



Installa utilizzando l'operatore Trident

Trident

NetApp
July 01, 2026

Sommario

Installa utilizzando l'operatore Trident	1
Distribuisci manualmente l'operatore Trident (modalità standard)	1
Informazioni critiche su Trident 26.02	1
Distribuire manualmente l'operatore Trident e installare Trident	1
Verificare l'installazione	4
Distribuisci manualmente l'operatore Trident (modalità offline)	6
Informazioni critiche su Trident	6
Distribuire manualmente l'operatore Trident e installare Trident	6
Verificare l'installazione	11
Distribuisci l'operatore Trident tramite Helm (modalità standard)	12
Informazioni critiche su Trident 25.10	12
Distribuire l'operatore Trident e installare Trident utilizzando Helm	12
Passare i dati di configurazione durante l'installazione	13
Opzioni di configurazione	13
Distribuire l'operatore Trident utilizzando Helm (Offline mode)	20
Informazioni critiche su Trident 26.02	20
Distribuire l'operatore Trident e installare Trident utilizzando Helm	21
Passare i dati di configurazione durante l'installazione	22
Opzioni di configurazione	22
Personalizzare l'installazione dell'operatore Trident	28
Comprendere i pod controller e i pod nodo	28
Opzioni di configurazione	28
Configurazioni di esempio	32

Installa utilizzando l'operatore Trident

Distribuisci manualmente l'operatore Trident (modalità standard)

È possibile distribuire manualmente l'operatore Trident per installare Trident. Questa procedura si applica alle installazioni in cui le immagini dei container richieste da Trident non sono archiviate in un registro privato. Se si dispone di un registro immagini privato, utilizzare il ["processo per la distribuzione offline"](#).

Informazioni critiche su Trident 26.02

È necessario leggere le seguenti informazioni critiche su Trident.

Informazioni critiche su Trident

- Kubernetes 1.35 è ora supportato in Trident. Aggiornare Trident prima di aggiornare Kubernetes.
- Trident impone rigorosamente l'uso della configurazione multipathing negli ambienti SAN, con un valore consigliato di `find_multipaths: no` nel file `multipath.conf`.

L'utilizzo di una configurazione non multipath o l'utilizzo di `find_multipaths: yes` o `find_multipaths: smart` nel file `multipath.conf` causerà errori di montaggio. Trident ha raccomandato l'utilizzo di `find_multipaths: no` dalla release 21.07.

Distribuire manualmente l'operatore Trident e installare Trident

Verificate ["la 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` in tutti gli esempi che seguono e accedere prima come **system:admin** eseguendo `oc login -u system:admin` o `oc login -u kube-admin`.

1. Verifica la versione di Kubernetes:

```
kubectl version
```

2. Verificare i privilegi di amministratore del cluster:

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

3. Verifica di poter avviare un pod che utilizza un'immagine da Docker Hub e raggiungere il tuo sistema storage tramite la rete del pod:

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

Passaggio 1: Scarica il pacchetto di installazione di Trident

Il pacchetto di installazione Trident contiene tutto ciò che serve per distribuire l'operatore Trident e installare Trident. Scarica ed estrai la versione più recente del programma di installazione di Trident da ["la sezione Assets su GitHub"](#).

```
wget https://github.com/NetApp/trident/releases/download/v26.02.0/trident-
installer-26.02.0.tar.gz
tar -xf trident-installer-26.02.0.tar.gz
cd trident-installer
```

Passaggio 2: crea la TridentOrchestrator CRD

Crea la TridentOrchestrator Definizione di Risorsa Personalizzata (CRD). Creerai una TridentOrchestrator Risorsa Personalizzata in seguito. Utilizza la versione YAML appropriata del CRD in [deploy/crds per creare la TridentOrchestrator CRD](#).

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

Fase 3: distribuisce l'operatore Trident

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

- 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, utilizzare `bundle_post_1_25.yaml`.

Prima di iniziare

- Per impostazione predefinita, il programma di installazione di Trident distribuisce l'operatore nel `trident` namespace. Se il `trident` namespace non esiste, crearlo utilizzando:

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

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

- a. Crea `kustomization.yaml` utilizzando il seguente comando dove `<bundle.yaml>` è `bundle_pre_1_25.yaml` o `bundle_post_1_25.yaml` in base alla tua versione di Kubernetes.

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

- b. Compila il bundle usando il seguente comando, dove `<bundle.yaml>` è `bundle_pre_1_25.yaml` o `bundle_post_1_25.yaml` in base alla tua versione di Kubernetes.

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

Passaggi

1. Crea le risorse e distribuisce l'operatore:

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

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

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



Dovrebbe esserci solo **un'istanza** dell'operatore in un cluster Kubernetes. Non creare distribuzioni multiple dell'operatore Trident.

Passaggio 4: crea `TridentOrchestrator` e installa Trident

Ora puoi creare il `TridentOrchestrator` e installare Trident. Facoltativamente, puoi ["personalizza la tua installazione Trident"](#) utilizzando gli attributi nella `TridentOrchestrator` spec.

```

kubect1 create -f deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created

kubect1 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:26.02
    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:26.02.0
  Message:              Trident installed Namespace:
trident
  Status:                Installed
  Version:               v26.02.0
Events:
  Type Reason Age From Message ---- -
  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.

Utilizzo `TridentOrchestrator` status

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 lo stato `Failed` e l'operatore non è in grado di ripristinarsi autonomamente, ["controlla i registri"](#).

Stato	Descrizione
Installazione	L'operatore sta installando Trident utilizzando questo <code>TridentOrchestrator</code> CR.
Installato	Trident è stato installato correttamente.
Disinstallazione	L'operatore sta disinstallando Trident, perché <code>spec.uninstall=true</code> .
Disinstallato	Trident è disinstallato.
Fallito	L'operatore non è riuscito a installare, applicare patch, aggiornare o disinstallare Trident; l'operatore tenterà automaticamente di recuperare da questo stato. Se questo stato persiste, sarà necessaria la risoluzione dei problemi.
Aggiornamento	L'operatore sta aggiornando un'installazione esistente.
Errore	The <code>TridentOrchestrator</code> non è utilizzato. Ne esiste già un altro.

Utilizzo dello stato di creazione del pod

Puoi confermare se l'installazione di Trident è stata completata esaminando 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
```

Utilizzando `tridentctl`

Puoi usare `tridentctl` per verificare la versione di Trident installata.

