USA Trident

Trident

NetApp
September 26, 2025

This PDF was generated from https://docs.netapp.com/it-it/trident/trident-use/worker-node-prep.html on September 26, 2025. Always check docs.netapp.com for the latest.

Sommario

USA Trident . . . 1
- Preparare il nodo di lavoro . . . 1
  - Selezionare gli strumenti giusti . . . 1
  - Rilevamento del servizio del nodo . . . 1
  - Volumi NFS . . . 2
  - Volumi iSCSI . . . 2
  - Volumi NVMe/TCP . . . 6
  - SCSI su volumi FC . . . 7
- Configurare e gestire i backend . . . 10
  - Configurare i backend . . . 10
  - Azure NetApp Files . . . 10
  - Google Cloud NetApp Volumes . . . 29
  - Configurare un Cloud Volumes Service per il backend di Google Cloud . . . 46
  - Configurare un backend NetApp HCI o SolidFire . . . 58
  - Driver SAN ONTAP . . . 63
  - Driver NAS ONTAP . . . 91
  - Amazon FSX per NetApp ONTAP . . . 127
  - Crea backend con kubectl . . . 162
  - Gestire i backend . . . 169
- Creare e gestire classi di archiviazione . . . 179
  - Creare una classe di storage . . . 179
  - Gestire le classi di storage . . . 182
- Provisioning e gestione dei volumi . . . 184
  - Provisioning di un volume . . . 184
  - Espandere i volumi . . . 188
  - Importa volumi . . . 199
  - Personalizzare i nomi e le etichette dei volumi . . . 207
  - Condividere un volume NFS tra spazi dei nomi . . . 210
  - Clona i volumi tra namespace . . . 214
  - Replica dei volumi con SnapMirror . . . 217
  - Utilizzare la topologia CSI . . . 223
  - Lavorare con le istantanee . . . 231
  - Lavorare con gli snapshot del gruppo di volumi . . . 239

USA Trident

Preparare il nodo di lavoro

Tutti i nodi di lavoro nel cluster Kubernetes devono essere in grado di montare i volumi forniti per i pod. Per preparare i nodi di lavoro, è necessario installare gli strumenti NFS, iSCSI, NVMe/TCP o FC in base alla selezione del driver.

Selezionare gli strumenti giusti

Se si utilizza una combinazione di driver, è necessario installare tutti gli strumenti necessari per i driver. Le versioni recenti di Red Hat Enterprise Linux CoreOS (RHCOS) hanno gli strumenti installati per impostazione predefinita.

Strumenti NFS

"Installare gli strumenti NFS" se si utilizza: `ontap-nas`, `ontap-nas-economy`, `ontap-nas-flexgroup`, `azure-netapp-files`, `gcp-cvs`.

Strumenti iSCSI

"Installare gli strumenti iSCSI" se si utilizza: `ontap-san`, `ontap-san-economy`, `solidfire-san`.

Strumenti NVMe

"Installazione degli strumenti NVMe" se si utilizza `ontap-san` Per il protocollo NVMe (nonvolatile Memory Express) su TCP (NVMe/TCP).

> NetApp consiglia ONTAP 9,12 o versione successiva per NVMe/TCP.

Strumenti SCSI su FC

Fare riferimento a "Modalità di configurazione degli host SAN FC FC-NVMe" per ulteriori informazioni sulla configurazione degli host SAN FC e FC-NVMe.

"Installare gli strumenti FC" Se si utilizza `ontap-san` con sanType `fcp` (SCSI su FC).

Punti da considerare: * SCSI su FC è supportato negli ambienti OpenShift e KubeVirt. * SCSI su FC non è supportato su Docker. * La riparazione automatica iSCSI non è applicabile a SCSI su FC.

Rilevamento del servizio del nodo

Trident tenta di rilevare automaticamente se il nodo può eseguire servizi iSCSI o NFS.

> Il rilevamento del servizio nodo identifica i servizi rilevati ma non garantisce che i servizi siano configurati correttamente. Al contrario, l'assenza di un servizio rilevato non garantisce il mancato funzionamento del montaggio del volume.

Rivedere gli eventi

Trident crea eventi per il nodo per identificare i servizi rilevati. Per rivedere questi eventi, eseguire:

```
kubectl get event -A --field-selector involvedObject.name=<Kubernetes node
name>
```

Esaminare i servizi rilevati

Trident identifica i servizi abilitati per ogni nodo sul nodo Trident CR. Per visualizzare i servizi rilevati, eseguire:

```
tridentctl get node -o wide -n <Trident namespace>
```

Volumi NFS

Installa gli strumenti NFS utilizzando i comandi del tuo sistema operativo. Assicurarsi che il servizio NFS venga avviato durante l'avvio.

RHEL 8+

```
sudo yum install -y nfs-utils
```

Ubuntu

```
sudo apt-get install -y nfs-common
```

> Riavviare i nodi di lavoro dopo aver installato gli strumenti NFS per evitare errori durante il collegamento dei volumi ai container.

Volumi iSCSI

Trident è in grado di stabilire automaticamente una sessione iSCSI, eseguire la scansione dei LUN e rilevare dispositivi multipath, formattarli e montarli su un pod.

Funzionalità di riparazione automatica di iSCSI

Per i sistemi ONTAP, Trident esegue la riparazione automatica iSCSI ogni cinque minuti per:

1. **Identificare** lo stato della sessione iSCSI desiderato e lo stato della sessione iSCSI corrente.
2. **Confrontare** lo stato desiderato con quello corrente per identificare le riparazioni necessarie. Trident determina le priorità di riparazione e quando prevenire le riparazioni.
3. **Eseguire le riparazioni** necessarie per riportare lo stato della sessione iSCSI corrente allo stato della sessione iSCSI desiderato.

> I log dell'attività di autoriparazione si trovano nel `trident-main` contenitore sul rispettivo pod Daemonset. Per visualizzare i log, è necessario impostare `debug` su "true" durante l'installazione di Trident.

Le funzionalità di riparazione automatica iSCSI di Trident possono contribuire a prevenire:

- Sessioni iSCSI obsolete o non funzionanti che potrebbero verificarsi dopo un problema di connettività di rete. Nel caso di una sessione obsoleta, Trident attende sette minuti prima di disconnettersi per ristabilire la connessione con un portale.

> Ad esempio, se i segreti CHAP sono stati ruotati sul controller di storage e la rete perde la connettività, i vecchi segreti CHAP (*stale*) potrebbero persistere. L'autoriparazione è in grado di riconoscerlo e ristabilire automaticamente la sessione per applicare i segreti CHAP aggiornati.

- Sessioni iSCSI mancanti
- LUN mancanti

Punti da considerare prima di aggiornare Trident

- Se sono in uso solo gli igroup per nodo (introdotti in 23,04+), la riparazione automatica iSCSI avvierà la riccansione SCSI per tutti i dispositivi nel bus SCSI.
- Se sono in uso solo gli igroup con ambito backend (deprecati da 23,04), la riparazione automatica iSCSI avvierà la riccansione SCSI per gli ID LUN esatti nel bus SCSI.
- Se si utilizza una combinazione di igroup per nodo e igroup con ambito backend, la riparazione automatica iSCSI avvierà la riccansione SCSI per gli ID LUN esatti nel bus SCSI.

Installare gli strumenti iSCSI

Installare gli strumenti iSCSI utilizzando i comandi del sistema operativo.

Prima di iniziare

- Ogni nodo del cluster Kubernetes deve avere un IQN univoco. **Questo è un prerequisito necessario**.
- Se si utilizza RHCOS versione 4.5 o successiva, o un'altra distribuzione Linux compatibile con RHEL, con `solidfire-san` Driver ed Element OS 12.5 o versioni precedenti, assicurarsi che l'algoritmo di autenticazione CHAP sia impostato su MD5 in `/etc/iscsi/iscsid.conf`. Gli algoritmi CHAP conformi a FIPS sicuri SHA1, SHA-256 e SHA3-256 sono disponibili con Element 12.7.

```
sudo sed -i 's/^\(node.session.auth.chap_algs\).*/\1 = MD5/'
/etc/iscsi/iscsid.conf
```

- Quando si utilizzano nodi di lavoro che eseguono RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) con iSCSI PVS, specificare il `discard` mount Option in StorageClass per eseguire il recupero dello spazio in linea. Fare riferimento alla "Documentazione di Red Hat".
- Assicurati di aver aggiornato all'ultima versione di `multipath-tools` .

RHEL 8+

1. Installare i seguenti pacchetti di sistema:

```
sudo yum install -y lsscsi iscsi-initiator-utils device-mapper-
multipath
```

2. Verificare che la versione di iscsi-initiator-utils sia 6.2.0.874-2.el7 o successiva:

```
rpm -q iscsi-initiator-utils
```

3. Impostare la scansione su manuale:

```
sudo sed -i 's/^\(node.session.scan\).*/\1 = manual/'
/etc/iscsi/iscsid.conf
```

4. Abilitare il multipathing:

```
sudo mpathconf --enable --with_multipathd y --find_multipaths n
```

> Assicurarsi che `/etc/multipath.conf` contenga `find_multipaths no` sotto `defaults`.

5. Assicurarsi che `iscsid` e. `multipathd` sono in esecuzione:

```
sudo systemctl enable --now iscsid multipathd
```

6. Attivare e avviare `iscsi`:

```
sudo systemctl enable --now iscsi
```

Ubuntu

1. Installare i seguenti pacchetti di sistema:

```
sudo apt-get install -y open-iscsi lsscsi sg3-utils multipath-tools
scsitools
```

2. Verificare che la versione Open-iscsi sia 2.0.874-5ubuntu2.10 o successiva (per il bionico) o 2.0.874-7.1ubuntu6.1 o successiva (per il focale):

```
dpkg -l open-iscsi
```

3. Impostare la scansione su manuale:

```
sudo sed -i 's/^\(node.session.scan\).*/\1 = manual/'
/etc/iscsi/iscsid.conf
```

4. Abilitare il multipathing:

```
sudo tee /etc/multipath.conf <<-EOF
defaults {
    user_friendly_names yes
    find_multipaths no
}
EOF
sudo systemctl enable --now multipath-tools.service
sudo service multipath-tools restart
```

> Assicurarsi che `/etc/multipath.conf` contenga `find_multipaths no` sotto `defaults`.

5. Assicurarsi che `open-iscsi` e. `multipath-tools` sono abilitati e in esecuzione:

```
sudo systemctl status multipath-tools
sudo systemctl enable --now open-iscsi.service
sudo systemctl status open-iscsi
```

> Per Ubuntu 18.04, è necessario rilevare le porte di destinazione con `iscsiadm` prima di iniziare `open-iscsi` Per avviare il daemon iSCSI. In alternativa, è possibile modificare `iscsi` servizio da avviare `iscsid` automaticamente.

Configurare o disattivare la riparazione automatica iSCSI

È possibile configurare le seguenti impostazioni di riparazione automatica iSCSI Trident per correggere le sessioni obsolete:

- **Intervallo di autoriparazione iSCSI**: Determina la frequenza con cui viene richiamata l'autoriparazione iSCSI (valore predefinito: 5 minuti). È possibile configurare l'esecuzione più frequente impostando un numero minore o meno frequente impostando un numero maggiore.

> Impostando l'intervallo di riparazione automatica iSCSI su 0 si arresta completamente la riparazione automatica iSCSI. Si sconsiglia di disattivare la funzionalità di riparazione automatica iSCSI; questa opzione deve essere disattivata solo in alcuni scenari quando la riparazione automatica iSCSI non funziona come previsto o a scopo di debug.

- **Tempo di attesa per la riparazione automatica iSCSI**: Determina la durata di attesa per la riparazione automatica iSCSI prima di uscire da una sessione non corretta e di tentare nuovamente l'accesso (valore predefinito: 7 minuti). È possibile configurarlo su un numero maggiore in modo che le sessioni identificate come non integre debbano attendere più a lungo prima di essere disconnesse e quindi venga effettuato un tentativo di riconnessione o un numero minore per disconnettersi e accedere in precedenza.

Timone

Per configurare o modificare le impostazioni di riparazione automatica iSCSI, passare il `iscsiSelfHealingInterval` e. `iscsiSelfHealingWaitTime` parametri durante l'installazione del timone o l'aggiornamento del timone.

Il seguente esempio imposta l'intervallo di riparazione automatica iSCSI su 3 minuti e il tempo di attesa di riparazione automatica su 6 minuti:

```
helm install trident trident-operator-100.2506.0.tgz --set
iscsiSelfHealingInterval=3m0s --set iscsiSelfHealingWaitTime=6m0s -n
trident
```

tridentctl

Per configurare o modificare le impostazioni di riparazione automatica iSCSI, passare il `iscsi-self-healing-interval` e. `iscsi-self-healing-wait-time` parametri durante l'installazione o l'aggiornamento di tridentctl.

Il seguente esempio imposta l'intervallo di riparazione automatica iSCSI su 3 minuti e il tempo di attesa di riparazione automatica su 6 minuti:

```
tridentctl install --iscsi-self-healing-interval=3m0s --iscsi-self
-healing-wait-time=6m0s -n trident
```

Volumi NVMe/TCP

Installa gli strumenti NVMe utilizzando i comandi del tuo sistema operativo.

> - NVMe richiede RHEL 9 o versione successiva.
> - Se la versione del kernel del nodo Kubernetes è troppo vecchia o se il pacchetto NVMe non è disponibile per la versione del kernel in uso, potrebbe essere necessario aggiornare la versione del kernel del nodo a una versione con il pacchetto NVMe.

RHEL 9

```
sudo yum install nvme-cli
sudo yum install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

Ubuntu

```
sudo apt install nvme-cli
sudo apt -y install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

Verificare l'installazione

Dopo l'installazione, verificare che ogni nodo nel cluster Kubernetes disponga di un NQN univoco utilizzando il comando:

```
cat /etc/nvme/hostnqn
```

> Trident modifica il `ctrl_device_tmo` valore per garantire che NVMe non ceda sul percorso in caso di arresti. Non modificare questa impostazione.

SCSI su volumi FC

Da oggi è possibile utilizzare il protocollo Fibre Channel (FC) con Trident per il provisioning e la gestione delle risorse di storage sul sistema ONTAP.

Prerequisiti

Configurare le impostazioni di rete e del nodo richieste per FC.

Impostazioni di rete

1. Ottenere il WWPN delle interfacce di destinazione. Per ulteriori informazioni, fare riferimento "visualizzazione dell'interfaccia di rete" a.
2. Ottenere il WWPN per le interfacce su iniziatore (host).

   Fare riferimento alle utility del sistema operativo host corrispondenti.

3. Configurare lo zoning sullo switch FC utilizzando i WWPN dell'host e della destinazione.

   Per informazioni, fare riferimento alla documentazione relativa del fornitore dell'interruttore.

   Per ulteriori informazioni, consultare la seguente documentazione di ONTAP:

   - "Panoramica dello zoning FCoE e Fibre Channel"

- "Modalità di configurazione degli host SAN FC FC-NVMe"

Installare gli strumenti FC

Installa gli strumenti FC utilizzando i comandi del tuo sistema operativo.

- Quando si utilizzano nodi di lavoro che eseguono RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) con FC PVS, specificare il `discard` mount Option in StorageClass per eseguire il recupero dello spazio in linea. Fare riferimento alla "Documentazione di Red Hat".

RHEL 8+

1. Installare i seguenti pacchetti di sistema:

```
sudo yum install -y lsscsi device-mapper-multipath
```

2. Abilitare il multipathing:

```
sudo mpathconf --enable --with_multipathd y --find_multipaths n
```

> Assicurarsi che `/etc/multipath.conf` contenga `find_multipaths no` sotto `defaults`.

3. Assicurarsi che `multipathd` sia in esecuzione:

```
sudo systemctl enable --now multipathd
```

Ubuntu

1. Installare i seguenti pacchetti di sistema:

```
sudo apt-get install -y lsscsi sg3-utils multipath-tools scsitools
```

2. Abilitare il multipathing:

```
sudo tee /etc/multipath.conf <<-EOF
defaults {
    user_friendly_names yes
    find_multipaths no
}
EOF
sudo systemctl enable --now multipath-tools.service
sudo service multipath-tools restart
```

> Assicurarsi che `/etc/multipath.conf` contenga `find_multipaths no` sotto `defaults`.

3. Assicurarsi che `multipath-tools` sia attivato e in esecuzione:

```
sudo systemctl status multipath-tools
```

Configurare e gestire i backend

Configurare i backend

Un backend definisce la relazione tra Trident e un sistema di storage. Spiega a Trident come comunicare con quel sistema storage e come Trident dovrebbe eseguire il provisioning dei volumi da esso.

Trident offre automaticamente i pool di storage dai backend che soddisfano i requisiti definiti da una classe storage. Scopri come configurare il back-end per il tuo sistema storage.

- "Configurare un backend Azure NetApp Files"
- "Configurare un backend Google Cloud NetApp Volumes"
- "Configurare un Cloud Volumes Service per il backend della piattaforma cloud Google"
- "Configurare un backend NetApp HCI o SolidFire"
- "Configurare un backend con driver NAS ONTAP o Cloud Volumes ONTAP"
- "Configurare un backend con i driver SAN ONTAP o Cloud Volumes ONTAP"
- "USA Trident con Amazon FSX per NetApp ONTAP"

Azure NetApp Files

Configurare un backend Azure NetApp Files

È possibile configurare Azure NetApp Files come backend per Trident. È possibile collegare volumi NFS e SMB utilizzando un backend Azure NetApp Files. Trident supporta inoltre la gestione delle credenziali utilizzando identità gestite per i cluster Azure Kubernetes Services (AKS).

Dettagli del driver Azure NetApp Files

Trident fornisce i seguenti driver di storage Azure NetApp Files per comunicare con il cluster. Le modalità di accesso supportate sono: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Driver	Protocollo	VolumeMode	Modalità di accesso supportate	File system supportati
`azure-netapp-files`	NFS PMI	Filesystem	RWO, ROX, RWX, RWOP	`nfs`, `smb`

Considerazioni

- Il servizio Azure NetApp Files non supporta volumi inferiori a 50 GiB. Trident crea automaticamente volumi 50-GiB se è richiesto un volume più piccolo.
- Trident supporta volumi SMB montati su pod in esecuzione solo sui nodi Windows.

Identità gestite per AKS

Trident supporta "identità gestite" i cluster di Azure Kubernetes Services. Per sfruttare al meglio la gestione semplificata delle credenziali offerta dalle identità gestite, è necessario disporre di:

- Un cluster Kubernetes implementato utilizzando AKS
- Identità gestite configurate sul cluster AKS kuBoost
- Trident installato che include `cloudProvider` per specificare `"Azure"`.

Operatore Trident

Per installare Trident utilizzando l'operatore Trident, `tridentorchestrator_cr.yaml` impostare su `cloudProvider` `"Azure"` . Ad esempio:

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

Timone

Nell'esempio seguente vengono installati i set Trident `cloudProvider` in Azure utilizzando la variabile di ambiente `$CP` :

```
helm install trident trident-operator-100.2506.0.tgz --create
-namespace --namespace <trident-namespace> --set cloudProvider=$CP
```

<code>tridentctl</code>

Nell'esempio seguente viene installato Trident e viene impostato il `cloudProvider` flag su `Azure`:

```
tridentctl install --cloud-provider="Azure" -n trident
```

Identità cloud per AKS

L'identità del cloud consente ai pod Kubernetes di accedere alle risorse Azure autenticandosi come identità del carico di lavoro invece di fornire credenziali Azure esplicite.

Per sfruttare l'identità cloud in Azure è necessario disporre di:

- Un cluster Kubernetes implementato utilizzando AKS

- Identità del workload e issuer oidc configurati nel cluster AKS Kubernetes
- Trident installato che include `cloudProvider` per specificare `"Azure"` e `cloudIdentity` specificare l'identità del workload

Operatore Trident

Per installare Trident utilizzando l'operatore Trident, `tridentorchestrator_cr.yaml` `"Azure"` impostare su `cloudProvider` e `cloudIdentity` su `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxx`.

Ad esempio:

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

Timone

Impostare i valori per i flag **cloud-provider (CP)** e **cloud-Identity (ci)** utilizzando le seguenti variabili di ambiente:

```
export CP="Azure"
export CI="'azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxx'"
```

Nell'esempio seguente viene installato Trident e impostato `cloudProvider` su Azure utilizzando la variabile di ambiente `$CP` e viene impostata la `cloudIdentity` variabile di ambiente Using the `$CI` :

```
helm install trident trident-operator-100.6.0.tgz --set
cloudProvider=$CP --set cloudIdentity="$CI"
```

<code>tridentctl</code>

Impostare i valori per i flag **cloud provider** e **cloud Identity** utilizzando le seguenti variabili di ambiente:

```
export CP="Azure"
export CI="azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxx"
```

Nell'esempio seguente viene installato Trident e viene impostato il `cloud-provider` flag su `$CP`, e `cloud-identity` su `$CI`:

```
tridentctl install --cloud-provider=$CP --cloud-identity="$CI" -n
trident
```

Prepararsi a configurare un backend Azure NetApp Files

Prima di poter configurare il backend Azure NetApp Files, è necessario assicurarsi che siano soddisfatti i seguenti requisiti.

Prerequisiti per volumi NFS e SMB

Se si utilizza Azure NetApp Files per la prima volta o in una nuova posizione, è necessaria una configurazione iniziale per configurare Azure NetApp Files e creare un volume NFS. Fare riferimento a. "Azure: Configura Azure NetApp Files e crea un volume NFS".

Per configurare e utilizzare un "Azure NetApp Files" back-end, sono necessari i seguenti elementi:

> - `subscriptionID`, `tenantID`, `clientID`, `location`, e. `clientSecret` Sono opzionali quando si utilizzano identità gestite su un cluster AKS.
> - `tenantID`, `clientID`, e. `clientSecret` Sono opzionali quando si utilizza un'identità cloud su un cluster AKS.

- Un pool di capacità. Fare riferimento a. "Microsoft: Creare un pool di capacità per Azure NetApp Files".
- Una subnet delegata a Azure NetApp Files. Fare riferimento a. "Microsoft: Delegare una subnet a Azure NetApp Files".
- `subscriptionID` Da un abbonamento Azure con Azure NetApp Files attivato.
- `tenantID`, `clientID`, e. `clientSecret` da un "Registrazione dell'app" In Azure Active Directory con autorizzazioni sufficienti per il servizio Azure NetApp Files. La registrazione dell'applicazione deve utilizzare:
  - Il ruolo di Proprietario o collaboratore "Predefinito da Azure".
  - A "Ruolo di collaboratore personalizzato" al livello di sottoscrizione (`assignableScopes`) con le seguenti autorizzazioni che sono limitate solo a ciò che Trident richiede. Dopo aver creato il ruolo personalizzato, "Assegnare il ruolo utilizzando il portale Azure".

Ruolo collaboratore personalizzato

```
{
  "id": "/subscriptions/<subscription-
id>/providers/Microsoft.Authorization/roleDefinitions/<role-
definition-id>",
  "properties": {
    "roleName": "custom-role-with-limited-perms",
    "description": "custom role providing limited permissions",
    "assignableScopes": [
      "/subscriptions/<subscription-id>"
    ],
    "permissions": [
      {
        "actions": [
          "Microsoft.NetApp/netAppAccounts/capacityPools/read",
          "Microsoft.NetApp/netAppAccounts/capacityPools/write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/read",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/delete",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
read",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
delete",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/MountTarge
ts/read",
          "Microsoft.Network/virtualNetworks/read",
          "Microsoft.Network/virtualNetworks/subnets/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/write",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/delete",
```

```
            "Microsoft.Features/features/read",
            "Microsoft.Features/operations/read",
            "Microsoft.Features/providers/features/read",

"Microsoft.Features/providers/features/register/action",

"Microsoft.Features/providers/features/unregister/action",

"Microsoft.Features/subscriptionFeatureRegistrations/read"
          ],
          "notActions": [],
          "dataActions": [],
          "notDataActions": []
        }
      ]
    }
}
```

- Azure `location` che ne contiene almeno uno "subnet delegata". A partire da Trident 22.01, il `location` parametro è un campo obbligatorio al livello superiore del file di configurazione back-end. I valori di posizione specificati nei pool virtuali vengono ignorati.
- Da utilizzare `Cloud Identity`, ottenere il `client ID` da un "identità gestita assegnata dall'utente" E specificare tale ID in `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx`.

Requisiti aggiuntivi per i volumi SMB

Per creare un volume SMB, è necessario disporre di:

- Active Directory configurato e connesso a Azure NetApp Files. Fare riferimento a. "Microsoft: Creazione e gestione delle connessioni Active Directory per Azure NetApp Files".
- Un cluster Kubernetes con un nodo controller Linux e almeno un nodo di lavoro Windows che esegue Windows Server 2022. Trident supporta volumi SMB montati su pod in esecuzione solo sui nodi Windows.
- Almeno un segreto Trident contenente le credenziali di Active Directory in modo che Azure NetApp Files possa autenticarsi ad Active Directory. Per generare segreto `smbcreds`:

  ```
  kubectl create secret generic smbcreds --from-literal username=user
  --from-literal password='password'
  ```

- Proxy CSI configurato come servizio Windows. Per configurare un `csi-proxy`, fare riferimento a. "GitHub: Proxy CSI" oppure "GitHub: Proxy CSI per Windows" Per i nodi Kubernetes in esecuzione su Windows.

Opzioni di configurazione back-end Azure NetApp Files ed esempi

Scopri le opzioni di configurazione di back-end NFS e SMB per Azure NetApp Files e consulta gli esempi di configurazione.

Opzioni di configurazione back-end

Trident utilizza la tua configurazione back-end (subnet, rete virtuale, livello di servizio e posizione) per creare volumi Azure NetApp Files su pool di capacità disponibili nel percorso richiesto e corrispondenti al livello di servizio e alla subnet richiesti.

> * A partire dalla versione NetApp Trident 25.06, i pool di capacità QoS manuali sono supportati come anteprima tecnologica.*

I backend Azure NetApp Files forniscono queste opzioni di configurazione.

Parametro	Descrizione	Predefinito
`version`		Sempre 1
`storageDriverName`	Nome del driver di storage	"azure-netapp-files"
`backendName`	Nome personalizzato o backend dello storage	Nome del driver + "_" + caratteri casuali
`subscriptionID`	L'ID dell'abbonamento dell'abbonamento Azure Opzionale quando le identità gestite sono abilitate su un cluster AKS.	
`tenantID`	L'ID tenant di una registrazione app Opzionale quando si utilizzano identità gestite o identità cloud su un cluster AKS.	
`clientID`	L'ID client di una registrazione dell'applicazione Opzionale quando si utilizzano identità gestite o identità cloud su un cluster AKS.	
`clientSecret`	Il segreto del client da una registrazione dell'applicazione Opzionale quando si utilizzano identità gestite o identità cloud su un cluster AKS.	
`serviceLevel`	Uno di `Standard`, `Premium`, o. `Ultra`	"" (casuale)
`location`	Nome della posizione di Azure in cui verranno creati i nuovi volumi Opzionale quando le identità gestite sono abilitate su un cluster AKS.	
`resourceGroups`	Elenco dei gruppi di risorse per filtrare le risorse rilevate	"[]" (nessun filtro)

Parametro	Descrizione	Predefinito
`netappAccounts`	Elenco degli account NetApp per il filtraggio delle risorse rilevate	"[]" (nessun filtro)
`capacityPools`	Elenco dei pool di capacità per filtrare le risorse rilevate	"[]" (nessun filtro, casuale)
`virtualNetwork`	Nome di una rete virtuale con una subnet delegata	""
`subnet`	Nome di una subnet delegata a. `Microsoft.Netapp/volumes`	""
`networkFeatures`	Serie di funzionalità VNET per un volume, potrebbe essere `Basic` oppure `Standard`. Le funzioni di rete non sono disponibili in tutte le regioni e potrebbero essere abilitate in un abbonamento. Specificare `networkFeatures` se la funzionalità non è attivata, il provisioning del volume non viene eseguito correttamente.	""
`nfsMountOptions`	Controllo dettagliato delle opzioni di montaggio NFS. Ignorato per i volumi SMB. Per montare i volumi utilizzando NFS versione 4.1, include `nfsvers=4` Nell'elenco delle opzioni di montaggio delimitate da virgole, scegliere NFS v4.1. Le opzioni di montaggio impostate in una definizione di classe di storage sovrascrivono le opzioni di montaggio impostate nella configurazione backend.	"nfsvers=3"
`limitVolumeSize`	Il provisioning non riesce se le dimensioni del volume richiesto sono superiori a questo valore	"" (non applicato per impostazione predefinita)
`debugTraceFlags`	Flag di debug da utilizzare per la risoluzione dei problemi. Esempio, `\{"api": false, "method": true, "discovery": true}`. Non utilizzare questa opzione a meno che non si stia eseguendo la risoluzione dei problemi e non si richieda un dump dettagliato del log.	nullo
`nasType`	Configurare la creazione di volumi NFS o SMB. Le opzioni sono `nfs`, `smb` o nullo. L'impostazione su Null consente di impostare i volumi NFS come predefiniti.	`nfs`

Parametro	Descrizione	Predefinito
`supportedTopologies`	Rappresenta un elenco di aree e zone supportate da questo backend. Per ulteriori informazioni, fare riferimento a "Utilizzare la topologia CSI".	
`qosType`	Rappresenta il tipo di QoS: automatico o manuale. **Anteprima tecnica per Trident 25.06**	Auto
`maxThroughput`	Imposta la velocità massima consentita. Supportato solo per pool di capacità QoS manuali. **Anteprima tecnica per Trident 25.06**	`4 MiB/sec`

Per ulteriori informazioni sulle funzioni di rete, fare riferimento a. "Configurare le funzionalità di rete per un volume Azure NetApp Files".

Autorizzazioni e risorse richieste

Se viene visualizzato l'errore "Nessun pool di capacità trovato" durante la creazione di un PVC, è probabile che la registrazione dell'app non disponga delle autorizzazioni e delle risorse necessarie (subnet, rete virtuale, pool di capacità) associate. Se il debug è attivato, Trident registrerà le risorse di Azure rilevate al momento della creazione del backend. Verificare che venga utilizzato un ruolo appropriato.

I valori per `resourceGroups`, `netappAccounts`, `capacityPools`, `virtualNetwork`, e. `subnet` può essere specificato utilizzando nomi brevi o completi. Nella maggior parte dei casi, si consiglia di utilizzare nomi completi, in quanto i nomi brevi possono corrispondere a più risorse con lo stesso nome.

