Gestire e monitorare Trident

Trident

NetApp
January 14, 2026

Sommario

Gestire e monitorare Trident

Upgrade Trident (Aggiorna server)

Upgrade Trident (Aggiorna server)

A partire dalla release 24,02, Trident segue una cadenza di quattro mesi, fornendo tre release principali ogni anno solare. Ogni nuova release si basa sulle release precedenti e offre nuove funzionalità, miglioramenti delle prestazioni, correzioni di bug e miglioramenti. Vi consigliamo di effettuare l'aggiornamento almeno una volta all'anno per usufruire delle nuove funzioni di Trident.

Considerazioni prima dell'aggiornamento

Quando si effettua l'aggiornamento alla versione più recente di Trident, tenere presente quanto segue:

- Dovrebbe essere installata una sola istanza di Trident in tutti gli spazi dei nomi di un determinato cluster Kubernetes.
- Trident 23,07 e versioni successive richiedono snapshot di volume v1 e non supportano più snapshot alfa o beta.
- Se è stato creato Cloud Volumes Service per Google Cloud in "Tipo di servizio CVS", è necessario aggiornare la configurazione backend per utilizzare il `standardsw` livello di servizio o `zoneredundantstandardsw` durante l'aggiornamento da Trident 23,01. Il mancato aggiornamento di `serviceLevel` nel backend potrebbe causare un errore dei volumi. Per ulteriori informazioni, fare riferimento alla "Esempi di tipo di servizio CVS" sezione.
- Quando si esegue l'aggiornamento, è importante fornire `parameter.fsType` in `StorageClasses` usato da Trident. Puoi eliminare e ricreare `StorageClasses` senza interrompere i volumi preesistenti.
  - Si tratta di un **requisito** per l'applicazione "contesti di sicurezza" di volumi SAN.
  - La directory sample input contiene esempi, come https://github.com/NetApp/Trident/blob/master/Trident-installer/sample-input/storage-class-samples/storage-class-Basic.yaml.templ ^] e link:https://github.com/NetApp/Trident/blob/master/Trident-installer/samples/storage-class-[`storage-class-bronze-default.yaml`yaml[`storage-class-basic.yaml.templ`
  - Per ulteriori informazioni, fare riferimento a "Problemi noti".

Fase 1: Selezionare una versione

Le versioni Trident seguono una convenzione di denominazione basata sulla data `YY.MM`, dove "YY" è l'ultima cifra dell'anno e "MM" è il mese. I rilasci di DOT seguono una `YY.MM.X` convenzione, dove "X" è il livello di patch. Selezionare la versione a cui eseguire l'aggiornamento in base alla versione da cui si sta eseguendo l'aggiornamento.

- È possibile eseguire un aggiornamento diretto a qualsiasi release di destinazione che si trova all'interno di una finestra di quattro release della versione installata. Ad esempio, è possibile aggiornare direttamente da 23,04 (o qualsiasi versione a 23,04 punti) a 24,06.
- Se si sta eseguendo l'aggiornamento da una release al di fuori della finestra a quattro release, eseguire un aggiornamento in più fasi. Utilizzare le istruzioni di aggiornamento per il "versione precedente" quale si sta eseguendo l'aggiornamento per passare alla versione più recente adatta alla finestra a quattro release. Ad esempio, se si utilizza 22,01 e si desidera eseguire l'aggiornamento a 24,06:

a. Primo aggiornamento da 22,07 a 23,04.

b. Quindi, eseguire l'aggiornamento da 23,04 a 24,06.

> Quando si esegue l'aggiornamento utilizzando l'operatore Trident su OpenShift Container Platform, è necessario eseguire l'aggiornamento a Trident 21.01.1 o versione successiva. L'operatore Trident rilasciato con 21.01.0 contiene un problema noto che è stato risolto nel 21.01.1. Per ulteriori informazioni, fare riferimento alla "Dettagli del problema su GitHub" .