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

```
+-----+-----+
| SERVER VERSION | CLIENT VERSION |
+-----+-----+
| 26.02.0        | 26.02.0        |
+-----+-----+
```

Distribuisci manualmente l'operatore Trident (modalità offline)

È possibile distribuire manualmente l'operatore Trident per installare Trident. Questa procedura si applica alle installazioni in cui le immagini dei container richieste da Trident sono archiviate in un registro privato. Se non si dispone di un registro immagini privato, utilizzare il ["processo per la distribuzione standard"](#).

Informazioni critiche su Trident

È necessario leggere le seguenti informazioni critiche su Trident.

Informazioni critiche su Trident

- Kubernetes 1.35 è ora supportato in Trident. Aggiornare Trident prima di aggiornare Kubernetes.
- Trident impone rigorosamente l'uso della configurazione multipathing negli ambienti SAN, con un valore consigliato di `find_multipaths: no` nel file `multipath.conf`.

L'utilizzo di una configurazione non multipath o l'utilizzo di `find_multipaths: yes` o `find_multipaths: smart` nel file `multipath.conf` causerà errori di montaggio. Trident ha raccomandato l'utilizzo di `find_multipaths: no` dalla release 21.07.

Distribuire manualmente l'operatore Trident e installare Trident

Verificate ["la 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` in tutti gli esempi che seguono e accedere prima come **system:admin** eseguendo `oc login -u system:admin` o `oc login -u kube-admin`.

1. Verifica la versione di Kubernetes:

```
kubectl version
```

2. Verificare i privilegi di amministratore del cluster:

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

3. Verifica di poter avviare un pod che utilizza un'immagine da Docker Hub e raggiungere il tuo sistema storage tramite la rete del pod:

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

Passaggio 1: Scarica il pacchetto di installazione di Trident

Il pacchetto di installazione Trident contiene tutto ciò che serve per distribuire l'operatore Trident e installare Trident. Scarica ed estrai la versione più recente del programma di installazione di Trident da ["la sezione Assets su GitHub"](#).

```
wget https://github.com/NetApp/trident/releases/download/v6.0/trident-
installer-26.02.0.tar.gz
tar -xf trident-installer-26.02.0.tar.gz
cd trident-installer
```

Passaggio 2: crea la TridentOrchestrator CRD

Crea la TridentOrchestrator Definizione di Risorsa Personalizzata (CRD). Creerai una TridentOrchestrator Custom Resources più tardi. Usa la versione YAML appropriata della CRD in [deploy/crds](#) per creare la TridentOrchestrator CRD:

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

Passaggio 3: Aggiorna la posizione del registro nell'operatore

In `/deploy/operator.yaml`, aggiorna `image: docker.io/netapp/trident-operator:26.02.0` per riflettere la posizione del tuo registro delle immagini. Le tue ["Trident e immagini CSI"](#) possono essere localizzate in un unico registro o in registri diversi, ma tutte le immagini CSI devono trovarsi nello stesso registro. Ad esempio:

- `image: <your-registry>/trident-operator:26.02.0` se tutte le immagini si trovano in un unico registro.

- `image: <your-registry>/netapp/trident-operator:26.02.0` se la tua immagine Trident si trova in un registro diverso rispetto alle tue immagini CSI.

Fase 4: distribuisce l'operatore Trident

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

- 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, utilizzare `bundle_post_1_25.yaml`.

Prima di iniziare

- Per impostazione predefinita, il programma di installazione di Trident distribuisce l'operatore nel `trident` namespace. Se il `trident` namespace non esiste, crearlo utilizzando:

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

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

- a. Crea `kustomization.yaml` utilizzando il seguente comando dove `<bundle.yaml>` è `bundle_pre_1_25.yaml` o `bundle_post_1_25.yaml` in base alla tua versione di Kubernetes.

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

- b. Compila il bundle usando il seguente comando, dove `<bundle.yaml>` è `bundle_pre_1_25.yaml` o `bundle_post_1_25.yaml` in base alla tua versione di Kubernetes.

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

Passaggi

1. Crea le risorse e distribuisce l'operatore:

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

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

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



Dovrebbe esserci solo **un'istanza** dell'operatore in un cluster Kubernetes. Non creare distribuzioni multiple dell'operatore Trident.

Fase 5: Aggiorna la posizione del registro delle immagini nel `TridentOrchestrator`

Le tue ["Trident e immagini CSI"](#) possono essere localizzate in un unico registro o in registri diversi, ma tutte le immagini CSI devono trovarsi nello stesso registro. Aggiorna `deploy/crds/tridentorchestrator_cr.yaml` per aggiungere le specifiche di posizione aggiuntive in base alla configurazione del tuo registro.

Immagini in un registry

```
imageRegistry: "<your-registry>"
autosupportImage: "<your-registry>/trident-autosupport:26.02"
tridentImage: "<your-registry>/trident:26.02.0"
```

Immagini in diversi registri

```
imageRegistry: "<your-registry>"
autosupportImage: "<your-registry>/trident-autosupport:26.02"
tridentImage: "<your-registry>/trident:26.02.0"
```

Passaggio 6: crea `TridentOrchestrator` e installa Trident

Ora puoi creare il `TridentOrchestrator` e installare Trident. Facoltativamente, puoi ["personalizza la tua installazione Trident"](#) ulteriormente utilizzando gli attributi nella `TridentOrchestrator` specifica. Il seguente esempio mostra un'installazione in cui le immagini Trident e CSI si trovano in registri diversi.

```

kubect1 create -f deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created

kubect1 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:26.02
  Debug:              true
  Image Registry:    <your-registry>
  Namespace:         trident
  Trident Image:     <your-registry>/trident:26.02.0
Status:
  Current Installation Params:
    IPv6:              false
    Autosupport Hostname:
    Autosupport Image: <your-registry>/trident-autosupport:26.02
    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:26.02.0
  Message:             Trident installed
  Namespace:           trident
  Status:               Installed
  Version:              v26.02.0
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.

Utilizzo `TridentOrchestrator` status

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 lo stato `Failed` e l'operatore non è in grado di ripristinarsi autonomamente, ["controlla i registri"](#).

Stato	Descrizione
Installazione	L'operatore sta installando Trident utilizzando questo <code>TridentOrchestrator</code> CR.
Installato	Trident è stato installato correttamente.
Disinstallazione	L'operatore sta disinstallando Trident, perché <code>spec.uninstall=true</code> .
Disinstallato	Trident è disinstallato.
Fallito	L'operatore non è riuscito a installare, applicare patch, aggiornare o disinstallare Trident; l'operatore tenterà automaticamente di recuperare da questo stato. Se questo stato persiste, sarà necessaria la risoluzione dei problemi.
Aggiornamento	L'operatore sta aggiornando un'installazione esistente.
Errore	The <code>TridentOrchestrator</code> non è utilizzato. Ne esiste già un altro.

Utilizzo dello stato di creazione del pod

Puoi confermare se l'installazione di Trident è stata completata esaminando 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
```