Il `resourceGroups`, `netappAccounts`, e. `capacityPools` i valori sono filtri che limitano l'insieme di risorse rilevate a quelle disponibili per questo backend di storage e possono essere specificati in qualsiasi combinazione. I nomi pienamente qualificati seguono questo formato:

Tipo	Formato
Gruppo di risorse	<resource group>
Account NetApp	<resource group>/<netapp account>
Pool di capacità	<resource group>/<netapp account>/<capacity pool>
Rete virtuale	<resource group>/<virtual network>
Subnet	<resource group>/<virtual network>/<subnet>

Provisioning di volumi

È possibile controllare il provisioning del volume predefinito specificando le seguenti opzioni in una sezione speciale del file di configurazione. Fare riferimento a. Configurazioni di esempio per ulteriori informazioni.

Parametro	Descrizione	Predefinito
`exportRule`	Regole di esportazione per nuovi volumi. `exportRule` Deve essere un elenco separato da virgole di qualsiasi combinazione di indirizzi IPv4 o subnet IPv4 nella notazione CIDR. Ignorato per i volumi SMB.	"0.0.0.0/0"
`snapshotDir`	Controlla la visibilità della directory .snapshot	"True" per NFSv4 "false" per NFSv3
`size`	La dimensione predefinita dei nuovi volumi	"100 G"
`unixPermissions`	Le autorizzazioni unix dei nuovi volumi (4 cifre ottali). Ignorato per i volumi SMB.	"" (funzione di anteprima, richiede la whitelist nell'abbonamento)

Configurazioni di esempio

Gli esempi seguenti mostrano le configurazioni di base che lasciano la maggior parte dei parametri predefiniti. Questo è il modo più semplice per definire un backend.

Configurazione minima

Questa è la configurazione backend minima assoluta. Con questa configurazione, Trident rileva tutti gli account NetApp, i pool di capacità e le subnet delegate a Azure NetApp Files nella posizione configurata e posiziona i nuovi volumi in uno di tali pool e subnet in modo casuale. Poiché `nasType` viene omesso, viene applicato il `nfs` valore predefinito e il backend esegue il provisioning dei volumi NFS.

Questa configurazione è l'ideale se stai iniziando a utilizzare Azure NetApp Files e provando qualcosa, ma in pratica vorresti fornire un ulteriore ambito per i volumi da te forniti.

```
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
  tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
  clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
  clientSecret: SECRET
  location: eastus
```

Identità gestite per AKS

Questa configurazione di backend omette `subscriptionID`, `tenantID`, `clientID`, e. `clientSecret`, che sono opzionali quando si utilizzano identità gestite.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  capacityPools:
    - ultra-pool
  resourceGroups:
    - aks-ami-eastus-rg
  netappAccounts:
    - smb-na
  virtualNetwork: eastus-prod-vnet
  subnet: eastus-anf-subnet
```

Identità cloud per AKS

Questa configurazione di backend omette `tenantID`, `clientID`, e. `clientSecret`, che sono opzionali quando si utilizza un'identità cloud.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  capacityPools:
    - ultra-pool
  resourceGroups:
    - aks-ami-eastus-rg
  netappAccounts:
    - smb-na
  virtualNetwork: eastus-prod-vnet
  subnet: eastus-anf-subnet
  location: eastus
  subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
```

Configurazione specifica del livello di servizio con filtri pool di capacità

Questa configurazione backend colloca i volumi nella posizione di Azure `eastus` in un `Ultra` pool di capacità. Trident rileva automaticamente tutte le subnet delegate a Azure NetApp Files in tale posizione e posiziona un nuovo volume su una di esse in modo casuale.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
serviceLevel: Ultra
capacityPools:
  - application-group-1/account-1/ultra-1
  - application-group-1/account-1/ultra-2
```

Esempio di backend con pool di capacità QoS manuali

Questa configurazione del backend posiziona i volumi in Azure `eastus` posizione con pool di capacità QoS manuali. **Anteprima tecnologica in NetApp Trident 25.06**.

```
---
version: 1
storageDriverName: azure-netapp-files
backendName: anf1
location: eastus
labels:
  clusterName: test-cluster-1
  cloud: anf
  nasType: nfs
defaults:
  qosType: Manual
storage:
  - serviceLevel: Ultra
    labels:
      performance: gold
    defaults:
      maxThroughput: 10
  - serviceLevel: Premium
    labels:
      performance: silver
    defaults:
      maxThroughput: 5
  - serviceLevel: Standard
    labels:
      performance: bronze
    defaults:
      maxThroughput: 3
```

Configurazione avanzata

Questa configurazione di back-end riduce ulteriormente l'ambito del posizionamento del volume in una singola subnet e modifica alcune impostazioni predefinite di provisioning del volume.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
serviceLevel: Ultra
capacityPools:
  - application-group-1/account-1/ultra-1
  - application-group-1/account-1/ultra-2
virtualNetwork: my-virtual-network
subnet: my-subnet
networkFeatures: Standard
nfsMountOptions: vers=3,proto=tcp,timeo=600
limitVolumeSize: 500Gi
defaults:
  exportRule: 10.0.0.0/24,10.0.1.0/24,10.0.2.100
  snapshotDir: "true"
  size: 200Gi
  unixPermissions: "0777"
```

Configurazione dei pool virtuali

Questa configurazione di back-end definisce più pool di storage in un singolo file. Ciò è utile quando si dispone di più pool di capacità che supportano diversi livelli di servizio e si desidera creare classi di storage in Kubernetes che ne rappresentano. Le etichette dei pool virtuali sono state utilizzate per differenziare i pool in base a. `performance`.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
resourceGroups:
  - application-group-1
networkFeatures: Basic
nfsMountOptions: vers=3,proto=tcp,timeo=600
labels:
  cloud: azure
storage:
  - labels:
      performance: gold
    serviceLevel: Ultra
    capacityPools:
      - ultra-1
      - ultra-2
    networkFeatures: Standard
  - labels:
      performance: silver
    serviceLevel: Premium
    capacityPools:
      - premium-1
  - labels:
      performance: bronze
    serviceLevel: Standard
    capacityPools:
      - standard-1
      - standard-2
```

Configurazione delle topologie supportate

Trident facilita il provisioning dei volumi per i workload in base a regioni e zone di disponibilità. Il `supportedTopologies` blocco in questa configurazione backend viene utilizzato per fornire un elenco di aree e zone per backend. I valori di regione e zona specificati qui devono corrispondere ai valori di regione e zona dalle etichette su ogni nodo del cluster Kubernetes. Queste regioni e zone rappresentano l'elenco dei valori consentiti che possono essere forniti in una classe di archiviazione. Per le classi di archiviazione che contengono un sottoinsieme delle aree e delle zone fornite in un backend, Trident crea volumi nell'area e nella zona menzionate. Per ulteriori informazioni, fare riferimento a "Utilizzare la topologia CSI".

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
serviceLevel: Ultra
capacityPools:
  - application-group-1/account-1/ultra-1
  - application-group-1/account-1/ultra-2
supportedTopologies:
  - topology.kubernetes.io/region: eastus
    topology.kubernetes.io/zone: eastus-1
  - topology.kubernetes.io/region: eastus
    topology.kubernetes.io/zone: eastus-2
```

Definizioni delle classi di storage

Quanto segue `StorageClass` le definizioni si riferiscono ai pool di storage sopra indicati.

Definizioni di esempio con `parameter.selector` campo

Utilizzo di `parameter.selector` è possibile specificare per ciascuno `StorageClass` il pool virtuale utilizzato per ospitare un volume. Gli aspetti del volume saranno definiti nel pool selezionato.

```yaml
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gold
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=gold
allowVolumeExpansion: true

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: silver
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver
allowVolumeExpansion: true

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: bronze
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=bronze
allowVolumeExpansion: true
```

Definizioni di esempio per volumi SMB

Utilizzo di `nasType`, `node-stage-secret-name`, e. `node-stage-secret-namespace`, È possibile specificare un volume SMB e fornire le credenziali Active Directory richieste.

Configurazione di base sullo spazio dei nomi predefinito

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
```

Utilizzo di segreti diversi per spazio dei nomi

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

Utilizzo di segreti diversi per volume

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: ${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

`nasType: smb` Filtri per pool che supportano volumi SMB. `nasType: nfs` oppure `nasType: null` Filtri per i pool NFS.

Creare il backend

Dopo aver creato il file di configurazione back-end, eseguire il seguente comando:

```
tridentctl create backend -f <backend-file>
```

Se la creazione del backend non riesce, si è verificato un errore nella configurazione del backend. È possibile visualizzare i log per determinare la causa eseguendo il seguente comando:

```
tridentctl logs
```

Dopo aver identificato e corretto il problema con il file di configurazione, è possibile eseguire nuovamente il comando create.

Google Cloud NetApp Volumes

Configurare un backend Google Cloud NetApp Volumes

Ora puoi configurare Google Cloud NetApp Volumes come back-end per Trident. È possibile collegare volumi NFS e SMB utilizzando un backend dei volumi Google Cloud NetApp.

Dettagli del driver di Google Cloud NetApp Volumes

Trident fornisce al `google-cloud-netapp-volumes` driver la comunicazione con il cluster. Le modalità di accesso supportate sono: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Driver	Protocollo	VolumeMode	Modalità di accesso supportate	File system supportati
`google-cloud-netapp-volumes`	NFS PMI	Filesystem	RWO, ROX, RWX, RWOP	`nfs`, `smb`

Identità cloud per GKE

L'identità del cloud consente ai pod Kubernetes di accedere alle risorse Google Cloud autenticandosi come identità di workload anziché fornendo credenziali esplicite di Google Cloud.

Per sfruttare l'identità cloud in Google Cloud, è necessario disporre di:

- Un cluster Kubernetes implementato usando GKE.
- Identità del carico di lavoro configurata sul cluster GKE e sul server dei metadati GKE configurato sui pool di nodi.

- Un account del servizio GCP con ruolo Google Cloud NetApp Volumes Admin (role/NetApp.admin) o un ruolo personalizzato.
- Trident installato che include il cloud Provider per specificare "GCP" e cloudIdentity specificando il nuovo account del servizio GCP. Di seguito viene riportato un esempio.

Operatore Trident

Per installare Trident utilizzando l'operatore Trident, `tridentorchestrator_cr.yaml` `"GCP"` impostare su `cloudProvider` e `cloudIdentity` su `iam.gke.io/gcp-service-account: cloudvolumes-admin-sa@mygcpproject.iam.gserviceaccount.com`.

Ad esempio:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "GCP"
  cloudIdentity: 'iam.gke.io/gcp-service-account: cloudvolumes-
admin-sa@mygcpproject.iam.gserviceaccount.com'
```

Timone

Impostare i valori per i flag **cloud-provider (CP)** e **cloud-Identity (ci)** utilizzando le seguenti variabili di ambiente:

```
export CP="GCP"
export ANNOTATION="'iam.gke.io/gcp-service-account: cloudvolumes-admin-
sa@mygcpproject.iam.gserviceaccount.com'"
```

Nell'esempio seguente viene installato Trident e impostato `cloudProvider` su GCP utilizzando la variabile di ambiente `$CP` e viene impostata la `cloudIdentity` variabile di ambiente Using the `$ANNOTATION` :

```
helm install trident trident-operator-100.6.0.tgz --set
cloudProvider=$CP --set cloudIdentity="$ANNOTATION"
```

<code>tridentctl</code>

Impostare i valori per i flag **cloud provider** e **cloud Identity** utilizzando le seguenti variabili di ambiente:

```
export CP="GCP"
export ANNOTATION="'iam.gke.io/gcp-service-account: cloudvolumes-admin-
sa@mygcpproject.iam.gserviceaccount.com'"
```

Nell'esempio seguente viene installato Trident e viene impostato il `cloud-provider` flag su `$CP`, e `cloud-identity` su `$ANNOTATION`:

```
tridentctl install --cloud-provider=$CP --cloud
-identity="$ANNOTATION" -n trident
```

Preparazione per la configurazione di un backend Google Cloud NetApp Volumes

Prima di poter configurare il back-end di Google Cloud NetApp Volumes, devi verificare che siano soddisfatti i seguenti requisiti.

Prerequisiti per i volumi NFS

Se stai utilizzando Google Cloud NetApp Volumes per la prima volta o in una nuova posizione, è necessaria una certa configurazione iniziale per configurare i volumi di Google Cloud NetApp e creare un volume NFS. Fare riferimento alla "Prima di iniziare".

Prima di configurare il back-end di Google Cloud NetApp Volumes, assicurati di disporre di quanto segue:

- Un account Google Cloud configurato con il servizio Google Cloud NetApp Volumes. Fare riferimento alla "Google Cloud NetApp Volumes".
- Numero di progetto dell'account Google Cloud. Fare riferimento alla "Identificazione dei progetti".
- Un account di servizio Google Cloud con il ruolo NetApp Volumes Admin (`roles/netapp.admin`). Fare riferimento alla "Ruoli e autorizzazioni di Identity and Access Management".
- File chiave API per il tuo account GCNV. Fare riferimento alla "Creare una chiave dell'account del servizio"
- Un pool di storage. Fare riferimento alla "Panoramica dei pool di storage".

Per ulteriori informazioni su come configurare l'accesso a Google Cloud NetApp Volumes, fare riferimento a "Configurare l'accesso a Google Cloud NetApp Volumes".

Opzioni ed esempi di configurazione di backend dei volumi Google Cloud NetApp

Scopri le opzioni di configurazione di back-end per Google Cloud NetApp Volumes e consulta gli esempi di configurazione.

Opzioni di configurazione back-end

Ogni back-end esegue il provisioning dei volumi in una singola area di Google Cloud. Per creare volumi in altre regioni, è possibile definire backend aggiuntivi.

Parametro	Descrizione	Predefinito
`version`		Sempre 1
`storageDriverName`	Nome del driver di storage	Il valore di `storageDriverName` deve essere specificato come "google-cloud-netapp-Volumes".
`backendName`	(Facoltativo) Nome personalizzato del backend dello storage	Nome del driver + "_" + parte della chiave API

Parametro	Descrizione	Predefinito
`storagePools`	Parametro facoltativo utilizzato per specificare i pool di storage per la creazione di volumi.	
`projectNumber`	Numero di progetto dell'account Google Cloud. Il valore si trova nella home page del portale Google Cloud.	
`location`	La posizione di Google Cloud in cui Trident crea volumi GCNV. Quando si creano cluster Kubernetes tra aree, i volumi creati in a `location` possono essere utilizzati nei carichi di lavoro pianificati sui nodi in più aree Google Cloud. Il traffico interregionale comporta un costo aggiuntivo.	
`apiKey`	Chiave API per l'account del servizio Google Cloud con il `netapp.admin` ruolo. Include il contenuto in formato JSON di un file di chiave privata dell'account di un servizio Google Cloud (copia integrale nel file di configurazione del backend). L' `apiKey` deve includere coppie chiave-valore per le seguenti chiavi: `type`, `project_id`, , `client_email`, , `client_id` `auth_uri` `token_uri` `auth_provider_x509_cert_url`, , e `client_x509_cert_url`.	
`nfsMountOptions`	Controllo dettagliato delle opzioni di montaggio NFS.	"nfsvers=3"
`limitVolumeSize`	Il provisioning non riesce se le dimensioni del volume richiesto sono superiori a questo valore.	"" (non applicato per impostazione predefinita)
`serviceLevel`	Il livello di servizio di un pool di storage e i relativi volumi. I valori sono `flex`, `standard`, , `premium`o `extreme`.	
`labels`	Set di etichette arbitrarie formattate con JSON da applicare sui volumi	""
`network`	Rete Google Cloud usata per GCNV Volumes.	
`debugTraceFlags`	Flag di debug da utilizzare per la risoluzione dei problemi. Esempio, `{"api":false, "method":true}`. Non utilizzare questa opzione a meno che non si stia eseguendo la risoluzione dei problemi e non si richieda un dump dettagliato del log.	nullo
`nasType`	Configurare la creazione di volumi NFS o SMB. Le opzioni sono `nfs`, `smb` o nullo. L'impostazione su Null consente di impostare i volumi NFS come predefiniti.	`nfs`

Parametro	Descrizione	Predefinito
`supportedTopologies`	Rappresenta un elenco di aree e zone supportate da questo backend. Per ulteriori informazioni, fare riferimento a "Utilizzare la topologia CSI". Ad esempio: `supportedTopologies: - topology.kubernetes.io/region: asia-east1 topology.kubernetes.io/zone: asia-east1-a`	

Opzioni di provisioning dei volumi

È possibile controllare il provisioning del volume predefinito in `defaults` del file di configurazione.

Parametro	Descrizione	Predefinito
`exportRule`	Le regole di esportazione per i nuovi volumi. Deve essere un elenco separato da virgole di qualsiasi combinazione di indirizzi IPv4.	"0.0.0.0/0"
`snapshotDir`	Accesso a `.snapshot` directory	"True" per NFSv4 "false" per NFSv3
`snapshotReserve`	Percentuale di volume riservato agli snapshot	"" (accettare l'impostazione predefinita di 0)
`unixPermissions`	Le autorizzazioni unix dei nuovi volumi (4 cifre ottali).	""

Configurazioni di esempio

Gli esempi seguenti mostrano le configurazioni di base che lasciano la maggior parte dei parametri predefiniti. Questo è il modo più semplice per definire un backend.

Configurazione minima

Questa è la configurazione backend minima assoluta. Con questa configurazione, Trident rileva tutti i pool di storage delegati ai volumi Google Cloud NetApp nella posizione configurata e posiziona nuovi volumi in uno di tali pool in modo casuale. Poiché `nasType` viene omesso, viene applicato il `nfs` valore predefinito e il backend esegue il provisioning dei volumi NFS.

Questa configurazione è ideale quando si inizia a usare Google Cloud NetApp Volumes e si tenta le cose, ma in pratica con tutta probabilità sarà necessario fornire un ambito aggiuntivo per i volumi da eseguire il provisioning.

```yaml
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----\n
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\n
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\n
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m\n
    XsYg6gyxy4zq7OlwWgLwGa==\n
    -----END PRIVATE KEY-----\n

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  serviceLevel: premium
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Configurazione per volumi SMB

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv1
  namespace: trident
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123456789"
  location: asia-east1
  serviceLevel: flex
  nasType: smb
  apiKey:
    type: service_account
    project_id: cloud-native-data
    client_email: trident-sample@cloud-native-
data.iam.gserviceaccount.com
    client_id: "123456789737813416734"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/trident-
sample%40cloud-native-data.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Configurazione con il filtro StoragePools

```yaml
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  serviceLevel: premium
  storagePools:
    - premium-pool1-europe-west6
    - premium-pool2-europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Configurazione dei pool virtuali

Questa configurazione backend definisce più pool virtuali in un singolo file. I pool virtuali sono definiti nella `storage` sezione . Sono utili quando disponi di più pool di storage che supportano diversi livelli di servizio e vuoi creare classi di storage in Kubernetes che ne rappresentano le caratteristiche. Le etichette dei pool virtuali vengono utilizzate per differenziare i pool. Ad esempio, nell'esempio riportato di seguito `performance` vengono utilizzate etichette e `serviceLevel` tipi per differenziare i pool virtuali.

È inoltre possibile impostare alcuni valori predefiniti applicabili a tutti i pool virtuali e sovrascrivere i valori predefiniti per i singoli pool virtuali. Nell'esempio seguente, `snapshotReserve` e `exportRule` fungono da impostazioni predefinite per tutti i pool virtuali.

Per ulteriori informazioni, fare riferimento a "Pool virtuali".

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
```

```
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
  defaults:
    snapshotReserve: "10"
    exportRule: 10.0.0.0/24
  storage:
    - labels:
        performance: extreme
      serviceLevel: extreme
      defaults:
        snapshotReserve: "5"
        exportRule: 0.0.0.0/0
    - labels:
        performance: premium
      serviceLevel: premium
    - labels:
        performance: standard
      serviceLevel: standard
```

Identità cloud per 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 topologie supportate

Trident facilita il provisioning dei volumi per i workload in base a regioni e zone di disponibilità. Il `supportedTopologies` blocco in questa configurazione backend viene utilizzato per fornire un elenco di aree e zone per backend. I valori di regione e zona specificati qui devono corrispondere ai valori di regione e zona dalle etichette su ogni nodo del cluster Kubernetes. Queste regioni e zone rappresentano l'elenco dei valori consentiti che possono essere forniti in una classe di archiviazione. Per le classi di archiviazione che contengono un sottoinsieme delle aree e delle zone fornite in un backend, Trident crea volumi nell'area e nella zona menzionate. Per ulteriori informazioni, fare riferimento a "Utilizzare la topologia CSI".

```
---
version: 1
storageDriverName: google-cloud-netapp-volumes
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: asia-east1
serviceLevel: flex
supportedTopologies:
  - topology.kubernetes.io/region: asia-east1
    topology.kubernetes.io/zone: asia-east1-a
  - topology.kubernetes.io/region: asia-east1
    topology.kubernetes.io/zone: asia-east1-b
```

Quali sono le prossime novità?

Dopo aver creato il file di configurazione back-end, eseguire il seguente comando:

```
kubectl create -f <backend-file>
```

Per verificare che il backend sia stato creato correttamente, eseguire il comando seguente:

```
kubectl get tridentbackendconfig

NAME                BACKEND NAME        BACKEND UUID
PHASE   STATUS
backend-tbc-gcnv    backend-tbc-gcnv    b2fd1ff9-b234-477e-88fd-713913294f65
Bound   Success
```

Se la creazione del backend non riesce, si è verificato un errore nella configurazione del backend. È possibile descrivere il backend utilizzando il `kubectl get tridentbackendconfig <backend-name>` comando oppure visualizzare i log per determinare la causa eseguendo il seguente comando:

```
tridentctl logs
```

Dopo aver identificato e corretto il problema con il file di configurazione, è possibile eliminare il backend ed eseguire nuovamente il comando create.

Definizioni delle classi di storage

Di seguito è riportata una definizione di base `StorageClass` che fa riferimento al backend riportato sopra.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-nfs-sc
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
```

Definizioni di esempio utilizzando il `parameter.selector` campo:

L'utilizzo `parameter.selector` consente di specificare per ciascun `StorageClass` "pool virtuale" sistema utilizzato per ospitare un volume. Gli aspetti del volume saranno definiti nel pool selezionato.

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: extreme-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=extreme
  backendType: google-cloud-netapp-volumes

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: premium-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=premium
  backendType: google-cloud-netapp-volumes

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: standard-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=standard
  backendType: google-cloud-netapp-volumes
```

Per ulteriori informazioni sulle classi di archiviazione, fare riferimento a "Creare una classe di storage".

Definizioni di esempio per volumi SMB

Utilizzando `nasType`, , `node-stage-secret-name` e `node-stage-secret-namespace`, è possibile specificare un volume SMB e fornire le credenziali di Active Directory richieste. Qualsiasi utente/password di Active Directory con autorizzazioni qualsiasi/nessuna può essere utilizzato per il segreto di fase del nodo.

Configurazione di base sullo spazio dei nomi predefinito

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
```

Utilizzo di segreti diversi per spazio dei nomi

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

Utilizzo di segreti diversi per volume

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: ${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

`nasType: smb` Filtri per pool che supportano volumi SMB. `nasType: nfs` oppure `nasType: null` Filtri per i pool NFS.

Esempio di definizione PVC

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: gcnv-nfs-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 100Gi
  storageClassName: gcnv-nfs-sc
```

Per verificare se il PVC è associato, eseguire il seguente comando:

```
kubectl get pvc gcnv-nfs-pvc

NAME          STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS AGE
gcnv-nfs-pvc  Bound    pvc-b00f2414-e229-40e6-9b16-ee03eb79a213  100Gi
RWX            gcnv-nfs-sc  1m
```

Configurare un Cloud Volumes Service per il backend di Google Cloud

Scopri come configurare NetApp Cloud Volumes Service per Google Cloud come back-end per l'installazione di Trident utilizzando le configurazioni di esempio fornite.

Dettagli del driver di Google Cloud

Trident fornisce al `gcp-cvs` driver la comunicazione con il cluster. Le modalità di accesso supportate sono: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Driver	Protocollo	VolumeMode	Modalità di accesso supportate	File system supportati
`gcp-cvs`	NFS	Filesystem	RWO, ROX, RWX, RWOP	`nfs`

Scopri il supporto Trident per Cloud Volumes Service per Google Cloud

Trident può creare volumi Cloud Volumes Service in uno dei due "tipi di servizio":

- **CVS-Performance**: Il tipo di servizio Trident predefinito. Questo tipo di servizio ottimizzato per le performance è più adatto per i carichi di lavoro di produzione che apprezzano le performance. Il tipo di servizio CVS-Performance è un'opzione hardware che supporta volumi con una dimensione minima di 100 GiB. È possibile scegliere tra"tre livelli di servizio":
  - `standard`
  - `premium`
  - `extreme`
- **CVS**: Il tipo di servizio CVS offre un'elevata disponibilità zonale con livelli di performance da limitati a moderati. Il tipo di servizio CVS è un'opzione software che utilizza pool di storage per supportare volumi di dimensioni pari a 1 GiB. Il pool di storage può contenere fino a 50 volumi in cui tutti i volumi condividono la capacità e le performance del pool. È possibile scegliere tra "due livelli di servizio":
  - `standardsw`
  - `zoneredundantstandardsw`

Di cosa hai bisogno

Per configurare e utilizzare "Cloud Volumes Service per Google Cloud" back-end, sono necessari i seguenti elementi:

- Un account Google Cloud configurato con NetApp Cloud Volumes Service
- Numero di progetto dell'account Google Cloud
- Account di servizio Google Cloud con `netappcloudvolumes.admin` ruolo
- File delle chiavi API per l'account Cloud Volumes Service

Opzioni di configurazione back-end

Ogni back-end esegue il provisioning dei volumi in una singola area di Google Cloud. Per creare volumi in altre regioni, è possibile definire backend aggiuntivi.

Parametro	Descrizione	Predefinito
`version`		Sempre 1
`storageDriverName`	Nome del driver di storage	"gcp-cvs"
`backendName`	Nome personalizzato o backend dello storage	Nome del driver + "_" + parte della chiave API
`storageClass`	Parametro facoltativo utilizzato per specificare il tipo di servizio CVS. Utilizzare `software` per selezionare il tipo di servizio CVS. In caso contrario, Trident assume il tipo di servizio CVS-Performance (`hardware`).	
`storagePools`	Solo tipo di servizio CVS. Parametro facoltativo utilizzato per specificare i pool di storage per la creazione di volumi.	
`projectNumber`	Numero di progetto dell'account Google Cloud. Il valore si trova nella home page del portale Google Cloud.	

Parametro	Descrizione	Predefinito
`hostProjectNumber`	Necessario se si utilizza una rete VPC condivisa. In questo scenario, `projectNumber` è il progetto di servizio, e. `hostProjectNumber` è il progetto host.	
`apiRegion`	La regione Google Cloud in cui Trident crea Cloud Volumes Service Volumes. Quando si creano cluster Kubernetes tra aree, i volumi creati in un `apiRegion` possono essere utilizzati nei carichi di lavoro pianificati sui nodi in più aree Google Cloud. Il traffico interregionale comporta un costo aggiuntivo.	
`apiKey`	Chiave API per l'account del servizio Google Cloud con `netappcloudvolumes.admin` ruolo. Include il contenuto in formato JSON di un file di chiave privata dell'account di un servizio Google Cloud (copia integrale nel file di configurazione del backend).	
`proxyURL`	URL del proxy se il server proxy ha richiesto di connettersi all'account CVS. Il server proxy può essere un proxy HTTP o un proxy HTTPS. Per un proxy HTTPS, la convalida del certificato viene ignorata per consentire l'utilizzo di certificati autofirmati nel server proxy. I server proxy con autenticazione abilitata non sono supportati.	
`nfsMountOptions`	Controllo dettagliato delle opzioni di montaggio NFS.	"nfsvers=3"
`limitVolumeSize`	Il provisioning non riesce se le dimensioni del volume richiesto sono superiori a questo valore.	"" (non applicato per impostazione predefinita)
`serviceLevel`	Livello di servizio CVS-Performance o CVS per i nuovi volumi. I valori CVS-Performance sono `standard`, `premium`, o. `extreme`. I valori CVS sono `standardsw` oppure `zoneredundantstandardsw`.	CVS-Performance (prestazioni CVS) è "standard". Il valore predefinito di CVS è "standardsw".
`network`	Rete Google Cloud utilizzata per i volumi Cloud Volumes Service.	"predefinito"
`debugTraceFlags`	Flag di debug da utilizzare per la risoluzione dei problemi. Esempio, `\{"api":false, "method":true}`. Non utilizzare questa opzione a meno che non si stia eseguendo la risoluzione dei problemi e non si richieda un dump dettagliato del log.	nullo

`allowedTopologies`	Per abilitare l'accesso multi-regione, la definizione StorageClass per `allowedTopologies` deve includere tutte le regioni. Ad esempio:
	`- key: topology.kubernetes.io/region`
	`values:`
	`- us-east1`
	`- europe-west1`	

Opzioni di provisioning dei volumi

È possibile controllare il provisioning del volume predefinito in `defaults` del file di configurazione.

Parametro	Descrizione	Predefinito
`exportRule`	Le regole di esportazione per i nuovi volumi. Deve essere un elenco separato da virgole di qualsiasi combinazione di indirizzi IPv4 o subnet IPv4 nella notazione CIDR.	"0.0.0.0/0"
`snapshotDir`	Accesso a `.snapshot` directory	"falso"
`snapshotReserve`	Percentuale di volume riservato agli snapshot	"" (accettare CVS come valore predefinito 0)
`size`	Le dimensioni dei nuovi volumi. Performance CVS minima: 100 GiB. CVS minimo: 1 GiB.	Per impostazione predefinita, il tipo di servizio CVS-Performance è "100GiB". Il tipo di servizio CVS non imposta un valore predefinito, ma richiede un minimo di 1 GiB.

Esempi di tipo di servizio CVS-Performance

I seguenti esempi forniscono configurazioni di esempio per il tipo di servizio CVS-Performance.

Esempio 1: Configurazione minima

Questa è la configurazione di backend minima che utilizza il tipo di servizio CVS-Performance predefinito con il livello di servizio "standard" predefinito.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: "012345678901"
apiRegion: us-west2
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: <id_value>
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: "123456789012345678901"
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
```

Esempio 2: Configurazione del livello di servizio

In questo esempio vengono illustrate le opzioni di configurazione back-end, inclusi il livello di servizio e i valori predefiniti del volume.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
apiRegion: us-west2
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: '123456789012345678901'
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
proxyURL: http://proxy-server-hostname/
nfsMountOptions: vers=3,proto=tcp,timeo=600
limitVolumeSize: 10Ti
serviceLevel: premium
defaults:
  snapshotDir: 'true'
  snapshotReserve: '5'
  exportRule: 10.0.0.0/24,10.0.1.0/24,10.0.2.100
  size: 5Ti
```

Esempio 3: Configurazione del pool virtuale

Questo esempio utilizza `storage` per configurare i pool virtuali e il `StorageClasses` che fanno riferimento a loro. Fare riferimento a. Definizioni delle classi di storage per vedere come sono state definite le classi di storage.

In questo caso, vengono impostati valori predefiniti specifici per tutti i pool virtuali, che impostano `snapshotReserve` al 5% e a. `exportRule` a 0.0.0.0/0. I pool virtuali sono definiti in `storage` sezione. Ogni singolo pool virtuale definisce il proprio `serviceLevel`e alcuni pool sovrascrivono i valori predefiniti. Le etichette dei pool virtuali sono state utilizzate per differenziare i pool in base a. `performance` e. `protection`.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
apiRegion: us-west2
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: '123456789012345678901'
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
nfsMountOptions: vers=3,proto=tcp,timeo=600
defaults:
  snapshotReserve: '5'
  exportRule: 0.0.0.0/0
labels:
  cloud: gcp
region: us-west2
storage:
- labels:
    performance: extreme
    protection: extra
  serviceLevel: extreme
```

```
  defaults:
    snapshotDir: 'true'
    snapshotReserve: '10'
    exportRule: 10.0.0.0/24
- labels:
    performance: extreme
    protection: standard
  serviceLevel: extreme
- labels:
    performance: premium
    protection: extra
  serviceLevel: premium
  defaults:
    snapshotDir: 'true'
    snapshotReserve: '10'
- labels:
    performance: premium
    protection: standard
  serviceLevel: premium
- labels:
    performance: standard
  serviceLevel: standard
```

Definizioni delle classi di storage

Le seguenti definizioni di StorageClass si applicano all'esempio di configurazione del pool virtuale. Utilizzo di `parameters.selector`, È possibile specificare per ogni StorageClass il pool virtuale utilizzato per ospitare un volume. Gli aspetti del volume saranno definiti nel pool selezionato.