Fase 2: Determinare il metodo di installazione originale

Per determinare quale versione è stata utilizzata per l'installazione originale di Trident:

1. Utilizzare `kubectl get pods -n trident` per esaminare i pod.
   - Se non è presente alcun pannello operatore, Trident è stato installato utilizzando `tridentctl`.
   - Se è presente un quadro di comando, Trident è stato installato utilizzando l'operatore Trident manualmente o utilizzando Helm.
2. Se è presente un pannello operatore, utilizzare `kubectl describe torc` per determinare se Trident è stato installato utilizzando 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: Selezionare un metodo di aggiornamento

In genere, è necessario eseguire l'aggiornamento utilizzando lo stesso metodo utilizzato per l'installazione iniziale, tuttavia è possibile "passare da un metodo di installazione all'altro". Sono disponibili due opzioni per aggiornare Trident.

- "Eseguire l'aggiornamento utilizzando l'operatore Trident"

> Si consiglia di eseguire la revisione "Comprendere il flusso di lavoro di aggiornamento dell'operatore" prima di effettuare l'aggiornamento con l'operatore.

*

Eseguire l'upgrade con l'operatore

Comprendere il flusso di lavoro di aggiornamento dell'operatore

Prima di utilizzare l'operatore Trident per aggiornare Trident, è necessario comprendere i processi in background che si verificano durante l'aggiornamento. Sono incluse le modifiche al controller Trident, al pod dei controller e ai pod dei nodi e al daemonSet dei nodi che consentono l'esecuzione degli aggiornamenti.

Gestione dell'aggiornamento dell'operatore Trident

Uno dei molti "Vantaggi dell'utilizzo dell'operatore Trident" da installare e aggiornare Trident è la gestione automatica degli oggetti Trident e Kubernetes senza interrompere i volumi montati esistenti. In questo modo, Trident è in grado di supportare gli aggiornamenti senza tempi di inattività, oppure "*rolling updates*". In

particolare, l'operatore Trident comunica con il cluster Kubernetes per:

- Eliminare e ricreare l'implementazione del controller Trident e il daemonSet del nodo.
- Sostituisci il Controller Pod Trident e i pod di nodi Trident con nuove versioni.
  - Se un nodo non viene aggiornato, non impedisce l'aggiornamento dei nodi rimanenti.
  - Solo i nodi con un pod nodo Trident in esecuzione possono montare volumi.

Per ulteriori informazioni sull'architettura Trident nel cluster Kubernetes, fare riferimento a "Architettura 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 CRD, RBAC e Trident SVC.
   c. Elimina l'implementazione del controller Trident per la versione *n*.
   d. Crea l'implementazione del controller Trident per la versione *n+1*.
2. **Kubernetes** crea il Pod controller Trident per *n+1*.
3. L'operatore **Trident**:
   a. Elimina il daemonSet del nodo Trident per *n*. L'operatore non attende la terminazione del nodo Pod.
   b. Crea il nodo Trident Daemonset per *n+1*.
4. **Kubernetes** crea pod di nodi Trident sui nodi che non eseguono il pod di nodi Trident *n*. In questo modo, si garantisce che non ci sia mai più di un Pod nodi Trident, di qualsiasi versione, su un nodo.

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

È possibile aggiornare Trident utilizzando l'operatore Trident manualmente o utilizzando Helm. È possibile eseguire l'aggiornamento da un'installazione dell'operatore Trident a un'altra installazione dell'operatore Trident o da un `tridentctl'installazione a una versione dell'operatore Trident. Prima di aggiornare l'installazione di un operatore Trident, rivedere la "Selezionare un metodo di aggiornamento"sezione.

Aggiornare un'installazione manuale

È possibile eseguire l'aggiornamento da un'installazione dell'operatore Trident definita dall'ambito del cluster a un'altra installazione dell'operatore Trident definita dal cluster. Tutte le versioni 21,01 e successive di Trident utilizzano un operatore cluster-scoped.

Per eseguire l'aggiornamento da Trident installato utilizzando l'operatore con spazio dei nomi (versioni da 20,07 a 20,10), utilizza le istruzioni di aggiornamento di "versione installata"Trident.