Utilizzando `tridentctl`

Puoi usare `tridentctl` per verificare la versione di Trident installata.

```
./tridentctl -n trident version

+-----+-----+
| SERVER VERSION | CLIENT VERSION |
+-----+-----+
| 26.02.0       | 26.02.0       |
+-----+-----+
```

Distribuisci l'operatore Trident tramite Helm (modalità standard)

È possibile distribuire l'operatore Trident e installare Trident utilizzando Helm. Questa procedura si applica alle installazioni in cui le immagini dei container richieste da Trident non sono archiviate in un registro privato. Se si dispone di un registro immagini privato, utilizzare il ["processo per la distribuzione offline"](#).

Informazioni critiche su Trident 25.10

È necessario leggere le seguenti informazioni critiche su Trident.

Informazioni critiche su Trident

- Kubernetes 1.35 è ora supportato in Trident. Aggiornare Trident prima di aggiornare Kubernetes.
- Trident impone rigorosamente l'uso della configurazione multipathing negli ambienti SAN, con un valore consigliato di `find_multipaths: no` nel file `multipath.conf`.

L'utilizzo di una configurazione non multipath o l'utilizzo di `find_multipaths: yes` o `find_multipaths: smart` nel file `multipath.conf` causerà errori di montaggio. Trident ha raccomandato l'utilizzo di `find_multipaths: no` dalla release 21.07.

Distribuire l'operatore Trident e installare Trident utilizzando Helm

Utilizzando il Trident ["Helm Chart"](#) puoi distribuire l'operatore Trident e installare Trident in un unico passaggio.

Verificate ["la 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 di deployment"](#) è necessario ["Helm versione 3"](#).

Passaggi

1. Aggiungi il repository Trident Helm:

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

2. Usa `helm install` e specifica un nome per la distribuzione come nel seguente esempio in cui `100..0` è la versione di Trident che stai installando.

```
helm install <name> netapp-trident/trident-operator --version 100.2602.0  
--create-namespace --namespace <trident-namespace>
```



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

È possibile utilizzare `helm list` per esaminare i dettagli dell'installazione, come nome, namespace, chart, 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
<code>--values (o -f)</code>	Specificare un file YAML con le sovrascritture. Questo può essere specificato più volte e il file più a destra avrà la precedenza.
<code>--set</code>	Specificare le sovrascritture sulla riga di comando.


Ad esempio, per modificare il valore predefinito di `debug`, eseguire il seguente comando dove `100.2602.0` è la versione di Trident che si sta installando:

```
helm install <name> netapp-trident/trident-operator --version 100.2602.0  
--create-namespace --namespace trident --set tridentDebug=true
```

Opzioni di configurazione

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


Opzione	Descrizione	Predefinito
<code>nodeSelector</code>	Etichette dei nodi per l'assegnazione dei pod	
<code>podAnnotations</code>	Annotazioni del pod	


Opzione	Descrizione	Predefinito
deploymentAnnotations	Annotazioni di distribuzione	
tolerations	Tolleranze per l'assegnazione dei pod	
affinity	Affinità per l'assegnazione dei pod	<pre> affinity: nodeAffinity: requiredDuringSchedulingIgnoredDuringExecution: nodeSelectorTerms: - matchExpressions: - key: kubernetes.io/arch operator: In values: - arm64 - amd64 - key: kubernetes.io/os operator: In values: - linux </pre> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;">  <p>Non rimuovere l'affinità predefinita dal file values.yaml. Quando vuoi fornire un'affinità personalizzata, estendi l'affinità predefinita.</p> </div>
tridentControllerPluginNodeSelector	Selettori di nodo aggiuntivi per i pod. Consultare Comprendere i pod controller e i pod nodo per i dettagli.	
tridentControllerPluginTolerations	Sostituisce le tolleranze di Kubernetes per i pod. Consultare Comprendere i pod controller e i pod nodo per i dettagli.	
tridentNodePluginNodeSelector	Selettori di nodo aggiuntivi per i pod. Consultare Comprendere i pod controller e i pod nodo per i dettagli.	

Opzione	Descrizione	Predefinito
<code>tridentNodePluginTolerations</code>	Sostituisce le tolleranze di Kubernetes per i pod. Consultare Comprendere i pod controller e i pod nodo per i dettagli.	
<code>imageRegistry</code>	Identifica il registro per le <code>trident-operator</code> , <code>trident</code> e altre immagini. Lascia vuoto per accettare l'impostazione predefinita. IMPORTANTE: Quando si installa Trident in un repository privato, se si utilizza l'opzione <code>imageRegistry</code> per specificare la posizione del repository, non usare <code>/netapp/</code> nel percorso del repository.	""
<code>imagePullPolicy</code>	Imposta il criterio di pull dell'immagine per il <code>trident-operator</code> .	<code>IfNotPresent</code>
<code>imagePullSecrets</code>	Imposta i segreti di estrazione delle immagini per le <code>trident-operator</code> , <code>trident</code> e altre immagini.	
<code>kubeletDir</code>	Consente di sovrascrivere la posizione host dello stato interno di kubelet.	<code>"/var/lib/kubelet"</code>
<code>operatorLogLevel</code>	Consente di impostare il livello di log dell'operatore Trident su: <code>trace</code> , <code>debug</code> , <code>info</code> , <code>warn</code> , <code>error</code> o <code>fatal</code> .	<code>"info"</code>
<code>operatorDebug</code>	Consente di impostare il livello di log dell'operatore Trident su <code>debug</code> .	<code>true</code>
<code>operatorImage</code>	Consente la completa sostituzione dell'immagine per <code>trident-operator</code> .	""
<code>operatorImageTag</code>	Permette di sovrascrivere il tag dell' <code>trident-operator</code> immagine.	""
<code>tridentIPv6</code>	Consente di abilitare Trident per funzionare in cluster IPv6.	<code>false</code>
<code>tridentK8sTimeout</code>	Sostituisce il timeout predefinito di 30 secondi per la maggior parte delle operazioni API di Kubernetes (se diverso da zero, in secondi).	<code>0</code>