Esempio di classe di storage

```yaml
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-extra-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=extreme; protection=extra
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-standard-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=premium; protection=standard
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-premium-extra-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=premium; protection=extra
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-premium
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=premium; protection=standard
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-standard
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=standard
```

```yaml
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extra-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: protection=extra
allowVolumeExpansion: true
```

- Il primo StorageClass (`cvs-extreme-extra-protection`) viene mappato al primo pool virtuale. Questo è l'unico pool che offre performance estreme con una riserva di snapshot del 10%.
- L'ultima StorageClass (`cvs-extra-protection`) richiama qualsiasi pool di archiviazione che fornisce una riserva snapshot del 10%. Trident decide quale pool virtuale è selezionato e garantisce che venga soddisfatto il requisito di riserva degli snapshot.

Esempi di tipo di servizio CVS

I seguenti esempi forniscono configurazioni di esempio per il tipo di servizio CVS.

Esempio 1: Configurazione minima

Questa è la configurazione backend minima utilizzata `storageClass` Per specificare il tipo di servizio CVS e il valore predefinito `standardsw` livello di servizio.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
storageClass: software
apiRegion: us-east4
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: '123456789012345678901'
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
serviceLevel: standardsw
```

Esempio 2: Configurazione del pool di storage

Questo esempio di configurazione di backend utilizza `storagePools` per configurare un pool di storage.

```
---
version: 1
storageDriverName: gcp-cvs
backendName: gcp-std-so-with-pool
projectNumber: '531265380079'
apiRegion: europe-west1
apiKey:
  type: service_account
  project_id: cloud-native-data
  private_key_id: "<id_value>"
  private_key: |-
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@cloud-native-
data.iam.gserviceaccount.com
  client_id: '107071413297115343396'
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40cloud-native-data.iam.gserviceaccount.com
storageClass: software
zone: europe-west1-b
network: default
storagePools:
- 1bc7f380-3314-6005-45e9-c7dc8c2d7509
serviceLevel: Standardsw
```

Quali sono le prossime novità?

Dopo aver creato il file di configurazione back-end, eseguire il seguente comando:

```
tridentctl create backend -f <backend-file>
```

Se la creazione del backend non riesce, si è verificato un errore nella configurazione del backend. È possibile visualizzare i log per determinare la causa eseguendo il seguente comando:

```
tridentctl logs
```

Dopo aver identificato e corretto il problema con il file di configurazione, è possibile eseguire nuovamente il comando create.

Configurare un backend NetApp HCI o SolidFire

Scoprite come creare e utilizzare un backend Element con l'installazione Trident.

Dettagli driver elemento

Trident fornisce il `solidfire-san` driver di storage per comunicare con il cluster. Le modalità di accesso supportate sono: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Il `solidfire-san` driver di archiviazione supporta le modalità di volume *file* e *block*. Per la `Filesystem` modalità volumeMode, Trident crea un volume e crea un filesystem. Il tipo di file system viene specificato da StorageClass.

Driver	Protocollo	VolumeMode	Modalità di accesso supportate	File system supportati
`solidfire-san`	ISCSI	Blocco	RWO, ROX, RWX, RWOP	Nessun filesystem. Dispositivo a blocchi raw.
`solidfire-san`	ISCSI	Filesystem	RWO, RWOP	`xfs`, `ext3`, `ext4`

Prima di iniziare

Prima di creare un backend elemento, è necessario quanto segue.

- Un sistema storage supportato che esegue il software Element.
- Credenziali per un amministratore del cluster NetApp HCI/SolidFire o un utente tenant in grado di gestire i volumi.
- Tutti i nodi di lavoro di Kubernetes devono disporre dei tool iSCSI appropriati. Fare riferimento a. "informazioni sulla preparazione del nodo di lavoro".

Opzioni di configurazione back-end

Per le opzioni di configurazione del backend, consultare la tabella seguente:

Parametro	Descrizione	Predefinito
`version`		Sempre 1
`storageDriverName`	Nome del driver di storage	Sempre "SolidFire-san"

Parametro	Descrizione	Predefinito
`backendName`	Nome personalizzato o backend dello storage	"SolidFire_" + indirizzo IP di storage (iSCSI)
`Endpoint`	MVIP per il cluster SolidFire con credenziali tenant	
`SVIP`	Porta e indirizzo IP dello storage (iSCSI)	
`labels`	Set di etichette arbitrarie formattate con JSON da applicare sui volumi.	""
`TenantName`	Nome tenant da utilizzare (creato se non trovato)	
`InitiatorIFace`	Limitare il traffico iSCSI a un'interfaccia host specifica	"predefinito"
`UseCHAP`	Utilizzare CHAP per autenticare iSCSI. Trident utilizza il protocollo CHAP.	vero
`AccessGroups`	Elenco degli ID del gruppo di accesso da utilizzare	Trova l'ID di un gruppo di accesso denominato "Trident"
`Types`	Specifiche QoS	
`limitVolumeSize`	Fallire il provisioning se la dimensione del volume richiesta è superiore a questo valore	"" (non applicato per impostazione predefinita)
`debugTraceFlags`	Flag di debug da utilizzare per la risoluzione dei problemi. Esempio, {"api":false, "method":true}	nullo

Non utilizzare `debugTraceFlags` a meno che non si stia eseguendo la risoluzione dei problemi e non si richieda un dump dettagliato del log.

Esempio 1: Configurazione back-end per `solidfire-san` driver con tre tipi di volume

Questo esempio mostra un file backend che utilizza l'autenticazione CHAP e modellazione di tre tipi di volume con specifiche garanzie di QoS. È molto probabile che si definiscano le classi di storage per utilizzarle utilizzando `IOPS` parametro della classe di storage.

```
---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0
SVIP: <svip>:3260
TenantName: <tenant>
labels:
  k8scluster: dev1
  backend: dev1-element-cluster
UseCHAP: true
Types:
  - Type: Bronze
    Qos:
      minIOPS: 1000
      maxIOPS: 2000
      burstIOPS: 4000
  - Type: Silver
    Qos:
      minIOPS: 4000
      maxIOPS: 6000
      burstIOPS: 8000
  - Type: Gold
    Qos:
      minIOPS: 6000
      maxIOPS: 8000
      burstIOPS: 10000
```

Esempio 2: Configurazione del backend e della classe di storage per `solidfire-san` driver con pool virtuali

Questo esempio mostra il file di definizione back-end configurato con i pool virtuali insieme a StorageClasses che fanno riferimento ad essi.

Trident copia le etichette presenti su un pool di storage al LUN di storage backend al momento del provisioning. Per comodità, gli amministratori dello storage possono definire le etichette per ogni pool virtuale e raggruppare i volumi per etichetta.

Nel file di definizione del backend di esempio mostrato di seguito, vengono impostati valori predefiniti specifici per tutti i pool di storage, che impostano `type` In Silver. I pool virtuali sono definiti in `storage` sezione. In questo esempio, alcuni pool di storage impostano il proprio tipo e alcuni pool sovrascrivono i valori predefiniti impostati in precedenza.

```
---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0
SVIP: <svip>:3260
```

```
TenantName: <tenant>
UseCHAP: true
Types:
  - Type: Bronze
    Qos:
      minIOPS: 1000
      maxIOPS: 2000
      burstIOPS: 4000
  - Type: Silver
    Qos:
      minIOPS: 4000
      maxIOPS: 6000
      burstIOPS: 8000
  - Type: Gold
    Qos:
      minIOPS: 6000
      maxIOPS: 8000
      burstIOPS: 10000
type: Silver
labels:
  store: solidfire
  k8scluster: dev-1-cluster
region: us-east-1
storage:
  - labels:
      performance: gold
      cost: "4"
    zone: us-east-1a
    type: Gold
  - labels:
      performance: silver
      cost: "3"
    zone: us-east-1b
    type: Silver
  - labels:
      performance: bronze
      cost: "2"
    zone: us-east-1c
    type: Bronze
  - labels:
      performance: silver
      cost: "1"
    zone: us-east-1d
```

Le seguenti definizioni di StorageClass si riferiscono ai pool virtuali sopra indicati. Utilizzando il `parameters.selector` Ciascun StorageClass richiama i pool virtuali che possono essere utilizzati per

ospitare un volume. Gli aspetti del volume saranno definiti nel pool virtuale scelto.

Il primo StorageClass (`solidfire-gold-four`) verrà mappato al primo pool virtuale. Questa è l'unica piscina che offre prestazioni d'oro con un `Volume Type QoS` di Gold. L'ultima StorageClass (`solidfire-silver`) richiama qualsiasi pool di storage che offre prestazioni eccezionali. Trident deciderà quale pool virtuale viene selezionato e garantirà che i requisiti di storage vengano soddisfatti.

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-gold-four
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=gold; cost=4
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-three
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver; cost=3
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-bronze-two
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=bronze; cost=2
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-one
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver; cost=1
  fsType: ext4

---
```

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver
  fsType: ext4
```

Trova ulteriori informazioni

- "Gruppi di accesso ai volumi"

Driver SAN ONTAP

Panoramica del driver SAN ONTAP

Informazioni sulla configurazione di un backend ONTAP con driver SAN ONTAP e Cloud Volumes ONTAP.

Dettagli del driver SAN ONTAP

Trident fornisce i seguenti driver di storage SAN per comunicare con il cluster ONTAP. Le modalità di accesso supportate sono: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Driver	Protocollo	VolumeMode	Modalità di accesso supportate	File system supportati
`ontap-san`	ISCSI SCSI su FC	Blocco	RWO, ROX, RWX, RWOP	Nessun file system; dispositivo a blocchi raw
`ontap-san`	ISCSI SCSI su FC	Filesystem	RWO, RWOP ROX e RWX non sono disponibili in modalità Volume filesystem.	`xfs`, `ext3`, `ext4`
`ontap-san`	NVMe/TCP Fare riferimento a. Considerazioni aggiuntive su NVMe/TCP.	Blocco	RWO, ROX, RWX, RWOP	Nessun file system; dispositivo a blocchi raw

Driver	Protocollo	VolumeMode	Modalità di accesso supportate	File system supportati
`ontap-san`	NVMe/TCP Fare riferimento a. Considerazioni aggiuntive su NVMe/TCP.	Filesystem	RWO, RWOP ROX e RWX non sono disponibili in modalità Volume filesystem.	`xfs`, `ext3`, `ext4`
`ontap-san-economy`	ISCSI	Blocco	RWO, ROX, RWX, RWOP	Nessun file system; dispositivo a blocchi raw
`ontap-san-economy`	ISCSI	Filesystem	RWO, RWOP ROX e RWX non sono disponibili in modalità Volume filesystem.	`xfs`, `ext3`, `ext4`

- Utilizzare `ontap-san-economy` solo se si prevede che il conteggio dell'utilizzo persistente del volume sia superiore a. "Limiti di volume ONTAP supportati".
- Utilizzare `ontap-nas-economy` solo se si prevede che il conteggio dell'utilizzo persistente del volume sia superiore a. "Limiti di volume ONTAP supportati" e a. `ontap-san-economy` impossibile utilizzare il driver.
- Non utilizzare `ontap-nas-economy` se prevedete la necessità di protezione dei dati, disaster recovery o mobilità.
- NetApp sconsiglia di utilizzare l'espansione automatica FlexVol in tutti i driver ONTAP, ad eccezione di ONTAP-san. Come soluzione alternativa, Trident supporta l'utilizzo di una riserva di snapshot e scala di conseguenza i volumi FlexVol.

Autorizzazioni utente

Trident può essere eseguito come amministratore di ONTAP o SVM, in genere utilizzando un utente del cluster o un utente SVM `admin` o `vsadmin` un utente con un nome diverso che svolge lo stesso ruolo. Per le implementazioni di Amazon FSX per NetApp ONTAP, Trident si aspetta un'esecuzione come amministratore ONTAP o SVM, con l'utente del cluster `fsxadmin`, un `vsadmin` utente SVM o un utente con un nome diverso che abbia lo stesso ruolo. L' `fsxadmin`utente sostituisce in modo limitato l'utente amministratore del cluster.

Se si utilizza il `limitAggregateUsage` parametro, sono necessarie le autorizzazioni di amministratore del cluster. Quando si utilizza Amazon FSX per NetApp ONTAP con Trident, il `limitAggregateUsage` parametro non funziona con `vsadmin` gli account utente e. `fsxadmin` L'operazione di configurazione non riesce se si specifica questo parametro.

Sebbene sia possibile creare un ruolo più restrittivo all'interno di ONTAP che un driver Trident può utilizzare, non lo consigliamo. La maggior parte delle nuove release di Trident chiamerà API aggiuntive che dovrebbero essere considerate, rendendo gli aggiornamenti difficili e soggetti a errori.

Considerazioni aggiuntive su NVMe/TCP

Trident supporta il protocollo non-volatile memory express (NVMe) utilizzando il `ontap-san` driver, tra cui:

- IPv6
- Snapshot e cloni di volumi NVMe
- Ridimensionamento di un volume NVMe
- Importazione di un volume NVMe creato al di fuori di Trident in modo che il suo ciclo di vita possa essere gestito da Trident
- Multipath nativo NVMe
- Arresto anomalo o anomalo dei K8s nodi (24,06)

Trident non supporta:

- DH-HMAC-CHAP supportato nativamente da NVMe
- Multipathing DM (Device mapper)
- Crittografia LUKS

> NVMe è supportato solo con le API REST ONTAP e non con ONTAPI (ZAPI).

Prepararsi a configurare il backend con i driver SAN ONTAP

Comprendere i requisiti e le opzioni di autenticazione per la configurazione di un backend ONTAP con i driver SAN ONTAP.

Requisiti

Per tutti i backend ONTAP, Trident richiede che almeno un aggregato sia assegnato all'SVM.

> "Sistemi ASA r2" differiscono dagli altri sistemi ONTAP (ASA, AFF e FAS) nell'implementazione del loro livello di storage. Nei sistemi ASA r2, vengono utilizzate zone di disponibilità dello storage al posto degli aggregati. Fare riferimento a "questo" Articolo della Knowledge Base su come assegnare aggregati alle SVM nei sistemi ASA r2.

È inoltre possibile eseguire più di un driver e creare classi di storage che puntino all'una o all'altra. Ad esempio, è possibile configurare un `san-dev` classe che utilizza `ontap-san` driver e a. `san-default` classe che utilizza `ontap-san-economy` uno.

Tutti i nodi di lavoro di Kubernetes devono disporre dei tool iSCSI appropriati. Fare riferimento a. "Preparare il nodo di lavoro" per ulteriori informazioni.

Autenticare il backend ONTAP

Trident offre due modalità di autenticazione di un backend ONTAP.

- Basato sulle credenziali: Nome utente e password di un utente ONTAP con le autorizzazioni richieste. Si consiglia di utilizzare un ruolo di accesso di sicurezza predefinito, ad esempio `admin` oppure `vsadmin` Per garantire la massima compatibilità con le versioni di ONTAP.
- Basato su certificato: Trident può anche comunicare con un cluster ONTAP utilizzando un certificato installato sul backend. In questo caso, la definizione di backend deve contenere i valori codificati in Base64

del certificato client, della chiave e del certificato CA attendibile, se utilizzato (consigliato).

È possibile aggiornare i backend esistenti per passare da un metodo basato su credenziali a un metodo basato su certificato. Tuttavia, è supportato un solo metodo di autenticazione alla volta. Per passare a un metodo di autenticazione diverso, è necessario rimuovere il metodo esistente dalla configurazione di back-end.

> Se si tenta di fornire **credenziali e certificati**, la creazione del backend non riesce e viene visualizzato un errore che indica che nel file di configurazione sono stati forniti più metodi di autenticazione.

Abilitare l'autenticazione basata su credenziali

Trident richiede le credenziali di un amministratore con ambito SVM/cluster per comunicare con il back-end ONTAP. Si consiglia di utilizzare ruoli standard predefiniti come `admin` o `vsadmin`. Ciò garantisce la compatibilità con le future versioni di ONTAP che potrebbero esporre le API delle funzioni da utilizzare nelle future versioni di Trident. È possibile creare e utilizzare un ruolo di accesso di protezione personalizzato con Trident, ma non è consigliabile.

Una definizione di back-end di esempio avrà un aspetto simile al seguente:

YAML

```yaml
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: password
```

JSON

```json
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password"
}
```

Tenere presente che la definizione di backend è l'unica posizione in cui le credenziali vengono memorizzate in testo normale. Una volta creato il backend, i nomi utente e le password vengono codificati con Base64 e memorizzati come segreti Kubernetes. La creazione o l'aggiornamento di un backend è l'unico passaggio che richiede la conoscenza delle credenziali. Pertanto, si tratta di un'operazione di sola amministrazione, che deve

essere eseguita dall'amministratore Kubernetes/storage.

Abilita l'autenticazione basata sul certificato

I backend nuovi ed esistenti possono utilizzare un certificato e comunicare con il backend ONTAP. Nella definizione di backend sono necessari tre parametri.

- ClientCertificate: Valore del certificato client codificato con base64.
- ClientPrivateKey: Valore codificato in base64 della chiave privata associata.
- TrustedCACertificate: Valore codificato in base64 del certificato CA attendibile. Se si utilizza una CA attendibile, è necessario fornire questo parametro. Questa operazione può essere ignorata se non viene utilizzata alcuna CA attendibile.

Un workflow tipico prevede i seguenti passaggi.

Fasi

1. Generare un certificato e una chiave del client. Durante la generazione, impostare il nome comune (CN) sull'utente ONTAP per l'autenticazione come.

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=admin"
```

2. Aggiungere un certificato CA attendibile al cluster ONTAP. Questo potrebbe essere già gestito dall'amministratore dello storage. Ignorare se non viene utilizzata alcuna CA attendibile.

```
security certificate install -type server -cert-name <trusted-ca-cert-
name> -vserver <vserver-name>
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled
true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca
<cert-authority>
```

3. Installare il certificato e la chiave del client (dal passaggio 1) sul cluster ONTAP.

```
security certificate install -type client-ca -cert-name <certificate-
name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

4. Verificare che il ruolo di accesso di sicurezza di ONTAP supporti `cert` metodo di autenticazione.

```
security login create -user-or-group-name admin -application ontapi
-authentication-method cert
security login create -user-or-group-name admin -application http
-authentication-method cert
```

5. Verifica dell'autenticazione utilizzando il certificato generato. Sostituire <LIF di gestione ONTAP> e <vserver name> con IP LIF di gestione e nome SVM.

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns="http://www.netapp.com/filer/admin" version="1.21"
vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. Codifica certificato, chiave e certificato CA attendibile con Base64.

```
base64 -w 0 k8senv.pem >> cert_base64
base64 -w 0 k8senv.key >> key_base64
base64 -w 0 trustedca.pem >> trustedca_base64
```

7. Creare il backend utilizzando i valori ottenuti dal passaggio precedente.

```
cat cert-backend.json
{
"version": 1,
"storageDriverName": "ontap-san",
"backendName": "SanBackend",
"managementLIF": "1.2.3.4",
"svm": "vserver_test",
"clientCertificate": "Faaaakkkkeeee...Vaaallllluuuueeee",
"clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
"trustedCACertificate": "QNFinfO...SiqOyN",
"storagePrefix": "myPrefix_"
}

tridentctl create backend -f cert-backend.json -n trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                  UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
online |       0 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Aggiornare i metodi di autenticazione o ruotare le credenziali

È possibile aggiornare un backend esistente per utilizzare un metodo di autenticazione diverso o per ruotare le credenziali. Questo funziona in entrambi i modi: I backend che utilizzano il nome utente/la password possono essere aggiornati per utilizzare i certificati; i backend che utilizzano i certificati possono essere aggiornati in base al nome utente/alla password. A tale scopo, è necessario rimuovere il metodo di autenticazione esistente e aggiungere il nuovo metodo di autenticazione. Quindi, utilizzare il file backend.json aggiornato contenente i parametri necessari per l'esecuzione `tridentctl backend update`.

```
cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-san",
"backendName": "SanBackend",
"managementLIF": "1.2.3.4",
"svm": "vserver_test",
"username": "vsadmin",
"password": "password",
"storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend SanBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

> Quando si ruotano le password, l'amministratore dello storage deve prima aggiornare la password per l'utente su ONTAP. Seguito da un aggiornamento back-end. Durante la rotazione dei certificati, è possibile aggiungere più certificati all'utente. Il backend viene quindi aggiornato per utilizzare il nuovo certificato, dopodiché il vecchio certificato può essere cancellato dal cluster ONTAP.

L'aggiornamento di un backend non interrompe l'accesso ai volumi già creati, né influisce sulle connessioni dei volumi effettuate successivamente. Un aggiornamento backend corretto indica che Trident può comunicare con il back-end ONTAP e gestire operazioni future sui volumi.

Creare un ruolo ONTAP personalizzato per Trident

Puoi creare un ruolo cluster ONTAP con Minimum Privileges in modo da non dover utilizzare il ruolo di

amministratore ONTAP per eseguire le operazioni in Trident. Quando si include il nome utente in una configurazione backend Trident, Trident utilizza il ruolo del cluster ONTAP creato per eseguire le operazioni.

Per ulteriori informazioni sulla creazione di ruoli personalizzati di Trident, fare riferimento a."Generatore di ruoli personalizzati Trident"

Utilizzo della CLI di ONTAP

1. Creare un nuovo ruolo utilizzando il seguente comando:

```
security login role create <role_name\> -cmddirname "command" -access all
-vserver <svm_name\>
```

2. Creare un nome utente per l'utente Trident:

```
security login create -username <user_name\> -application ontapi
-authmethod <password\> -role <name_of_role_in_step_1\> -vserver
<svm_name\> -comment "user_description"
```

3. Associare il ruolo all'utente:

```
security login modify username <user_name\> -vserver <svm_name\> -role
<role_name\> -application ontapi -application console -authmethod
<password\>
```

Utilizzo di System Manager

In Gestione sistema di ONTAP, eseguire le seguenti operazioni:

1. **Crea un ruolo personalizzato**:
   a. Per creare un ruolo personalizzato a livello di cluster, selezionare **Cluster > Impostazioni**.

      (Oppure) per creare un ruolo personalizzato a livello di SVM, selezionare **Storage > Storage VM `required SVM` > > Impostazioni > utenti e ruoli**.

   b. Selezionare l'icona a freccia (→) accanto a **utenti e ruoli**.
   c. Selezionare **+Aggiungi** in **ruoli**.
   d. Definire le regole per il ruolo e fare clic su **Salva**.
2. **Associare il ruolo all'utente Trident**: + eseguire i seguenti passaggi nella pagina **utenti e ruoli**:
   a. Selezionare icona Aggiungi **+** in **utenti**.
   b. Selezionare il nome utente richiesto e scegliere un ruolo nel menu a discesa **ruolo**.
   c. Fare clic su **Save** (Salva).

Per ulteriori informazioni, fare riferimento alle pagine seguenti:

- "Ruoli personalizzati per l'amministrazione di ONTAP" o. "Definire ruoli personalizzati"
- "Lavorare con ruoli e utenti"

Autenticare le connessioni con CHAP bidirezionale

Trident può autenticare le sessioni iSCSI con CHAP bidirezionale per i `ontap-san` driver e. `ontap-san-economy` Ciò richiede l'attivazione dell' `useCHAP`opzione nella definizione di backend. Quando è impostato su `true`, Trident configura la protezione dell'iniziatore predefinito della SVM su CHAP bidirezionale e imposta il nome utente e i segreti dal file backend. NetApp consiglia di utilizzare CHAP bidirezionale per autenticare le connessioni. Vedere la seguente configurazione di esempio:

```
---
version: 1
storageDriverName: ontap-san
backendName: ontap_san_chap
managementLIF: 192.168.0.135
svm: ontap_iscsi_svm
useCHAP: true
username: vsadmin
password: password
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
```

> Il `useCHAP` Parameter è un'opzione booleana che può essere configurata una sola volta. L'impostazione predefinita è false. Una volta impostato su true, non è possibile impostarlo su false.

Oltre a `useCHAP=true`, il `chapInitiatorSecret`, `chapTargetInitiatorSecret`, `chapTargetUsername`, e. `chapUsername` i campi devono essere inclusi nella definizione di backend. I segreti possono essere modificati dopo la creazione di un backend mediante l'esecuzione `tridentctl update`.

Come funziona

Impostando `useCHAP` su true, l'amministratore dello storage richiede a Trident di configurare CHAP sul backend dello storage. Ciò include quanto segue:

- Impostazione di CHAP su SVM:
  - Se il tipo di protezione iniziatore predefinito della SVM è nessuno (impostato per impostazione predefinita) **e** non sono già presenti LUN preesistenti nel volume, Trident imposterà il tipo di protezione predefinito su `CHAP` e procederà alla configurazione dell'iniziatore CHAP e del nome utente e dei segreti di destinazione.
  - Se la SVM contiene LUN, Trident non attiva il protocollo CHAP nella SVM. In questo modo, l'accesso ai LUN già presenti nella SVM non è limitato.
- Configurazione dell'iniziatore CHAP e del nome utente e dei segreti di destinazione; queste opzioni devono essere specificate nella configurazione del backend (come mostrato sopra).

Una volta creato il backend, Trident crea un CRD corrispondente `tridentbackend` e memorizza i segreti CHAP e i nomi utente come segreti Kubernetes. Tutti i PVS creati da Trident su questo backend verranno

montati e collegati tramite CHAP.

Ruota le credenziali e aggiorna i backend

È possibile aggiornare le credenziali CHAP aggiornando i parametri CHAP in `backend.json` file. Per eseguire questa operazione, è necessario aggiornare i segreti CHAP e utilizzare `tridentctl update` per riflettere queste modifiche.

> Quando si aggiornano i segreti CHAP per un backend, è necessario utilizzare `tridentctl` per aggiornare il backend. Non aggiornare le credenziali sul cluster di storage utilizzando l'interfaccia a riga di comando di ONTAP o ONTAP System Manager poiché Trident non sarà in grado di accettare queste modifiche.