A proposito di questa attività

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

- Per i cluster che eseguono Kubernetes 1,24, utilizzare "bundle_pre_1_25.yaml".
- Per i cluster che eseguono Kubernetes 1,25 o versione successiva, utilizzare "bundle_post_1_25.yaml".

Prima di iniziare

Assicurarsi di utilizzare un cluster Kubernetes in esecuzione "Una versione di Kubernetes supportata".

Fasi

1. Verificare la versione di Trident:

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

2. Eliminare l'operatore Trident utilizzato per installare l'istanza Trident corrente. Ad esempio, se si sta eseguendo l'aggiornamento da 23,07, eseguire il seguente comando:

```
kubectl delete -f 23.07.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

3. Se l'installazione iniziale è stata personalizzata utilizzando `TridentOrchestrator` gli attributi, è possibile modificare l' `TridentOrchestrator`oggetto per modificare i parametri di installazione. Ciò potrebbe includere le modifiche apportate per specificare i registri di immagini Trident e CSI mirrorati per la modalità offline, abilitare i registri di debug o specificare i segreti di pull delle immagini.
4. Installa Trident usando il file YAML del bundle corretto per il tuo ambiente, dove <*bundle.yaml*> si trova `bundle_pre_1_25.yaml` o `bundle_post_1_25.yaml` si basa sulla tua versione di Kubernetes. Ad esempio, se si sta installando Trident 24,10, eseguire il seguente comando:

```
kubectl create -f 24.10.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

Aggiornare un'installazione Helm

È possibile aggiornare un'installazione di Trident Helm.

> Quando si aggiorna un cluster Kubernetes da 1,24 a 1,25 o versione successiva su `true` cui è installato Trident, è necessario aggiornare Values.yaml per impostarlo `excludePodSecurityPolicy` o aggiungerlo `--set excludePodSecurityPolicy=true` al `helm upgrade` comando prima di poter aggiornare il cluster.

Se hai già aggiornato il tuo cluster Kubernetes dalla 1,24 alla 1,25 senza aggiornare il timone Trident, l'aggiornamento del timone non riuscirà. Per eseguire l'aggiornamento del timone, eseguire questi passaggi come prerequisiti:

1. Installare il plugin helm-mapkubeapis da https://github.com/helm/helm-mapkubeapis.
2. Eseguire un ciclo di asciugatura per la release Trident nello spazio dei nomi in cui è installato Trident. In questo modo vengono elencate le risorse che verranno ripulite.

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

3. Eseguire una corsa completa con il timone per eseguire la pulizia.

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

Fasi

1. Se si "Installato Trident utilizzando Helm"utilizza , è possibile utilizzare `helm upgrade trident netapp-trident/trident-operator --version 100.2410.0` per eseguire l'aggiornamento in un solo passaggio. Se non è stato aggiunto il repo Helm o non è possibile utilizzarlo per l'aggiornamento:
   a. Scaricare la versione più recente di Trident dal sito "La sezione *Assets* su GitHub".
   b. Utilizzare il `helm upgrade` comando dove riflette la versione a cui `trident-operator-24.10.0.tgz` si desidera eseguire l'aggiornamento.

```
helm upgrade <name> trident-operator-24.10.0.tgz
```

> Se si impostano opzioni personalizzate durante l'installazione iniziale (ad esempio specificando registri privati e speculari per le immagini Trident e CSI), aggiungere il `helm upgrade` comando utilizzando `--set` per assicurarsi che tali opzioni siano incluse nel comando di aggiornamento, altrimenti i valori verranno ripristinati ai valori predefiniti.

2. Eseguire `helm list` per verificare che la versione di carta e app sia stata aggiornata. Esegui `tridentctl logs` per esaminare eventuali messaggi di debug.

Aggiornamento da un'installazione a un `tridentctl` operatore Trident

È possibile eseguire l'aggiornamento alla versione più recente 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, vedere "Passaggio da un metodo di installazione all'altro".