Opzione	Descrizione	Predefinito
tridentHttpRequestTimeout	Sovrascrive il timeout predefinito di 90 secondi per le richieste HTTP, con 0s che rappresenta una durata infinita per il timeout. Non sono ammessi valori negativi.	"90s"
tridentSilenceAutoSupport	Consente di disabilitare la segnalazione periodica AutoSupport di Trident.	false
tridentAutoSupportImageTag	Consente di sovrascrivere il tag dell'immagine per il container Trident AutoSupport.	<version>
tridentAutoSupportProxy	Consente al container Trident AutoSupport di telefonare a casa tramite un proxy HTTP.	""
tridentLogFormat	Imposta il formato di logging Trident (text o json).	"text"
tridentDisableAuditLog	Disattiva il logger di audit di Trident.	true
tridentLogLevel	Consente di impostare il livello di log di Trident su: trace, debug, info, warn, error, o fatal.	"info"
tridentDebug	Consente di impostare il livello di log di Trident su debug. È possibile automatizzare il processo di distacco forzato tramite integrazione con node health check (NHC) operator. Per informazioni, vedi "Automatizzare il failover delle applicazioni stateful con Trident" .	false
tridentLogWorkflows	Consente di abilitare flussi di lavoro Trident specifici per la registrazione delle tracce o la soppressione dei log.	""
tridentLogLayers	Consente di abilitare specifici livelli Trident per la registrazione delle tracce o la soppressione dei log.	""
tridentImage	Consente la completa sostituzione dell'immagine per Trident.	""
tridentImageTag	Consente di sovrascrivere il tag dell'immagine per Trident.	""
tridentProbePort	Consente di sovrascrivere la porta predefinita utilizzata per le sonde liveness/readiness di Kubernetes.	""

Opzione	Descrizione	Predefinito
windows	Consente l'installazione di Trident sul nodo worker di Windows.	false
enableForceDetach	Consente di abilitare la funzione force detach.	false
excludePodSecurityPolicy	Esclude la pod security policy dell'operatore dalla creazione.	false
cloudProvider	Impostare su "Azure" quando si utilizzano identità gestite o un'identità cloud su un cluster AKS. Impostare su "AWS" quando si utilizza un'identità cloud su un cluster EKS.	""
cloudIdentity	Impostare su workload identity ("azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxx") quando si utilizza l'identità cloud su un cluster AKS. Impostare su AWS IAM role ("eks.amazonaws.com/role-arn: arn:aws:iam::123456:role/trident-role") quando si utilizza l'identità cloud su un cluster EKS.	""
iscsiSelfHealingInterval	L'intervallo al quale viene invocata la funzione di auto-riparazione iSCSI.	5m0s
iscsiSelfHealingWaitTime	La durata dopo la quale la funzione di auto-riparazione iSCSI avvia un tentativo di risolvere una sessione non aggiornata eseguendo un logout e un successivo login.	7m0s
nodePrep	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. NOTA: a partire da OpenShift 4.19, la versione minima di Trident supportata per questa funzionalità è la 25.06.1.	

Opzione	Descrizione	Predefinito
enableConcurrency	<p>Consente operazioni simultanee del controller Trident per un throughput migliorato.</p> <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;">  <p>Anteprima tecnica: questa funzionalità è sperimentale e attualmente supporta flussi di lavoro paralleli limitati con i driver ONTAP-NAS (solo NFS) e ONTAP-SAN (NVMe per unified ONTAP 9), oltre all'anteprima tecnica esistente per il driver ONTAP-SAN (protocolli iSCSI e FCP in unified ONTAP 9).</p> </div>	falso
k8sAPIQPS	<p>Limite di query al secondo (QPS) utilizzato dal controller durante la comunicazione con il server API Kubernetes. Il valore Burst viene impostato automaticamente in base al valore QPS.</p>	100; facoltativo

Opzione	Descrizione	Predefinito
resources	<p>Imposta i limiti delle risorse Kubernetes e le richieste per i pod del controller, del nodo e dell'operatore Trident. Puoi configurare CPU e memoria per ogni container e sidecar per gestire l'allocazione delle risorse in Kubernetes.</p> <p>Per ulteriori informazioni sulla configurazione delle richieste e dei limiti delle risorse, fare riferimento a "Gestione delle risorse per pod e container".</p> <div style="border: 1px solid #ccc; padding: 10px; margin-top: 20px;">  <ul style="list-style-type: none"> NON modificare i nomi di alcun container o campo. NON modificare l'indentazione - l'indentazione YAML è fondamentale per una corretta analisi. </div>	<pre>resources: controller: trident-main: requests: cpu: 10m memory: 80Mi limits: cpu: memory: csi-provisioner: requests: cpu: 2m memory: 20Mi limits: cpu: memory: csi-attacher: requests: cpu: 2m memory: 20Mi limits: cpu: memory: csi-resizer: requests: cpu: 3m memory: 20Mi limits: cpu: memory: csi-snapshotter: requests: cpu: 2m memory: 20Mi limits: cpu: memory: trident-autosupport: requests: cpu: 1m memory: 30Mi limits: cpu: memory: node: linux: trident-main:</pre>

Opzione	Descrizione	Predefinito
httpsMetrics	Abilita HTTPS per l'endpoint delle metriche Prometheus.	falso
hostNetwork	Abilita la rete host per il controller Trident. Questa funzionalità è utile quando si desidera separare il traffico frontend e backend in una rete multi-home.	falso

Comprendere i pod controller e i pod nodo

Trident viene eseguito come un singolo pod controller, più un pod nodo su ciascun nodo worker del cluster. Il pod nodo deve essere in esecuzione su qualsiasi host in cui si desidera potenzialmente montare un volume Trident.