```
cat backend-san.json
{
    "version": 1,
    "storageDriverName": "ontap-san",
    "backendName": "ontap_san_chap",
    "managementLIF": "192.168.0.135",
    "svm": "ontap_iscsi_svm",
    "useCHAP": true,
    "username": "vsadmin",
    "password": "password",
    "chapInitiatorSecret": "cl9qxUpDaTeD",
    "chapTargetInitiatorSecret": "rqxigXgkeUpDaTeD",
    "chapTargetUsername": "iJF4heBRT0TCwxyz",
    "chapUsername": "uh2aNCLSd6cNwxyz",
}

./tridentctl update backend ontap_san_chap -f backend-san.json -n trident
+----------------+----------------+--------------------------------------
+--------+---------+
|   NAME         | STORAGE DRIVER |                  UUID                |
STATE   | VOLUMES |
+----------------+----------------+--------------------------------------
+--------+---------+
| ontap_san_chap | ontap-san      | aa458f3b-ad2d-4378-8a33-1a472ffbeb5c |
online |       7 |
+----------------+----------------+--------------------------------------
+--------+---------+
```

Le connessioni esistenti non subiranno alcun problema e continueranno a rimanere attive se le credenziali vengono aggiornate da Trident sulla SVM. Le nuove connessioni utilizzano le credenziali aggiornate e le connessioni esistenti continuano a rimanere attive. Disconnettendo e riconnettendo il vecchio PVS, verranno utilizzate le credenziali aggiornate.

Opzioni ed esempi di configurazione DELLA SAN ONTAP

Informazioni su come creare e utilizzare i driver SAN ONTAP con l'installazione Trident. In questa sezione vengono forniti esempi di configurazione backend e dettagli per la mappatura dei backend a StorageClasses.

"Sistemi ASA r2" differiscono dagli altri sistemi ONTAP (ASA, AFF e FAS) nell'implementazione del loro livello di archiviazione. Queste variazioni influiscono sull'utilizzo di determinati parametri, come indicato. "Scopri di più sulle differenze tra i sistemi ASA r2 e gli altri sistemi ONTAP".

> Solo il `ontap-san` driver (con protocolli iSCSI e NVMe/TCP) è supportato per i sistemi ASA r2.

Nella configurazione del backend Trident, non è necessario specificare che il sistema sia ASA r2. Quando si seleziona `ontap-san` come il `storageDriverName` Trident rileva automaticamente il sistema ASA r2 o il sistema ONTAP tradizionale. Alcuni parametri di configurazione del backend non sono applicabili ai sistemi ASA r2, come indicato nella tabella seguente.

Opzioni di configurazione back-end

Per le opzioni di configurazione del backend, consultare la tabella seguente:

Parametro	Descrizione	Predefinito
`version`		Sempre 1
`storageDriverName`	Nome del driver di storage	`ontap-san` o. `ontap-san-economy`
`backendName`	Nome personalizzato o backend dello storage	Nome del driver + "_" + dataLIF
`managementLIF`	Indirizzo IP di un cluster o di una LIF di gestione SVM. È possibile specificare un nome di dominio completo (FQDN). Può essere impostato in modo da utilizzare gli indirizzi IPv6 se Trident è stato installato utilizzando il flag IPv6. Gli indirizzi IPv6 devono essere definiti tra parentesi quadre, ad esempio `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . Per lo switchover di MetroCluster senza problemi, vedere la Esempio MetroCluster. Se stai utilizzando credenziali "vsadmin", `managementLIF` devi essere quelle della SVM; se utilizzi credenziali "admin", `managementLIF` devi essere quelle del cluster.	"10,0.0,1", "[2001:1234:abcd::fefe]"

Parametro	Descrizione	Predefinito
`dataLIF`	Indirizzo IP del protocollo LIF. Può essere impostato in modo da utilizzare gli indirizzi IPv6 se Trident è stato installato utilizzando il flag IPv6. Gli indirizzi IPv6 devono essere definiti tra parentesi quadre, ad esempio `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . **Non specificare per iSCSI.** Trident utilizza "Mappa LUN selettiva ONTAP" per rilevare le LIF iSCSI necessarie per stabilire una sessione multi-path. Viene generato un avviso se `dataLIF` è definito esplicitamente. **Omettere per MetroCluster.** Consultare la Esempio MetroCluster.	Derivato dalla SVM
`svm`	Macchina virtuale per lo storage da utilizzare **Ometti per MetroCluster.** vedere la Esempio MetroCluster.	Derivato se un SVM `managementLIF` è specificato
`useCHAP`	Utilizzare CHAP per autenticare iSCSI per i driver SAN ONTAP [booleano]. Impostare su `true` for Trident per configurare e utilizzare il protocollo CHAP bidirezionale come autenticazione predefinita per la SVM fornita nel backend. Per ulteriori informazioni, fare riferimento alla "Prepararsi a configurare il backend con i driver SAN ONTAP" sezione. **Non supportato per FCP o NVMe/TCP.**	`false`
`chapInitiatorSecret`	Segreto iniziatore CHAP. Necessario se `useCHAP=true`	""
`labels`	Set di etichette arbitrarie formattate con JSON da applicare sui volumi	""
`chapTargetInitiatorSecret`	CHAP target Initiator secret. Necessario se `useCHAP=true`	""
`chapUsername`	Nome utente inbound. Necessario se `useCHAP=true`	""
`chapTargetUsername`	Nome utente di destinazione. Necessario se `useCHAP=true`	""
`clientCertificate`	Valore del certificato client codificato con base64. Utilizzato per l'autenticazione basata su certificato	""
`clientPrivateKey`	Valore codificato in base64 della chiave privata del client. Utilizzato per l'autenticazione basata su certificato	""
`trustedCACertificate`	Valore codificato in base64 del certificato CA attendibile. Opzionale. Utilizzato per l'autenticazione basata su certificato.	""
`username`	Nome utente necessario per comunicare con il cluster ONTAP. Utilizzato per l'autenticazione basata su credenziali.	""

Parametro	Descrizione	Predefinito
`password`	Password necessaria per comunicare con il cluster ONTAP. Utilizzato per l'autenticazione basata su credenziali.	""
`svm`	Macchina virtuale per lo storage da utilizzare	Derivato se un SVM `managementLIF` è specificato
`storagePrefix`	Prefisso utilizzato per il provisioning di nuovi volumi nella SVM. Non può essere modificato in seguito. Per aggiornare questo parametro, è necessario creare un nuovo backend.	`trident`
`aggregate`	Aggregato per il provisioning (facoltativo; se impostato, deve essere assegnato alla SVM). Per il `ontap-nas-flexgroup` driver, questa opzione viene ignorata. Se non viene assegnato, è possibile utilizzare qualsiasi aggregato disponibile per il provisioning di un volume FlexGroup. Una volta aggiornato l'aggregato in SVM, viene aggiornato automaticamente in Trident eseguendo un polling della SVM senza riavviare il controller Trident. Dopo aver configurato un aggregato specifico in Trident per il provisioning dei volumi, in caso di ridenominazione o spostamento dell'aggregato dalla SVM, il back-end passa allo stato di errore in Trident durante il polling dell'aggregato della SVM. È necessario modificare l'aggregato in uno presente nella SVM o rimuoverlo del tutto per riportare online il back-end. **Non specificare per i sistemi ASA r2**.	""
`limitAggregateUsage`	Il provisioning non riesce se l'utilizzo è superiore a questa percentuale. Se si utilizza un backend Amazon FSX per NetApp ONTAP, non specificare `limitAggregateUsage`. Fornito `fsxadmin` e `vsadmin` non contiene le autorizzazioni necessarie per recuperare l'utilizzo dell'aggregato e limitarlo mediante Trident. **Non specificare per i sistemi ASA r2**.	"" (non applicato per impostazione predefinita)
`limitVolumeSize`	Fallire il provisioning se la dimensione del volume richiesta è superiore a questo valore. Limita anche le dimensioni massime dei volumi che gestisce per i LUN.	"" (non applicato per impostazione predefinita)
`lunsPerFlexvol`	LUN massimi per FlexVol, devono essere compresi nell'intervallo [50, 200]	`100`

Parametro	Descrizione	Predefinito
`debugTraceFlags`	Flag di debug da utilizzare per la risoluzione dei problemi. Esempio, {"api":false, "method":true} Non utilizzare a meno che non si stia eseguendo la risoluzione dei problemi e non si richieda un dump dettagliato del log.	`null`
`useREST`	Parametro booleano per utilizzare le API REST ONTAP . `` `useREST`Quando impostato su `true` , Trident utilizza le API REST ONTAP per comunicare con il backend; quando impostato su `false` Trident utilizza chiamate ONTAPI (ZAPI) per comunicare con il backend. Questa funzionalità richiede ONTAP 9.11.1 e versioni successive. Inoltre, il ruolo di accesso ONTAP utilizzato deve avere accesso a `ontapi` applicazione. Ciò è soddisfatto dal predefinito `vsadmin` E `cluster-admin` ruoli. A partire dalla versione Trident 24.06 e ONTAP 9.15.1 o successiva, `useREST` è impostato su `true` per impostazione predefinita; modifica `useREST` A `false` per utilizzare le chiamate ONTAPI (ZAPI). `` `useREST`è completamente qualificato per NVMe/TCP. NVMe è supportato solo con le API REST ONTAP e non con ONTAPI (ZAPI). **Se specificato, impostare sempre su `true` per sistemi ASA r2**.	`true` Per ONTAP 9.15.1 o versioni successive, altrimenti `false`.
`sanType`	Utilizzare per selezionare `iscsi` iSCSI, `nvme` NVMe/TCP o `fcp` SCSI over Fibre Channel (FC).	`iscsi` se vuoto

Parametro	Descrizione	Predefinito
`formatOptions`	Consente `formatOptions` di specificare gli argomenti della riga di comando per il `mkfs` comando, che verranno applicati ogni volta che un volume viene formattato. In questo modo è possibile formattare il volume in base alle proprie preferenze. Assicurarsi di specificare le opzioni formatOptions simili a quelle del comando mkfs, escludendo il percorso del dispositivo. Esempio: "-e nodiscard" **Supportato per `ontap-san` E `ontap-san-economy` driver con protocollo iSCSI. Inoltre, supportati per sistemi ASA r2 quando si utilizzano i protocolli iSCSI e NVMe/TCP.**	
`limitVolumePoolSize`	Dimensioni massime degli FlexVol richiedibili quando si utilizzano le LUN di un backend ONTAP-san-economy.	"" (non applicato per impostazione predefinita)
`denyNewVolumePools`	Limita `ontap-san-economy` i backend dalla creazione di nuovi volumi FlexVol per contenere le proprie LUN. Per il provisioning di nuovi PVS vengono utilizzati solo i FlexVol preesistenti.	

Consigli per l'uso di formatOptions

Per accelerare il processo di formattazione, Trident consiglia di utilizzare la seguente opzione:

-e nodose:

- Keep, non tentare di scartare i blocchi al momento di mkfs (scartare inizialmente i blocchi è utile sui dispositivi a stato solido e sullo storage sparse/thin-provisioning). Questo sostituisce l'opzione obsoleta "-K" ed è applicabile a tutti i file system (xfs, ext3 e ext4).

Opzioni di configurazione back-end per il provisioning dei volumi

È possibile controllare il provisioning predefinito utilizzando queste opzioni in `defaults` della configurazione. Per un esempio, vedere gli esempi di configurazione riportati di seguito.

Parametro	Descrizione	Predefinito
`spaceAllocation`	Allocazione dello spazio per LUN	"true" **Se specificato, impostare su `true` per sistemi ASA r2**.
`spaceReserve`	Modalità di prenotazione dello spazio; "nessuno" (sottile) o "volume" (spesso). **Impostato su `none` per sistemi ASA r2**.	"nessuno"
`snapshotPolicy`	Policy Snapshot da utilizzare. **Impostato su `none` per sistemi ASA r2**.	"nessuno"

Parametro	Descrizione	Predefinito
`qosPolicy`	Gruppo di criteri QoS da assegnare per i volumi creati. Scegliere tra qosPolicy o adaptiveQosPolicy per pool di storage/backend. L'utilizzo di gruppi di criteri QoS con Trident richiede ONTAP 9.8 o versioni successive. È necessario utilizzare un gruppo di criteri QoS non condiviso e garantire che il gruppo di criteri venga applicato singolarmente a ciascun componente. Un gruppo di policy QoS condiviso impone un limite massimo per il throughput totale di tutti i carichi di lavoro.	""
`adaptiveQosPolicy`	Gruppo di criteri QoS adattivi da assegnare per i volumi creati. Scegliere tra qosPolicy o adaptiveQosPolicy per pool di storage/backend	""
`snapshotReserve`	Percentuale del volume riservato alle snapshot. **Non specificare per i sistemi ASA r2**.	"0" se `snapshotPolicy` è "nessuno", altrimenti ""
`splitOnClone`	Separare un clone dal suo padre al momento della creazione	"falso"
`encryption`	Abilitare la crittografia del volume NetApp (NVE) sul nuovo volume; il valore predefinito è `false`. NVE deve essere concesso in licenza e abilitato sul cluster per utilizzare questa opzione. Se NAE è abilitato sul backend, qualsiasi volume sottoposto a provisioning in Trident sarà abilitato NAE. Per ulteriori informazioni, fare riferimento a: "Come funziona Trident con NVE e NAE".	"false" **Se specificato, impostare su `true` per sistemi ASA r2**.
`luksEncryption`	Attivare la crittografia LUKS. Fare riferimento alla "Utilizzo di Linux Unified Key Setup (LUKS)".	"" **Impostato su `false` per sistemi ASA r2**.
`tieringPolicy`	Criterio di suddivisione in livelli per utilizzare "none" **Non specificare per i sistemi ASA r2**.	
`nameTemplate`	Modello per creare nomi di volume personalizzati.	""

Esempi di provisioning di volumi

Ecco un esempio con i valori predefiniti definiti:

```yaml
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: trident_svm
username: admin
password: <password>
labels:
  k8scluster: dev2
  backend: dev2-sanbackend
storagePrefix: alternate-trident
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: standard
  spaceAllocation: 'false'
  snapshotPolicy: default
  snapshotReserve: '10'
```

> Per tutti i volumi creati utilizzando il `ontap-san` driver, Trident aggiunge un ulteriore 10% di capacità alla FlexVol per ospitare i metadati LUN. Il LUN viene fornito con le dimensioni esatte richieste dall'utente nel PVC. Trident aggiunge il 10% al FlexVol (mostra come dimensioni disponibili in ONTAP). A questo punto, gli utenti otterranno la quantità di capacità utilizzabile richiesta. Questa modifica impedisce inoltre che le LUN diventino di sola lettura, a meno che lo spazio disponibile non sia completamente utilizzato. Ciò non si applica a ontap-san-Economy.

Per i backend che definiscono `snapshotReserve`, Trident calcola le dimensioni dei volumi come segue:

```
Total volume size = [(PVC requested size) / (1 - (snapshotReserve
percentage) / 100)] * 1.1
```

L'1,1 è il 10 percento in più Trident aggiunge al FlexVol per ospitare i metadati LUN. Per `snapshotReserve` = 5% e richiesta PVC = 5 GiB, la dimensione totale del volume è 5,79 GiB e la dimensione disponibile è 5,5 GiB . `volume show` il comando dovrebbe mostrare risultati simili a questo esempio:

```
Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- -----
                  _pvc_89f1c156_3801_4de4_9f9d_034d54c395f4
                                   online     RW         10GB     5.00GB    0%
                  _pvc_e42ec6fe_3baa_4af6_996d_134adbbb8e6d
                                   online     RW       5.79GB     5.50GB    0%
                  _pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba
                                   online     RW          1GB    511.8MB    0%
3 entries were displayed.
```

Attualmente, il ridimensionamento è l'unico modo per utilizzare il nuovo calcolo per un volume esistente.

Esempi di configurazione minimi

Gli esempi seguenti mostrano le configurazioni di base che lasciano la maggior parte dei parametri predefiniti. Questo è il modo più semplice per definire un backend.

> Se utilizzi Amazon FSX su NetApp ONTAP con Trident, NetApp consiglia di specificare i nomi DNS per le LIF invece degli indirizzi IP.

Esempio DI SAN ONTAP

Si tratta di una configurazione di base che utilizza `ontap-san` driver.

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
labels:
  k8scluster: test-cluster-1
  backend: testcluster1-sanbackend
username: vsadmin
password: <password>
```

Esempio MetroCluster

È possibile configurare il backend per evitare di dover aggiornare manualmente la definizione del backend dopo lo switchover e lo switchback durante "Replica e recovery di SVM".

Per uno switchover e uno switchback perfetto, specifica la SVM utilizzando `managementLIF` ed omette i `svm` parametri. Ad esempio:

```
version: 1
storageDriverName: ontap-san
managementLIF: 192.168.1.66
username: vsadmin
password: password
```

Esempio di economia SAN ONTAP

```
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
username: vsadmin
password: <password>
```

Esempio di autenticazione basata su certificato

In questo esempio di configurazione di base `clientCertificate`, `clientPrivateKey`, e. `trustedCACertificate` (Facoltativo, se si utilizza una CA attendibile) sono inseriti in `backend.json` E prendere rispettivamente i valori codificati base64 del certificato client, della chiave privata e del certificato CA attendibile.

```
---
version: 1
storageDriverName: ontap-san
backendName: DefaultSANBackend
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
```

Esempi CHAP bidirezionali

Questi esempi creano un backend con `useCHAP` impostare su `true`.

Esempio di SAN ONTAP CHAP

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
labels:
  k8scluster: test-cluster-1
  backend: testcluster1-sanbackend
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
username: vsadmin
password: <password>
```

Esempio di ONTAP SAN economy CHAP

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
username: vsadmin
password: <password>
```

Esempio NVMe/TCP

Devi disporre di una SVM configurata con NVMe sul back-end ONTAP. Si tratta di una configurazione backend di base per NVMe/TCP.

```
---
version: 1
backendName: NVMeBackend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_nvme
username: vsadmin
password: password
sanType: nvme
useREST: true
```

Esempio di SCSI su FC (FCP)

Devi disporre di una SVM configurata con FC sul back-end ONTAP. Configurazione backend di base per FC.

```
---
version: 1
backendName: fcp-backend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_fc
username: vsadmin
password: password
sanType: fcp
useREST: true
```

Esempio di configurazione backend con nameTemplate

```
---
version: 1
storageDriverName: ontap-san
backendName: ontap-san-backend
managementLIF: <ip address>
svm: svm0
username: <admin>
password: <password>
defaults:
  nameTemplate:
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.vo\
    lume.RequestName}}"
labels:
  cluster: ClusterA
  PVC: "{{.volume.Namespace}}_{{.volume.RequestName}}"
```

Esempio di formattoOpzioni per il driver ONTAP-san-economy

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: ""
svm: svm1
username: ""
password: "!"
storagePrefix: whelk_
debugTraceFlags:
  method: true
  api: true
defaults:
  formatOptions: -E nodiscard
```

Esempi di backend con pool virtuali

In questi file di definizione back-end di esempio, vengono impostati valori predefiniti specifici per tutti i pool di storage, ad esempio `spaceReserve` a nessuno, `spaceAllocation` a false, e. `encryption` a falso. I pool virtuali sono definiti nella sezione storage.

Trident imposta le etichette di provisioning nel campo "commenti". I commenti vengono impostati sulle copie FlexVol volume Trident. Tutte le etichette presenti su un pool virtuale nel volume di storage al momento del provisioning. Per comodità, gli amministratori dello storage possono definire le etichette per ogni pool virtuale e raggruppare i volumi per etichetta.

In questi esempi, alcuni dei pool di storage sono impostati in modo personalizzato `spaceReserve`, `spaceAllocation`, e. `encryption` e alcuni pool sovrascrivono i valori predefiniti.

Esempio DI SAN ONTAP

```yaml
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
username: vsadmin
password: <password>
defaults:
  spaceAllocation: "false"
  encryption: "false"
  qosPolicy: standard
labels:
  store: san_store
  kubernetes-cluster: prod-cluster-1
region: us_east_1
storage:
  - labels:
      protection: gold
      creditpoints: "40000"
    zone: us_east_1a
    defaults:
      spaceAllocation: "true"
      encryption: "true"
      adaptiveQosPolicy: adaptive-extreme
  - labels:
      protection: silver
      creditpoints: "20000"
    zone: us_east_1b
    defaults:
      spaceAllocation: "false"
      encryption: "true"
      qosPolicy: premium
  - labels:
      protection: bronze
      creditpoints: "5000"
    zone: us_east_1c
    defaults:
      spaceAllocation: "true"
      encryption: "false"
```

Esempio di economia SAN ONTAP

```yaml
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
username: vsadmin
password: <password>
defaults:
  spaceAllocation: "false"
  encryption: "false"
labels:
  store: san_economy_store
region: us_east_1
storage:
  - labels:
      app: oracledb
      cost: "30"
    zone: us_east_1a
    defaults:
      spaceAllocation: "true"
      encryption: "true"
  - labels:
      app: postgresdb
      cost: "20"
    zone: us_east_1b
    defaults:
      spaceAllocation: "false"
      encryption: "true"
  - labels:
      app: mysqldb
      cost: "10"
    zone: us_east_1c
    defaults:
      spaceAllocation: "true"
      encryption: "false"
  - labels:
      department: legal
      creditpoints: "5000"
    zone: us_east_1c
```

```
    defaults:
      spaceAllocation: "true"
      encryption: "false"
```

Esempio NVMe/TCP

```
---
version: 1
storageDriverName: ontap-san
sanType: nvme
managementLIF: 10.0.0.1
svm: nvme_svm
username: vsadmin
password: <password>
useREST: true
defaults:
  spaceAllocation: "false"
  encryption: "true"
storage:
  - labels:
      app: testApp
      cost: "20"
    defaults:
      spaceAllocation: "false"
      encryption: "false"
```

Mappare i backend in StorageClasses

Le seguenti definizioni di StorageClass fanno riferimento a. Esempi di backend con pool virtuali. Utilizzando il `parameters.selector` Ciascun StorageClass richiama i pool virtuali che possono essere utilizzati per ospitare un volume. Gli aspetti del volume saranno definiti nel pool virtuale scelto.

- Il `protection-gold` StorageClass verrà mappato al primo pool virtuale in `ontap-san` back-end. Questo è l'unico pool che offre una protezione di livello gold.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=gold"
  fsType: "ext4"
```

- Il `protection-not-gold` StorageClass eseguirà il mapping al secondo e al terzo pool virtuale in `ontap-san` back-end. Questi sono gli unici pool che offrono un livello di protezione diverso dall'oro.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
```

- Il `app-mysqldb` StorageClass eseguirà il mapping al terzo pool virtuale in `ontap-san-economy` back-end. Questo è l'unico pool che offre la configurazione del pool di storage per l'applicazione di tipo mysqldb.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
```

- Il `protection-silver-creditpoints-20k` StorageClass eseguirà il mapping al secondo pool virtuale in `ontap-san` back-end. Questo è l'unico pool che offre una protezione di livello Silver e 20000 punti di credito.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
```

- Il `creditpoints-5k` StorageClass eseguirà il mapping al terzo pool virtuale in `ontap-san` il back-end e il quarto pool virtuale in `ontap-san-economy` back-end. Queste sono le uniche offerte di pool con 5000 punti di credito.

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: csi.trident.netapp.io
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"
```

- Il `my-test-app-sc` StorageClass verrà mappato su `testAPP` pool virtuale in `ontap-san` conducente con `sanType: nvme`. Si tratta dell'unica offerta di piscina `testApp`.

```yaml
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: my-test-app-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=testApp"
  fsType: "ext4"
```

Trident deciderà quale pool virtuale viene selezionato e garantirà che i requisiti di storage vengano soddisfatti.

Driver NAS ONTAP

Panoramica del driver NAS ONTAP

Informazioni sulla configurazione di un backend ONTAP con driver NAS ONTAP e Cloud Volumes ONTAP.

Dettagli del driver NAS ONTAP

Trident fornisce i seguenti driver di storage NAS per comunicare con il cluster ONTAP. Le modalità di accesso supportate sono: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Driver	Protocollo	VolumeMode	Modalità di accesso supportate	File system supportati
`ontap-nas`	NFS PMI	Filesystem	RWO, ROX, RWX, RWOP	"", `nfs`, `smb`
`ontap-nas-economy`	NFS PMI	Filesystem	RWO, ROX, RWX, RWOP	"", `nfs`, `smb`

Driver	Protocollo	VolumeMode	Modalità di accesso supportate	File system supportati
`ontap-nas-flexgroup`	NFS PMI	Filesystem	RWO, ROX, RWX, RWOP	"", `nfs`, `smb`

- Utilizzare `ontap-san-economy` solo se si prevede che il conteggio dell'utilizzo persistente del volume sia superiore a. "Limiti di volume ONTAP supportati".
- Utilizzare `ontap-nas-economy` solo se si prevede che il conteggio dell'utilizzo persistente del volume sia superiore a. "Limiti di volume ONTAP supportati" e a. `ontap-san-economy` impossibile utilizzare il driver.
- Non utilizzare `ontap-nas-economy` se prevedete la necessità di protezione dei dati, disaster recovery o mobilità.
- NetApp sconsiglia di utilizzare l'espansione automatica FlexVol in tutti i driver ONTAP, ad eccezione di ONTAP-san. Come soluzione alternativa, Trident supporta l'utilizzo di una riserva di snapshot e scala di conseguenza i volumi FlexVol.

Autorizzazioni utente

Trident può essere eseguito come amministratore di ONTAP o SVM, in genere utilizzando un utente del cluster o un utente SVM `admin` o `vsadmin` un utente con un nome diverso che svolge lo stesso ruolo.

Per le implementazioni di Amazon FSX per NetApp ONTAP, Trident si aspetta un'esecuzione come amministratore ONTAP o SVM, con l'utente del cluster `fsxadmin`, un `vsadmin` utente SVM o un utente con un nome diverso che abbia lo stesso ruolo. L' `fsxadmin`utente sostituisce in modo limitato l'utente amministratore del cluster.

Se si utilizza il `limitAggregateUsage` parametro, sono necessarie le autorizzazioni di amministratore del cluster. Quando si utilizza Amazon FSX per NetApp ONTAP con Trident, il `limitAggregateUsage` parametro non funziona con `vsadmin` gli account utente e. `fsxadmin` L'operazione di configurazione non riesce se si specifica questo parametro.

Sebbene sia possibile creare un ruolo più restrittivo all'interno di ONTAP che un driver Trident può utilizzare, non lo consigliamo. La maggior parte delle nuove release di Trident chiamerà API aggiuntive che dovrebbero essere considerate, rendendo gli aggiornamenti difficili e soggetti a errori.

Prepararsi a configurare un backend con i driver NAS ONTAP

Comprendere i requisiti, le opzioni di autenticazione e le policy di esportazione per la configurazione di un backend ONTAP con i driver NAS ONTAP.

Requisiti

- Per tutti i backend ONTAP, Trident richiede che almeno un aggregato sia assegnato all'SVM.
- È possibile eseguire più di un driver e creare classi di storage che puntano all'una o all'altra. Ad esempio, è possibile configurare una classe Gold che utilizza `ontap-nas` Driver e una classe Bronze che utilizza `ontap-nas-economy` uno.
- Tutti i nodi di lavoro di Kubernetes devono avere installati gli strumenti NFS appropriati. Fare riferimento a. "qui" per ulteriori dettagli.

- Trident supporta volumi SMB montati su pod in esecuzione solo sui nodi Windows. Per ulteriori informazioni, fare riferimento alla Preparatevi al provisioning dei volumi SMB sezione.

Autenticare il backend ONTAP

Trident offre due modalità di autenticazione di un backend ONTAP.

- Basato sulle credenziali: Questa modalità richiede autorizzazioni sufficienti per il backend ONTAP. Si consiglia di utilizzare un account associato a un ruolo di accesso di sicurezza predefinito, ad esempio `admin` oppure `vsadmin` Per garantire la massima compatibilità con le versioni di ONTAP.
- Basato su certificato: Questa modalità richiede l'installazione di un certificato sul backend affinché Trident possa comunicare con un cluster ONTAP. In questo caso, la definizione di backend deve contenere i valori codificati in Base64 del certificato client, della chiave e del certificato CA attendibile, se utilizzato (consigliato).

È possibile aggiornare i backend esistenti per passare da un metodo basato su credenziali a un metodo basato su certificato. Tuttavia, è supportato un solo metodo di autenticazione alla volta. Per passare a un metodo di autenticazione diverso, è necessario rimuovere il metodo esistente dalla configurazione di back-end.

> Se si tenta di fornire **credenziali e certificati**, la creazione del backend non riesce e viene visualizzato un errore che indica che nel file di configurazione sono stati forniti più metodi di autenticazione.

Abilitare l'autenticazione basata su credenziali

Trident richiede le credenziali di un amministratore con ambito SVM/cluster per comunicare con il back-end ONTAP. Si consiglia di utilizzare ruoli standard predefiniti come `admin` o `vsadmin`. Ciò garantisce la compatibilità con le future versioni di ONTAP che potrebbero esporre le API delle funzioni da utilizzare nelle future versioni di Trident. È possibile creare e utilizzare un ruolo di accesso di protezione personalizzato con Trident, ma non è consigliabile.

Una definizione di back-end di esempio avrà un aspetto simile al seguente:

YAML

```yaml
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
credentials:
  name: secret-backend-creds
```

JSON

```json
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "credentials": {
        "name": "secret-backend-creds"
    }
}
```

Tenere presente che la definizione di backend è l'unica posizione in cui le credenziali vengono memorizzate in testo normale. Una volta creato il backend, i nomi utente e le password vengono codificati con Base64 e memorizzati come segreti Kubernetes. La creazione/l'updation di un backend è l'unico passaggio che richiede la conoscenza delle credenziali. Pertanto, si tratta di un'operazione di sola amministrazione, che deve essere eseguita dall'amministratore Kubernetes/storage.

Abilitare l'autenticazione basata su certificato

I backend nuovi ed esistenti possono utilizzare un certificato e comunicare con il backend ONTAP. Nella definizione di backend sono necessari tre parametri.

- ClientCertificate: Valore del certificato client codificato con base64.
- ClientPrivateKey: Valore codificato in base64 della chiave privata associata.
- TrustedCACertificate: Valore codificato in base64 del certificato CA attendibile. Se si utilizza una CA attendibile, è necessario fornire questo parametro. Questa operazione può essere ignorata se non viene utilizzata alcuna CA attendibile.

Un workflow tipico prevede i seguenti passaggi.

Fasi

1. Generare un certificato e una chiave del client. Durante la generazione, impostare il nome comune (CN) sull'utente ONTAP per l'autenticazione come.

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=vsadmin"
```

2. Aggiungere un certificato CA attendibile al cluster ONTAP. Questo potrebbe essere già gestito dall'amministratore dello storage. Ignorare se non viene utilizzata alcuna CA attendibile.

```
security certificate install -type server -cert-name <trusted-ca-cert-
name> -vserver <vserver-name>
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled
true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca
<cert-authority>
```

3. Installare il certificato e la chiave del client (dal passaggio 1) sul cluster ONTAP.

```
security certificate install -type client-ca -cert-name <certificate-
name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

4. Verificare che il ruolo di accesso di sicurezza di ONTAP supporti `cert` metodo di autenticazione.

```
security login create -user-or-group-name vsadmin -application ontapi
-authentication-method cert -vserver <vserver-name>
security login create -user-or-group-name vsadmin -application http
-authentication-method cert -vserver <vserver-name>
```

5. Verifica dell'autenticazione utilizzando il certificato generato. Sostituire <LIF di gestione ONTAP> e <vserver name> con IP LIF di gestione e nome SVM. Assicurarsi che la politica di servizio di LIF sia impostata su `default-data-management`.

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns="http://www.netapp.com/filer/admin" version="1.21"
vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. Codifica certificato, chiave e certificato CA attendibile con Base64.

```
base64 -w 0 k8senv.pem >> cert_base64
base64 -w 0 k8senv.key >> key_base64
base64 -w 0 trustedca.pem >> trustedca_base64
```

7. Creare il backend utilizzando i valori ottenuti dal passaggio precedente.

```
cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "NasBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"clientCertificate": "Faaaakkkkeeee...Vaaallllluuuueeee",
"clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
"storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Aggiornare i metodi di autenticazione o ruotare le credenziali

È possibile aggiornare un backend esistente per utilizzare un metodo di autenticazione diverso o per ruotare le credenziali. Questo funziona in entrambi i modi: I backend che utilizzano il nome utente/la password possono essere aggiornati per utilizzare i certificati; i backend che utilizzano i certificati possono essere aggiornati in base al nome utente/alla password. A tale scopo, è necessario rimuovere il metodo di autenticazione esistente e aggiungere il nuovo metodo di autenticazione. Quindi, utilizzare il file backend.json aggiornato contenente i parametri necessari per l'esecuzione `tridentctl update backend`.