Fasi

1. Scarica la versione più recente di Trident.

```
# Download the release required [24.10.0]
mkdir 24.10.0
cd 24.10.0
wget
https://github.com/NetApp/trident/releases/download/v24.10.0/trident-
installer-24.10.0.tar.gz
tar -xf trident-installer-24.10.0.tar.gz
cd trident-installer
```

2. Creare il `tridentorchestrator` CRD dal manifesto.

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

3. Implementare l'operatore 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. Creare una `TridentOrchestrator` CR per l'installazione di 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. Confermare che Trident è stato aggiornato alla versione prevista.

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

Message:                Trident installed
Namespace:              trident
Status:                 Installed
Version:                v24.10.0
```

Upgrade con tridentctl

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

A proposito di questa attività

La disinstallazione e la reinstallazione di Trident funge da aggiornamento. Quando si disinstalla Trident, il PVC (Persistent Volume Claim) e il PV (Persistent Volume Claim) utilizzati dall'implementazione Trident non vengono eliminati. I PVC che sono già stati sottoposti a provisioning rimarranno disponibili mentre Trident è offline, e Trident eseguirà il provisioning dei volumi per qualsiasi PVC creato nel frattempo dopo il ritorno online.

Prima di iniziare

"Selezionare un metodo di aggiornamento"Prima di eseguire l'aggiornamento mediante `tridentctl`.

Fasi

1. Eseguire 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 alla "Installare Trident usando tridentctl".

> Non interrompere il processo di aggiornamento. Assicurarsi che il programma di installazione venga completato.

Gestisci Trident usando tridentctl

`https://github.com/NetApp/trident/releases["Pacchetto di installazione Trident"^]Include l' `tridentctl`utilità della riga di comando per fornire un semplice accesso a Trident. Gli utenti Kubernetes con Privileges sufficiente possono usarlo per installare Trident o gestire il namespace che contiene l'pod Trident.`

Comandi e flag globali

Si può eseguire `tridentctl help` per ottenere un elenco di comandi disponibili per o aggiungere il `--help` flag a qualsiasi comando per `tridentctl` 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`
Aggiungere una risorsa a Trident.

`delete`
Rimuovere una o più risorse da Trident.

`get`
Ottieni una o più risorse da Trident.

`help`
Aiuto su qualsiasi comando.

`images`
Stampare una tabella delle immagini contenitore richieste da Trident.

`import`
Importare una risorsa esistente in Trident.

`install`
Installare Trident.

`logs`
Stampare i registri da Trident.

`send`
Inviare una risorsa da Trident.

`uninstall`
Disinstallare Trident.

`update`
Modificare una risorsa in Trident.

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

`upgrade`
Aggiornare una risorsa in Trident.

`version`
Stampare la versione di Trident.

Flag globali

`-d, --debug`
Output di debug.

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

`-k, --kubeconfig string`
Specifica il `KUBECONFIG` percorso per eseguire comandi in locale o da un cluster Kubernetes a un altro.

In alternativa, puoi esportare la `KUBECONFIG` variabile in modo da puntare a un cluster Kubernetes specifico ed emettere `tridentctl` comandi a quel cluster.

`-n, --namespace string`
Namespace delle implementazioni Trident.

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

`-s, --server string`
Indirizzo/porta dell'interfaccia REST 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).

Opzioni di comando e flag

creare

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

```
tridentctl create [option]
```

Opzioni

`backend`: Aggiungere un backend a Trident.

eliminare

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

```
tridentctl delete [option]
```

Opzioni

`backend`: Eliminare uno o più backend di archiviazione da Trident.
`snapshot`: Eliminare uno o più snapshot di volume da Trident.
`storageclass`: Eliminare una o più classi di archiviazione da Trident.

`volume`: Eliminare uno o più volumi di archiviazione da Trident.

ottieni

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

```
tridentctl get [option]
```

Opzioni

`backend`: Ottenere uno o più backend di archiviazione da Trident.
`snapshot`: Ottenere uno o più snapshot da Trident.
`storageclass`: Ottenere una o più classi di archiviazione da Trident.
`volume`: Ottenere uno o più volumi da Trident.

Allarmi

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

immagini

Utilizzare `images` i flag per stampare una tabella delle immagini contenitore richieste da Trident.

