Installare utilizzando l'operatore Trident

Trident

NetApp
November 03, 2026

This PDF was generated from https://docs.netapp.com/it-it/trident-2410/trident-install/kubernetes-deploy-operator.html on November 03, 2026. Always check docs.netapp.com for the latest.

Sommario

Installare utilizzando l'operatore Trident . . . 1
- Implementare manualmente l'operatore Trident (modalità standard) . . . 1
  - Informazioni critiche su Trident 24,10 . . . 1
  - Implementare manualmente l'operatore Trident e installare Trident . . . 1
  - Verificare l'installazione . . . 4
- Implementare manualmente l'operatore Trident (modalità offline) . . . 6
  - Informazioni critiche su Trident 24,10 . . . 6
  - Implementare manualmente l'operatore Trident e installare Trident . . . 6
  - Verificare l'installazione . . . 10
- Implementare l'operatore Trident utilizzando Helm (modalità standard) . . . 11
  - Informazioni critiche su Trident 24,10 . . . 11
  - Distribuire l'operatore Trident e installare Trident utilizzando Helm . . . 12
  - Passare i dati di configurazione durante l'installazione . . . 12
  - Opzioni di configurazione . . . 13
- Implementare l'operatore Trident utilizzando Helm (modalità offline) . . . 17
  - Informazioni critiche su Trident 24,10 . . . 17
  - Distribuire l'operatore Trident e installare Trident utilizzando Helm . . . 17
  - Passare i dati di configurazione durante l'installazione . . . 18
  - Opzioni di configurazione . . . 19
- Personalizzare l'installazione dell'operatore Trident . . . 23
  - Comprensione dei pod controller e dei pod di nodi . . . 23
  - Opzioni di configurazione . . . 23
  - Configurazioni di esempio . . . 26

Installare utilizzando l'operatore Trident

Implementare manualmente l'operatore Trident (modalità standard)

È possibile distribuire manualmente l'operatore Trident per installare Trident. Questo processo si applica alle installazioni in cui le immagini contenitore richieste da Trident non sono memorizzate in un registro privato. Se si dispone di un registro delle immagini privato, utilizzare "processo per l'implementazione offline".

Informazioni critiche su Trident 24,10

È necessario leggere le seguenti informazioni critiche su Trident.

**▼ informazioni aggiornate su Trident **

- Kubernetes 1,32 è ora supportato in Trident. Aggiornare Trident prima di aggiornare Kubernetes.
- Trident impone rigorosamente l'uso della configurazione multipath negli ambienti SAN, con un valore consigliato di `find_multipaths: no` nel file multipath.conf.

  L'utilizzo di una configurazione non multipathing o di `find_multipaths: yes` un valore OR `find_multipaths: smart` nel file multipath.conf determinerà errori di montaggio. Trident ha consigliato l'uso di `find_multipaths: no` fin dalla versione 21,07.

Implementare manualmente l'operatore Trident e installare Trident

Verificate "panoramica dell'installazione" per assicurarvi di aver soddisfatto i prerequisiti di installazione e di aver selezionato l'opzione di installazione corretta per il vostro ambiente.

Prima di iniziare

Prima di iniziare l'installazione, accedi all'host Linux e verifica che stia gestendo un "Cluster Kubernetes supportato" funzionante e che tu disponga dei privilegi necessari.

Con OpenShift, utilizzare `oc` invece di `kubectl` tutti gli esempi che seguono e accedere come **system:admin** eseguendo prima `oc login -u system:admin` o `oc login -u kube-admin`.

▼ Details

1. Verificare la versione di Kubernetes:

```
kubectl version
```

2. Verificare i privilegi di amministratore del cluster:

```
kubectl auth can-i '*' '*' --all-namespaces
```

3. Verificare che sia possibile avviare un pod che utilizza un'immagine da Docker Hub e raggiungere il sistema di storage tramite la rete pod:

```
kubectl run -i --tty ping --image=busybox --restart=Never --rm -- \
  ping <management IP>
```

Fase 1: Scaricare il pacchetto di installazione di Trident

Il pacchetto di installazione di Trident contiene tutto ciò che serve per distribuire l'operatore Trident e installare Trident. Scaricare ed estrarre la versione più recente del programma di installazione di Trident da "La sezione *Assets* su GitHub".

```
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
```

Fase 2: Creare il `TridentOrchestrator` CRD

Crea il `TridentOrchestrator` Definizione di risorse personalizzate (CRD). Creerai un `TridentOrchestrator` Risorsa personalizzata in seguito. Utilizzare la versione CRD YAML appropriata in `deploy/crds` per creare il `TridentOrchestrator` CRD.

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

Fase 3: Implementare l'operatore Trident

Il programma di installazione di Trident fornisce un file bundle che può essere utilizzato per installare l'operatore e creare oggetti associati. Il file bundle è un modo semplice per distribuire l'operatore e installare Trident utilizzando una configurazione predefinita.

- 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

- Per impostazione predefinita, il programma di installazione di Trident implementa l'operatore nello `trident` spazio dei nomi. Se lo `trident` spazio dei nomi non esiste, crearlo utilizzando:

```
kubectl apply -f deploy/namespace.yaml
```

- Per distribuire l'operatore in uno spazio dei nomi diverso dallo `trident` spazio dei nomi, aggiornare `serviceaccount.yaml` `clusterrolebinding.yaml` e `operator.yaml` generare il file bundle utilizzando `kustomization.yaml` .

a. Crea il `kustomization.yaml` usando il seguente comando dove *<bundle.yaml>* è `bundle_pre_1_25.yaml` o `bundle_post_1_25.yaml` si basa sulla tua versione di Kubernetes.

```
cp deploy/kustomization_<bundle.yaml> deploy/kustomization.yaml
```

b. Compila il bundle usando il seguente comando dove *<bundle.yaml>* si trova `bundle_pre_1_25.yaml` o `bundle_post_1_25.yaml` si basa sulla tua versione di Kubernetes.

```
kubectl kustomize deploy/ > deploy/<bundle.yaml>
```

Fasi

1. Creare le risorse e implementare l'operatore:

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

2. Verificare che l'operatore, l'implementazione e i replicaset siano stati creati.

```
kubectl get all -n <operator-namespace>
```

In un cluster Kubernetes dovrebbe esserci solo **un'istanza** dell'operatore. Non creare implementazioni multiple dell'operatore Trident.