```
cat cert-backend-updated.json
```

```
{
"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "NasBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"username": "vsadmin",
"password": "password",
"storagePrefix": "myPrefix_"
}
```

```
#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                  UUID                |
STATE   | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

> Quando si ruotano le password, l'amministratore dello storage deve prima aggiornare la password per l'utente su ONTAP. Seguito da un aggiornamento back-end. Durante la rotazione dei certificati, è possibile aggiungere più certificati all'utente. Il backend viene quindi aggiornato per utilizzare il nuovo certificato, dopodiché il vecchio certificato può essere cancellato dal cluster ONTAP.

L'aggiornamento di un backend non interrompe l'accesso ai volumi già creati, né influisce sulle connessioni dei volumi effettuate successivamente. Un aggiornamento backend corretto indica che Trident può comunicare con il back-end ONTAP e gestire operazioni future sui volumi.

Creare un ruolo ONTAP personalizzato per Trident

Puoi creare un ruolo cluster ONTAP con Minimum Privileges in modo da non dover utilizzare il ruolo di amministratore ONTAP per eseguire le operazioni in Trident. Quando si include il nome utente in una configurazione backend Trident, Trident utilizza il ruolo del cluster ONTAP creato per eseguire le operazioni.

Per ulteriori informazioni sulla creazione di ruoli personalizzati di Trident, fare riferimento a."Generatore di ruoli personalizzati Trident"

Utilizzo della CLI di ONTAP

1. Creare un nuovo ruolo utilizzando il seguente comando:

```
security login role create <role_name\> -cmddirname "command" -access all
-vserver <svm_name\>
```

2. Creare un nome utente per l'utente Trident:

```
security login create -username <user_name\> -application ontapi
-authmethod <password\> -role <name_of_role_in_step_1\> -vserver
<svm_name\> -comment "user_description"
```

3. Associare il ruolo all'utente:

```
security login modify username <user_name\> -vserver <svm_name\> -role
<role_name\> -application ontapi -application console -authmethod
<password\>
```

Utilizzo di System Manager

In Gestione sistema di ONTAP, eseguire le seguenti operazioni:

1. **Crea un ruolo personalizzato**:
   a. Per creare un ruolo personalizzato a livello di cluster, selezionare **Cluster > Impostazioni**.

      (Oppure) per creare un ruolo personalizzato a livello di SVM, selezionare **Storage > Storage VM `required SVM` > > Impostazioni > utenti e ruoli**.

   b. Selezionare l'icona a freccia (→) accanto a **utenti e ruoli**.
   c. Selezionare **+Aggiungi** in **ruoli**.
   d. Definire le regole per il ruolo e fare clic su **Salva**.
2. **Associare il ruolo all'utente Trident**: + eseguire i seguenti passaggi nella pagina **utenti e ruoli**:
   a. Selezionare icona Aggiungi **+** in **utenti**.
   b. Selezionare il nome utente richiesto e scegliere un ruolo nel menu a discesa **ruolo**.
   c. Fare clic su **Save** (Salva).

Per ulteriori informazioni, fare riferimento alle pagine seguenti:

- "Ruoli personalizzati per l'amministrazione di ONTAP" o. "Definire ruoli personalizzati"
- "Lavorare con ruoli e utenti"

Gestire le policy di esportazione NFS

Trident utilizza le policy di esportazione NFS per controllare l'accesso ai volumi forniti.

Trident fornisce due opzioni quando si utilizzano i criteri di esportazione:

- Trident è in grado di gestire in modo dinamico il criterio di esportazione; in questa modalità operativa, l'amministratore dello storage specifica un elenco di blocchi CIDR che rappresentano indirizzi IP consentiti.

Trident aggiunge automaticamente al criterio di esportazione gli indirizzi IP dei nodi applicabili che rientrano in questi intervalli al momento della pubblicazione. In alternativa, quando non vengono specificate CIDR, tutti gli IP unicast con ambito globale trovati nel nodo in cui il volume pubblicato viene aggiunto al criterio di esportazione.

- Gli amministratori dello storage possono creare una policy di esportazione e aggiungere regole manualmente. Trident utilizza il criterio di esportazione predefinito, a meno che non venga specificato un nome di criterio di esportazione diverso nella configurazione.

Gestione dinamica delle policy di esportazione

Trident consente di gestire in modo dinamico le policy di esportazione per i backend ONTAP. In questo modo, l'amministratore dello storage può specificare uno spazio di indirizzi consentito per gli IP dei nodi di lavoro, invece di definire manualmente regole esplicite. Semplifica notevolmente la gestione delle policy di esportazione; le modifiche alle policy di esportazione non richiedono più l'intervento manuale sul cluster di storage. Inoltre, ciò consente di limitare l'accesso al cluster di storage solo ai nodi di lavoro che montano volumi e hanno IP nell'intervallo specificato, supportando una gestione dettagliata e automatizzata.

> Non utilizzare NAT (Network Address Translation) quando si utilizzano criteri di esportazione dinamici. Con NAT, il controller di archiviazione rileva l'indirizzo NAT di frontend e non l'indirizzo host IP effettivo, pertanto l'accesso viene negato quando non viene trovata alcuna corrispondenza nelle regole di esportazione.

Esempio

È necessario utilizzare due opzioni di configurazione. Ecco un esempio di definizione di backend:

```
---
version: 1
storageDriverName: ontap-nas-economy
backendName: ontap_nas_auto_export
managementLIF: 192.168.0.135
svm: svm1
username: vsadmin
password: password
autoExportCIDRs:
  - 192.168.0.0/24
autoExportPolicy: true
```

> Quando si utilizza questa funzione, è necessario assicurarsi che la giunzione root di SVM disponga di un criterio di esportazione creato in precedenza con una regola di esportazione che consenta il blocco CIDR del nodo (ad esempio il criterio di esportazione predefinito). Segui sempre le Best practice consigliate da NetApp per dedicare una SVM a Trident.

Ecco una spiegazione del funzionamento di questa funzione utilizzando l'esempio precedente:

- `autoExportPolicy` è impostato su `true`. In questo modo, Trident crea una policy di esportazione per ogni volume sottoposto a provisioning con questo backend per la `svm1` SVM e gestisce l'aggiunta e l'eliminazione di regole utilizzando `autoexportCIDRs` i blocchi di indirizzi. Fino al collegamento di un volume a un nodo, il volume utilizza un criterio di esportazione vuoto senza regole per impedire l'accesso

indesiderato a tale volume. Quando un volume viene pubblicato in un nodo, Trident crea una policy di esportazione con lo stesso nome del qtree sottostante contenente l'IP del nodo all'interno del blocco CIDR specificato. Questi IP verranno aggiunti anche al criterio di esportazione utilizzato dal FlexVol volume padre

  - Ad esempio:
    - Backend UUUID 403b5326-8482-40dB-96d0-d83fb3f4daec
    - `autoExportPolicy` impostare su `true`
    - prefisso di memorizzazione `trident`
    - UUUID PVC a79bcf5f-7b6d-4a40-9876-e2551f159c1c
    - Il qtree denominato Trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c crea una policy di esportazione per il FlexVol Named , una policy di esportazione per il qtree Named e `trident-403b5326-8482-40db96d0-d83fb3f4daec`una policy di esportazione vuota `trident_empty` denominata `trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c` nella SVM. Le regole per la policy di esportazione di FlexVol saranno un superset di regole contenute nelle policy di esportazione dei qtree. Il criterio di esportazione vuoto verrà riutilizzato da tutti i volumi non collegati.

- `autoExportCIDRs` contiene un elenco di blocchi di indirizzi. Questo campo è opzionale e per impostazione predefinita è ["0.0.0.0/0", "::/0"]. Se non definito, Trident aggiunge tutti gli indirizzi unicast con ambito globale trovati nei nodi di lavoro con pubblicazioni.

In questo esempio, `192.168.0.0/24` viene fornito lo spazio degli indirizzi. Questo indica che gli IP dei nodi Kubernetes che rientrano in questo intervallo di indirizzi con pubblicazioni verranno aggiunti alla policy di esportazione creata da Trident. Quando Trident registra un nodo su cui viene eseguito, recupera gli indirizzi IP del nodo e li controlla in base ai blocchi di indirizzi forniti in. al momento della pubblicazione, dopo aver filtrato gli indirizzi `autoExportCIDRs` IP, Trident crea le regole dei criteri di esportazione per gli indirizzi IP del client per il nodo in cui viene pubblicato.

È possibile eseguire l'aggiornamento `autoExportPolicy` e. `autoExportCIDRs` per i backend dopo la creazione. È possibile aggiungere nuovi CIDR a un backend gestito automaticamente o eliminare i CIDR esistenti. Prestare attenzione quando si eliminano i CIDR per assicurarsi che le connessioni esistenti non vengano interrotte. È anche possibile scegliere di disattivare `autoExportPolicy` per un backend e tornare a una policy di esportazione creata manualmente. Questa operazione richiede l'impostazione di `exportPolicy` nella configurazione del backend.

Dopo che Trident crea o aggiorna un backend, è possibile controllare il backend utilizzando `tridentctl` o il CRD corrispondente `tridentbackend`:

```
./tridentctl get backends ontap_nas_auto_export -n trident -o yaml
items:
- backendUUID: 403b5326-8482-40db-96d0-d83fb3f4daec
  config:
    aggregate: ""
    autoExportCIDRs:
    - 192.168.0.0/24
    autoExportPolicy: true
    backendName: ontap_nas_auto_export
    chapInitiatorSecret: ""
    chapTargetInitiatorSecret: ""
    chapTargetUsername: ""
    chapUsername: ""
    dataLIF: 192.168.0.135
    debug: false
    debugTraceFlags: null
    defaults:
      encryption: "false"
      exportPolicy: <automatic>
      fileSystemType: ext4
```

Quando viene rimosso un nodo, Trident controlla tutte le policy di esportazione per rimuovere le regole di accesso corrispondenti al nodo. Rimuovendo questo IP nodo dalle policy di esportazione dei backend gestiti, Trident impedisce i montaggi non autorizzati, a meno che questo IP non venga riutilizzato da un nuovo nodo nel cluster.

Per i backend esistenti in precedenza, l'aggiornamento del backend con `tridentctl update backend` assicura che Trident gestisca automaticamente i criteri di esportazione. In questo modo, vengono create due nuove policy di esportazione denominate in base all'UUID e al nome del qtree del backend, quando necessario. I volumi presenti sul backend utilizzeranno i criteri di esportazione appena creati dopo essere stati smontati e montati nuovamente.

> L'eliminazione di un backend con policy di esportazione gestite automaticamente elimina la policy di esportazione creata dinamicamente. Se il backend viene ricreato, viene trattato come un nuovo backend e si otterrà la creazione di una nuova policy di esportazione.

Se l'indirizzo IP di un nodo attivo viene aggiornato, è necessario riavviare il pod Trident sul nodo. Trident aggiornerà quindi il criterio di esportazione per i backend che gestisce in modo da riflettere questa modifica dell'IP.

Preparatevi al provisioning dei volumi SMB

Con un po' di preparazione aggiuntiva, puoi eseguire il provisioning dei volumi SMB utilizzando `ontap-nas` driver.

> Devi configurare i protocolli NFS e SMB/CIFS nella SVM per creare un `ontap-nas-economy` volume SMB per i cluster on-premise ONTAP. La mancata configurazione di uno di questi protocolli causerà un errore nella creazione del volume SMB.

`autoExportPolicy` Non è supportato per i volumi SMB.

Prima di iniziare

Prima di eseguire il provisioning di volumi SMB, è necessario disporre di quanto segue.

- Un cluster Kubernetes con un nodo controller Linux e almeno un nodo di lavoro Windows che esegue Windows Server 2022. Trident supporta volumi SMB montati su pod in esecuzione solo sui nodi Windows.
- Almeno un segreto Trident contenente le credenziali di Active Directory. Per generare segreto `smbcreds`:

```
kubectl create secret generic smbcreds --from-literal username=user
--from-literal password='password'
```

- Proxy CSI configurato come servizio Windows. Per configurare un `csi-proxy`, fare riferimento a. "GitHub: Proxy CSI" oppure "GitHub: Proxy CSI per Windows" Per i nodi Kubernetes in esecuzione su Windows.

Fasi

1. Per ONTAP on-premise, puoi facoltativamente creare una condivisione SMB oppure Trident può crearne una.

   Le condivisioni SMB sono richieste per Amazon FSX per ONTAP.

   È possibile creare le condivisioni amministrative SMB in due modi utilizzando "Console di gestione Microsoft" Snap-in cartelle condivise o utilizzo dell'interfaccia CLI di ONTAP. Per creare le condivisioni SMB utilizzando la CLI ONTAP:

   a. Se necessario, creare la struttura del percorso di directory per la condivisione.

      Il `vserver cifs share create` il comando controlla il percorso specificato nell'opzione -path durante la creazione della condivisione. Se il percorso specificato non esiste, il comando non riesce.

   b. Creare una condivisione SMB associata alla SVM specificata:

      ```
      vserver cifs share create -vserver vserver_name -share-name
      share_name -path path [-share-properties share_properties,...]
      [other_attributes] [-comment text]
      ```

   c. Verificare che la condivisione sia stata creata:

      ```
      vserver cifs share show -share-name share_name
      ```

      Fare riferimento a. "Creare una condivisione SMB" per informazioni dettagliate.

2. Quando si crea il backend, è necessario configurare quanto segue per specificare i volumi SMB. Per tutte le opzioni di configurazione backend FSX per ONTAP, fare riferimento a. "FSX per le opzioni di configurazione e gli esempi di ONTAP".

Parametro	Descrizione	Esempio
`smbShare`	È possibile specificare uno dei seguenti elementi: Il nome di una condivisione SMB creata utilizzando la console di gestione Microsoft o l'interfaccia CLI di ONTAP; un nome che consente a Trident di creare la condivisione SMB; oppure è possibile lasciare vuoto il parametro per impedire l'accesso condiviso ai volumi. Questo parametro è facoltativo per ONTAP on-premise. Questo parametro è obbligatorio per i backend Amazon FSX per ONTAP e non può essere vuoto.	`smb-share`
`nasType`	**Deve essere impostato su `smb`.** se null, il valore predefinito è `nfs`.	`smb`
`securityStyle`	Stile di sicurezza per nuovi volumi. **Deve essere impostato su `ntfs` oppure `mixed` Per volumi SMB.**	`ntfs` oppure `mixed` Per volumi SMB
`unixPermissions`	Per i nuovi volumi. **Deve essere lasciato vuoto per i volumi SMB.**	""

Abilita SMB sicuro

A partire dalla versione 25.06, NetApp Trident supporta il provisioning sicuro dei volumi SMB creati utilizzando `ontap-nas` E `ontap-nas-economy` backend. Quando l'SMB sicuro è abilitato, è possibile fornire un accesso controllato alle condivisioni SMB per utenti e gruppi di utenti di Active Directory (AD) utilizzando gli elenchi di controllo di accesso (ACL).

Punti da ricordare

- Importazione `ontap-nas-economy` volumi non è supportato.
- Sono supportati solo i cloni di sola lettura per `ontap-nas-economy` volumi.
- Se Secure SMB è abilitato, Trident ignorerà la condivisione SMB menzionata nel backend.
- L'aggiornamento dell'annotazione PVC, dell'annotazione della classe di archiviazione e del campo backend non aggiorna l'ACL della condivisione SMB.
- L'ACL di condivisione SMB specificato nell'annotazione del PVC clone avrà la precedenza su quelli presenti nel PVC di origine.
- Assicurati di fornire utenti AD validi quando attivi SMB sicuro. Gli utenti non validi non verranno aggiunti all'ACL.
- Se si forniscono allo stesso utente AD nel backend, nella classe di archiviazione e nel PVC autorizzazioni diverse, la priorità delle autorizzazioni sarà: PVC, classe di archiviazione e quindi backend.
- SMB sicuro è supportato per `ontap-nas` importazioni di volumi gestiti e non applicabile alle importazioni di volumi non gestiti.

Fasi

1. Specificare adAdminUser in TridentBackendConfig come mostrato nel seguente esempio:

```yaml
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.193.176.x
  svm: svm0
  useREST: true
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

2. Aggiungere l'annotazione nella classe di archiviazione.

   Aggiungere il `trident.netapp.io/smbShareAdUser` Annotazione alla classe di archiviazione per abilitare SMB sicuro senza errori. Il valore utente specificato per l'annotazione `trident.netapp.io/smbShareAdUser` dovrebbe essere uguale al nome utente specificato in `smbcreds` segreto. è `full_control` .

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

1. Creare un PVC.

   L'esempio seguente crea un PVC:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/snapshotDirectory: "true"
    trident.netapp.io/smbShareAccessControl: |
      read:
        - tridentADtest
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

Opzioni ed esempi di configurazione del NAS ONTAP

Scopri come creare e utilizzare i driver NAS ONTAP con l'installazione Trident. In questa sezione vengono forniti esempi di configurazione backend e dettagli per la mappatura dei backend a StorageClasses.

Opzioni di configurazione back-end

Per le opzioni di configurazione del backend, consultare la tabella seguente:

Parametro	Descrizione	Predefinito
`version`		Sempre 1
`storageDriverName`	Nome del driver di storage	`ontap-nas`, , `ontap-nas-economy` o `ontap-nas-flexgroup`
`backendName`	Nome personalizzato o backend dello storage	Nome del driver + "_" + dataLIF
`managementLIF`	Indirizzo IP di un cluster o LIF di gestione SVM È possibile specificare Un nome di dominio completo (FQDN). Può essere impostato in modo da utilizzare gli indirizzi IPv6 se Trident è stato installato utilizzando il flag IPv6. Gli indirizzi IPv6 devono essere definiti tra parentesi quadre, ad esempio `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . Per lo switchover di MetroCluster senza problemi, vedere la Esempio MetroCluster.	"10,0.0,1", "[2001:1234:abcd::fefe]"

Parametro	Descrizione	Predefinito
`dataLIF`	Indirizzo IP del protocollo LIF. NetApp consiglia di specificare `dataLIF`. Se non viene fornita, Trident recupera le LIF dati dalla SVM. È possibile specificare un nome di dominio completo (FQDN) da utilizzare per le operazioni di montaggio NFS, consentendo di creare un DNS round-robin per bilanciare il carico su più LIF dati. Può essere modificato dopo l'impostazione iniziale. Fare riferimento alla . Può essere impostato in modo da utilizzare gli indirizzi IPv6 se Trident è stato installato utilizzando il flag IPv6. Gli indirizzi IPv6 devono essere definiti tra parentesi quadre, ad esempio `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . **Omettere per MetroCluster.** Consultare la Esempio MetroCluster.	Indirizzo specificato o derivato da SVM, se non specificato (non consigliato)
`svm`	Macchina virtuale per lo storage da utilizzare **Ometti per MetroCluster.** vedere la Esempio MetroCluster.	Derivato se un SVM `managementLIF` è specificato
`autoExportPolicy`	Abilita la creazione e l'aggiornamento automatici dei criteri di esportazione [booleano]. Utilizzando le `autoExportPolicy` opzioni e `autoExportCIDRs`, Trident può gestire automaticamente i criteri di esportazione.	falso
`autoExportCIDRs`	Elenco di CIDR per filtrare gli IP dei nodi di Kubernetes rispetto a quando `autoExportPolicy` è attivato. Utilizzando le `autoExportPolicy` opzioni e `autoExportCIDRs`, Trident può gestire automaticamente i criteri di esportazione.	["0,0.0,0/0", ":/0"]»
`labels`	Set di etichette arbitrarie formattate con JSON da applicare sui volumi	""
`clientCertificate`	Valore del certificato client codificato con base64. Utilizzato per l'autenticazione basata su certificato	""
`clientPrivateKey`	Valore codificato in base64 della chiave privata del client. Utilizzato per l'autenticazione basata su certificato	""
`trustedCACertificate`	Valore codificato in base64 del certificato CA attendibile. Opzionale. Utilizzato per l'autenticazione basata su certificato	""
`username`	Nome utente per la connessione al cluster/SVM. Utilizzato per l'autenticazione basata sulle credenziali	
`password`	Password per la connessione al cluster/SVM. Utilizzato per l'autenticazione basata sulle credenziali	

Parametro	Descrizione	Predefinito
`storagePrefix`	Prefisso utilizzato per il provisioning di nuovi volumi nella SVM. Non può essere aggiornato dopo l'impostazione Quando si utilizza ONTAP-nas-Economy e un prefisso di archiviazione di 24 o più caratteri, i qtree non avranno il prefisso di archiviazione incorporato, anche se sarà nel nome del volume.	"trident"
`aggregate`	Aggregato per il provisioning (facoltativo; se impostato, deve essere assegnato alla SVM). Per il `ontap-nas-flexgroup` driver, questa opzione viene ignorata. Se non viene assegnato, è possibile utilizzare qualsiasi aggregato disponibile per il provisioning di un volume FlexGroup. Una volta aggiornato l'aggregato in SVM, viene aggiornato automaticamente in Trident eseguendo un polling della SVM senza riavviare il controller Trident. Dopo aver configurato un aggregato specifico in Trident per il provisioning dei volumi, in caso di ridenominazione o spostamento dell'aggregato dalla SVM, il back-end passa allo stato di errore in Trident durante il polling dell'aggregato della SVM. È necessario modificare l'aggregato in uno presente nella SVM o rimuoverlo del tutto per riportare online il back-end.	""
`limitAggregateUsage`	Il provisioning non riesce se l'utilizzo è superiore a questa percentuale. **Non si applica ad Amazon FSx per ONTAP**.	"" (non applicato per impostazione predefinita)

Parametro	Descrizione	Predefinito
FlexgroupAggregateList	Elenco di aggregati per il provisioning (facoltativo; se impostato, deve essere assegnato alla SVM). Tutti gli aggregati assegnati alla SVM vengono utilizzati per il provisioning di un volume FlexGroup. Supportato per il driver di archiviazione **ONTAP-nas-FlexGroup**. Una volta aggiornato l'elenco degli aggregati all'interno della SVM, l'elenco viene aggiornato automaticamente in Trident eseguendo un polling della SVM senza dover riavviare il controller Trident. Dopo aver configurato un elenco di aggregati specifici in Trident per il provisioning dei volumi, se l'elenco degli aggregati viene rinominato o spostato fuori dalla SVM, il back-end passa allo stato di errore in Trident durante il polling dell'aggregato della SVM. È necessario modificare l'elenco degli aggregati in uno presente nella SVM o rimuoverlo del tutto per riportare online il back-end.	""
`limitVolumeSize`	Fallire il provisioning se la dimensione del volume richiesta è superiore a questo valore. Limita anche la dimensione massima dei volumi gestiti per i qtree e l' `qtreesPerFlexvol`opzione consente di personalizzare il numero massimo di qtree per FlexVol volume	"" (non applicato per impostazione predefinita)
`debugTraceFlags`	Flag di debug da utilizzare per la risoluzione dei problemi. Esempio, {"api":false, "method":true} Non utilizzare `debugTraceFlags` a meno che non si stia eseguendo la risoluzione dei problemi e non si richieda un dump dettagliato del log.	nullo
`nasType`	Configurare la creazione di volumi NFS o SMB. Le opzioni sono `nfs`, `smb` o nullo. L'impostazione su Null consente di impostare i volumi NFS come predefiniti.	`nfs`
`nfsMountOptions`	Elenco separato da virgole delle opzioni di montaggio NFS. Le opzioni di montaggio per volumi persistenti di Kubernetes vengono normalmente specificate in classi di storage, ma se non sono specificate opzioni di montaggio in una classe di storage, Trident tornerà all'utilizzo delle opzioni di montaggio specificate nel file di configurazione del backend di storage. Se non sono specificate opzioni di montaggio nella classe di storage o nel file di configurazione, Trident non imposterà alcuna opzione di montaggio su un volume persistente associato.	""

Parametro	Descrizione	Predefinito
`qtreesPerFlexvol`	Qtree massimi per FlexVol, devono essere compresi nell'intervallo [50, 300]	"200"
`smbShare`	È possibile specificare uno dei seguenti elementi: Il nome di una condivisione SMB creata utilizzando la console di gestione Microsoft o l'interfaccia CLI di ONTAP; un nome che consente a Trident di creare la condivisione SMB; oppure è possibile lasciare vuoto il parametro per impedire l'accesso condiviso ai volumi. Questo parametro è facoltativo per ONTAP on-premise. Questo parametro è obbligatorio per i backend Amazon FSX per ONTAP e non può essere vuoto.	`smb-share`
`useREST`	Parametro booleano per l'utilizzo delle API REST di ONTAP. `useREST` Quando è impostato su `true`, Trident utilizza le API REST ONTAP per comunicare con il backend; quando è impostato su `false`, Trident utilizza le chiamate ONTAPI (ZAPI) per comunicare con il backend. Questa funzione richiede ONTAP 9.11.1 e versioni successive. Inoltre, il ruolo di accesso ONTAP utilizzato deve avere accesso all' `ontapi` applicazione. Ciò è soddisfatto dai ruoli predefiniti `vsadmin` e `cluster-admin` . A partire da Trident 24,06 e ONTAP 9.15.1 o versioni successive, `useREST` è impostato su `true` per impostazione predefinita; passare `useREST` a `false` per utilizzare le chiamate ONTAPI (ZAPI).	`true` Per ONTAP 9.15.1 o versioni successive, altrimenti `false`.
`limitVolumePoolSize`	Dimensioni FlexVol massime richiedibili quando si utilizzano Qtree nel backend ONTAP-nas-Economy.	"" (non applicato per impostazione predefinita)
`denyNewVolumePools`	Limita `ontap-nas-economy` i backend dalla creazione di nuovi volumi FlexVol per contenere i propri Qtree. Per il provisioning di nuovi PVS vengono utilizzati solo i FlexVol preesistenti.	
`adAdminUser`	Utente o gruppo di utenti amministratore di Active Directory con accesso completo alle condivisioni SMB. Utilizzare questo parametro per fornire diritti di amministratore alla condivisione SMB con controllo completo.	

Opzioni di configurazione back-end per il provisioning dei volumi

È possibile controllare il provisioning predefinito utilizzando queste opzioni in `defaults` della configurazione. Per un esempio, vedere gli esempi di configurazione riportati di seguito.

Parametro	Descrizione	Predefinito
`spaceAllocation`	Allocazione dello spazio per Qtree	"vero"

Parametro	Descrizione	Predefinito
`spaceReserve`	Modalità di prenotazione dello spazio; "nessuno" (sottile) o "volume" (spesso)	"nessuno"
`snapshotPolicy`	Policy di Snapshot da utilizzare	"nessuno"
`qosPolicy`	Gruppo di criteri QoS da assegnare per i volumi creati. Scegliere tra qosPolicy o adaptiveQosPolicy per pool di storage/backend	""
`adaptiveQosPolicy`	Gruppo di criteri QoS adattivi da assegnare per i volumi creati. Scegliere tra qosPolicy o adaptiveQosPolicy per pool di storage/backend. Non supportato da ontap-nas-Economy.	""
`snapshotReserve`	Percentuale di volume riservato agli snapshot	"0" se `snapshotPolicy` è "nessuno", altrimenti ""
`splitOnClone`	Separare un clone dal suo padre al momento della creazione	"falso"
`encryption`	Abilitare la crittografia del volume NetApp (NVE) sul nuovo volume; il valore predefinito è `false`. NVE deve essere concesso in licenza e abilitato sul cluster per utilizzare questa opzione. Se NAE è abilitato sul backend, qualsiasi volume sottoposto a provisioning in Trident sarà abilitato NAE. Per ulteriori informazioni, fare riferimento a: "Come funziona Trident con NVE e NAE".	"falso"
`tieringPolicy`	Criterio di tiering da utilizzare "nessuno"	
`unixPermissions`	Per i nuovi volumi	"777" per i volumi NFS; vuoto (non applicabile) per i volumi SMB
`snapshotDir`	Controlla l'accesso a. `.snapshot` directory	"True" per NFSv4 "false" per NFSv3
`exportPolicy`	Policy di esportazione da utilizzare	"predefinito"
`securityStyle`	Stile di sicurezza per nuovi volumi. Supporto di NFS `mixed` e. `unix` stili di sicurezza. Supporto SMB `mixed` e. `ntfs` stili di sicurezza.	Il valore predefinito di NFS è `unix`. Il valore predefinito di SMB è `ntfs`.
`nameTemplate`	Modello per creare nomi di volume personalizzati.	""

> L'utilizzo di gruppi di criteri QoS con Trident richiede ONTAP 9.8 o versioni successive. È necessario utilizzare un gruppo di criteri QoS non condiviso e assicurarsi che il gruppo di criteri venga applicato singolarmente a ciascun componente. Un gruppo di policy QoS condiviso impone un limite massimo per il throughput totale di tutti i carichi di lavoro.

Esempi di provisioning di volumi

Ecco un esempio con i valori predefiniti definiti:

```yaml
---
version: 1
storageDriverName: ontap-nas
backendName: customBackendName
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
labels:
  k8scluster: dev1
  backend: dev1-nasbackend
svm: trident_svm
username: cluster-admin
password: <password>
limitAggregateUsage: 80%
limitVolumeSize: 50Gi
nfsMountOptions: nfsvers=4
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: premium
  exportPolicy: myk8scluster
  snapshotPolicy: default
  snapshotReserve: "10"
```

Per `ontap-nas` E `ontap-nas-flexgroups` Trident ora utilizza un nuovo calcolo per garantire che il FlexVol sia dimensionato correttamente con la percentuale di snapshotReserve e PVC. Quando l'utente richiede un PVC, Trident crea il FlexVol originale con più spazio utilizzando il nuovo calcolo. Questo calcolo garantisce che l'utente riceva lo spazio scrivibile richiesto nel PVC e non uno spazio inferiore a quello richiesto. Prima della versione 21.07, quando l'utente richiedeva un PVC (ad esempio, 5 GiB), con snapshotReserve al 50%, otteneva solo 2,5 GiB di spazio scrivibile. Questo perché ciò che l'utente richiedeva era l'intero volume e `snapshotReserve` è una percentuale di quello. Con Trident 21.07, ciò che l'utente richiede è lo spazio scrivibile e Trident definisce lo `snapshotReserve` numero come percentuale del volume totale. Questo non si applica a `ontap-nas-economy`. Per vedere come funziona, vedere l'esempio seguente

Il calcolo è il seguente:

```
Total volume size = (PVC requested size) / (1 - (snapshotReserve
percentage) / 100)
```

Per snapshotReserve = 50% e richiesta PVC = 5 GiB, la dimensione totale del volume è 5/.5 = 10 GiB e la dimensione disponibile è 5 GiB, che è ciò che l'utente ha richiesto nella richiesta PVC . `volume show` il comando dovrebbe mostrare risultati simili a questo esempio:

```
Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- -----
                 _pvc_89f1c156_3801_4de4_9f9d_034d54c395f4
                                      online     RW         10GB     5.00GB    0%
                 _pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba
                                      online     RW          1GB    511.8MB    0%
2 entries were displayed.
```

I backend esistenti delle installazioni precedenti eseguiranno il provisioning dei volumi come spiegato sopra durante l'aggiornamento Trident. Per i volumi creati prima dell'aggiornamento, è necessario ridimensionarli affinché la modifica venga visualizzata. Ad esempio, un PVC da 2 GiB con `snapshotReserve=50` In precedenza, il risultato era un volume che forniva 1 GiB di spazio scrivibile. Ridimensionando il volume a 3 GiB, ad esempio, l'applicazione ottiene 3 GiB di spazio scrivibile su un volume da 6 GiB.

Esempi di configurazione minimi

Gli esempi seguenti mostrano le configurazioni di base che lasciano la maggior parte dei parametri predefiniti. Questo è il modo più semplice per definire un backend.

> Se si utilizza Amazon FSX su NetApp ONTAP con Trident, si consiglia di specificare i nomi DNS per le LIF anziché gli indirizzi IP.

Esempio di economia NAS ONTAP

```yaml
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

Esempio di FlexGroup NAS ONTAP

```yaml
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

Esempio MetroCluster

È possibile configurare il backend per evitare di dover aggiornare manualmente la definizione del backend dopo lo switchover e lo switchback durante "Replica e recovery di SVM".

Per uno switchover e uno switchback perfetto, specifica la SVM utilizzando `managementLIF` e omettere `dataLIF` e. `svm` parametri. Ad esempio:

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 192.168.1.66
username: vsadmin
password: password
```

Esempio di volumi SMB

```
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
nasType: smb
securityStyle: ntfs
unixPermissions: ""
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

Esempio di autenticazione basata su certificato

Si tratta di un esempio minimo di configurazione di back-end. `clientCertificate`, `clientPrivateKey`, e. `trustedCACertificate` (Facoltativo, se si utilizza una CA attendibile) sono inseriti in `backend.json` E prendere rispettivamente i valori codificati base64 del certificato client, della chiave privata e del certificato CA attendibile.