```
tridentctl images [flags]
```

Allarmi

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

importa volume

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

```
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`: Creare solo PV/PVC. Non presupporre la gestione del ciclo di vita dei volumi.

installare

Utilizzare i `install` flag per installare Trident.

```
tridentctl install [flags]
```

Allarmi

`--autosupport-image string`: L'immagine contenitore per la telemetria AutoSupport (predefinita "NetApp/Trident AutoSupport:<current-version>").
`--autosupport-proxy string`: L'indirizzo/porta di un proxy per l'invio della telemetria AutoSupport.
`--enable-node-prep`: Tentativo di installare i pacchetti richiesti sui nodi.
`--generate-custom-yaml`: Generare file YAML senza installare nulla.
`-h, --help`: Guida per l'installazione.
`--http-request-timeout`: Ignorare il timeout della richiesta HTTP per l'API REST del controller Trident (valore predefinito 1m30).
`--image-registry string`: L'indirizzo/porta di un registro interno dell'immagine.
`--k8s-timeout duration`: Il timeout per tutte le operazioni Kubernetes (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 Trident (text, json) (default "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.**
`--pv string`: il nome del PV esistente utilizzato da Trident, garantisce che non esista (default "Trident").
`--pvc string`: Il nome del PVC legacy utilizzato da Trident, garantisce che non esista (default "Trident").
`--silence-autosupport`: Non inviare pacchetti AutoSupport a NetApp automaticamente (default true).
`--silent`: Consente di disattivare la maggior parte dell'output durante l'installazione.
`--trident-image string`: L'immagine Trident da installare.
`--use-custom-yaml`: Utilizzare i file YAML esistenti nella directory di installazione.
`--use-ipv6`: Utilizzare IPv6 per la comunicazione di Trident.

registri

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

```
tridentctl logs [flags]
```

Allarmi

`-a, --archive`: Creare un archivio di supporto con tutti i registri, se non diversamente specificato.
`-h, --help`: Guida per i registri.
`-l, --log string`: Registro Trident da visualizzare. Uno di 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`: Ottiene i log per l'istanza contenitore precedente, se esiste.
`--sidecars`: Ottenere i tronchi per i contenitori del sidecar.

invia

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

```
tridentctl send [option]
```

Opzioni

`autosupport`: Inviare un archivio AutoSupport a NetApp.

disinstallazione

Utilizzare `uninstall` i flag per disinstallare Trident.

```
tridentctl uninstall [flags]
```

Allarmi

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

aggiornamento

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

```
tridentctl update [option]
```

Opzioni

`backend`: Aggiornare un backend in Trident.

aggiorna stato backend

Utilizzare il `update backend state` comando 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), non è possibile aggiornare il backend utilizzando un `backend.json` file.
- Se il `userState` è stato impostato in un tbc, non può essere modificato utilizzando il `tridentctl update backend state <backend-name> --user-state suspended/normal` comando .
- Per recuperare la capacità di impostare il `userState` tridentctl via dopo che è stato impostato tramite tbc, il `userState` campo deve essere rimosso dal tbc. Questo può essere fatto usando il `kubectl edit tbc` comando. Una volta rimosso il `userState` campo, è possibile utilizzare il `tridentctl update backend state` comando per modificare il `userState` di un backend.
- Utilizzare il `tridentctl update backend state` per modificare il `userState`. È anche possibile aggiornare il `userState` file Using `TridentBackendConfig` o `backend.json` ; questo attiva una reinizializzazione completa del backend e può richiedere molto tempo.

  #### Allarmi

  `-h, --help`: Guida per lo stato backend.
  `--user-state`: Impostare su `suspended` per sospendere le operazioni di backend. Impostare su `normal` per riprendere le operazioni di backend. Quando impostato su `suspended`:

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

È inoltre possibile aggiornare lo stato backend utilizzando il `userState` campo nel file di configurazione backend `TridentBackendConfig` o `backend.json`. Per ulteriori informazioni, fare riferimento a "Opzioni per la gestione dei backend" e "Eseguire la gestione del back-end con kubectl".