Fase 4: Creare `TridentOrchestrator` e installare Trident

A questo punto è possibile creare `TridentOrchestrator` e installare Trident. Facoltativamente, è possibile "Personalizzare l'installazione di Trident" utilizzare gli attributi nella `TridentOrchestrator` specifica.

```
kubectl create -f deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created

kubectl describe torc trident

Name:         trident
Namespace:
Labels:       <none>
Annotations:  <none>
API Version:  trident.netapp.io/v1
Kind:         TridentOrchestrator
...
Spec:
  Debug:      true
  Namespace:  trident
```

```
  nodePrep:
  - iscsi
Status:
  Current Installation Params:
    IPv6:                      false
    Autosupport Hostname:
    Autosupport Image:         netapp/trident-autosupport:24.10
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                     true
    Image Pull Secrets:
    Image Registry:
    k8sTimeout:            30
    Kubelet Dir:           /var/lib/kubelet
    Log Format:            text
    Silence Autosupport:   false
    Trident Image:         netapp/trident:24.10.0
  Message:                   Trident installed  Namespace:
trident
  Status:                    Installed
  Version:                   v24.10.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

Verificare l'installazione

Esistono diversi modi per verificare l'installazione.

Uso dello `TridentOrchestrator` stato

Lo stato di `TridentOrchestrator` indica se l'installazione è riuscita e visualizza la versione di Trident installata. Durante l'installazione, lo stato di `TridentOrchestrator` cambia da `Installing` a `Installed`. Se si osserva `Failed` lo stato e l'operatore non è in grado di recuperare da solo, "controllare i registri".

Stato	Descrizione
Installazione in corso	L'operatore sta installando Trident utilizzando questo `TridentOrchestrator` CR.
Installato	Installazione di Trident riuscita.
Disinstallazione in corso	L'operatore sta disinstallando Trident, perché `spec.uninstall=true`.
Disinstallato	Trident è disinstallato.

Stato	Descrizione
Non riuscito	L'operatore non ha potuto installare, applicare patch, aggiornare o disinstallare Trident; l'operatore tenterà automaticamente di ripristinare da questo stato. Se lo stato persiste, è necessario eseguire la risoluzione dei problemi.
Aggiornamento in corso	L'operatore sta aggiornando un'installazione esistente.
Errore	`TridentOrchestrator`Non viene utilizzato. Un'altra esiste già.

Utilizzo dello stato di creazione del pod

È possibile verificare se l'installazione di Trident è stata completata controllando lo stato dei pod creati:

```
kubectl get pods -n trident

NAME                                  READY   STATUS    RESTARTS
AGE
trident-controller-7d466bf5c7-v4cpw   6/6     Running   0
1m
trident-node-linux-mr6zc              2/2     Running   0
1m
trident-node-linux-xrp7w              2/2     Running   0
1m
trident-node-linux-zh2jt              2/2     Running   0
1m
trident-operator-766f7b8658-ldzsv     1/1     Running   0
3m
```

Uso `tridentctl`

È possibile utilizzare `tridentctl` per controllare la versione di Trident installata.

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 24.10.0        | 24.10.0        |
+----------------+----------------+
```

Implementare manualmente l'operatore Trident (modalità offline)

È possibile distribuire manualmente l'operatore Trident per installare Trident. Questo processo si applica alle installazioni in cui le immagini contenitore richieste da Trident sono memorizzate in un registro privato. Se non si dispone di un registro delle immagini private, utilizzare "processo per l'implementazione standard".

Informazioni critiche su Trident 24,10

È necessario leggere le seguenti informazioni critiche su Trident.

**▼ informazioni aggiornate su Trident **

- Kubernetes 1,32 è ora supportato in Trident. Aggiornare Trident prima di aggiornare Kubernetes.
- Trident impone rigorosamente l'uso della configurazione multipath negli ambienti SAN, con un valore consigliato di `find_multipaths: no` nel file multipath.conf.

  L'utilizzo di una configurazione non multipathing o di `find_multipaths: yes` un valore OR `find_multipaths: smart` nel file multipath.conf determinerà errori di montaggio. Trident ha consigliato l'uso di `find_multipaths: no` fin dalla versione 21,07.

Implementare manualmente l'operatore Trident e installare Trident

Verificate "panoramica dell'installazione" per assicurarvi di aver soddisfatto i prerequisiti di installazione e di aver selezionato l'opzione di installazione corretta per il vostro ambiente.

Prima di iniziare

Accedi all'host Linux e verifica che stia gestendo un sistema operativo funzionante "Cluster Kubernetes supportato" e che tu disponga dei privilegi necessari.

Con OpenShift, utilizzare `oc` invece di `kubectl` tutti gli esempi che seguono e accedere come **system:admin** eseguendo prima `oc login -u system:admin` o `oc login -u kube-admin`.

▼ Details

1. Verificare la versione di Kubernetes:

```
kubectl version
```

2. Verificare i privilegi di amministratore del cluster:

```
kubectl auth can-i '*' '*' --all-namespaces
```

3. Verificare che sia possibile avviare un pod che utilizza un'immagine da Docker Hub e raggiungere il sistema di storage tramite la rete pod:

```
kubectl run -i --tty ping --image=busybox --restart=Never --rm -- \
  ping <management IP>
```

Fase 1: Scaricare il pacchetto di installazione di Trident

Il pacchetto di installazione di Trident contiene tutto ciò che serve per distribuire l'operatore Trident e installare Trident. Scaricare ed estrarre la versione più recente del programma di installazione di Trident da "La sezione *Assets* su GitHub".

```
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
```

Fase 2: Creare il `TridentOrchestrator` CRD

Crea il `TridentOrchestrator` Definizione di risorse personalizzate (CRD). Creerai un `TridentOrchestrator` Risorsa personalizzata in seguito. Utilizzare la versione CRD YAML appropriata in `deploy/crds` per creare il `TridentOrchestrator` CRD:

```
kubectl create -f deploy/crds/<VERSION>.yaml
```

Fase 3: Aggiornare la posizione del registro nell'operatore

In `/deploy/operator.yaml`, eseguire l'aggiornamento `image: docker.io/netapp/trident-operator:24.10.0` per riflettere la posizione del registro delle immagini. Lo "Immagini Trident e CSI" può trovarsi in un registro o in registri diversi, ma tutte le immagini CSI devono trovarsi nello stesso registro. Ad esempio:

- `image: <your-registry>/trident-operator:24.10.0` se tutte le immagini si trovano in un unico registro.
- `image: <your-registry>/netapp/trident-operator:24.10.0` Se l'immagine Trident si trova in un registro diverso dalle immagini CSI.