Kubernetes "selettori di nodi" e "tolerazioni" sono utilizzati per vincolare un pod a essere eseguito su un nodo specifico o preferito. Utilizzando i NodeAffinityPlugin e NodeTolerationPlugin, puoi specificare vincoli e override.

- Il plugin del controller gestisce il provisioning e la gestione dei volumi, come snapshot e ridimensionamento.
- Il plugin del nodo gestisce il collegamento dello storage al nodo.

Distribuire l'operatore Trident utilizzando Helm (Offline mode)

È possibile distribuire l'operatore Trident e installare Trident utilizzando Helm. Questa procedura si applica alle installazioni in cui le immagini dei container richieste da Trident sono archiviate in un registro immagini. Se non si dispone di un registro immagini privato, utilizzare il "processo per la distribuzione standard".

Informazioni critiche su Trident 26.02

È necessario leggere le seguenti informazioni critiche su Trident.

Informazioni critiche su Trident

- Kubernetes 1.35 è ora supportato in Trident. Aggiornare Trident prima di aggiornare Kubernetes.
- Trident impone rigorosamente l'uso della configurazione multipathing negli ambienti SAN, con un valore consigliato di `find_multipaths: no` nel file `multipath.conf`.

L'utilizzo di una configurazione non `multipath` o l'utilizzo di `find_multipaths: yes` o `find_multipaths: smart` nel file `multipath.conf` causerà errori di montaggio. Trident ha raccomandato l'utilizzo di `find_multipaths: no` dalla release 21.07.

Distribuire l'operatore Trident e installare Trident utilizzando Helm

Utilizzando il Trident ["Helm Chart"](#) puoi distribuire l'operatore Trident e installare Trident in un unico passaggio.

Verificate ["la 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 di deployment"](#) è necessario ["Helm versione 3"](#).



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

Passaggi

1. Aggiungi il repository Trident Helm:

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

2. Usa `helm install` e specifica un nome per il deployment e la posizione del registro delle immagini. Le tue ["Trident e immagini CSI"](#) possono essere localizzate in un unico registro o in registri diversi, ma tutte le immagini CSI devono trovarsi nello stesso registro. Negli esempi, `100.2602.0` è la versione di Trident che stai installando.

Immagini in un registry

```
helm install <name> netapp-trident/trident-operator --version  
100.2602.0 --set imageRegistry=<your-registry> --create-namespace  
--namespace <trident-namespace> --set nodePrep={iscsi}
```

Immagini in diversi registri

```
helm install <name> netapp-trident/trident-operator --version  
100.2602.0 --set imageRegistry=<your-registry> --set operatorImage  
=<your-registry>/trident-operator:26.02.0 --set  
tridentAutosupportImage=<your-registry>/trident-autosupport:26.02  
--set tridentImage=<your-registry>/trident:26.02.0 --create  
-namespace --namespace <trident-namespace> --set nodePrep={iscsi}
```



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

È possibile utilizzare `helm list` per esaminare i dettagli dell'installazione, come nome, namespace, chart, 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
<code>--values (o -f)</code>	Specificare un file YAML con le sovrascritture. Questo può essere specificato più volte e il file più a destra avrà la precedenza.
<code>--set</code>	Specificare le sovrascritture sulla riga di comando.

Ad esempio, per modificare il valore predefinito di `debug`, eseguire il seguente comando dove `100.2602.0` è la versione di Trident che si sta installando:

```
helm install <name> netapp-trident/trident-operator --version 100.2602.0  
--create-namespace --namespace trident --set tridentDebug=true
```

Per aggiungere il valore `nodePrep`, eseguire il seguente comando:

```
helm install <name> netapp-trident/trident-operator --version 100.2602.0  
--create-namespace --namespace trident --set nodePrep={iscsi}
```


Opzioni di configurazione

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





Non rimuovere l'affinità predefinita dal file `values.yaml`. Quando vuoi fornire un'affinità personalizzata, estendi l'affinità predefinita.


Opzione	Descrizione	Predefinito
<code>nodeSelector</code>	Etichette dei nodi per l'assegnazione dei pod	
<code>podAnnotations</code>	Annotazioni del pod	
<code>deploymentAnnotations</code>	Annotazioni di distribuzione	
<code>tolerations</code>	Tolleranze per l'assegnazione dei pod	

Opzione	Descrizione	Predefinito
affinity	Affinità per l'assegnazione dei pod	<pre data-bbox="1047 157 1485 1144"> affinity: nodeAffinity: requiredDuringSchedulingIgnoredDuringExecution: nodeSelectorTerms: - matchExpressions: - key: kubernetes.io/arch operator: In values: - arm64 - amd64 - key: kubernetes.io/os operator: In values: - linux </pre> <div data-bbox="1071 1176 1485 1480" style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;">  <p>Non rimuovere l'affinità predefinita dal file values.yaml. Quando vuoi fornire un'affinità personalizzata, estendi l'affinità predefinita.</p> </div>
tridentControllerPluginNodeSelector	Selettori di nodo aggiuntivi per i pod. Consultare "Comprendere i pod controller e i pod nodo" per i dettagli.	
tridentControllerPluginTolerations	Sostituisce le tolleranze di Kubernetes per i pod. Consultare "Comprendere i pod controller e i pod nodo" per i dettagli.	