```
---
version: 1
backendName: DefaultNASBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.15
svm: nfs_svm
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

Esempio di policy di esportazione automatica

In questo esempio viene illustrato come impostare Trident in modo che utilizzi i criteri di esportazione dinamici per creare e gestire automaticamente i criteri di esportazione. Funziona allo stesso modo per i `ontap-nas-economy` driver e `ontap-nas-flexgroup` .

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
labels:
  k8scluster: test-cluster-east-1a
  backend: test1-nasbackend
autoExportPolicy: true
autoExportCIDRs:
- 10.0.0.0/24
username: admin
password: password
nfsMountOptions: nfsvers=4
```

Esempio di indirizzi IPv6

Questo esempio mostra `managementLIF` Utilizzando un indirizzo IPv6.

```
---
version: 1
storageDriverName: ontap-nas
backendName: nas_ipv6_backend
managementLIF: "[5c5d:5edf:8f:7657:bef8:109b:1b41:d491]"
labels:
  k8scluster: test-cluster-east-1a
  backend: test1-ontap-ipv6
svm: nas_ipv6_svm
username: vsadmin
password: password
```

Esempio di Amazon FSX per ONTAP con volumi SMB

Il `smbShare` Il parametro è obbligatorio per FSX per ONTAP che utilizza volumi SMB.

```
---
version: 1
backendName: SMBBackend
storageDriverName: ontap-nas
managementLIF: example.mgmt.fqdn.aws.com
nasType: smb
dataLIF: 10.0.0.15
svm: nfs_svm
smbShare: smb-share
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

Esempio di configurazione backend con nameTemplate

```
---
version: 1
storageDriverName: ontap-nas
backendName: ontap-nas-backend
managementLIF: <ip address>
svm: svm0
username: <admin>
password: <password>
defaults:
  nameTemplate:
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.vo\
    lume.RequestName}}"
labels:
  cluster: ClusterA
  PVC: "{{.volume.Namespace}}_{{.volume.RequestName}}"
```

Esempi di backend con pool virtuali

Nei file di definizione back-end di esempio illustrati di seguito, vengono impostati valori predefiniti specifici per tutti i pool di storage, ad esempio `spaceReserve` a nessuno, `spaceAllocation` a false, e. `encryption` a falso. I pool virtuali sono definiti nella sezione storage.

Trident imposta le etichette di provisioning nel campo "commenti". I commenti sono impostati su FlexVol for `ontap-nas` o FlexGroup for `ontap-nas-flexgroup`. Trident copia tutte le etichette presenti su un pool virtuale nel volume di storage al momento del provisioning. Per comodità, gli amministratori dello storage possono definire le etichette per ogni pool virtuale e raggruppare i volumi per etichetta.

In questi esempi, alcuni dei pool di storage sono impostati in modo personalizzato `spaceReserve`, `spaceAllocation`, e. `encryption` e alcuni pool sovrascrivono i valori predefiniti.

Esempio di NAS ONTAP

```yaml
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
svm: svm_nfs
username: admin
password: <password>
nfsMountOptions: nfsvers=4
defaults:
  spaceReserve: none
  encryption: "false"
  qosPolicy: standard
labels:
  store: nas_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
  - labels:
      app: msoffice
      cost: "100"
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: "true"
      unixPermissions: "0755"
      adaptiveQosPolicy: adaptive-premium
  - labels:
      app: slack
      cost: "75"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      department: legal
      creditpoints: "5000"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      app: wordpress
```

```
      cost: "50"
    zone: us_east_1c
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0775"
  - labels:
      app: mysqldb
      cost: "25"
    zone: us_east_1d
    defaults:
      spaceReserve: volume
      encryption: "false"
      unixPermissions: "0775"
```

Esempio di NAS FlexGroup ONTAP

```yaml
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: <password>
defaults:
  spaceReserve: none
  encryption: "false"
labels:
  store: flexgroup_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
  - labels:
      protection: gold
      creditpoints: "50000"
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      protection: gold
      creditpoints: "30000"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      protection: silver
      creditpoints: "20000"
    zone: us_east_1c
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0775"
  - labels:
      protection: bronze
      creditpoints: "10000"
    zone: us_east_1d
    defaults:
```

```
        spaceReserve: volume
        encryption: "false"
        unixPermissions: "0775"
```

Esempio di economia NAS ONTAP

```yaml
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: <password>
defaults:
  spaceReserve: none
  encryption: "false"
labels:
  store: nas_economy_store
region: us_east_1
storage:
  - labels:
      department: finance
      creditpoints: "6000"
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      protection: bronze
      creditpoints: "5000"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      department: engineering
      creditpoints: "3000"
    zone: us_east_1c
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0775"
  - labels:
      department: humanresource
      creditpoints: "2000"
    zone: us_east_1d
    defaults:
      spaceReserve: volume
```

```
      encryption: "false"
      unixPermissions: "0775"
```

Mappare i backend in StorageClasses

Le seguenti definizioni di StorageClass fanno riferimento a. Esempi di backend con pool virtuali. Utilizzando il `parameters.selector` Ciascun StorageClass richiama i pool virtuali che possono essere utilizzati per ospitare un volume. Gli aspetti del volume saranno definiti nel pool virtuale scelto.

- Il `protection-gold` StorageClass eseguirà il mapping al primo e al secondo pool virtuale in `ontap-nas-flexgroup` back-end. Questi sono gli unici pool che offrono una protezione di livello gold.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=gold"
  fsType: "ext4"
```

- Il `protection-not-gold` StorageClass eseguirà il mapping al terzo e al quarto pool virtuale in `ontap-nas-flexgroup` back-end. Questi sono gli unici pool che offrono un livello di protezione diverso dall'oro.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
```

- Il `app-mysqldb` StorageClass eseguirà il mapping al quarto pool virtuale in `ontap-nas` back-end. Questo è l'unico pool che offre la configurazione del pool di storage per l'applicazione di tipo mysqldb.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
```

- Til `protection-silver-creditpoints-20k` StorageClass eseguirà il mapping al terzo pool virtuale in `ontap-nas-flexgroup` back-end. Questo è l'unico pool che offre una protezione di livello Silver e 20000 punti di credito.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
```

- Il `creditpoints-5k` StorageClass eseguirà il mapping al terzo pool virtuale in `ontap-nas` il back-end e il secondo pool virtuale in `ontap-nas-economy` back-end. Queste sono le uniche offerte di pool con 5000 punti di credito.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: csi.trident.netapp.io
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"
```

Trident deciderà quale pool virtuale viene selezionato e garantirà che i requisiti di storage vengano soddisfatti.

Aggiornare `dataLIF` dopo la configurazione iniziale

Puoi modificare la dataLIF dopo la configurazione iniziale eseguendo il seguente comando per fornire il nuovo file JSON di backend con i dati LIF aggiornati.

```
tridentctl update backend <backend-name> -f <path-to-backend-json-file-
with-updated-dataLIF>
```

> Se sono collegati a uno o più pod, è necessario abbassare tutti i pod corrispondenti e quindi riportarli in posizione per rendere effettiva la nuova data LIF.

Esempi di SMB sicuri

Configurazione backend con driver ontap-nas

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Configurazione backend con driver ontap-nas-economy

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas-economy
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Configurazione backend con pool di archiviazione

```yaml
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm0
  useREST: false
  storage:
  - labels:
      app: msoffice
    defaults:
      adAdminUser: tridentADuser
  nasType: smb
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Esempio di classe di archiviazione con driver ontap-nas

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADtest
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

Assicurati di aggiungere `annotations` Per abilitare SMB sicuro. SMB sicuro non funziona senza annotazioni, indipendentemente dalle configurazioni impostate nel Backend o nel PVC.

Esempio di classe di archiviazione con driver ontap-nas-economy

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser3
parameters:
  backendType: ontap-nas-economy
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

Esempio di PVC con un singolo utente AD

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      change:
        - tridentADtest
      read:
        - tridentADuser
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

Esempio di PVC con più utenti AD

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-test-pvc
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      full_control:
        - tridentTestuser
        - tridentuser
        - tridentTestuser1
        - tridentuser1
      change:
        - tridentADuser
        - tridentADuser1
        - tridentADuser4
        - tridentTestuser2
      read:
        - tridentTestuser2
        - tridentTestuser3
        - tridentADuser2
        - tridentADuser3
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

Amazon FSX per NetApp ONTAP

USA Trident con Amazon FSX per NetApp ONTAP

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

Puoi integrare il tuo file system Amazon FSX per NetApp ONTAP con Trident per garantire che i cluster Kubernetes in esecuzione in Amazon Elastic Kubernetes Service (EKS) possano effettuare il provisioning di volumi persistenti di file e blocchi supportati da ONTAP.

Un file system è la risorsa principale di Amazon FSX, simile a un cluster ONTAP on-premise. All'interno di ogni SVM è possibile creare uno o più volumi, ovvero contenitori di dati che memorizzano i file e le cartelle nel file

system. Con Amazon FSX per NetApp ONTAP verrà fornito come file system gestito nel cloud. Il nuovo tipo di file system è denominato **NetApp ONTAP**.

Utilizzando Trident con Amazon FSX per NetApp ONTAP, puoi assicurarti che i cluster Kubernetes in esecuzione in Amazon Elastic Kubernetes Service (EKS) possano effettuare il provisioning di volumi persistenti di file e blocchi supportati da ONTAP.

Requisiti

Oltre a "Requisiti Trident", per integrare FSX for ONTAP con Trident, hai bisogno di:

- Un cluster Amazon EKS esistente o un cluster Kubernetes autogestito con `kubectl` installato.
- Una macchina virtuale di storage e file system Amazon FSX per NetApp ONTAP esistente raggiungibile dai nodi di lavoro del cluster.
- Nodi di lavoro preparati per "NFS o iSCSI".

> Assicurati di seguire la procedura di preparazione del nodo richiesta per Amazon Linux e Ubuntu "Immagini Amazon Machine" (Amis) a seconda del tipo di AMI EKS.

Considerazioni

- Volumi SMB:
  - I volumi SMB sono supportati utilizzando `ontap-nas` solo driver.
  - I volumi SMB non sono supportati con i componenti aggiuntivi Trident EKS.
  - Trident supporta volumi SMB montati su pod in esecuzione solo sui nodi Windows. Per ulteriori informazioni, fare riferimento alla "Preparatevi al provisioning dei volumi SMB" sezione.
- Prima di Trident 24,02, Trident non ha potuto eliminare i volumi creati su file system Amazon FSX con backup automatici abilitati. Per evitare questo problema in Trident 24,02 o versioni successive, specificare `fsxFilesystemID` , AWS , AWS `apiRegion` `apikey` e AWS `secretKey` nel file di configurazione backend per AWS FSX for ONTAP.

> Se si specifica un ruolo IAM in Trident, è possibile omettere esplicitamente i `apiRegion` campi , `apiKey` e `secretKey` in Trident. Per ulteriori informazioni, fare riferimento a "FSX per le opzioni di configurazione e gli esempi di ONTAP".

Utilizzo simultaneo del driver Trident SAN/iSCSI ed EBS-CSI

Se si prevede di utilizzare i driver ontap-san (ad esempio, iSCSI) con AWS (EKS, ROSA, EC2 o qualsiasi altra istanza), la configurazione multipath richiesta sui nodi potrebbe entrare in conflitto con il driver CSI di Amazon Elastic Block Store (EBS). Per garantire che il multipathing funzioni senza interferire con i dischi EBS sullo stesso nodo, è necessario escludere EBS dalla configurazione del multipathing. Questo esempio mostra un `multipath.conf` file che include le impostazioni Trident richieste escludendo i dischi EBS dal multipathing:

```
defaults {
    find_multipaths no
}
blacklist {
    device {
        vendor "NVME"
        product "Amazon Elastic Block Store"
    }
}
```

Autenticazione

Trident offre due modalità di autenticazione.

- Basato su credenziali (consigliato): Memorizza le credenziali in modo sicuro in AWS Secrets Manager. Puoi utilizzare l' `fsxadmin` utente per il tuo file system o quello `vsadmin` configurato per la tua SVM.

> Trident si aspetta di essere eseguito come `vsadmin` utente SVM o come utente con un nome diverso che abbia lo stesso ruolo. Amazon FSX per NetApp ONTAP include un `fsxadmin` utente che sostituisce in modo limitato l'utente del cluster ONTAP `admin`. Si consiglia vivamente di utilizzare `vsadmin` con Trident.

- Basato su certificato: Trident comunica con la SVM sul file system FSX utilizzando un certificato installato nella SVM.

Per ulteriori informazioni sull'attivazione dell'autenticazione, fare riferimento all'autenticazione per il tipo di driver in uso:

- "Autenticazione NAS ONTAP"
- "Autenticazione SAN ONTAP"

Immagini Amazon Machine testate (AMI)

Il cluster EKS supporta vari sistemi operativi, ma AWS ha ottimizzato alcuni Amazon Machine Images (AMI) per container ed EKS. Le seguenti AMI sono state testate con NetApp Trident 25.02.

AMI	NAS	Economia NAS	ISCSI	iSCSI-economy
AL2023_x86_64_STANDARD	Sì	Sì	Sì	Sì
AL2_x86_64	Sì	Sì	Sì*	Sì*
BOTTLEROCKET_x86_64	Sì**	Sì	N/A.	N/A.
AL2023_ARM_64_STANDARD	Sì	Sì	Sì	Sì
AL2_ARM_64	Sì	Sì	Sì*	Sì*

BOTTLEROCKET_ARM_64	Sì**	Sì	N/A.	N/A.

- * Impossibile eliminare il PV senza riavviare il nodo
- ** Non funziona con NFSv3 con Trident versione 25.02.

> Se il vostro AMI desiderato non è elencato qui, non significa che non è supportato; significa semplicemente che non è stato testato. Questo elenco serve da guida per le AMI di cui è noto il funzionamento.

Prove eseguite con:

- Versione EKS: 1.32
- Metodo di installazione: Helm 25.06 e come componente aggiuntivo AWS 25.06
- Per le NAS sono stati testati sia NFSv3 che NFSv4,1.
- Per SAN è stato testato solo iSCSI, non NVMe-of.

Prove eseguite:

- Creare: Classe di archiviazione, pvc, pod
- Eliminazione: Pod, pvc (normale, qtree/lun – economia, NAS con backup AWS)

Trova ulteriori informazioni

- "Documentazione di Amazon FSX per NetApp ONTAP"
- "Post del blog su Amazon FSX per NetApp ONTAP"

Creare un ruolo IAM e un segreto AWS

Puoi configurare i pod Kubernetes in modo che accedano alle risorse AWS autenticandosi come ruolo AWS IAM invece di fornire credenziali AWS esplicite.

> Per eseguire l'autenticazione usando un ruolo AWS IAM, devi disporre di un cluster Kubernetes implementato utilizzando EKS.

Crea un segreto per AWS Secrets Manager

Poiché Trident emetterà API su un vserver FSX per gestire lo storage in modo automatico, saranno necessarie le credenziali per farlo. Il modo sicuro per passare queste credenziali è tramite un segreto di AWS Secrets Manager. Pertanto, se non ne hai già uno, dovrai creare un segreto di AWS Secrets Manager che contenga le credenziali per l'account vsadmin.

Questo esempio crea un segreto di Gestore segreti AWS per memorizzare le credenziali Trident CSI:

```
aws secretsmanager create-secret --name trident-secret --description
"Trident CSI credentials"\
  --secret-string
"{\"username\":\"vsadmin\",\"password\":\"<svmpassword>\"}"
```

Crea criterio IAM

Trident necessita anche delle autorizzazioni AWS per funzionare correttamente. Pertanto, è necessario creare un criterio che fornisca a Trident le autorizzazioni necessarie.

I seguenti esempi creano una policy IAM utilizzando l'interfaccia a riga di comando di AWS:

```
aws iam create-policy --policy-name AmazonFSxNCSIDriverPolicy --policy
-document file://policy.json
  --description "This policy grants access to Trident CSI to FSxN and
Secrets manager"
```

Policy JSON esempio:

```
{
  "Statement": [
    {
      "Action": [
        "fsx:DescribeFileSystems",
        "fsx:DescribeVolumes",
        "fsx:CreateVolume",
        "fsx:RestoreVolumeFromSnapshot",
        "fsx:DescribeStorageVirtualMachines",
        "fsx:UntagResource",
        "fsx:UpdateVolume",
        "fsx:TagResource",
        "fsx:DeleteVolume"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Action": "secretsmanager:GetSecretValue",
      "Effect": "Allow",
      "Resource": "arn:aws:secretsmanager:<aws-region>:<aws-account-
id>:secret:<aws-secret-manager-name>*"
    }
  ],
  "Version": "2012-10-17"
}
```

Crea l'identità del pod o il ruolo IAM per l'associazione dell'account di servizio (IRSA)

È possibile configurare un account di servizio Kubernetes per assumere un ruolo AWS Identity and Access Management (IAM) con EKS Pod Identity o un ruolo IAM per l'associazione dell'account di servizio (IRSA). Tutti i Pod configurati per utilizzare l'account di servizio possono quindi accedere a qualsiasi servizio AWS per

il quale il ruolo dispone delle autorizzazioni di accesso.

Identità del pod

Le associazioni di identità dei pod Amazon EKS offrono la possibilità di gestire le credenziali per le applicazioni, in modo simile a come i profili delle istanze Amazon EC2 forniscono le credenziali alle istanze Amazon EC2.

Installa Pod Identity sul tuo cluster EKS:

Puoi creare l'identità del Pod tramite la console AWS o utilizzando il seguente comando AWS CLI:

```
aws eks create-addon --cluster-name <EKS_CLUSTER_NAME> --addon-name
eks-pod-identity-agent
```

Per maggiori informazioni fare riferimento a "Configurare l'agente di identità del pod Amazon EKS" .

Crea trust-relationship.json:

Crea trust-relationship.json per consentire al Service Principal EKS di assumere questo ruolo per l'identità del Pod. Quindi crea un ruolo con questa policy di attendibilità:

```
aws iam create-role \
  --role-name fsxn-csi-role --assume-role-policy-document file://trust-
relationship.json \
  --description "fsxn csi pod identity role"
```

file trust-relationship.json:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "pods.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

Associare la policy del ruolo al ruolo IAM:

Associa il criterio di ruolo del passaggio precedente al ruolo IAM creato:

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:111122223333:policy/fsxn-csi-policy \
  --role-name fsxn-csi-role
```

Crea un'associazione di identità pod:

Crea un'associazione di identità pod tra il ruolo IAM e l'account del servizio Trident (trident-controller)

```
aws eks create-pod-identity-association \
  --cluster-name <EKS_CLUSTER_NAME> \
  --role-arn arn:aws:iam::111122223333:role/fsxn-csi-role \
  --namespace trident --service-account trident-controller
```

Ruolo IAM per l'associazione dell'account di servizio (IRSA)
Utilizzando l'AWS CLI:

```
aws iam create-role --role-name AmazonEKS_FSxN_CSI_DriverRole \
  --assume-role-policy-document file://trust-relationship.json
```

file trust-relation.json:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::<account_id>:oidc-
provider/<oidc_provider>"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "<oidc_provider>:aud": "sts.amazonaws.com",
          "<oidc_provider>:sub":
"system:serviceaccount:trident:trident-controller"
        }
      }
    }
  ]
}
```

Aggiornare i seguenti valori nel `trust-relationship.json` file:

- **<account_id>** - il tuo ID account AWS
- **<oidc_provider>** - l'OIDC del tuo cluster EKS. È possibile ottenere oidc_provider eseguendo:

```
aws eks describe-cluster --name my-cluster --query
"cluster.identity.oidc.issuer"\
  --output text | sed -e "s/^https:\/\///"
```

Associare il ruolo IAM alla policy IAM:

Una volta creato il ruolo, allegare il criterio (creato nel passaggio precedente) al ruolo utilizzando questo comando:

```
aws iam attach-role-policy --role-name my-role --policy-arn <IAM policy
ARN>
```

Verificare che il provider OICD sia associato:

Verifica che il tuo provider OIDC sia associato al cluster. È possibile verificarlo utilizzando il seguente comando:

```
aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
```

Se l'output è vuoto, utilizzare il seguente comando per associare IAM OIDC al cluster:

```
eksctl utils associate-iam-oidc-provider --cluster $cluster_name
--approve
```

Se si utilizza eksctl, utilizzare il seguente esempio per creare un ruolo IAM per l'account di servizio in EKS:

```
eksctl create iamserviceaccount --name trident-controller --namespace
trident \
  --cluster <my-cluster> --role-name AmazonEKS_FSxN_CSI_DriverRole
--role-only \
  --attach-policy-arn <IAM-Policy ARN> --approve
```

Installare Trident

Trident ottimizza la gestione dello storage di Amazon FSX per NetApp ONTAP in Kubernetes per permettere a sviluppatori e amministratori di concentrarsi

sull’implementazione dell’applicazione.

È possibile installare Trident utilizzando uno dei seguenti metodi:

- Timone
- Componente aggiuntivo EKS

Se si desidera utilizzare la funzionalità snapshot, installare il componente aggiuntivo del controller snapshot CSI. Per ulteriori informazioni, fare riferimento "Attiva la funzionalità snapshot per volumi CSI" a.

Installare Trident tramite helm

Identità del pod

1. Aggiungere il repository Trident Helm:

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

2. Installa Trident utilizzando il seguente esempio:

```
helm install trident-operator netapp-trident/trident-operator
--version 100.2502.1 --namespace trident --create-namespace
```

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

```
helm list -n trident
```

```
NAME                NAMESPACE   REVISION    UPDATED
STATUS       CHART                          APP VERSION

trident-operator    trident     1           2024-10-14
14:31:22.463122 +0300 IDT    deployed     trident-operator-
100.2502.0    25.02.0
```

Associazione dell'account di servizio (IRSA)

1. Aggiungere il repository Trident Helm:

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

2. Imposta i valori per **cloud provider** e **cloud identity**:

```
helm install trident-operator netapp-trident/trident-operator
--version 100.2502.1  \
--set cloudProvider="AWS" \
--set cloudIdentity="'eks.amazonaws.com/role-arn:
arn:aws:iam::<accountID>:role/<AmazonEKS_FSxN_CSI_DriverRole>'" \
--namespace trident \
--create-namespace
```

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

```
helm list -n trident
```

```
NAME                NAMESPACE   REVISION    UPDATED
STATUS       CHART                          APP VERSION

trident-operator    trident     1           2024-10-14
14:31:22.463122 +0300 IDT    deployed     trident-operator-
100.2506.0    25.06.0
```

Se prevedi di utilizzare iSCSI, assicurati che iSCSI sia abilitato sul computer client. Se utilizzi il sistema operativo AL2023 Worker node, puoi automatizzare l'installazione del client iSCSI aggiungendo il parametro node prep nell'installazione di helm:

```
helm install trident-operator netapp-trident/trident-operator
--version 100.2502.1 --namespace trident --create-namespace --
set nodePrep={iscsi}
```

Installare Trident tramite il componente aggiuntivo EKS

Il componente aggiuntivo Trident EKS include le più recenti patch di sicurezza, correzioni di bug ed è convalidato da AWS per funzionare con Amazon EKS. Il componente aggiuntivo EKS ti consente di garantire in modo coerente che i tuoi cluster Amazon EKS siano sicuri e stabili e di ridurre la quantità di lavoro da svolgere per installare, configurare e aggiornare i componenti aggiuntivi.

Prerequisiti

Prima di configurare il componente aggiuntivo Trident per AWS EKS, assicurati di disporre di quanto segue:

- Un account cluster Amazon EKS con abbonamento add-on
- Autorizzazioni AWS nel marketplace AWS:
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- Tipo di ami: Amazon Linux 2 (AL2_x86_64) o Amazon Linux 2 Arm (AL2_ARM_64)
- Tipo di nodo: AMD o ARM
- Un file system Amazon FSX per NetApp ONTAP esistente

Attiva il componente aggiuntivo Trident per AWS

Console di gestione

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel riquadro di spostamento di sinistra, selezionare **Cluster**.
3. Selezionare il nome del cluster per il quale si desidera configurare il componente aggiuntivo NetApp Trident CSI.
4. Selezionare **componenti aggiuntivi**, quindi selezionare **Ottieni altri componenti aggiuntivi**.
5. Per selezionare il componente aggiuntivo, segui questi passaggi:
   a. Scorri verso il basso fino alla sezione **Componenti aggiuntivi di AWS Marketplace** e digita **"Trident"** nella casella di ricerca.
   b. Selezionare la casella di controllo nell'angolo in alto a destra della casella Trident by NetApp.
   c. Selezionare **Avanti**.
6. Nella pagina Impostazioni **Configura componenti aggiuntivi selezionati**, effettuare le seguenti operazioni:

   Salta questi passaggi se utilizzi l'associazione Pod Identity.

   a. Selezionare la **versione** che si desidera utilizzare.
   b. Se si utilizza l'autenticazione IRSA, assicurarsi di impostare i valori di configurazione disponibili nelle impostazioni di configurazione facoltative:
      - Selezionare la **versione** che si desidera utilizzare.
      - Segui lo **schema di configurazione del componente aggiuntivo** e imposta il parametro **configurationValues** nella sezione **Valori di configurazione** sul role-arn creato nel passaggio precedente (il valore deve essere nel seguente formato):

```
{

  "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'",
  "cloudProvider": "AWS"

}
```

   +
   Se si seleziona Sovrascrivi per il metodo di risoluzione dei conflitti, una o più impostazioni per il componente aggiuntivo esistente possono essere sovrascritte con le impostazioni del componente aggiuntivo Amazon EKS. Se non si attiva questa opzione e si verifica un conflitto con le impostazioni esistenti, l'operazione non riesce. È possibile utilizzare il messaggio di errore risultante per risolvere il conflitto. Prima di selezionare questa opzione, assicurati che il componente aggiuntivo Amazon EKS non gestisca le impostazioni da gestire in autonomia.

7. Scegliere **Avanti**.
8. Nella pagina **Rivedi e Aggiungi**, scegliere **Crea**.

   Al termine dell'installazione del componente aggiuntivo, viene visualizzato il componente aggiuntivo installato.

CLI AWS

1. Crea il `add-on.json` file:

Per l'identità del pod, utilizzare il seguente formato:

```
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v25.6.0-eksbuild.1",
}
```

Per l'autenticazione IRSA, utilizzare il seguente formato:

```
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v25.6.0-eksbuild.1",
  "serviceAccountRoleArn": "<role ARN>",
  "configurationValues": {
    "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'",
    "cloudProvider": "AWS"
  }
}
```

Sostituire `<role ARN>` con l'ARN del ruolo creato nel passaggio precedente.

2. Installa il componente aggiuntivo Trident EKS.

```
aws eks create-addon --cli-input-json file://add-on.json
```

eksctl

Il seguente comando di esempio installa il componente aggiuntivo Trident EKS:

```
eksctl create addon --name netapp_trident-operator --cluster
<cluster_name> --force
```

Aggiornare il componente aggiuntivo Trident EKS

Console di gestione

1. Aprire la console Amazon EKS https://console.aws.amazon.com/eks/home#/clusters.
2. Nel riquadro di spostamento di sinistra, selezionare **Cluster**.
3. Selezionare il nome del cluster per il quale si desidera aggiornare il componente aggiuntivo NetApp Trident CSI.
4. Selezionare la scheda **componenti aggiuntivi**.
5. Selezionare **Trident by NetApp**, quindi selezionare **Modifica**.
6. Nella pagina **Configure Trident by** (Configura server tramite NetApp*), procedere come segue:
   a. Selezionare la **versione** che si desidera utilizzare.
   b. Espandere le **impostazioni di configurazione opzionali** e modificarle secondo necessità.
   c. Selezionare **Save Changes** (Salva modifiche).