Fase 4: Implementare l'operatore Trident

Il programma di installazione di Trident fornisce un file bundle che può essere utilizzato per installare l'operatore e creare oggetti associati. Il file bundle è un modo semplice per distribuire l'operatore e installare Trident utilizzando una configurazione predefinita.

- 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

- Per impostazione predefinita, il programma di installazione di Trident implementa l'operatore nello `trident` spazio dei nomi. Se lo `trident` spazio dei nomi non esiste, crearlo utilizzando:

```
kubectl apply -f deploy/namespace.yaml
```

- Per distribuire l'operatore in uno spazio dei nomi diverso dallo `trident` spazio dei nomi, aggiornare `serviceaccount.yaml` `clusterrolebinding.yaml` e `operator.yaml` generare il file bundle utilizzando `kustomization.yaml` .

  a. Crea il `kustomization.yaml` usando il seguente comando dove *<bundle.yaml>* è `bundle_pre_1_25.yaml` o `bundle_post_1_25.yaml` si basa sulla tua versione di Kubernetes.

  ```
  cp deploy/kustomization_<bundle.yaml> deploy/kustomization.yaml
  ```

  b. Compila il bundle usando il seguente comando dove *<bundle.yaml>* si trova `bundle_pre_1_25.yaml` o `bundle_post_1_25.yaml` si basa sulla tua versione di Kubernetes.

  ```
  kubectl kustomize deploy/ > deploy/<bundle.yaml>
  ```

Fasi

1. Creare le risorse e implementare l'operatore:

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

2. Verificare che l'operatore, l'implementazione e i replicaset siano stati creati.

```
kubectl get all -n <operator-namespace>
```

In un cluster Kubernetes dovrebbe esserci solo **un'istanza** dell'operatore. Non creare implementazioni multiple dell'operatore Trident.

Passaggio 5: Aggiornare la posizione del Registro di sistema dell'immagine in `TridentOrchestrator`

Lo "Immagini Trident e CSI" può trovarsi in un registro o in registri diversi, ma tutte le immagini CSI devono trovarsi nello stesso registro. Aggiornare `deploy/crds/tridentorchestrator_cr.yaml` per aggiungere le specifiche di posizione aggiuntive in base alla configurazione del Registro di sistema.

Immagini in un registro

```
imageRegistry: "<your-registry>"
autosupportImage: "<your-registry>/trident-autosupport:24.10"
tridentImage: "<your-registry>/trident:24.10.0"
```

Immagini in diversi registri

```
imageRegistry: "<your-registry>"
autosupportImage: "<your-registry>/trident-autosupport:24.10"
tridentImage: "<your-registry>/trident:24.10.0"
```

Fase 6: Creare `TridentOrchestrator` e installare Trident

A questo punto è possibile creare `TridentOrchestrator` e installare Trident. Facoltativamente, è possibile utilizzare ulteriormente "Personalizzare l'installazione di Trident"gli attributi nella `TridentOrchestrator` specifica. L'esempio seguente mostra un'installazione in cui le immagini Trident e CSI si trovano in diversi registri.

```
kubectl create -f deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created

kubectl describe torc trident

Name:         trident
Namespace:
Labels:       <none>
Annotations:  <none>
API Version:  trident.netapp.io/v1
Kind:         TridentOrchestrator
...
Spec:
  Autosupport Image:  <your-registry>/trident-autosupport:24.10
  Debug:              true
  Image Registry:     <your-registry>
  Namespace:          trident
  Trident Image:      <your-registry>/trident:24.10.0
Status:
  Current Installation Params:
    IPv6:                      false
    Autosupport Hostname:
    Autosupport Image:         <your-registry>/trident-autosupport:24.10
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                     true
    Http Request Timeout:      90s
    Image Pull Secrets:
    Image Registry:       <your-registry>
    k8sTimeout:           30
    Kubelet Dir:          /var/lib/kubelet
    Log Format:           text
```

```
    Probe Port:           17546
    Silence Autosupport:  false
    Trident Image:        <your-registry>/trident:24.10.0
  Message:                Trident installed
  Namespace:              trident
  Status:                 Installed
  Version:                v24.10.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

Verificare l'installazione

Esistono diversi modi per verificare l'installazione.

Uso dello `TridentOrchestrator` stato

Lo stato di `TridentOrchestrator` indica se l'installazione è riuscita e visualizza la versione di Trident installata. Durante l'installazione, lo stato di `TridentOrchestrator` cambia da `Installing` a `Installed`. Se si osserva `Failed` lo stato e l'operatore non è in grado di recuperare da solo, "controllare i registri".

Stato	Descrizione
Installazione in corso	L'operatore sta installando Trident utilizzando questo `TridentOrchestrator` CR.
Installato	Installazione di Trident riuscita.
Disinstallazione in corso	L'operatore sta disinstallando Trident, perché `spec.uninstall=true`.
Disinstallato	Trident è disinstallato.
Non riuscito	L'operatore non ha potuto installare, applicare patch, aggiornare o disinstallare Trident; l'operatore tenterà automaticamente di ripristinare da questo stato. Se lo stato persiste, è necessario eseguire la risoluzione dei problemi.
Aggiornamento in corso	L'operatore sta aggiornando un'installazione esistente.
Errore	`TridentOrchestrator`Non viene utilizzato. Un'altra esiste già.

Utilizzo dello stato di creazione del pod

È possibile verificare se l'installazione di Trident è stata completata controllando lo stato dei pod creati:

```
kubectl get pods -n trident
```

```
NAME                                READY   STATUS    RESTARTS
AGE
trident-controller-7d466bf5c7-v4cpw  6/6     Running   0
1m
trident-node-linux-mr6zc             2/2     Running   0
1m
trident-node-linux-xrp7w             2/2     Running   0
1m
trident-node-linux-zh2jt             2/2     Running   0
1m
trident-operator-766f7b8658-ldzsv    1/1     Running   0
3m
```

Uso `tridentctl`

È possibile utilizzare `tridentctl` per controllare la versione di Trident installata.

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 24.10.0        | 24.10.0        |
+----------------+----------------+
```

Implementare l'operatore Trident utilizzando Helm (modalità standard)

È possibile distribuire l'operatore Trident e installare Trident utilizzando Helm. Questo processo si applica alle installazioni in cui le immagini contenitore richieste da Trident non sono memorizzate in un registro privato. Se si dispone di un registro delle immagini privato, utilizzare "processo per l'implementazione offline".

Informazioni critiche su Trident 24,10

È necessario leggere le seguenti informazioni critiche su Trident.

**▼ informazioni aggiornate su Trident **

- Kubernetes 1,32 è ora supportato in Trident. Aggiornare Trident prima di aggiornare Kubernetes.
- Trident impone rigorosamente l'uso della configurazione multipath negli ambienti SAN, con un valore consigliato di `find_multipaths: no` nel file multipath.conf.

  L'utilizzo di una configurazione non multipathing o di `find_multipaths: yes` un valore OR `find_multipaths: smart` nel file multipath.conf determinerà errori di montaggio. Trident ha

consigliato l'uso di `find_multipaths: no` fin dalla versione 21,07.

Distribuire l'operatore Trident e installare Trident utilizzando Helm

Utilizzando Trident "Grafico di comando"è possibile distribuire l'operatore Trident e installare Trident in un'unica fase.

Verificate "panoramica dell'installazione" per assicurarvi di aver soddisfatto i prerequisiti di installazione e di aver selezionato l'opzione di installazione corretta per il vostro ambiente.

Prima di iniziare

Oltre al "prerequisiti per l'implementazione" è necessario "Helm versione 3".

Fasi

1. Aggiungere il repository Trident Helm:

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
```

2. Utilizzare `helm install` e specificare un nome per la distribuzione come nell'esempio seguente dove si trova `100.2410.0` la versione di Trident che si sta installando.

```
helm install <name> netapp-trident/trident-operator --version 100.2410.0
--create-namespace --namespace <trident-namespace>
```

> Se è già stato creato uno spazio dei nomi per Trident, il `--create-namespace` parametro non creerà uno spazio dei nomi aggiuntivo.

È possibile utilizzare `helm list` per esaminare i dettagli dell'installazione come nome, spazio dei nomi, grafico, stato, versione dell'app, e numero di revisione.

Passare i dati di configurazione durante l'installazione

Esistono due modi per passare i dati di configurazione durante l'installazione:

Opzione	Descrizione
`--values` (o `-f`)	Specificare un file YAML con override. Questo valore può essere specificato più volte e il file più a destra avrà la precedenza.
`--set`	Specificare le sostituzioni sulla riga di comando.

Ad esempio, per modificare il valore predefinito di `debug`, eseguire il comando seguente dove si trova `100.2410.0` la versione di Trident che si sta installando:

```
helm install <name> netapp-trident/trident-operator --version 100.2410.0
```

```
--create-namespace --namespace trident --set tridentDebug=true
```

Opzioni di configurazione

Questa tabella e il `values.yaml` file, che fa parte del grafico Helm, forniscono l'elenco delle chiavi e i loro valori predefiniti.

Opzione	Descrizione	Predefinito
`nodeSelector`	Etichette dei nodi per l'assegnazione dei pod	
`podAnnotations`	Annotazioni Pod	
`deploymentAnnotations`	Annotazioni di implementazione	
`tolerations`	Pedaggi per l'assegnazione del pod	

`affinity`	Affinità per l'assegnazione del pod	`affinity:`
		`  nodeAffinity:`
		
		`requiredDuringSchedulingIgnoredDuringExecution:`
		`  nodeSelectorTerms:`
		`  - matchExpressions:`
		`  - key: kubernetes.io/arch`
		`  operator: In`
		`  values:`
		`  - arm64`
		`  - amd64`
		`  - key: kubernetes.io/os`
		`  operator: In`
		`  values:`
		`  - linux`
		
		Non rimuovere l'affinità predefinita dal file Values.yaml. Se si desidera fornire un'affinità personalizzata, estendere l'affinità predefinita.

`tridentControllerPluginNodeSelector`	Selettori di nodi aggiuntivi per i pod. Per ulteriori informazioni, fare riferimento alla Comprensione dei pod controller e dei pod di nodi sezione.	

Opzione	Descrizione	Predefinito
`tridentControllerPluginTolerations`	Ignora le tolleranze Kubernetes per i pod. Per ulteriori informazioni, fare riferimento alla Comprensione dei pod controller e dei pod di nodi sezione.	
`tridentNodePluginNodeSelector`	Selettori di nodi aggiuntivi per i pod. Per ulteriori informazioni, fare riferimento alla Comprensione dei pod controller e dei pod di nodi sezione.	
`tridentNodePluginTolerations`	Ignora le tolleranze Kubernetes per i pod. Per ulteriori informazioni, fare riferimento alla Comprensione dei pod controller e dei pod di nodi sezione.	
`imageRegistry`	Identifica il Registro di sistema per `trident-operator` , `trident` e altre immagini. Lasciare vuoto per accettare l'impostazione predefinita. IMPORTANTE: Quando si installa Trident in un repository privato, se si utilizza `imageRegistry` lo switch per specificare la posizione del repository, non utilizzare `/netapp/` nel percorso del repository.	`""`
`imagePullPolicy`	Consente di impostare il criterio di pull dell'immagine per `trident-operator` .	`IfNotPresent`
`imagePullSecrets`	Consente di impostare i segreti di estrazione delle immagini per `trident-operator` , `trident` e altre immagini.	
`kubeletDir`	Consente di ignorare la posizione host dello stato interno del kubelet.	`"/var/lib/kubelet"`
`operatorLogLevel`	Consente di impostare il livello di registrazione dell'operatore Trident su: `trace`, `debug`, , `info`, `warn` `error` O `fatal`.	`"info"`
`operatorDebug`	Consente di impostare il livello di log dell'operatore Trident su debug.	`true`
`operatorImage`	Consente la sostituzione completa dell'immagine per `trident-operator`.	`""`
`operatorImageTag`	Consente di sovrascrivere il tag dell' `trident-operator`immagine.	`""`

Opzione	Descrizione	Predefinito
`tridentIPv6`	Permette a Trident di lavorare in cluster IPv6.	`false`
`tridentK8sTimeout`	Esegue l'override del timeout predefinito di 30 secondi per la maggior parte delle operazioni API di Kubernetes (se diverso da zero, in secondi).	`0`
`tridentHttpRequestTimeout`	Ignora il timeout predefinito di 90 secondi per le richieste HTTP, con `0s` una durata infinita per il timeout. Non sono consentiti valori negativi.	`"90s"`
`tridentSilenceAutosupport`	Consente di disattivare la creazione di rapporti periodici AutoSupport Trident.	`false`
`tridentAutosupportImageTag`	Consente di sovrascrivere il tag dell'immagine per il contenitore Trident AutoSupport.	`<version>`
`tridentAutosupportProxy`	Consente al contenitore Trident AutoSupport di telefonare a casa tramite un proxy HTTP.	`""`
`tridentLogFormat`	Consente di impostare il formato di registrazione Trident (`text`o `json`).	`"text"`
`tridentDisableAuditLog`	Disattiva il registro di controllo Trident.	`true`
`tridentLogLevel`	Consente di impostare il livello di registrazione di Trident su: `trace`, `debug`, `info`, , `warn` `error` O `fatal`.	`"info"`
`tridentDebug`	Consente di impostare il livello di registro di Trident su `debug`.	`false`
`tridentLogWorkflows`	Consente di attivare flussi di lavoro Trident specifici per la registrazione delle tracce o l'eliminazione dei registri.	`""`
`tridentLogLayers`	Consente di attivare livelli Trident specifici per la registrazione delle tracce o la soppressione dei registri.	`""`
`tridentImage`	Consente la sostituzione completa dell'immagine per Trident.	`""`
`tridentImageTag`	Consente di sovrascrivere il tag dell'immagine per Trident.	`""`

Opzione	Descrizione	Predefinito
`tridentProbePort`	Consente di ignorare la porta predefinita utilizzata per le sonde liveness/readiness Kubernetes.	`""`
`windows`	Consente di installare Trident sul nodo di lavoro Windows.	`false`
`enableForceDetach`	Consente di attivare la funzione di distacco forzato.	`false`
`excludePodSecurityPolicy`	Esclude la creazione della policy di sicurezza del pod operatore.	`false`
`cloudProvider`	Impostato su `"Azure"` quando si utilizzano identità gestite o un'identità cloud su un cluster AKS. Impostato su "AWS" quando si utilizza un'identità cloud su un cluster EKS.	`""`
`cloudIdentity`	Impostare sull'identità del workload ("Azure.workload.Identity/client-id: Xxxxxxxx-xxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx") quando si utilizza l'identità cloud su un cluster AKS. Impostato sul ruolo AWS IAM ("'eks.amazonaws.com/role-arn: arn:AWS:iam::123456:role/Trident-role'") quando si utilizza l'identità cloud su un cluster EKS.	`""`
`iscsiSelfHealingInterval`	Intervallo in cui viene richiamata la riparazione automatica iSCSI.	`5m0s`
`iscsiSelfHealingWaitTime`	La durata dopo la quale l'autoriparazione iSCSI avvia un tentativo di risolvere una sessione obsoleta eseguendo una disconnessione e un login successivo.	`7m0s`
`nodePrep`	Permette 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.**	

Comprensione dei pod controller e dei pod di nodi

Trident funziona come un singolo pod controller, più un pod di nodo su ciascun nodo di lavoro nel cluster. Il pod di nodo deve essere in esecuzione su qualsiasi host in cui si desidera montare potenzialmente un volume Trident.

Kubernetes "selettori di nodi" e "tollerazioni e contamini" sono utilizzati per vincolare un pod all'esecuzione su

un nodo specifico o preferito. Utilizzando `ControllerPlugin` e `NodePlugin`, è possibile specificare vincoli e sostituzioni.

- Il plug-in del controller gestisce il provisioning e la gestione dei volumi, ad esempio snapshot e ridimensionamento.
- Il plug-in del nodo gestisce il collegamento dello storage al nodo.

Implementare l'operatore Trident utilizzando Helm (modalità offline)

È possibile distribuire l'operatore Trident e installare Trident utilizzando Helm. Questo processo si applica alle installazioni in cui le immagini contenitore richieste da Trident sono memorizzate in un registro privato. Se non si dispone di un registro delle immagini private, utilizzare "processo per l'implementazione standard".

Informazioni critiche su Trident 24,10

È necessario leggere le seguenti informazioni critiche su Trident.

**▼ informazioni aggiornate su Trident **

- Kubernetes 1,32 è ora supportato in Trident. Aggiornare Trident prima di aggiornare Kubernetes.
- Trident impone rigorosamente l'uso della configurazione multipath negli ambienti SAN, con un valore consigliato di `find_multipaths: no` nel file multipath.conf.

  L'utilizzo di una configurazione non multipathing o di `find_multipaths: yes` un valore OR `find_multipaths: smart` nel file multipath.conf determinerà errori di montaggio. Trident ha consigliato l'uso di `find_multipaths: no` fin dalla versione 21,07.

Distribuire l'operatore Trident e installare Trident utilizzando Helm

Utilizzando Trident "Grafico di comando"è possibile distribuire l'operatore Trident e installare Trident in un'unica fase.

Verificate "panoramica dell'installazione" per assicurarvi di aver soddisfatto i prerequisiti di installazione e di aver selezionato l'opzione di installazione corretta per il vostro ambiente.

Prima di iniziare

Oltre al "prerequisiti per l'implementazione" è necessario "Helm versione 3".

> Quando si installa Trident in un repository privato, se si utilizza `imageRegistry` lo switch per specificare la posizione del repository, non utilizzare `/netapp/` nel percorso del repository.

Fasi

1. Aggiungere il repository Trident Helm:

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
```