Opzione	Descrizione	Predefinito
<code>tridentNodePluginNodeSelector</code>	Selettori di nodo aggiuntivi per i pod. Consultare "Comprendere i pod controller e i pod nodo" per i dettagli.	
<code>tridentNodePluginTolerations</code>	Sostituisce le tolleranze di Kubernetes per i pod. Consultare "Comprendere i pod controller e i pod nodo" per i dettagli.	
<code>imageRegistry</code>	Identifica il registro per le <code>trident-operator</code> , <code>trident</code> e altre immagini. Lascia vuoto per accettare l'impostazione predefinita. IMPORTANTE: Quando si installa Trident in un repository privato, se si utilizza l'opzione <code>imageRegistry</code> per specificare la posizione del repository, non usare <code>/netapp/</code> nel percorso del repository.	""
<code>imagePullPolicy</code>	Imposta il criterio di pull dell'immagine per il <code>trident-operator</code> .	<code>IfNotPresent</code>
<code>imagePullSecrets</code>	Imposta i segreti di estrazione delle immagini per le <code>trident-operator</code> , <code>trident</code> e altre immagini.	
<code>kubeletDir</code>	Consente di sovrascrivere la posizione host dello stato interno di kubelet.	<code>"/var/lib/kubelet"</code>
<code>operatorLogLevel</code>	Consente di impostare il livello di log dell'operatore Trident su: <code>trace</code> , <code>debug</code> , <code>info</code> , <code>warn</code> , <code>error</code> o <code>fatal</code> .	<code>"info"</code>
<code>operatorDebug</code>	Consente di impostare il livello di log dell'operatore Trident su <code>debug</code> .	<code>true</code>
<code>operatorImage</code>	Consente la completa sostituzione dell'immagine per <code>trident-operator</code> .	""
<code>operatorImageTag</code>	Permette di sovrascrivere il tag dell' <code>trident-operator</code> immagine.	""
<code>tridentIPv6</code>	Consente di abilitare Trident per funzionare in cluster IPv6.	<code>false</code>

Opzione	Descrizione	Predefinito
tridentK8sTimeout	<p>Sostituisce il timeout predefinito di 180 secondi per la maggior parte delle operazioni API di Kubernetes (se diverso da zero, in secondi).</p> <div style="border: 1px solid gray; padding: 5px; margin: 10px 0;">  Il tridentK8sTimeout parametro è applicabile solo per l'installazione Trident. </div>	180
tridentHttpRequestTimeout	Sovrascrive il timeout predefinito di 90 secondi per le richieste HTTP, con 0s che rappresenta una durata infinita per il timeout. Non sono ammessi valori negativi.	"90s"
tridentSilenceAutosupport	Consente di disabilitare la segnalazione periodica AutoSupport di Trident.	false
tridentAutosupportImageTag	Consente di sovrascrivere il tag dell'immagine per il container Trident AutoSupport.	<version>
tridentAutosupportProxy	Consente al container Trident AutoSupport di telefonare a casa tramite un proxy HTTP.	""
tridentLogFormat	Imposta il formato di logging Trident (text o json).	"text"
tridentDisableAuditLog	Disattiva il logger di audit di Trident.	true
tridentLogLevel	Consente di impostare il livello di log di Trident su: trace, debug, info, warn, error, o fatal.	"info"
tridentDebug	Consente di impostare il livello di log di Trident su debug.	false
tridentLogWorkflows	Consente di abilitare flussi di lavoro Trident specifici per la registrazione delle tracce o la soppressione dei log.	""
tridentLogLayers	Consente di abilitare specifici livelli Trident per la registrazione delle tracce o la soppressione dei log.	""
tridentImage	Consente la completa sostituzione dell'immagine per Trident.	""
tridentImageTag	Consente di sovrascrivere il tag dell'immagine per Trident.	""

Opzione	Descrizione	Predefinito
tridentProbePort	Consente di sovrascrivere la porta predefinita utilizzata per le sonde liveness/readiness di Kubernetes.	""
windows	Consente l'installazione di Trident sul nodo worker di Windows.	false
enableForceDetach	Consente di abilitare la funzione force detach. È possibile automatizzare il processo di distacco forzato tramite integrazione con node health check (NHC) operator. Per informazioni, vedi "Automatizzare il failover delle applicazioni stateful con Trident" .	false
excludePodSecurityPolicy	Esclude la pod security policy dell'operatore dalla creazione.	false
nodePrep	<p>Consente a Trident di preparare i nodi del cluster Kubernetes per gestire i volumi utilizzando il protocollo storage specificato.</p> <p>Attualmente, <code>iscsi</code> è l'unico valore supportato.</p> <div style="border-left: 1px solid #ccc; padding-left: 10px; margin-top: 10px;">  <p>A partire da OpenShift 4.19, la versione minima di Trident supportata per questa funzionalità è la 25.06.1.</p> </div>	

Opzione	Descrizione	Predefinito
resources	<p>Imposta i limiti delle risorse Kubernetes e le richieste per i pod del controller, del nodo e dell'operatore Trident. Puoi configurare CPU e memoria per ogni container e sidecar per gestire l'allocazione delle risorse in Kubernetes.</p> <p>Per ulteriori informazioni sulla configurazione delle richieste e dei limiti delle risorse, fare riferimento a "Gestione delle risorse per pod e container".</p> <div style="border-left: 1px solid #ccc; padding-left: 10px; margin-left: 20px;">  <ul style="list-style-type: none"> NON modificare i nomi di alcun container o campo. NON modificare l'indentazione - l'indentazione YAML è fondamentale per una corretta analisi. </div>	<pre>resources: controller: trident-main: requests: cpu: 10m memory: 80Mi limits: cpu: memory: csi-provisioner: requests: cpu: 2m memory: 20Mi limits: cpu: memory: csi-attacher: requests: cpu: 2m memory: 20Mi limits: cpu: memory: csi-resizer: requests: cpu: 3m memory: 20Mi limits: cpu: memory: csi-snapshotter: requests: cpu: 2m memory: 20Mi limits: cpu: memory: trident- autosupport: requests: cpu: 1m memory: 30Mi limits: cpu: memory: node: linux:</pre>

Personalizzare l'installazione dell'operatore Trident

L'operatore Trident permette di personalizzare l'installazione Trident usando gli attributi nella TridentOrchestrator spec. Se vuoi personalizzare l'installazione oltre ciò che gli argomenti di TridentOrchestrator consentono, considera l'utilizzo di tridentctl per generare manifesti YAML personalizzati da modificare secondo necessità.

Comprendere i pod controller e i pod nodo

Trident viene eseguito come un singolo controller pod e un node pod su ogni nodo worker nel cluster. Il pod nodo deve essere in esecuzione su qualsiasi host in cui si desidera potenzialmente montare un volume Trident.

Kubernetes "selettori di nodi" e "tolleranze e taint" sono utilizzati per vincolare un pod a essere eseguito su un nodo specifico o preferito. Utilizzando ControllerPlugin e NodePlugin, puoi specificare vincoli e override.

- Il plugin del controller gestisce il provisioning e la gestione dei volumi, come snapshot e ridimensionamento.
- Il plugin del nodo gestisce il collegamento dello storage al nodo.