Esempio:

JSON

Per aggiornare utilizzando il file, procedere come segue `userState` `backend.json` :

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

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

```
{
    "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 `kubectl edit <tbc-name> -n <namespace>` comando . Nell'esempio riportato di seguito viene aggiornato lo stato backend per la sospensione mediante l' `userState: suspended` opzione:

```
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]
```

Allarmi

`--client`: Solo versione client (non è richiesto alcun server).
`-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 di ambiente PATH. Tutti i plugin rilevati sono elencati nella sezione dei plugin della guida tridentctl. In alternativa, è possibile limitare la ricerca specificando una cartella di plugin nella variabile Environment TRIDENTCTL_PLUGIN_PATH (esempio: `TRIDENTCTL_PLUGIN_PATH=~/tridentctl-plugins/`). Se si utilizza la variabile, tridenctl ricerca solo nella cartella specificata.

Monitor Trident

Trident fornisce un set di endpoint di misurazione Prometheus che è possibile utilizzare per monitorare le prestazioni Trident.

Panoramica

Le metriche fornite da Trident consentono di:

- Tenere sotto controllo lo stato di salute e la configurazione di Trident. È possibile esaminare il successo delle operazioni e se è in grado di comunicare con i back-end come previsto.
- Esaminare le informazioni sull'utilizzo del back-end e comprendere il numero di volumi sottoposti a provisioning su un back-end, la quantità di spazio consumato e così via.
- Mantenere una mappatura della quantità di volumi forniti sui backend disponibili.
- Tenere traccia delle performance. È possibile esaminare il tempo necessario a Trident per comunicare con i backend ed eseguire le operazioni.

> Per impostazione predefinita, le metriche di Trident sono esposte sulla porta di destinazione `8001` all' `/metrics`endpoint. Queste metriche sono **abilitate per impostazione predefinita** quando Trident è installato.

Di cosa hai bisogno

- Un cluster Kubernetes con Trident installato.
- Un'istanza Prometheus. Questo può essere un "Implementazione di Prometheus in container" o si può scegliere di eseguire Prometheus come un "applicazione nativa".

Fase 1: Definire un target Prometheus

È necessario definire un target Prometheus per raccogliere le metriche e ottenere informazioni sui backend gestiti da Trident, sui volumi creati e così via. Questo "blog" spiega come utilizzare Prometheus e Grafana con Trident per recuperare le metriche. Il blog spiega come eseguire Prometheus come operatore nel cluster

Kubernetes e la creazione di un ServiceMonitor per ottenere le metriche Trident.

Fase 2: Creazione di un ServiceMonitor Prometheus

Per utilizzare le metriche Trident, è necessario creare un Prometheus ServiceMonitor che controlla il `trident-csi` servizio e ascolta sulla `metrics` porta. Un esempio di ServiceMonitor è simile al 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 ricerca specificamente l' `metrics`endpoint 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 di misurazione. Kubelet può fornire informazioni sui volumi collegati, sui pod e sulle altre operazioni interne gestite. Fare riferimento alla "qui".

Fase 3: Eseguire una query sulle metriche di Trident con PromQL

PromQL è utile per la creazione di espressioni che restituiscono dati di serie temporali o tabulari.

Di seguito sono riportate alcune query PromQL che è possibile utilizzare:

Ottieni informazioni sulla 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 A RIPOSO 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"})
```

Ottenere 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 di singoli volumi

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

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

```
kubelet_volume_stats_used_bytes / kubelet_volume_stats_capacity_bytes *
100
```

Ulteriori informazioni sulla telemetria di Trident AutoSupport

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

- Per impedire a Trident di inviare metriche Prometheus e informazioni di base di backend a NetApp, passare il `--silence-autosupport` flag durante l'installazione di Trident.
- Trident può inoltre inviare i log dei container al supporto NetApp on-demand tramite `tridentctl send autosupport`. Sarà necessario attivare Trident per caricare i suoi registri. Prima di inviare i log, è necessario accettare NetApp "direttiva sulla privacy".
- Se non specificato, Trident recupera i registri dalle ultime 24 ore.