2. Utilizzare `helm install` e specificare un nome per la distribuzione e la posizione del Registro di sistema dell'immagine. Lo "Immagini Trident e CSI" può trovarsi in un registro o in registri diversi, ma tutte le immagini CSI devono trovarsi nello stesso registro. Negli esempi, `100.2410.0` è la versione di Trident che si sta installando.

Immagini in un registro

```
helm install <name> netapp-trident/trident-operator --version
100.2410.0 --set imageRegistry=<your-registry> --create-namespace
--namespace <trident-namespace> --set nodePrep={iscsi}
```

Immagini in diversi registri

```
helm install <name> netapp-trident/trident-operator --version
100.2410.0 --set imageRegistry=<your-registry> --set
operatorImage=<your-registry>/trident-operator:24.10.0 --set
tridentAutosupportImage=<your-registry>/trident-autosupport:24.10
--set tridentImage=<your-registry>/trident:24.10.0 --create
-namespace --namespace <trident-namespace> --set nodePrep={iscsi}
```

Se è già stato creato uno spazio dei nomi per Trident, il `--create-namespace` parametro non creerà uno spazio dei nomi aggiuntivo.

È possibile utilizzare `helm list` per esaminare i dettagli dell'installazione come nome, spazio dei nomi, grafico, stato, versione dell'app, e numero di revisione.

Passare i dati di configurazione durante l'installazione

Esistono due modi per passare i dati di configurazione durante l'installazione:

Opzione	Descrizione
`--values` (o `-f`)	Specificare un file YAML con override. Questo valore può essere specificato più volte e il file più a destra avrà la precedenza.
`--set`	Specificare le sostituzioni sulla riga di comando.

Ad esempio, per modificare il valore predefinito di `debug`, eseguire il comando seguente dove si trova `100.2410.0` la versione di Trident che si sta installando:

```
helm install <name> netapp-trident/trident-operator --version 100.2410.0
--create-namespace --namespace trident --set tridentDebug=true
```

Per aggiungere il valore nodePrep, eseguire il comando seguente:

```
helm install <name> netapp-trident/trident-operator --version 100.2406.0
--create-namespace --namespace trident --set nodePrep={iscsi}
```

Opzioni di configurazione

Questa tabella e il `values.yaml` file, che fa parte del grafico Helm, forniscono l'elenco delle chiavi e i loro valori predefiniti.

Non rimuovere l'affinità predefinita dal file Values.yaml. Se si desidera fornire un'affinità personalizzata, estendere l'affinità predefinita.

Opzione	Descrizione	Predefinito
`nodeSelector`	Etichette dei nodi per l'assegnazione dei pod	
`podAnnotations`	Annotazioni Pod	
`deploymentAnnotations`	Annotazioni di implementazione	
`tolerations`	Pedaggi per l'assegnazione del pod	

Opzione	Descrizione	Predefinito

`affinity`	Affinità per l'assegnazione del pod	`affinity:`
		`  nodeAffinity:`
		
		`requiredDuringSchedul`
		`ingIgnoredDuringExecu`
		`tion:`
		
		`nodeSelectorTerms:`
		`  -`
		`matchExpressions:`
		`  - key:`
		`kubernetes.io/arch`
		
		`operator: In`
		`  values:`
		`  - arm64`
		`  - amd64`
		`  - key:`
		`kubernetes.io/os`
		
		`operator: In`
		`  values:`
		`  - linux`
		
		Non rimuovere l'affinità predefinita dal file Values.yaml. Se si desidera fornire un'affinità personalizzata, estendere l'affinità predefinita.

`tridentControllerPluginNodeSelector`	Selettori di nodi aggiuntivi per i pod. Per ulteriori informazioni, fare riferimento alla "Comprensione dei pod controller e dei pod di nodi" sezione.	
`tridentControllerPluginTolerations`	Ignora le tolleranze Kubernetes per i pod. Per ulteriori informazioni, fare riferimento alla "Comprensione dei pod controller e dei pod di nodi" sezione.	

Opzione	Descrizione	Predefinito
`tridentNodePluginNodeSelector`	Selettori di nodi aggiuntivi per i pod. Per ulteriori informazioni, fare riferimento alla "Comprensione dei pod controller e dei pod di nodi" sezione.	
`tridentNodePluginTolerations`	Ignora le tolleranze Kubernetes per i pod. Per ulteriori informazioni, fare riferimento alla "Comprensione dei pod controller e dei pod di nodi" sezione.	
`imageRegistry`	Identifica il Registro di sistema per `trident-operator` , `trident` e altre immagini. Lasciare vuoto per accettare l'impostazione predefinita. IMPORTANTE: Quando si installa Trident in un repository privato, se si utilizza `imageRegistry` lo switch per specificare la posizione del repository, non utilizzare `/netapp/` nel percorso del repository.	""
`imagePullPolicy`	Consente di impostare il criterio di pull dell'immagine per `trident-operator` .	`IfNotPresent`
`imagePullSecrets`	Consente di impostare i segreti di estrazione delle immagini per `trident-operator` , `trident` e altre immagini.	
`kubeletDir`	Consente di ignorare la posizione host dello stato interno del kubelet.	`"/var/lib/kubelet"`
`operatorLogLevel`	Consente di impostare il livello di registrazione dell'operatore Trident su: `trace`, `debug`, , `info`, `warn` `error` O `fatal`.	`"info"`
`operatorDebug`	Consente di impostare il livello di log dell'operatore Trident su debug.	`true`
`operatorImage`	Consente la sostituzione completa dell'immagine per `trident-operator`.	""
`operatorImageTag`	Consente di sovrascrivere il tag dell' `trident-operator`immagine.	""
`tridentIPv6`	Permette a Trident di lavorare in cluster IPv6.	`false`