CLI AWS

Nell'esempio seguente viene aggiornato il componente aggiuntivo EKS:

```
aws eks update-addon --cluster-name <eks_cluster_name> --addon-name
netapp_trident-operator --addon-version v25.6.0-eksbuild.1 \
  --service-account-role-arn <role-ARN> --resolve-conflict preserve \
  --configuration-values "{\"cloudIdentity\":
\"'eks.amazonaws.com/role-arn: <role ARN>'\"}"
```

eksctl

- Controllare la versione corrente del componente aggiuntivo FSxN Trident CSI. Sostituire `my-cluster` con il nome del cluster.

```
eksctl get addon --name netapp_trident-operator --cluster my-cluster
```

Esempio di output:

```
NAME                        VERSION             STATUS    ISSUES
IAMROLE    UPDATE AVAILABLE    CONFIGURATION VALUES
netapp_trident-operator    v25.6.0-eksbuild.1    ACTIVE    0
{"cloudIdentity":"'eks.amazonaws.com/role-arn:
arn:aws:iam::139763910815:role/AmazonEKS_FSXN_CSI_DriverRole'"}
```

- Aggiornare il componente aggiuntivo alla versione restituita in AGGIORNAMENTO DISPONIBILE nell'output del passaggio precedente.

```
eksctl update addon --name netapp_trident-operator --version
v25.6.0-eksbuild.1 --cluster my-cluster --force
```

Se si rimuove l' `--force` opzione e una delle impostazioni del componente aggiuntivo Amazon EKS è in conflitto con le impostazioni esistenti, l'aggiornamento del componente aggiuntivo Amazon EKS non viene eseguito correttamente; viene visualizzato un messaggio di errore che aiuta a risolvere il conflitto. Prima di specificare questa opzione, assicurati che il componente aggiuntivo Amazon EKS non gestisca le impostazioni da gestire, perché queste impostazioni vengono sovrascritte con questa opzione. Per ulteriori informazioni sulle altre opzioni per questa impostazione, vedere "Componenti aggiuntivi". Per ulteriori informazioni su Amazon EKS Kubernetes Field management, consulta "Gestione sul campo di Kubernetes".

Disinstallare/rimuovere il componente aggiuntivo Trident EKS

Hai due opzioni per rimuovere un add-on Amazon EKS:

- **Mantieni il software aggiuntivo sul tuo cluster** – questa opzione rimuove la gestione Amazon EKS di qualsiasi impostazione. Inoltre, rimuove la possibilità per Amazon EKS di informarti degli aggiornamenti e di aggiornare automaticamente il componente aggiuntivo Amazon EKS dopo l'avvio di un aggiornamento. Tuttavia, mantiene il software add-on sul cluster. Questa opzione rende il componente aggiuntivo un'installazione a gestione autonoma, piuttosto che un componente aggiuntivo Amazon EKS. Con questa opzione, il componente aggiuntivo non presenta tempi di inattività. Mantenere l' `--preserve` opzione nel comando per mantenere il componente aggiuntivo.
- **Rimozione del software aggiuntivo interamente dal cluster** – NetApp consiglia di rimuovere il componente aggiuntivo Amazon EKS dal cluster solo se non sono presenti risorse del cluster che dipendono da esso. Rimuovere l' `--preserve` opzione dal `delete` comando per rimuovere il componente aggiuntivo.

Se al componente aggiuntivo è associato un account IAM, l'account IAM non viene rimosso.

Console di gestione

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel riquadro di spostamento di sinistra, selezionare **cluster**.
3. Selezionare il nome del cluster per il quale si desidera rimuovere il componente aggiuntivo NetApp Trident CSI.
4. Selezionare la scheda **componenti aggiuntivi**, quindi selezionare **Trident by NetApp**.*
5. Selezionare **Rimuovi**.
6. Nella finestra di dialogo **Rimuovi conferma netapp_trident-operator**, esegui quanto segue:
   a. Se si desidera che Amazon EKS smetta di gestire le impostazioni del componente aggiuntivo, selezionare **conserva su cluster**. Questa operazione consente di conservare il software aggiuntivo nel cluster in modo da poter gestire da soli tutte le impostazioni del componente aggiuntivo.
   b. Immettere **netapp_trident-operator**.
   c. Selezionare **Rimuovi**.

CLI AWS

Sostituisci `my-cluster` con il nome del cluster ed esegui il seguente comando.

```
aws eks delete-addon --cluster-name my-cluster --addon-name
netapp_trident-operator --preserve
```

eksctl

Il seguente comando disinstalla il componente aggiuntivo Trident EKS:

```
eksctl delete addon --cluster K8s-arm --name netapp_trident-operator
```

Configurare il backend di archiviazione

Integrazione dei driver ONTAP SAN e NAS

Per creare un backend di archiviazione, è necessario creare un file di configurazione in formato JSON o YAML. Il file deve specificare il tipo di storage desiderato (NAS o SAN), il file system e la SVM per ottenerlo e come eseguirne l'autenticazione. Il seguente esempio illustra come definire lo storage basato su NAS e utilizzare un segreto AWS per memorizzare le credenziali nella SVM che desideri utilizzare:

YAML

```yaml
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  backendName: tbc-ontap-nas
  svm: svm-name
  aws:
    fsxFilesystemID: fs-xxxxxxxxxx
  credentials:
    name: "arn:aws:secretsmanager:us-west-2:xxxxxxxx:secret:secret-
name"
    type: awsarn
```

JSON

```json
{
  "apiVersion": "trident.netapp.io/v1",
  "kind": "TridentBackendConfig",
  "metadata": {
    "name": "backend-tbc-ontap-nas"
    "namespace": "trident"
  },
  "spec": {
    "version": 1,
    "storageDriverName": "ontap-nas",
    "backendName": "tbc-ontap-nas",
    "svm": "svm-name",
    "aws": {
      "fsxFilesystemID": "fs-xxxxxxxxxx"
    },
    "managementLIF": null,
    "credentials": {
      "name": "arn:aws:secretsmanager:us-west-2:xxxxxxxx:secret:secret-
name",
      "type": "awsarn"
    }
  }
}
```

Eseguire i seguenti comandi per creare e convalidare la configurazione del backend Trident (TBC):

- Creare la configurazione back-end Trident (TBC) dal file yaml ed eseguire il comando seguente:

```
kubectl create -f backendconfig.yaml -n trident
```

```
tridentbackendconfig.trident.netapp.io/backend-tbc-ontap-nas created
```

- Verificare che la configurazione back-end Trident (TBC) sia stata creata correttamente:

```
Kubectl get tbc -n trident
```

```
NAME                          BACKEND NAME         BACKEND UUID
PHASE   STATUS

backend-tbc-ontap-nas         tbc-ontap-nas        933e0071-66ce-4324-
b9ff-f96d916ac5e9   Bound   Success
```

Dettagli del driver FSX per ONTAP

Puoi integrare Trident con Amazon FSX per NetApp ONTAP utilizzando i seguenti driver:

- `ontap-san`: Ogni PV sottoposto a provisioning è una LUN all'interno del proprio volume Amazon FSX per NetApp ONTAP. Consigliato per la conservazione dei blocchi.
- `ontap-nas`: Ogni PV sottoposto a provisioning è un volume Amazon FSX completo per NetApp ONTAP. Consigliato per NFS e SMB.
- `ontap-san-economy`: Ogni PV fornito è un LUN con un numero configurabile di LUN per volume Amazon FSX per NetApp ONTAP.
- `ontap-nas-economy`: Ogni PV fornito è un qtree, con un numero configurabile di qtree per ogni volume Amazon FSX per NetApp ONTAP.
- `ontap-nas-flexgroup`: Ogni PV fornito è un volume Amazon FSX completo per NetApp ONTAP FlexGroup.

Per informazioni dettagliate sul conducente, fare riferimento a. "Driver NAS" e. "Driver SAN".

Una volta creato il file di configurazione, esegui questo comando per crearlo all'interno del tuo EKS:

```
kubectl create -f configuration_file
```

Per verificare lo stato, eseguire questo comando:

```
kubectl get tbc -n trident
```

```
NAME                     BACKEND NAME            BACKEND UUID
PHASE   STATUS
backend-fsx-ontap-nas   backend-fsx-ontap-nas   7a551921-997c-4c37-a1d1-
f2f4c87fa629   Bound   Success
```

Configurazione avanzata backend ed esempi

Per le opzioni di configurazione del backend, consultare la tabella seguente:

Parametro	Descrizione	Esempio
`version`		Sempre 1
`storageDriverName`	Nome del driver di storage	`ontap-nas`, `ontap-nas-economy`, `ontap-nas-flexgroup`, `ontap-san`, `ontap-san-economy`
`backendName`	Nome personalizzato o backend dello storage	Nome del driver + "_" + dataLIF
`managementLIF`	Indirizzo IP di un cluster o LIF di gestione SVM È possibile specificare Un nome di dominio completo (FQDN). Può essere impostato in modo da utilizzare gli indirizzi IPv6 se Trident è stato installato utilizzando il flag IPv6. Gli indirizzi IPv6 devono essere definiti tra parentesi quadre, ad esempio [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]. Se fornisci il `fsxFilesystemID` sotto `aws` il campo, non devi fornire il `managementLIF`, perché Trident recupera le informazioni SVM `managementLIF` da AWS. Pertanto, devi fornire le credenziali a un utente sotto la SVM (ad esempio, vsadmin) e tale utente deve avere un `vsadmin` ruolo.	"10,0.0,1", "[2001:1234:abcd::fefe]"

Parametro	Descrizione	Esempio
`dataLIF`	Indirizzo IP del protocollo LIF. **Driver NAS ONTAP**: NetApp consiglia di specificare dataLIF. Se non viene fornita, Trident recupera le LIF dati dalla SVM. È possibile specificare un nome di dominio completo (FQDN) da utilizzare per le operazioni di montaggio NFS, consentendo di creare un DNS round-robin per bilanciare il carico su più LIF dati. Può essere modificato dopo l'impostazione iniziale. Fare riferimento alla . **Driver SAN ONTAP**: Non specificare iSCSI. Trident utilizza la mappa selettiva delle LUN di ONTAP per scoprire le LIF di isci necessarie per stabilire una sessione multi-path. Viene generato un avviso se dataLIF è esplicitamente definito. Può essere impostato in modo da utilizzare gli indirizzi IPv6 se Trident è stato installato utilizzando il flag IPv6. Gli indirizzi IPv6 devono essere definiti tra parentesi quadre, ad esempio [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	
`autoExportPolicy`	Abilita la creazione e l'aggiornamento automatici dei criteri di esportazione [booleano]. Utilizzando le `autoExportPolicy` opzioni e `autoExportCIDRs`, Trident può gestire automaticamente i criteri di esportazione.	`false`
`autoExportCIDRs`	Elenco di CIDR per filtrare gli IP dei nodi di Kubernetes rispetto a quando `autoExportPolicy` è attivato. Utilizzando le `autoExportPolicy` opzioni e `autoExportCIDRs`, Trident può gestire automaticamente i criteri di esportazione.	"["0,0.0,0/0", "::/0"]"
`labels`	Set di etichette arbitrarie formattate con JSON da applicare sui volumi	""
`clientCertificate`	Valore del certificato client codificato con base64. Utilizzato per l'autenticazione basata su certificato	""

Parametro	Descrizione	Esempio
`clientPrivateKey`	Valore codificato in base64 della chiave privata del client. Utilizzato per l'autenticazione basata su certificato	""
`trustedCACertificate`	Valore codificato in base64 del certificato CA attendibile. Opzionale. Utilizzato per l'autenticazione basata su certificato.	""
`username`	Nome utente per la connessione al cluster o alla SVM. Utilizzato per l'autenticazione basata su credenziali. Ad esempio, vsadmin.	
`password`	Password per la connessione al cluster o alla SVM. Utilizzato per l'autenticazione basata su credenziali.	
`svm`	Macchina virtuale per lo storage da utilizzare	Derivato se viene specificato un LIF di gestione SVM.
`storagePrefix`	Prefisso utilizzato per il provisioning di nuovi volumi nella SVM. Impossibile modificare dopo la creazione. Per aggiornare questo parametro, è necessario creare un nuovo backend.	`trident`
`limitAggregateUsage`	**Non specificare Amazon FSX per NetApp ONTAP.** Fornito `fsxadmin` e `vsadmin` non contiene le autorizzazioni necessarie per recuperare l'utilizzo dell'aggregato e limitarlo mediante Trident.	Non utilizzare.
`limitVolumeSize`	Fallire il provisioning se la dimensione del volume richiesta è superiore a questo valore. Limita anche le dimensioni massime dei volumi gestiti per qtree e LUN e l' `qtreesPerFlexvol`opzione consente di personalizzare il numero massimo di qtree per FlexVol volume	"" (non applicato per impostazione predefinita)
`lunsPerFlexvol`	Il numero massimo di LUN per FlexVol volume deve essere compreso nell'intervallo [50, 200]. Solo SAN.	"`100'"

Parametro	Descrizione	Esempio
`debugTraceFlags`	Flag di debug da utilizzare per la risoluzione dei problemi. Esempio, {"api":false, "method":true} Non utilizzare `debugTraceFlags` a meno che non si stia eseguendo la risoluzione dei problemi e non si richieda un dump dettagliato del log.	nullo
`nfsMountOptions`	Elenco separato da virgole delle opzioni di montaggio NFS. Le opzioni di montaggio per volumi persistenti di Kubernetes vengono normalmente specificate in classi di storage, ma se non sono specificate opzioni di montaggio in una classe di storage, Trident tornerà all'utilizzo delle opzioni di montaggio specificate nel file di configurazione del backend di storage. Se non sono specificate opzioni di montaggio nella classe di storage o nel file di configurazione, Trident non imposterà alcuna opzione di montaggio su un volume persistente associato.	""
`nasType`	Configurare la creazione di volumi NFS o SMB. Le opzioni sono `nfs`, `smb`o nullo. **Deve essere impostato su `smb`** Per i volumi SMB. l'impostazione su Null imposta come predefinita i volumi NFS.	`nfs`
`qtreesPerFlexvol`	Qtree massimi per FlexVol volume, devono essere compresi nell'intervallo [50, 300]	`"200"`
`smbShare`	È possibile specificare uno dei seguenti elementi: Il nome di una condivisione SMB creata utilizzando la console di gestione Microsoft o l'interfaccia CLI di ONTAP oppure un nome per consentire a Trident di creare la condivisione SMB. Questo parametro è obbligatorio per i backend Amazon FSX per ONTAP.	`smb-share`

Parametro	Descrizione	Esempio
`useREST`	Parametro booleano per l'utilizzo delle API REST di ONTAP. Quando è impostato su `true`, Trident utilizza le API REST ONTAP per comunicare con il backend. Questa funzione richiede ONTAP 9.11.1 e versioni successive. Inoltre, il ruolo di accesso ONTAP utilizzato deve avere accesso all' `ontap` applicazione. Ciò è soddisfatto dai ruoli predefiniti `vsadmin` e `cluster-admin` .	`false`
`aws`	Puoi specificare quanto segue nel file di configurazione per AWS FSX per ONTAP: - `fsxFilesystemID`: Specificare l'ID del file system AWS FSX. - `apiRegion`: Nome regione API AWS. - `apikey`: Chiave API AWS. - `secretKey`: Chiave segreta AWS.	"" "" ""
`credentials`	Specifica le credenziali di FSX SVM da memorizzare in AWS Secrets Manager. - `name`: Amazon Resource Name (ARN) del segreto, che contiene le credenziali di SVM. - `type`: Impostare su `awsarn`. Per ulteriori informazioni, fare riferimento "Creare un segreto AWS Secrets Manager" a.	

Opzioni di configurazione back-end per il provisioning dei volumi

È possibile controllare il provisioning predefinito utilizzando queste opzioni in `defaults` della configurazione. Per un esempio, vedere gli esempi di configurazione riportati di seguito.

Parametro	Descrizione	Predefinito
`spaceAllocation`	Allocazione dello spazio per LUN	`true`
`spaceReserve`	Modalità di prenotazione dello spazio; "nessuno" (sottile) o "volume" (spesso)	`none`
`snapshotPolicy`	Policy di Snapshot da utilizzare	`none`

Parametro	Descrizione	Predefinito
`qosPolicy`	Gruppo di criteri QoS da assegnare per i volumi creati. Scegliere una delle opzioni qosPolicy o adaptiveQosPolicy per pool di storage o backend. L'utilizzo di gruppi di criteri QoS con Trident richiede ONTAP 9.8 o versioni successive. È necessario utilizzare un gruppo di criteri QoS non condiviso e garantire che il gruppo di criteri venga applicato singolarmente a ciascun componente. Un gruppo di policy QoS condiviso impone un limite massimo per il throughput totale di tutti i carichi di lavoro.	""
`adaptiveQosPolicy`	Gruppo di criteri QoS adattivi da assegnare per i volumi creati. Scegliere una delle opzioni qosPolicy o adaptiveQosPolicy per pool di storage o backend. Non supportato da ontap-nas-Economy.	""
`snapshotReserve`	Percentuale di volume riservato agli snapshot "0"	Se `snapshotPolicy` è `none`, `else` ""
`splitOnClone`	Separare un clone dal suo padre al momento della creazione	`false`
`encryption`	Abilitare la crittografia del volume NetApp (NVE) sul nuovo volume; il valore predefinito è `false`. NVE deve essere concesso in licenza e abilitato sul cluster per utilizzare questa opzione. Se NAE è abilitato sul backend, qualsiasi volume sottoposto a provisioning in Trident sarà abilitato NAE. Per ulteriori informazioni, fare riferimento a: "Come funziona Trident con NVE e NAE".	`false`
`luksEncryption`	Attivare la crittografia LUKS. Fare riferimento a. "Utilizzo di Linux Unified Key Setup (LUKS)". Solo SAN.	""
`tieringPolicy`	Policy di tiering da utilizzare `none`	
`unixPermissions`	Per i nuovi volumi. **Lasciare vuoto per i volumi SMB.**	""

Parametro	Descrizione	Predefinito
`securityStyle`	Stile di sicurezza per nuovi volumi. Supporto di NFS `mixed` e. `unix` stili di sicurezza. Supporto SMB `mixed` e. `ntfs` stili di sicurezza.	Il valore predefinito di NFS è `unix`. Il valore predefinito di SMB è `ntfs`.

Preparatevi al provisioning dei volumi SMB

È possibile eseguire il provisioning dei volumi SMB utilizzando `ontap-nas` driver. Prima di completare Integrazione dei driver ONTAP SAN e NAS completare i seguenti passaggi.

Prima di iniziare

Prima di eseguire il provisioning di volumi SMB utilizzando `ontap-nas` driver, è necessario disporre di quanto segue.

- Un cluster Kubernetes con un nodo controller Linux e almeno un nodo di lavoro Windows che esegue Windows Server 2019. Trident supporta volumi SMB montati su pod in esecuzione solo sui nodi Windows.
- Almeno un segreto Trident contenente le credenziali di Active Directory. Per generare segreto `smbcreds`:

```
kubectl create secret generic smbcreds --from-literal username=user
--from-literal password='password'
```

- Proxy CSI configurato come servizio Windows. Per configurare un `csi-proxy`, fare riferimento a. "GitHub: Proxy CSI" oppure "GitHub: Proxy CSI per Windows" Per i nodi Kubernetes in esecuzione su Windows.

Fasi

1. Creare condivisioni SMB. È possibile creare le condivisioni amministrative SMB in due modi utilizzando "Console di gestione Microsoft" Snap-in cartelle condivise o utilizzo dell'interfaccia CLI di ONTAP. Per creare le condivisioni SMB utilizzando la CLI ONTAP:
   a. Se necessario, creare la struttura del percorso di directory per la condivisione.

      Il `vserver cifs share create` il comando controlla il percorso specificato nell'opzione -path durante la creazione della condivisione. Se il percorso specificato non esiste, il comando non riesce.

   b. Creare una condivisione SMB associata alla SVM specificata:

```
vserver cifs share create -vserver vserver_name -share-name
share_name -path path [-share-properties share_properties,...]
[other_attributes] [-comment text]
```

   c. Verificare che la condivisione sia stata creata:

```
vserver cifs share show -share-name share_name
```

> Fare riferimento a. "Creare una condivisione SMB" per informazioni dettagliate.

2. Quando si crea il backend, è necessario configurare quanto segue per specificare i volumi SMB. Per tutte le opzioni di configurazione backend FSX per ONTAP, fare riferimento a. "FSX per le opzioni di configurazione e gli esempi di ONTAP".

Parametro	Descrizione	Esempio
`smbShare`	È possibile specificare uno dei seguenti elementi: Il nome di una condivisione SMB creata utilizzando la console di gestione Microsoft o l'interfaccia CLI di ONTAP oppure un nome per consentire a Trident di creare la condivisione SMB. Questo parametro è obbligatorio per i backend Amazon FSX per ONTAP.	`smb-share`
`nasType`	**Deve essere impostato su `smb`.** se null, il valore predefinito è `nfs`.	`smb`
`securityStyle`	Stile di sicurezza per nuovi volumi. **Deve essere impostato su `ntfs` oppure `mixed` Per volumi SMB.**	`ntfs` oppure `mixed` Per volumi SMB
`unixPermissions`	Per i nuovi volumi. **Deve essere lasciato vuoto per i volumi SMB.**	""

Configurare una classe di storage e PVC

Configurare un oggetto Kubernetes StorageClass e creare la classe storage per istruire Trident su come eseguire il provisioning dei volumi. Creare un PersistentVolumeClaim (PVC) che utilizzi Kubernetes StorageClass configurato per richiedere l'accesso al PV. È quindi possibile montare il PV su un pod.

Creare una classe di storage

Configurare un oggetto Kubernetes StorageClass

IL "Oggetto Kubernetes StorageClass" L'oggetto identifica Trident come il provisioner utilizzato per quella classe e indica a Trident come effettuare il provisioning di un volume. Utilizzare questo esempio per configurare Storageclass per i volumi tramite NFS (fare riferimento alla sezione Attributi Trident di seguito per l'elenco completo degli attributi):

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  provisioningType: "thin"
  snapshots: "true"
```

Utilizzare questo esempio per configurare Storageclass per i volumi che utilizzano iSCSI:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
  provisioningType: "thin"
  snapshots: "true"
```

Per eseguire il provisioning di volumi NFSv3 su AWS Bottlerocket, aggiungere i necessari `mountOptions` alla classe storage:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  media: "ssd"
  provisioningType: "thin"
  snapshots: "true"
mountOptions:
  - nfsvers=3
  - nolock
```

Per ulteriori informazioni sull'interazione delle classi di archiviazione con i `PersistentVolumeClaim` parametri e per il controllo del provisioning dei volumi da parte di Trident, fare riferimento "Kubernetes e Trident Objects"a.

Creare una classe di storage

Fasi

1. Si tratta di un oggetto Kubernetes, lo utilizza `kubectl` Per crearlo in Kubernetes.

```
kubectl create -f storage-class-ontapnas.yaml
```

2. Ora dovresti vedere una classe storage **Basic-csi** in Kubernetes e Trident, e Trident dovrebbe aver scoperto i pool nel back-end.

```
kubectl get sc basic-csi
```

```
NAME          PROVISIONER             AGE
basic-csi     csi.trident.netapp.io   15h
```

Creare il PVC

Un "*PersistentVolumeClaim*" (PVC) è una richiesta di accesso a PersistentVolume sul cluster.

Il PVC può essere configurato per richiedere la memorizzazione di una determinata dimensione o modalità di accesso. Utilizzando StorageClass associato, l'amministratore del cluster può controllare più delle dimensioni di PersistentVolume e della modalità di accesso, ad esempio le prestazioni o il livello di servizio.

Dopo aver creato il PVC, è possibile montare il volume in un pod.

Manifesti campione

Manifesti di campioni PersistentVolumeClaim

Questi esempi mostrano le opzioni di configurazione di base del PVC.

PVC con accesso RWX

Questo esempio mostra un PVC di base con accesso RWX associato a un StorageClass denominato `basic-csi`.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-storage
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-gold
```

PVC utilizzando l'esempio iSCSI

Questo esempio mostra un PVC di base per iSCSI con accesso RWO associato a una StorageClass denominata `protection-gold` .

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: pvc-san
spec:
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 1Gi
storageClassName: protection-gold
```

Crea PVC

Fasi

1. Creare il PVC.

```
kubectl create -f pvc.yaml
```

2. Verificare lo stato del PVC.

```
kubectl get pvc
```

```
NAME         STATUS VOLUME     CAPACITY ACCESS MODES STORAGECLASS AGE
pvc-storage Bound  pv-name 2Gi      RWO                          5m
```

Per ulteriori informazioni sull'interazione delle classi di archiviazione con i `PersistentVolumeClaim` parametri e per il controllo del provisioning dei volumi da parte di Trident, fare riferimento "Kubernetes e Trident Objects"a.

Attributi Trident

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

Attributo	Tipo	Valori	Offerta	Richiesta	Supportato da
supporti[1]	stringa	hdd, ibrido, ssd	Il pool contiene supporti di questo tipo; ibridi significa entrambi	Tipo di supporto specificato	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san
ProvisioningType	stringa	sottile, spesso	Il pool supporta questo metodo di provisioning	Metodo di provisioning specificato	thick: all ONTAP; thin: all ONTAP e solidfire-san
BackendType	stringa	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san, gcp-cvs, azure-netapp-files, ontap-san-economy	Il pool appartiene a questo tipo di backend	Backend specificato	Tutti i driver
snapshot	bool	vero, falso	Il pool supporta volumi con snapshot	Volume con snapshot attivate	ontap-nas, ontap-san, solidfire-san, gcp-cvs
cloni	bool	vero, falso	Il pool supporta la clonazione dei volumi	Volume con cloni attivati	ontap-nas, ontap-san, solidfire-san, gcp-cvs

Attributo	Tipo	Valori	Offerta	Richiesta	Supportato da
crittografia	bool	vero, falso	Il pool supporta volumi crittografati	Volume con crittografia attivata	ontap-nas, ontap-nas-economy, ontap-nas-flexgroups, ontap-san
IOPS	int	intero positivo	Il pool è in grado di garantire IOPS in questa gamma	Volume garantito per questi IOPS	solidfire-san

[1]: Non supportato dai sistemi ONTAP Select

Distribuire l'applicazione di esempio

Una volta creata la classe di archiviazione e il PVC, è possibile montare il PV su un pod. Questa sezione elenca il comando e la configurazione di esempio per collegare il PV a un pod.

Fasi

1. Montare il volume in un pod.

```
kubectl create -f pv-pod.yaml
```

Questi esempi mostrano le configurazioni di base per collegare il PVC a un pod: **Configurazione di base**:

```
kind: Pod
apiVersion: v1
metadata:
  name: pv-pod
spec:
  volumes:
    - name: pv-storage
      persistentVolumeClaim:
       claimName: basic
  containers:
    - name: pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: pv-storage
```

È possibile monitorare l'avanzamento utilizzando `kubectl get pod --watch`.

2. Verificare che il volume sia montato su `/my/mount/path`.

```
kubectl exec -it pv-pod -- df -h /my/mount/path
```

```
Filesystem                                                      Size
Used  Avail Use%  Mounted on
192.168.188.78:/trident_pvc_ae45ed05_3ace_4e7c_9080_d2a83ae03d06  1.1G
320K  1.0G   1%   /my/mount/path
```

A questo punto è possibile eliminare il pod. L'applicazione Pod non esisterà più, ma il volume rimarrà.

