.

Supporto plugin

Tridentctl supporta plugin simili a kubectl. Tridentctl rileva un plugin se il nome del file binario del plugin segue lo schema "tridentctl-<plugin>" e il binario si trova in una cartella elencata nella variabile d'ambiente PATH. Tutti i plugin rilevati sono elencati nella sezione plugin della guida di tridentctl. Facoltativamente, puoi anche limitare la ricerca specificando una cartella di plugin nella variabile d'ambiente TRIDENTCTL_PLUGIN_PATH (Example: `TRIDENTCTL_PLUGIN_PATH=~/tridentctl-plugins/`). Se la variabile viene utilizzata, tridentctl cerca solo nella cartella specificata.

Monitorare Trident

Trident fornisce una serie di endpoint di metriche Prometheus che puoi utilizzare per monitorare le prestazioni di Trident.

Panoramica

Le metriche fornite da Trident consentono di fare quanto segue:

- Tieni d'occhio lo stato di salute e la configurazione di Trident. Puoi verificare il successo delle operazioni e se riesce a comunicare con i backend come previsto.
- Esaminare le informazioni sull'utilizzo del backend e comprendere quanti volumi sono provisionati su un backend, la quantità di spazio consumata e così via.
- Mantieni una mappatura della quantità di volumi forniti sui backend disponibili.
- Monitora le prestazioni. Puoi osservare quanto tempo impiega Trident a comunicare con i backend ed eseguire operazioni.

> Per impostazione predefinita, le metriche di Trident sono esposte sulla porta di destinazione `8001` all'endpoint `/metrics`. Queste metriche sono **abilitate per impostazione predefinita** quando Trident è installato. È possibile configurare il consumo delle metriche di Trident anche tramite HTTPS sulla porta `8444`.

Cosa ti servirà

- Un cluster Kubernetes con Trident installato.
- Un'istanza di Prometheus. Questo può essere un "distribuzione containerizzata di Prometheus" oppure puoi scegliere di eseguire Prometheus come un "applicazione nativa".

Fase 1: definire un target Prometheus

Dovresti definire un target Prometheus per raccogliere le metriche e ottenere informazioni sui backend gestiti

da Trident, sui volumi che crea e così via. Vedi "Documentazione Prometheus Operator".

Fase 2: crea un Prometheus ServiceMonitor

Per consumare le metriche Trident, è necessario creare un Prometheus ServiceMonitor che monitora il `trident-csi` servizio e ascolta sulla porta `metrics`. Un esempio di ServiceMonitor è il seguente:

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

Questa definizione di ServiceMonitor recupera le metriche restituite dal `trident-csi` servizio e cerca specificamente l'endpoint `metrics` del servizio. Di conseguenza, Prometheus è ora configurato per comprendere le metriche di Trident.

Oltre alle metriche disponibili direttamente da Trident, kubelet espone molte `kubelet_volume_*` metriche tramite il proprio endpoint delle metriche. Kubelet può fornire informazioni sui volumi collegati, sui pod e su altre operazioni interne che gestisce. Fare riferimento a "qui".

Utilizza le metriche di Trident tramite HTTPS

Per utilizzare le metriche Trident tramite HTTPS (porta 8444), è necessario modificare la definizione ServiceMonitor per includere la configurazione TLS. È inoltre necessario copiare il `trident-csi` secret dallo `trident` namespace allo namespace in cui è in esecuzione Prometheus. È possibile farlo utilizzando il seguente comando:

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

Un esempio di ServiceMonitor per metriche HTTPS si presenta così:

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: trident-sm
  namespace: monitoring
  labels:
    release: prom-operator
spec:
  jobLabel: trident
  selector:
    matchLabels:
      app: controller.csi.trident.netapp.io
  namespaceSelector:
    matchNames:
      - trident
  endpoints:
    - interval: 15s
      path: /metrics
      port: https-metrics
      scheme: https
      tlsConfig:
        ca:
          secret:
            key: caCert
            name: trident-csi
        cert:
          secret:
            key: clientCert
            name: trident-csi
        keySecret:
          key: clientKey
          name: trident-csi
        serverName: trident-csi
```

Trident supporta metriche HTTPS in tutti i metodi di installazione: tridentctl, Helm chart e Operator:

- Se si utilizza il `tridentctl install` comando, è possibile passare il `--https-metrics` flag per abilitare le metriche HTTPS.
- Se si utilizza il grafico Helm, è possibile impostare il parametro `httpsMetrics` per abilitare le metriche HTTPS.
- Se si utilizzano file YAML, è possibile aggiungere il `--https_metrics` flag al `trident-main` container nel `trident-deployment.yaml` file.

Passaggio 3: interrogare le metriche di Trident con PromQL

PromQL è utile per creare espressioni che restituiscono dati in serie temporali o tabulari.

Ecco alcune query PromQL che puoi utilizzare:

Ottieni informazioni sullo stato di salute di Trident

- **Percentuale di risposte HTTP 2XX da Trident**

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

- **Percentuale di risposte REST da Trident tramite codice di stato**

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

- **Durata media in ms delle operazioni eseguite da Trident**

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

Ottieni informazioni sull'utilizzo di Trident

- **Dimensione media del volume**

```
trident_volume_allocated_bytes/trident_volume_count
```

- **Spazio totale del volume fornito da ciascun backend**

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

Ottieni l'utilizzo del volume individuale

Questa opzione è abilitata solo se vengono raccolte anche le metriche kubelet.

- **Percentuale di spazio utilizzato per ciascun volume**