Opzione	Descrizione	Predefinito
`tridentK8sTimeout`	Esegue l'override del timeout predefinito di 30 secondi per la maggior parte delle operazioni API di Kubernetes (se diverso da zero, in secondi).	`0`
`tridentHttpRequestTimeout`	Ignora il timeout predefinito di 90 secondi per le richieste HTTP, con `0s` una durata infinita per il timeout. Non sono consentiti valori negativi.	`"90s"`
`tridentSilenceAutosupport`	Consente di disattivare la creazione di rapporti periodici AutoSupport Trident.	`false`
`tridentAutosupportImageTag`	Consente di sovrascrivere il tag dell'immagine per il contenitore Trident AutoSupport.	`<version>`
`tridentAutosupportProxy`	Consente al contenitore Trident AutoSupport di telefonare a casa tramite un proxy HTTP.	""
`tridentLogFormat`	Consente di impostare il formato di registrazione Trident (`text`o `json`).	`"text"`
`tridentDisableAuditLog`	Disattiva il registro di controllo Trident.	`true`
`tridentLogLevel`	Consente di impostare il livello di registrazione di Trident su: `trace`, `debug`, `info`, , `warn` `error` O `fatal`.	`"info"`
`tridentDebug`	Consente di impostare il livello di registro di Trident su `debug`.	`false`
`tridentLogWorkflows`	Consente di attivare flussi di lavoro Trident specifici per la registrazione delle tracce o l'eliminazione dei registri.	""
`tridentLogLayers`	Consente di attivare livelli Trident specifici per la registrazione delle tracce o la soppressione dei registri.	""
`tridentImage`	Consente la sostituzione completa dell'immagine per Trident.	""
`tridentImageTag`	Consente di sovrascrivere il tag dell'immagine per Trident.	""
`tridentProbePort`	Consente di ignorare la porta predefinita utilizzata per le sonde liveness/readiness Kubernetes.	""

Opzione	Descrizione	Predefinito
`windows`	Consente di installare Trident sul nodo di lavoro Windows.	`false`
`enableForceDetach`	Consente di attivare la funzione di distacco forzato.	`false`
`excludePodSecurityPolicy`	Esclude la creazione della policy di sicurezza del pod operatore.	`false`
`nodePrep`	Permette 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.**	

Personalizzare l'installazione dell'operatore Trident

L'operatore Trident consente di personalizzare l'installazione di Trident utilizzando gli attributi della `TridentOrchestrator` specifica. Se si desidera personalizzare l'installazione oltre gli `TridentOrchestrator` argomenti consentiti, utilizzare `tridentctl` per generare manifesti YAML personalizzati da modificare in base alle necessità.

Comprensione dei pod controller e dei pod di nodi

Trident funziona come un singolo pod controller, più un pod di nodo su ciascun nodo di lavoro nel cluster. Il pod di nodo deve essere in esecuzione su qualsiasi host in cui si desidera montare potenzialmente un volume Trident.

Kubernetes "selettori di nodi" e "tollerazioni e contamini" sono utilizzati per vincolare un pod all'esecuzione su un nodo specifico o preferito. Utilizzando `ControllerPlugin` e `NodePlugin`, è possibile specificare vincoli e sostituzioni.

- Il plug-in del controller gestisce il provisioning e la gestione dei volumi, ad esempio snapshot e ridimensionamento.
- Il plug-in del nodo gestisce il collegamento dello storage al nodo.

Opzioni di configurazione

> `spec.namespace` Viene specificato in per indicare lo spazio dei nomi in `TridentOrchestrator` cui è installato Trident. Questo parametro **non può essere aggiornato dopo l'installazione di Trident**. Se si tenta di eseguire questa operazione, `TridentOrchestrator` lo stato viene modificato in `Failed`. Trident non deve essere migrato tra namespace.

Questa tabella fornisce i dettagli degli `TridentOrchestrator` attributi.

Parametro	Descrizione	Predefinito
`namespace`	Spazio dei nomi in cui installare Trident	`"default"`
`debug`	Attivare il debug per Trident	`false`
`enableForceDetach`	`ontap-san`, , `ontap-san-economy` e `ontap-nas-economy` solo. Funziona con Kubernetes non-Graged Node Shutdown (NGNS) per consentire agli amministratori del cluster di migrare in sicurezza i carichi di lavoro con volumi montati su nuovi nodi in caso di problemi di integrità di un nodo.	`false`
`windows`	L'impostazione su `true` attiva l'installazione sui nodi di lavoro Windows.	`false`
`cloudProvider`	Impostato su `"Azure"` quando si utilizzano identità gestite o un'identità cloud su un cluster AKS. Impostato su "AWS" quando si utilizza un'identità cloud su un cluster EKS.	`""`
`cloudIdentity`	Impostare sull'identità del workload ("Azure.workload.Identity/client-id: Xxxxxxxx-xxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx") quando si utilizza l'identità cloud su un cluster AKS. Impostato sul ruolo AWS IAM ("'eks.amazonaws.com/role-arn: arn:AWS:iam::123456:role/Trident-role") quando si utilizza l'identità cloud su un cluster EKS.	`""`
`IPv6`	Installare Trident su IPv6	falso
`k8sTimeout`	Timeout per le operazioni Kubernetes	`30sec`
`silenceAutosupport`	Non inviare pacchetti AutoSupport automaticamente a NetApp	`false`
`autosupportImage`	L'immagine del contenitore per la telemetria AutoSupport	`"netapp/trident-autosupport:24.10"`
`autosupportProxy`	Indirizzo/porta di un proxy per l'invio di telemetria AutoSupport	`"http://proxy.example.com:8888"`
`uninstall`	Un flag utilizzato per disinstallare Trident	`false`
`logFormat`	Formato di registrazione Trident da utilizzare [text,json]	`"text"`
`tridentImage`	Immagine Trident da installare	`"netapp/trident:24.10"`
`imageRegistry`	Percorso al registro interno, del formato `<registry FQDN>[:port][/subpath]`	`"k8s.gcr.io"` (Kubernetes 1,19+) o. `"quay.io/k8scsi"`
`kubeletDir`	Percorso della directory del kubelet sull'host	`"/var/lib/kubelet"`
`wipeout`	Un elenco di risorse da eliminare per eseguire la rimozione completa di Trident	