Opzioni di configurazione



spec.namespace è specificato in TridentOrchestrator per indicare il namespace in cui Trident è installato. Questo parametro non può essere aggiornato dopo che Trident è installato. Il tentativo di farlo fa sì che lo stato TridentOrchestrator cambi in Failed. Trident non è destinato a essere migrato tra namespace.

Questa tabella illustra TridentOrchestrator gli attributi.

Parametro	Descrizione	Predefinito
namespace	Spazio dei nomi in cui installare Trident	"default"
debug	Abilita il debug per Trident	false
enableForceDetach	ontap-san, ontap-san-economy, ontap-nas e ontap-nas-economy solo. Funziona con Kubernetes Non-Graceful Node Shutdown (NGNS) per garantire agli amministratori cluster la possibilità di migrare in modo sicuro i carichi di lavoro con volumi montati su nuovi nodi nel caso in cui un nodo diventi non sano. Per informazioni, vedi "Automatizzare il failover delle applicazioni stateful con Trident".	false
windows	L'impostazione su true abilita l'installazione sui nodi worker Windows.	false

```


trident-main:
  requests:
    cpu: 10m
  memory: 30Mi
  limits:
    cpu:
    memory:
node-driver-
  registrar:
  requests:
    cpu: 1m
  limits:
    cpu:
    memory:
windows:
  trident-main:
  requests:
    cpu: 6m
    memory: 40Mi
  cpu:
  memory:
node-driver-
  registrar:



```

```

cpu:
memory:
operator:
requests:
  cpu: 10m
  memory: 40Mi
limits:

```

Parametro	Descrizione	Predefinito
cloudProvider	Impostare su "Azure" quando si utilizzano identità gestite o un'identità cloud su un cluster AKS. Impostare su "AWS" quando si utilizza una cloud identity su un cluster EKS. Impostare su "GCP" quando si utilizza una cloud identity su un cluster GKE.	""
cloudIdentity	Impostare su workload identity ("azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx") quando si utilizza l'identità cloud su un cluster AKS. Impostare su AWS IAM role ("eks.amazonaws.com/role-arn: arn:aws:iam::123456:role/trident-role") quando si utilizza l'identità cloud su un cluster EKS. Impostare su cloud identity ("iam.gke.io/gcp-service-account: xxx@mygcpproject.iam.gserviceaccount.com") quando si utilizza cloud identity su un cluster GKE.	""
IPv6	Install Trident su IPv6	false
k8sTimeout	Timeout per le operazioni Kubernetes.  Il k8sTimeout parametro è applicabile solo per l'installazione Trident.	180sec
silenceAutosupport	Non inviare automaticamente i bundle di autosupport a NetApp	false
autosupportImage	L'immagine container per Autosupport Telemetry	"netapp/trident-autosupport10"
autosupportProxy	L'indirizzo/porta di un proxy per l'invio della Telemetria di 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	Trident image da installare	"netapp/trident:26.02"
imageRegistry	Percorso al registro interno, del formato <registry FQDN>[:port][subpath]	"registry.k8s.io"
kubeletDir	Percorso della directory kubelet sull'host	"/var/lib/kubelet"
wipeout	Un elenco di risorse da eliminare per eseguire una rimozione completa di Trident	
imagePullSecrets	Segreti per prelevare immagini da un registro interno	

Parametro	Descrizione	Predefinito
imagePullPolicy	Imposta la policy di pull dell'immagine per il Trident operator. I valori validi sono: Always per estrarre sempre l'immagine. IfNotPresent per estrarre l'immagine solo se non esiste già sul nodo. Never per non estrarre mai l'immagine.	IfNotPresent
controllerPluginNodeSelector	Selettori di nodo aggiuntivi per i pod. Segue lo stesso formato come <code>pod.spec.nodeSelector</code> .	Nessun valore predefinito; facoltativo
controllerPluginTolerations	Sostituisce le tolleranze di Kubernetes per i pod. Segue lo stesso formato di <code>pod.spec.Tolerations</code> .	Nessun valore predefinito; facoltativo
nodePluginNodeSelector	Selettori di nodo aggiuntivi per i pod. Segue lo stesso formato come <code>pod.spec.nodeSelector</code> .	Nessun valore predefinito; facoltativo
nodePluginTolerations	Sostituisce le tolleranze di Kubernetes per i pod. Segue lo stesso formato di <code>pod.spec.Tolerations</code> .	Nessun valore predefinito; facoltativo
nodePrep	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. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;">  A partire da OpenShift 4.19, la versione minima di Trident supportata per questa funzionalità è la 25.06.1. </div>	
k8sAPIQPS	Limite di query al secondo (QPS) utilizzato dal controller durante la comunicazione con il server API Kubernetes. Il valore Burst viene impostato automaticamente in base al valore QPS.	100; facoltativo
enableConcurrency	Consente operazioni simultanee del controller Trident per un throughput migliorato. <div style="border: 1px solid #ccc; padding: 5px; margin-top: 10px;">  Anteprima tecnica: questa funzionalità è sperimentale e attualmente supporta flussi di lavoro paralleli limitati con i driver ONTAP-NAS (solo NFS) e ONTAP-SAN (NVMe per unified ONTAP 9), oltre all'anteprima tecnica esistente per il driver ONTAP-SAN (protocolli iSCSI e FCP in unified ONTAP 9). </div>	falso

Parametro	Descrizione	Predefinito
resources	<p>Imposta i limiti delle risorse Kubernetes e le richieste per il controller Trident e i pod dei nodi. Puoi configurare CPU e memoria per ogni container e sidecar per gestire l'allocazione delle risorse in Kubernetes.</p> <p>Per ulteriori informazioni sulla configurazione delle richieste e dei limiti delle risorse, fare riferimento a "Gestione delle risorse per pod e container".</p> <ul style="list-style-type: none"> • NON modificare i nomi di alcun container o campo. • NON modificare l'indentazione - l'indentazione YAML è fondamentale per una corretta analisi. • Per impostazione predefinita non vengono applicati limiti: solo le richieste hanno valori predefiniti e sono applicate automaticamente se non specificate. • I nomi dei container sono elencati così come appaiono nelle specifiche del pod. • I sidecar sono elencati sotto ogni container principale. • Controllare il campo <code>status.CurrentInstallationParams.TORC</code> per visualizzare i valori attualmente applicati. 	<pre>resources: controller: trident- main: requests: cpu: 10m memory: 80Mi limits: cpu: memory: csi- provisioner: requests: cpu: 2m memory: 20Mi limits: cpu: memory: csi- attacher: requests: cpu: 2m memory: 20Mi limits: cpu: memory: csi- resizer: requests: cpu: 3m memory: 20Mi limits: cpu: memory: csi- snapshotter: requests: cpu: 2m memory: 20Mi limits:</pre>