- È possibile specificare il periodo di conservazione del registro con il `--since` flag. Ad esempio: `tridentctl send autosupport --since=1h`. Queste informazioni vengono raccolte e inviate tramite un `trident-autosupport` contenitore installato insieme a Trident. È possibile ottenere l'immagine contenitore in "Trident AutoSupport".
- Trident AutoSupport non raccoglie né trasmette dati personali o di identificazione personale (PII). Viene fornito con un "EULA" che non è applicabile all'immagine contenitore Trident stessa. Puoi saperne di più sull'impegno di NetApp nei confronti della sicurezza e della fiducia dei dati "qui".

Un esempio di payload inviato da Trident è simile al seguente:

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

- I messaggi AutoSupport vengono inviati all'endpoint AutoSupport di NetApp. Se si utilizza un registro privato per memorizzare immagini contenitore, è possibile utilizzare il `--image-registry` flag.
- È inoltre possibile configurare gli URL proxy generando i file YAML di installazione. A tale scopo, è possibile utilizzare `tridentctl install --generate-custom-yaml` per creare i file YAML e aggiungere l' `--proxy-url`argomento per il `trident-autosupport` contenitore in `trident-deployment.yaml`.

Disattiva metriche Trident

Per disabilitare** le metriche da riportare, è necessario generare YAML personalizzati (utilizzando il `--generate-custom-yaml` flag) e modificarli per rimuovere il `--metrics` flag da richiamare per il `trident-main` contenitore.

Disinstallare Trident

Utilizzare lo stesso metodo per disinstallare Trident utilizzato per installare Trident.

A proposito di questa attività

- Se è necessaria una correzione per i bug osservati dopo un aggiornamento, problemi di dipendenza o un aggiornamento non riuscito o incompleto, è necessario disinstallare Trident e reinstallare la versione precedente utilizzando le istruzioni specifiche per tale aggiornamento"versione". Questo è l'unico modo

consigliato per *eseguire il downgrade* a una versione precedente.

- Per semplificare 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 relativi dati, fare riferimento alla sezione "Rimuovere completamente Trident e CRD".

Prima di iniziare

Se stai decommissionando i cluster Kubernetes, devi eliminare tutte le applicazioni che utilizzano i volumi creati da Trident prima della disinstallazione. In questo modo, si garantisce che i PVC non siano pubblicati sui nodi Kubernetes prima di essere eliminati.

Determinare il metodo di installazione originale

Utilizzare lo stesso metodo per disinstallare Trident utilizzato per installarlo. Prima di disinstallare, verificare quale versione è stata utilizzata per installare Trident in origine.

1. Utilizzare `kubectl get pods -n trident` per esaminare i pod.
   - Se non è presente alcun pannello operatore, Trident è stato installato utilizzando `tridentctl`.
   - Se è presente un quadro di comando, Trident è stato installato utilizzando l'operatore Trident manualmente o utilizzando Helm.
2. Se è presente un pannello operatore, utilizzare `kubectl describe tproc trident` per determinare se Trident è stato installato utilizzando 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

È possibile disinstallare manualmente un'installazione dell'operatore tridente o utilizzando Helm.

Disinstallare l'installazione manuale

Se Trident è stato installato utilizzando l'operatore, è possibile disinstallarlo effettuando 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 lo stesso TridentOrchestrator. Se si desidera installare di nuovo Trident, è necessario ripulire TridentOrchestrator e crearne uno nuovo.

2. **Elimina `TridentOrchestrator`**: Rimuovendo il `TridentOrchestrator` CR utilizzato per distribuire Trident, si istruisce l'operatore a disinstallare Trident. L'operatore elabora la rimozione `TridentOrchestrator` e procede alla rimozione della distribuzione Trident e del daemonset, eliminando i pod Trident creati durante l'installazione.

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

Disinstallare l'installazione di Helm

Se Trident è stato installato utilizzando Helm, è possibile disinstallarlo utilizzando `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
```

Disinstallare 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>
```

Informazioni sul copyright

Informazioni sul marchio commerciale