Parametro	Descrizione	Predefinito
`imagePullSecrets`	Secrets (segreti) per estrarre immagini da un registro interno	
`imagePullPolicy`	Imposta il criterio di pull dell'immagine per l'operatore Trident. I valori validi sono: `Always` Per tirare sempre l'immagine. `IfNotPresent` per estrarre l'immagine solo se non esiste già sul nodo. `Never` per non tirare mai l'immagine.	`IfNotPresent`
`controllerPluginNodeSelector`	Selettori di nodi aggiuntivi per i pod. Segue lo stesso formato di `pod.spec.nodeSelector`.	Nessuna impostazione predefinita; opzionale
`controllerPluginTolerations`	Ignora le tolleranze Kubernetes per i pod. Segue lo stesso formato di `pod.spec.Tolerations`.	Nessuna impostazione predefinita; opzionale
`nodePluginNodeSelector`	Selettori di nodi aggiuntivi per i pod. Segue lo stesso formato di `pod.spec.nodeSelector`.	Nessuna impostazione predefinita; opzionale
`nodePluginTolerations`	Ignora le tolleranze Kubernetes per i pod. Segue lo stesso formato di `pod.spec.Tolerations`.	Nessuna impostazione predefinita; opzionale
`nodePrep`	Permette 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.**	

Per ulteriori informazioni sulla formattazione dei parametri del pod, fare riferimento alla sezione "Assegnazione di pod ai nodi".

Dettagli sulla forza di distacco

Il distacco forzato è disponibile per `ontap-san` , `ontap-san-economy` E `ontap-nas-economy` soltanto. Prima di abilitare la disconnessione forzata, è necessario abilitare l'arresto non regolare del nodo (NGNS) sul cluster Kubernetes. Per ulteriori informazioni, fare riferimento a "Kubernetes: Shutdown del nodo non aggraziato" .

Quando si utilizza il `ontap-nas-economy` driver, è necessario impostare il `autoExportPolicy` parametro nella configurazione backend su in `true` modo che Trident possa limitare l'accesso dal nodo Kubernetes con il tag applicato utilizzando policy di esportazione gestite.

Poiché Trident fa affidamento su Kubernetes NGNS, non rimuovere i `out-of-service` tag da un nodo non integro fino a quando tutti i carichi di lavoro non tollerabili non vengono ripianificati. L'applicazione o la rimozione sconsiderata della contaminazione può compromettere la protezione dei dati back-end.

Quando l'amministratore del cluster Kubernetes ha applicato il `node.kubernetes.io/out-of-service=nodeshutdown:NoExecute` tag al nodo ed `enableForceDetach` è impostato su `true`, Trident determinerà lo stato del nodo e:

1. Interrompere l'accesso i/o back-end per i volumi montati su quel nodo.

2. Contrassegnare l'oggetto nodo Trident come `dirty` (non sicuro per le nuove pubblicazioni).

> Il controller Trident rifiuterà le nuove richieste di volume di pubblicazione finché il nodo non viene riqualificato (dopo essere stato contrassegnato come `dirty`) dal pod di nodo Trident. Tutti i carichi di lavoro pianificati con un PVC montato (anche dopo che il nodo del cluster è integro e pronto) non saranno accettati fino a quando Trident non sarà in grado di verificare il nodo `clean` (sicuro per le nuove pubblicazioni).

Quando l'integrità del nodo viene ripristinata e il tag viene rimosso, Trident:

1. Identificare e pulire i percorsi pubblicati obsoleti sul nodo.
2. Se il nodo si trova in uno `cleanable` stato (il tag out-of-service è stato rimosso e il nodo è nello `Ready` stato) e tutti i percorsi obsoleti e pubblicati sono puliti, Trident riammetterà il nodo come `clean` e consentirà ai nuovi volumi pubblicati di accedere al nodo.

Configurazioni di esempio

È possibile utilizzare gli attributi in Opzioni di configurazione durante la definizione `TridentOrchestrator` per personalizzare l'installazione.

▼ Configurazione personalizzata di base

Questo è un esempio per un'installazione personalizzata di base.

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

▼ Selettori di nodo

In questo esempio viene installato Trident con i selettori di nodo.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  controllerPluginNodeSelector:
    nodetype: master
```

```
  nodePluginNodeSelector:
    storage: netapp
```

▼ **Nodi di lavoro Windows**

In questo esempio viene installato Trident su un nodo di lavoro Windows.

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

▼ **Identità gestite su un cluster AKS**

In questo esempio viene installato Trident per abilitare le identità gestite su un cluster AKS.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  cloudProvider: "Azure"
```

▼ **Identità cloud su un cluster AKS**

Questo esempio installa Trident per l'utilizzo con un'identità cloud su un cluster AKS.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  cloudProvider: "Azure"
  cloudIdentity: 'azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-
xxxx-xxxxxxxxxxx'
```

▼ **Identità cloud su un cluster EKS**

Questo esempio installa Trident per l'utilizzo con un'identità cloud su un cluster AKS.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  cloudProvider: "AWS"
  cloudIdentity: "'eks.amazonaws.com/role-arn:
arn:aws:iam::123456:role/trident-role'"
```

▼ **Identità cloud per GKE**

Questo esempio installa Trident per l'utilizzo con un'identità cloud su un cluster GKE.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcp-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: '012345678901'
  network: gcnv-network
  location: us-west2
  serviceLevel: Premium
  storagePool: pool-premium1
```

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.