Parametro	Descrizione	Predefinito
httpsMetrics	Abilita HTTPS per l'endpoint delle metriche Prometheus.	falso
hostNetwork	Abilita la rete host per il controller Trident. Questa funzionalità è utile quando si desidera separare il traffico frontend e backend in una rete multi-home.	falso



Per ulteriori informazioni sulla formattazione dei parametri del pod, consulta ["Assegnazione dei pod ai nodi"](#).

Configurazioni di esempio

È possibile utilizzare gli attributi in [Opzioni di configurazione](#) quando si definisce `TridentOrchestrator` per personalizzare l'installazione.

Configurazione personalizzata di base

Questo esempio, creato dopo l'esecuzione del comando `cat deploy/crds/tridentorchestrator_cr_imagepullsecrets.yaml`, rappresenta un'installazione personalizzata di base:

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

```
memory: 30Mi
limits:
  cpu:
  memory:
node:
linux:
  trident-
main:
```

```
1m
memory: 10Mi
limits:
  cpu:
memory:
  windows:
  trident-
main:
requests:
  cpu:
6m
memory: 40Mi
limits:
```

Selettori di nodo

Questo esempio installa Trident con node selectors.

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

```
memory:
liveness-
```

Nodi worker Windows

Questo esempio, creato dopo aver eseguito il `cat deploy/crds/tridentorchestrator_cr.yaml` comando, installa Trident su un nodo worker Windows.

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

Identità gestite su un cluster AKS

Questo esempio installa 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'uso 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-xxxxxxxxxxxxx'
```

Identità cloud su un cluster EKS

Questo esempio installa Trident per l'uso 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'uso 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
```

Configurazione delle richieste di risorse Kubernetes e dei limiti per il controller Trident e i pod dei nodi Linux Trident

Questo esempio configura le richieste e i limiti di risorse Kubernetes per il controller Trident e i pod Trident del nodo Linux.



Disclaimer: I valori di richiesta e limite forniti in questo esempio sono solo a scopo dimostrativo. Adatta questi valori in base al tuo ambiente e ai requisiti del carico di lavoro.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullSecrets:
  - thisisasecret
  resources:
    controller:
      trident-main:
        requests:
          cpu: 10m
          memory: 80Mi
        limits:
          cpu: 200m
          memory: 256Mi
    # sidecars
    csi-provisioner:
      requests:
        cpu: 2m
        memory: 20Mi
      limits:
        cpu: 100m
        memory: 64Mi
    csi-attacher:
      requests:
        cpu: 2m
        memory: 20Mi
      limits:
        cpu: 100m
        memory: 64Mi
    csi-resizer:
      requests:
        cpu: 3m
        memory: 20Mi
```

```
limits:
  cpu: 100m
  memory: 64Mi
csi-snapshotter:
  requests:
    cpu: 2m
    memory: 20Mi
  limits:
    cpu: 100m
    memory: 64Mi
trident-autosupport:
  requests:
    cpu: 1m
    memory: 30Mi
  limits:
    cpu: 50m
    memory: 128Mi
node:
  linux:
    trident-main:
      requests:
        cpu: 10m
        memory: 60Mi
      limits:
        cpu: 200m
        memory: 256Mi
    # sidecars
    node-driver-registrar:
      requests:
        cpu: 1m
        memory: 10Mi
      limits:
        cpu: 50m
        memory: 32Mi
```

Configurazione delle richieste di risorse Kubernetes e dei limiti per il controller Trident e per i pod dei nodi Trident Windows e Linux

Questo esempio configura le richieste e i limiti di risorse Kubernetes per il controller Trident e i pod Trident dei nodi Windows e Linux.



Disclaimer: I valori di richiesta e limite forniti in questo esempio sono solo a scopo dimostrativo. Adatta questi valori in base al tuo ambiente e ai requisiti del carico di lavoro.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullSecrets:
  - thisisasecret
  windows: true
  resources:
    controller:
      trident-main:
        requests:
          cpu: 10m
          memory: 80Mi
        limits:
          cpu: 200m
          memory: 256Mi
      # sidecars
      csi-provisioner:
        requests:
          cpu: 2m
          memory: 20Mi
        limits:
          cpu: 100m
          memory: 64Mi
      csi-attacher:
        requests:
          cpu: 2m
          memory: 20Mi
        limits:
          cpu: 100m
          memory: 64Mi
      csi-resizer:
        requests:
          cpu: 3m
```

```
    memory: 20Mi
  limits:
    cpu: 100m
    memory: 64Mi
csi-snapshotter:
  requests:
    cpu: 2m
    memory: 20Mi
  limits:
    cpu: 100m
    memory: 64Mi
trident-autosupport:
  requests:
    cpu: 1m
    memory: 30Mi
  limits:
    cpu: 50m
    memory: 128Mi
node:
  linux:
    trident-main:
      requests:
        cpu: 10m
        memory: 60Mi
      limits:
        cpu: 200m
        memory: 256Mi
    # sidecars
    node-driver-registrar:
      requests:
        cpu: 1m
        memory: 10Mi
      limits:
        cpu: 50m
        memory: 32Mi
  windows:
    trident-main:
      requests:
        cpu: 6m
        memory: 40Mi
      limits:
        cpu: 200m
        memory: 128Mi
    # sidecars
    node-driver-registrar:
      requests:
```

```
    cpu: 6m
    memory: 40Mi
  limits:
    cpu: 100m
    memory: 128Mi
  liveness-probe:
    requests:
      cpu: 2m
      memory: 40Mi
  limits:
    cpu: 50m
    memory: 64Mi
```

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

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

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

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

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

Informazioni sul marchio commerciale

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