```
kubelet_volume_stats_used_bytes / kubelet_volume_stats_capacity_bytes *
100
```

Scopri la telemetria Trident AutoSupport

Per impostazione predefinita, Trident invia metriche Prometheus e le informazioni di base del backend a NetApp su base giornaliera.

- Per impedire a Trident di inviare metriche Prometheus e informazioni di base sul backend a NetApp, passare il `--silence-autosupport` flag durante l'installazione di Trident.
- Trident può anche inviare i log dei container a NetApp Support su richiesta tramite `tridentctl send autosupport`. Sarà necessario attivare Trident per caricare i suoi log. Prima di inviare i log, è necessario accettare le NetApp "informativa sulla privacy".
- Se non diversamente specificato, Trident recupera i log delle ultime 24 ore.
- È possibile specificare il periodo di conservazione del log con il `--since` flag. Ad esempio: `tridentctl send autosupport --since=1h`. Queste informazioni vengono raccolte e inviate tramite un `trident-autosupport` container che viene installato insieme a Trident. È possibile ottenere l'immagine del container su "Trident AutoSupport".
- Trident AutoSupport non raccoglie né trasmette Personally Identifiable Information (PII) o Personal Information. Viene fornito con un "EULA" che non è applicabile all'immagine del container Trident stessa. Puoi saperne di più sull'impegno di NetApp per la sicurezza e la fiducia dei dati "qui".

Un esempio di payload inviato da Trident si presenta così:

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

- I messaggi AutoSupport vengono inviati all'endpoint AutoSupport di NetApp. Se stai utilizzando un registro privato per memorizzare le immagini container, puoi utilizzare il flag `--image-registry`.
- È anche possibile configurare gli URL proxy generando i file YAML di installazione. Questo può essere fatto utilizzando `tridentctl install --generate-custom-yaml` per creare i file YAML e aggiungendo l' `--proxy-url` argomento per il `trident-autosupport` container in `trident-`

`deployment.yaml`.

Disabilita metriche Trident

Per **disabilitare** le metriche dalla segnalazione, è necessario generare file YAML personalizzati (utilizzando il `--generate-custom-yaml` flag) e modificarli per rimuovere il `--metrics` flag dall'essere invocato per il `trident-main` container.

Disinstalla Trident

Dovresti usare lo stesso metodo per disinstallare Trident che hai usato per installare Trident.

Informazioni su questa attività

- Se hai bisogno di una correzione per bug riscontrati dopo un aggiornamento, problemi di dipendenza o un aggiornamento non riuscito o incompleto, dovresti disinstallare Trident e reinstallare la versione precedente seguendo le istruzioni specifiche per quella versione. Questo è l'unico metodo consigliato per effettuare il *downgrade* a una versione precedente.
- Per facilitare l'aggiornamento e la reinstallazione, la disinstallazione di Trident non rimuove i CRD o gli oggetti correlati creati da Trident. Se è necessario rimuovere completamente Trident e tutti i suoi dati, fare riferimento a "Rimuovere completamente Trident e CRDs".

Prima di iniziare

Se si stanno dismettendo i cluster Kubernetes, è necessario eliminare tutte le applicazioni che utilizzano volumi creati da Trident prima della disinstallazione. Questo garantisce che le PVC siano annullate sui nodi Kubernetes prima di essere eliminate.

Determinare il metodo di installazione originale

Dovresti usare lo stesso metodo per disinstallare Trident che hai usato per installarlo. Prima di disinstallare, verifica quale versione hai usato per installare originariamente Trident.

1. Usa `kubectl get pods -n trident` per esaminare i pod.
   - Se non è presente alcun pod operatore, Trident è stato installato usando `tridentctl`.
   - Se è presente un pod operatore, Trident è stato installato utilizzando l'operatore Trident manualmente o tramite Helm.
2. Se è presente un pod operatore, utilizzare `kubectl describe tproc trident` per determinare se Trident è stato installato tramite Helm.
   - Se è presente un'etichetta Helm, Trident è stato installato utilizzando Helm.
   - Se non è presente alcuna etichetta Helm, Trident è stato installato manualmente utilizzando l'operatore Trident.

Disinstallare un'installazione dell'operatore Trident

Puoi disinstallare un'installazione dell'operatore Trident manualmente o usando Helm.

Disinstalla installazione manuale

Se hai installato Trident usando l'operatore, puoi disinstallarlo eseguendo una delle seguenti operazioni:

1. **Modifica `TridentOrchestrator` CR e imposta il flag di disinstallazione**:

   ```
   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. Dovresti pulire il TridentOrchestrator e crearne uno nuovo se vuoi installare nuovamente Trident.

2. **Elimina `TridentOrchestrator`**: Rimuovendo la `TridentOrchestrator` CR utilizzata per distribuire Trident, si chiede all'operatore di disinstallare Trident. L'operatore elabora la rimozione di `TridentOrchestrator` e procede a rimuovere la distribuzione e il daemonset di Trident, eliminando i pod Trident creati durante l'installazione.

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

Disinstallare l'installazione di Helm

Se hai installato Trident tramite Helm, puoi disinstallarlo usando `helm uninstall`.

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

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

Disinstalla un' `tridentctl`installazione

Utilizzare il `uninstall` comando in `tridentctl` per rimuovere tutte le risorse associate a Trident, ad eccezione dei CRD e degli oggetti correlati:

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