```
kubectl delete pod pv-pod
```

Configurare il componente aggiuntivo Trident EKS su un cluster EKS

NetApp Trident ottimizza la gestione dello storage di Amazon FSX per NetApp ONTAP in Kubernetes per permettere a sviluppatori e amministratori di concentrarsi sull'implementazione dell'applicazione. Il componente aggiuntivo NetApp Trident EKS include le più recenti patch di sicurezza, correzioni di bug ed è convalidato da AWS per funzionare con Amazon EKS. Il componente aggiuntivo EKS ti consente di garantire in modo coerente che i tuoi cluster Amazon EKS siano sicuri e stabili e di ridurre la quantità di lavoro da svolgere per installare, configurare e aggiornare i componenti aggiuntivi.

Prerequisiti

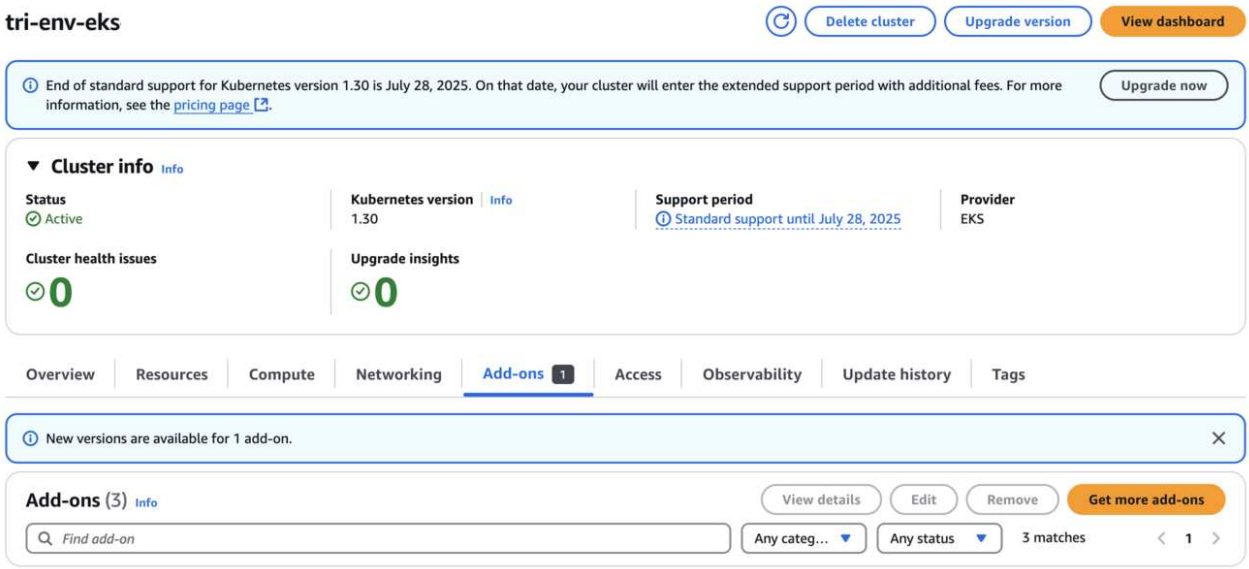
Prima di configurare il componente aggiuntivo Trident per AWS EKS, assicurati di disporre di quanto segue:

- Un account cluster Amazon EKS con autorizzazioni per l'uso dei componenti aggiuntivi. Fare riferimento alla "Componenti aggiuntivi Amazon EKS".
- Autorizzazioni AWS nel marketplace AWS:
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- Tipo di ami: Amazon Linux 2 (AL2_x86_64) o Amazon Linux 2 Arm (AL2_ARM_64)
- Tipo di nodo: AMD o ARM
- Un file system Amazon FSX per NetApp ONTAP esistente

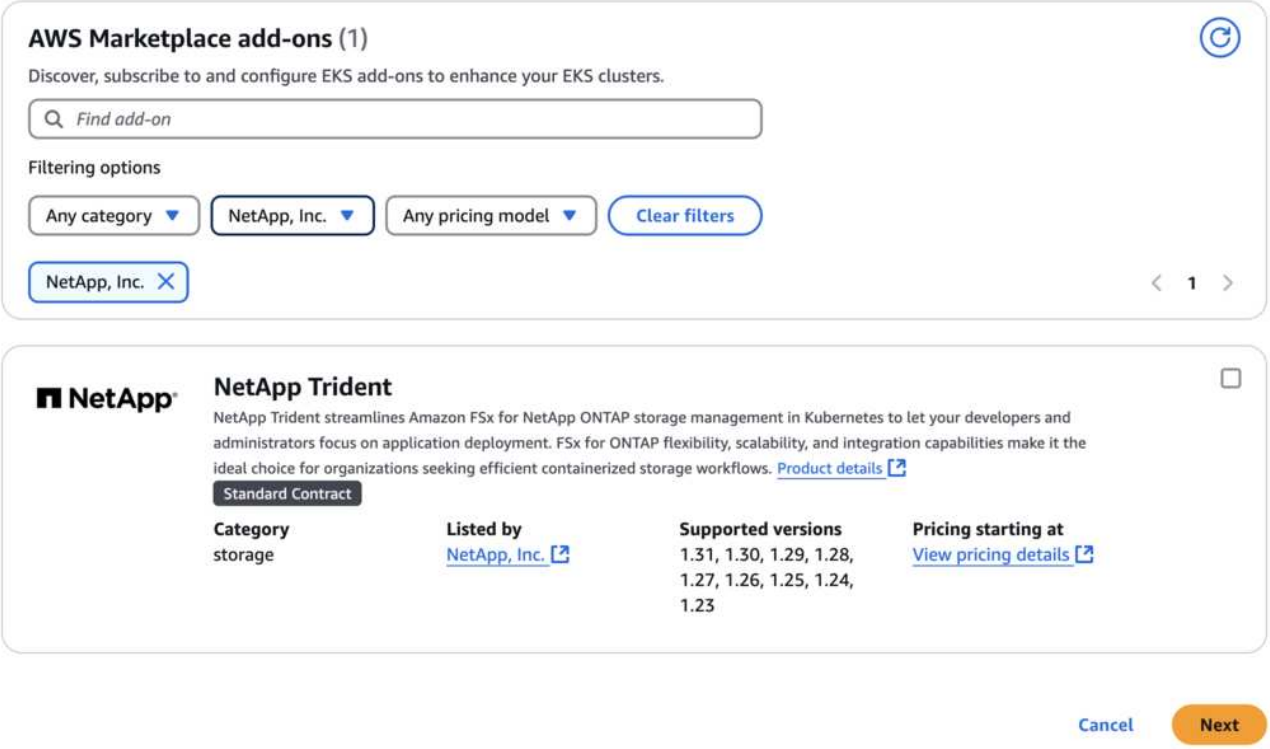
Fasi

1. Assicurati di creare il ruolo IAM e il segreto AWS per abilitare i pod EKS per accedere alle risorse AWS. Per istruzioni, vedere "Creare un ruolo IAM e un segreto AWS".

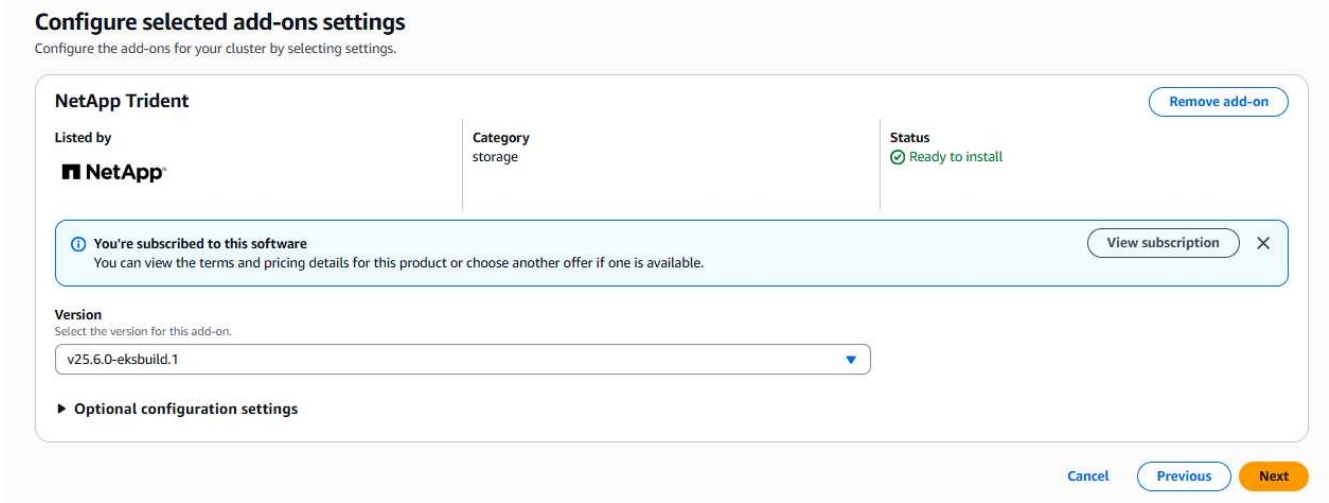
2. Sul tuo cluster EKS Kubernetes, accedi alla scheda **Add-on**.



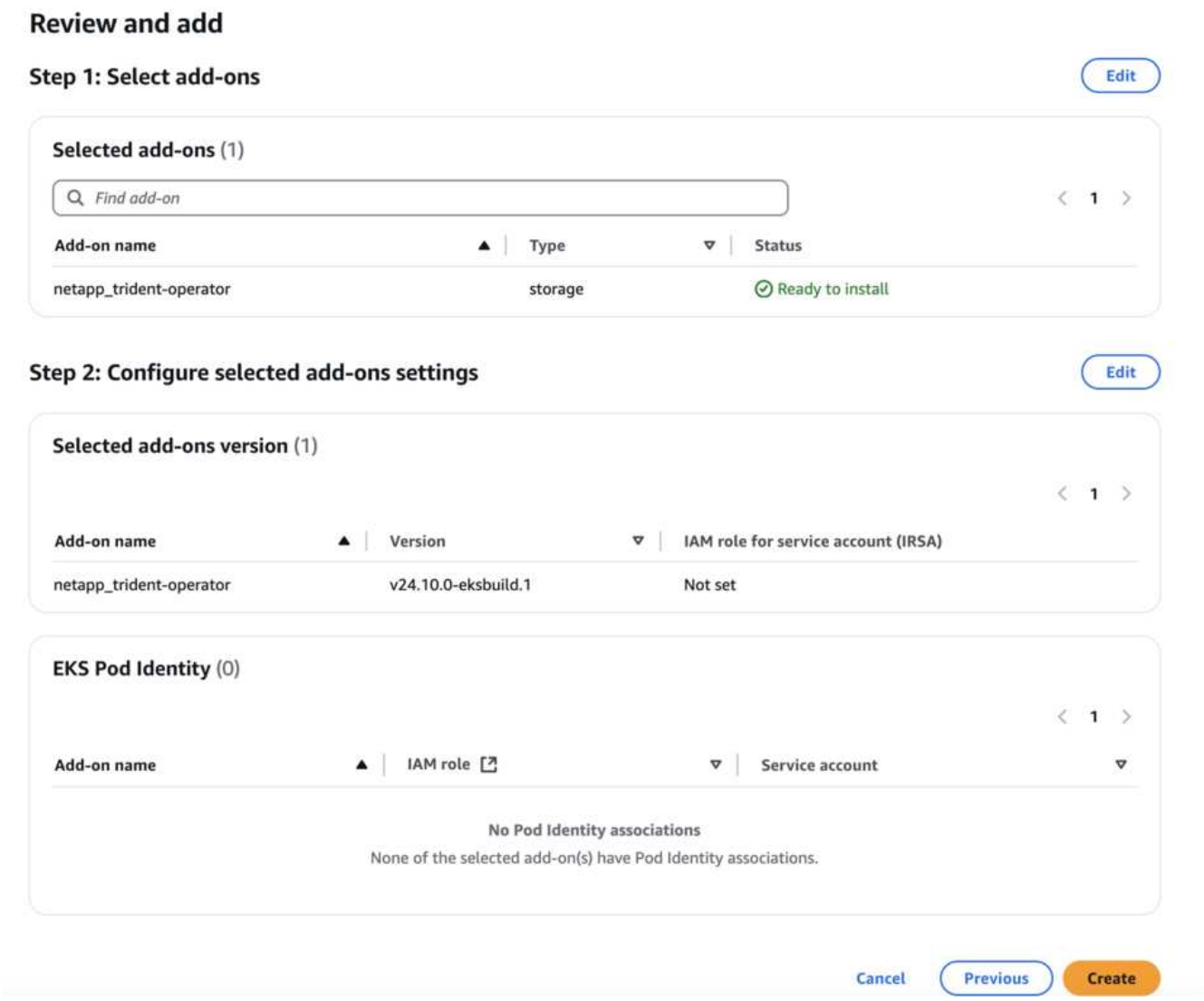
3. Vai su **componenti aggiuntivi di AWS Marketplace** e scegli la categoria *storage*.



4. Individuare **NetApp Trident** e selezionare la casella di controllo del componente aggiuntivo Trident, quindi fare clic su **Avanti**.
5. Scegliere la versione desiderata del componente aggiuntivo.

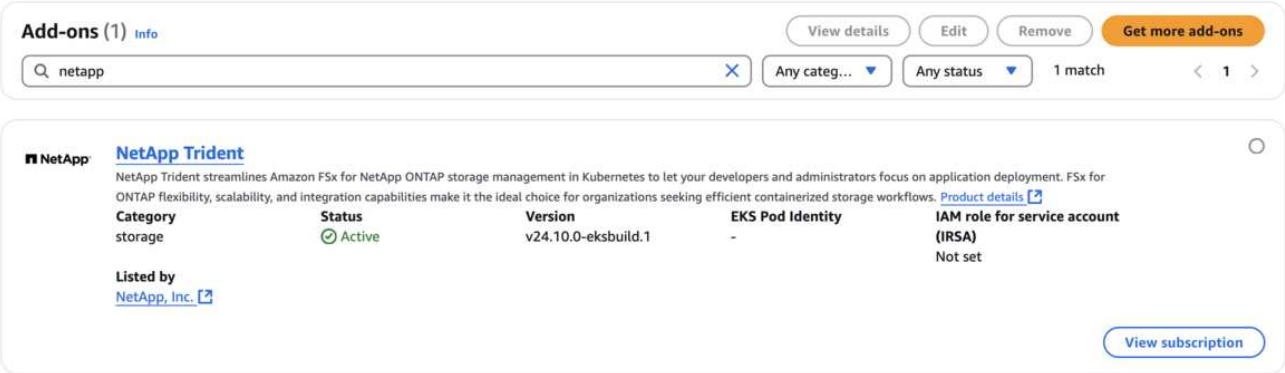


6. Configurare le impostazioni aggiuntive richieste.



7. Se si utilizza IRSA (ruoli IAM per l'account di servizio), fare riferimento ai passaggi di configurazione aggiuntivi"qui" .
8. Selezionare **Crea**.

9. Verificare che lo stato del componente aggiuntivo sia *attivo*.



10. Eseguire il seguente comando per verificare che Trident sia installato correttamente nel cluster:

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

11. Continuare l'installazione e configurare il backend di archiviazione. Per informazioni, vedere "Configurare il backend di archiviazione".

Installare/disinstallare il componente aggiuntivo Trident EKS utilizzando la CLI

Installare il componente aggiuntivo NetApp Trident EKS utilizzando la CLI:

Il seguente comando di esempio installa il componente aggiuntivo Trident EKS: `eksctl create addon --cluster clusterName --name netapp_trident-operator --version v25.6.0-eksbuild.1` (con una versione dedicata)

Disinstallare il componente aggiuntivo NetApp Trident EKS utilizzando CLI:

Il seguente comando disinstalla il componente aggiuntivo Trident EKS:

```
eksctl delete addon --cluster K8s-arm --name netapp_trident-operator
```

Crea backend con kubectl

Un backend definisce la relazione tra Trident e un sistema di storage. Spiega a Trident come comunicare con quel sistema storage e come Trident dovrebbe eseguire il provisioning dei volumi da esso. Dopo l'installazione di Trident, il passaggio successivo consiste nella creazione di un backend. La `TridentBackendConfig` definizione risorsa personalizzata (CRD) ti consente di creare e gestire i backend Trident direttamente attraverso l'interfaccia di Kubernetes. Puoi farlo utilizzando `kubectl` o l'equivalente strumento CLI per la tua distribuzione Kubernetes.

`TridentBackendConfig`

`TridentBackendConfig` (`tbc`, , `tbconfig` `tbackendconfig`) È un CRD in primo piano, con nome, che consente di gestire backend Trident utilizzando `kubectl`. Gli amministratori di Kubernetes e dello storage possono ora creare e gestire i backend direttamente attraverso l'interfaccia a riga di comando di Kubernetes

senza richiedere un'utility a riga di comando dedicata (`tridentctl`).

Alla creazione di un `TridentBackendConfig` oggetto, si verifica quanto segue:

- Trident crea automaticamente un backend in base alla configurazione fornita. Questo è rappresentato internamente come a `TridentBackend` (`tbe`, `tridentbackend`) CR.
- Il `TridentBackendConfig` è associato in modo univoco a un `TridentBackend` creato da Trident.

Ciascuno `TridentBackendConfig` mantiene una mappatura uno a uno con un `TridentBackend`. Il primo è l'interfaccia fornita all'utente per progettare e configurare i backend; il secondo è il modo in cui Trident rappresenta l'oggetto backend effettivo.

> `TridentBackend` I CRS vengono creati automaticamente da Trident. Non è possibile modificarle. Se si desidera aggiornare i backend, modificare l' `TridentBackendConfig`oggetto.

Vedere l'esempio seguente per il formato di `TridentBackendConfig` CR:

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret
```

È inoltre possibile esaminare gli esempi in "trident-installer" directory per configurazioni di esempio per la piattaforma/servizio di storage desiderato.

Il `spec` utilizza parametri di configurazione specifici per il back-end. In questo esempio, il backend utilizza `ontap-san` storage driver e utilizza i parametri di configurazione riportati in tabella. Per un elenco delle opzioni di configurazione del driver di archiviazione desiderato, consultare la "informazioni di configurazione back-end per il driver di storage".

Il `spec` la sezione include anche `credentials` e. `deletionPolicy` i campi, che sono stati introdotti di recente in `TridentBackendConfig` CR:

- `credentials`: Questo parametro è un campo obbligatorio e contiene le credenziali utilizzate per l'autenticazione con il sistema/servizio di storage. Questo è impostato su un Kubernetes Secret creato dall'utente. Le credenziali non possono essere passate in testo normale e si verificherà un errore.
- `deletionPolicy`: Questo campo definisce cosa deve accadere quando `TridentBackendConfig` viene cancellato. Può assumere uno dei due valori possibili:
  - `delete`: Questo comporta l'eliminazione di entrambi `TridentBackendConfig` CR e il backend

associato. Questo è il valore predefinito.

- `retain`: Quando un `TridentBackendConfig` La CR viene eliminata, la definizione di back-end rimane presente e può essere gestita con `tridentctl`. Impostazione del criterio di eliminazione su `retain` consente agli utenti di eseguire il downgrade a una release precedente (precedente alla 21.04) e conservare i backend creati. Il valore di questo campo può essere aggiornato dopo un `TridentBackendConfig` viene creato.

> Il nome di un backend viene impostato utilizzando `spec.backendName`. Se non specificato, il nome del backend viene impostato sul nome di `TridentBackendConfig` oggetto (metadata.name). Si consiglia di impostare esplicitamente i nomi backend utilizzando `spec.backendName`.

> I backend creati con `tridentctl` non hanno un oggetto associato `TridentBackendConfig`. È possibile scegliere di gestire tali backend con `kubectl` creando una `TridentBackendConfig` CR. Occorre prestare attenzione a specificare parametri di configurazione identici (come `spec.backendName`, , `spec.storagePrefix`, `spec.storageDriverName` e così via). Trident associa automaticamente il nuovo creato `TridentBackendConfig` al backend preesistente.

Panoramica dei passaggi

Per creare un nuovo backend utilizzando `kubectl`, eseguire le seguenti operazioni:

1. Crea un "Kubernetes Secret". il segreto contiene le credenziali che Trident deve avere per comunicare con il cluster/servizio di archiviazione.
2. Creare un `TridentBackendConfig` oggetto. Contiene specifiche relative al cluster/servizio di storage e fa riferimento al segreto creato nel passaggio precedente.

Dopo aver creato un backend, è possibile osservarne lo stato utilizzando `kubectl get tbc <tbc-name> -n <trident-namespace>` e raccogliere ulteriori dettagli.

Fase 1: Creare un Kubernetes Secret

Creare un segreto contenente le credenziali di accesso per il backend. Si tratta di una caratteristica esclusiva di ogni piattaforma/servizio di storage. Ecco un esempio:

```
kubectl -n trident create -f backend-tbc-ontap-san-secret.yaml
```

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-san-secret
type: Opaque
stringData:
  username: cluster-admin
  password: password
```

Questa tabella riassume i campi che devono essere inclusi nel Secret per ciascuna piattaforma di storage:

Descrizione dei campi segreti della piattaforma di storage	Segreto	Descrizione dei campi
Azure NetApp Files	ID cliente	L'ID client dalla registrazione di un'applicazione
Cloud Volumes Service per GCP	id_chiave_privata	ID della chiave privata. Parte della chiave API per l'account di servizio GCP con ruolo di amministratore CVS
Cloud Volumes Service per GCP	private_key	Chiave privata. Parte della chiave API per l'account di servizio GCP con ruolo di amministratore CVS
Elemento (NetApp HCI/SolidFire)	Endpoint	MVIP per il cluster SolidFire con credenziali tenant
ONTAP	nome utente	Nome utente per la connessione al cluster/SVM. Utilizzato per l'autenticazione basata su credenziali
ONTAP	password	Password per la connessione al cluster/SVM. Utilizzato per l'autenticazione basata su credenziali
ONTAP	ClientPrivateKey	Valore codificato in base64 della chiave privata del client. Utilizzato per l'autenticazione basata su certificato
ONTAP	ChapNomeUtente	Nome utente inbound. Obbligatorio se useCHAP=true. Per `ontap-san` e. `ontap-san-economy`
ONTAP	ChapInitialatorSecret	Segreto iniziatore CHAP. Obbligatorio se useCHAP=true. Per `ontap-san` e. `ontap-san-economy`
ONTAP	ChapTargetNomeUtente	Nome utente di destinazione. Obbligatorio se useCHAP=true. Per `ontap-san` e. `ontap-san-economy`

Descrizione dei campi segreti della piattaforma di storage	Segreto	Descrizione dei campi
ONTAP	ChapTargetInitialatorSecret	CHAP target Initiator secret. Obbligatorio se useCHAP=true. Per `ontap-san` e. `ontap-san-economy`

Il Segreto creato in questo passaggio verrà indicato in `spec.credentials` campo di `TridentBackendConfig` oggetto creato nel passaggio successivo.

Fase 2: Creare `TridentBackendConfig` CR

A questo punto, è possibile creare il `TridentBackendConfig` CR. In questo esempio, un backend che utilizza `ontap-san` il driver viene creato utilizzando `TridentBackendConfig` oggetto mostrato di seguito:

```
kubectl -n trident create -f backend-tbc-ontap-san.yaml
```

```yaml
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret
```

Fase 3: Verificare lo stato di `TridentBackendConfig` CR

Ora che è stato creato il `TridentBackendConfig` CR, è possibile verificare lo stato. Vedere il seguente esempio:

```
kubectl -n trident get tbc backend-tbc-ontap-san
NAME                     BACKEND NAME          BACKEND UUID
PHASE   STATUS
backend-tbc-ontap-san   ontap-san-backend     8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8   Bound   Success
```

Un backend è stato creato e associato a `TridentBackendConfig` CR.

La fase può assumere uno dei seguenti valori:

- `Bound`: Il `TridentBackendConfig` CR è associato a un backend e contiene tale backend `configRef` impostare su `TridentBackendConfig` Uid di CR.
- `Unbound`: Rappresentato utilizzando `""`. Il `TridentBackendConfig` l'oggetto non è associato a un backend. Tutti creati di recente `TridentBackendConfig` I CRS sono in questa fase per impostazione predefinita. Una volta modificata la fase, non sarà più possibile tornare a Unbound.
- `Deleting`: Il `TridentBackendConfig` CR `deletionPolicy` è stato impostato per l'eliminazione. Quando il `TridentBackendConfig` La CR viene eliminata, passa allo stato di eliminazione.
  - Se sul backend non sono presenti PVC (Persistent Volume Request), l'eliminazione di `TridentBackendConfig` comporterà l'eliminazione del back-end e della CR da parte di Trident `TridentBackendConfig`.
  - Se uno o più PVC sono presenti sul backend, passa a uno stato di eliminazione. Il `TridentBackendConfig` Successivamente, la CR entra anche nella fase di eliminazione. Il backend e. `TridentBackendConfig` Vengono eliminati solo dopo l'eliminazione di tutti i PVC.
- `Lost`: Il backend associato a `TridentBackendConfig` La CR è stata eliminata accidentalmente o deliberatamente e il `TridentBackendConfig` CR ha ancora un riferimento al backend cancellato. Il `TridentBackendConfig` La CR può comunque essere eliminata indipendentemente da `deletionPolicy` valore.
- `Unknown`: Trident non è in grado di determinare lo stato o l'esistenza del backend associato al `TridentBackendConfig` CR. Ad esempio, se il server API non risponde o se manca il `tridentbackends.trident.netapp.io` CRD. Ciò potrebbe richiedere l'intervento dell'utente.

In questa fase, viene creato un backend. È possibile gestire anche diverse operazioni, ad esempio "aggiornamenti back-end ed eliminazioni back-end".

(Facoltativo) fase 4: Ulteriori informazioni

È possibile eseguire il seguente comando per ottenere ulteriori informazioni sul backend:

```
kubectl -n trident get tbc backend-tbc-ontap-san -o wide
```

```
NAME                     BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8   Bound   Success   ontap-san        delete
```

Inoltre, è possibile ottenere un dump YAML/JSON di `TridentBackendConfig`.

```
kubectl -n trident get tbc backend-tbc-ontap-san -o yaml
```

```yaml
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  creationTimestamp: 2021-04-21T20:45:11Z
  finalizers:
    - trident.netapp.io
  generation: 1
  name: backend-tbc-ontap-san
  namespace: trident
  resourceVersion: "947143"
  uid: 35b9d777-109f-43d5-8077-c74a4559d09c
spec:
  backendName: ontap-san-backend
  credentials:
    name: backend-tbc-ontap-san-secret
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  storageDriverName: ontap-san
  svm: trident_svm
  version: 1
status:
  backendInfo:
    backendName: ontap-san-backend
    backendUUID: 8d24fce7-6f60-4d4a-8ef6-bab2699e6ab8
  deletionPolicy: delete
  lastOperationStatus: Success
  message: Backend 'ontap-san-backend' created
  phase: Bound
```

`backendInfo` Contiene il `backendName` e il `backendUUID` del backend creato in risposta al `TridentBackendConfig` CR. Il `lastOperationStatus` campo rappresenta lo stato dell'ultima operazione del `TridentBackendConfig` CR, che può essere attivata dall'utente (ad esempio, un elemento modificato dall'utente in) o attivata da Trident (ad esempio, `spec` durante il riavvio di Trident). Può essere riuscito o non riuscito. `phase` Rappresenta lo stato della relazione tra `TridentBackendConfig` CR e backend. Nell'esempio precedente, `phase` ha il valore associato, il che significa che la `TridentBackendConfig` CR è associata al backend.

È possibile eseguire `kubectl -n trident describe tbc <tbc-cr-name>` per ottenere i dettagli dei registri degli eventi.

> Non è possibile aggiornare o eliminare un backend che contiene un associato `TridentBackendConfig` utilizzo di oggetti `tridentctl`. Comprendere le fasi necessarie per passare da un'operazione all'altra `tridentctl` e. `TridentBackendConfig`, "vedi qui".

Gestire i backend

Eseguire la gestione del back-end con kubectl

Scopri come eseguire operazioni di gestione back-end utilizzando `kubectl`.

Eliminare un backend

Eliminando un `TridentBackendConfig`, si ordina a Trident di eliminare/conservare i backend (in base a `deletionPolicy`). Per eliminare un backend, assicurarsi che `deletionPolicy` sia impostato su Elimina. Per eliminare solo il `TridentBackendConfig`, assicurarsi che `deletionPolicy` sia impostato su Mantieni. In questo modo si garantisce che il backend sia ancora presente e che possa essere gestito utilizzando `tridentctl`.

Eseguire il seguente comando:

```
kubectl delete tbc <tbc-name> -n trident
```

Trident non elimina i segreti di Kubernetes utilizzati da `TridentBackendConfig`. L'utente Kubernetes è responsabile della pulizia dei segreti. Prestare attenzione quando si eliminano i segreti. È necessario eliminare i segreti solo se non vengono utilizzati dai backend.

Visualizzare i backend esistenti

Eseguire il seguente comando:

```
kubectl get tbc -n trident
```

Puoi anche correre `tridentctl get backend -n trident` oppure `tridentctl get backend -o yaml -n trident` per ottenere un elenco di tutti i backend esistenti. Questo elenco includerà anche i backend creati con `tridentctl`.

Aggiornare un backend

Possono esserci diversi motivi per aggiornare un backend:

- Le credenziali del sistema storage sono state modificate. Per aggiornare le credenziali, è necessario aggiornare il segreto Kubernetes utilizzato nell' `TridentBackendConfig`oggetto. Trident aggiornerà automaticamente il backend con le credenziali più recenti fornite. Eseguire il seguente comando per aggiornare Kubernetes Secret:

  ```
  kubectl apply -f <updated-secret-file.yaml> -n trident
  ```

- È necessario aggiornare i parametri (ad esempio il nome della SVM ONTAP utilizzata).
  - È possibile eseguire l'aggiornamento `TridentBackendConfig` Oggetti direttamente tramite Kubernetes usando il seguente comando:

```
kubectl apply -f <updated-backend-file.yaml>
```

- In alternativa, è possibile apportare modifiche all'esistente `TridentBackendConfig` CR utilizzando il seguente comando:

```
kubectl edit tbc <tbc-name> -n trident
```

> - Se un aggiornamento back-end non riesce, il back-end continua a rimanere nella sua ultima configurazione nota. È possibile visualizzare i log per determinare la causa eseguendo `kubectl get tbc <tbc-name> -o yaml -n trident` oppure `kubectl describe tbc <tbc-name> -n trident`.
> - Dopo aver identificato e corretto il problema con il file di configurazione, è possibile eseguire nuovamente il comando update.

Eseguire la gestione back-end con tridentctl

Scopri come eseguire operazioni di gestione back-end utilizzando `tridentctl`.

Creare un backend

Dopo aver creato un "file di configurazione back-end", eseguire il seguente comando:

```
tridentctl create backend -f <backend-file> -n trident
```

Se la creazione del back-end non riesce, si è verificato un errore nella configurazione del back-end. È possibile visualizzare i log per determinare la causa eseguendo il seguente comando:

```
tridentctl logs -n trident
```

Dopo aver identificato e corretto il problema con il file di configurazione, è possibile eseguire semplicemente `create` di nuovo comando.

Eliminare un backend

Per eliminare un backend da Trident, procedere come segue:

1. Recuperare il nome del backend:

   ```
   tridentctl get backend -n trident
   ```

2. Eliminare il backend:

```
tridentctl delete backend <backend-name> -n trident
```

> Se Trident ha eseguito il provisioning di volumi e Snapshot da questo backend che ancora esistono, l'eliminazione del backend impedisce il provisioning di nuovi volumi da parte dell'IT. Il backend continuerà ad esistere in uno stato di "eliminazione".

Visualizzare i backend esistenti

Per visualizzare i backend di cui Trident è a conoscenza, procedere come segue:

- Per ottenere un riepilogo, eseguire il seguente comando:

```
tridentctl get backend -n trident
```

- Per ottenere tutti i dettagli, eseguire il seguente comando:

```
tridentctl get backend -o json -n trident
```

Aggiornare un backend

Dopo aver creato un nuovo file di configurazione back-end, eseguire il seguente comando:

```
tridentctl update backend <backend-name> -f <backend-file> -n trident
```

Se l'aggiornamento del back-end non riesce, si è verificato un errore nella configurazione del back-end o si è tentato di eseguire un aggiornamento non valido. È possibile visualizzare i log per determinare la causa eseguendo il seguente comando:

```
tridentctl logs -n trident
```

Dopo aver identificato e corretto il problema con il file di configurazione, è possibile eseguire semplicemente `update` di nuovo comando.

Identificare le classi di storage che utilizzano un backend

Questo è un esempio del tipo di domande a cui puoi rispondere con il JSON che `tridentctl` output per oggetti backend. Viene utilizzato il `jq` che è necessario installare.

```
tridentctl get backend -o json | jq '[.items[] | {backend: .name,
storageClasses: [.storage[].storageClasses]|unique}]'
```

Questo vale anche per i backend creati con `TridentBackendConfig`.

Passare da un'opzione di gestione back-end all'altra

Scopri i diversi modi di gestire i backend in Trident.

Opzioni per la gestione dei backend

Con l'introduzione di `TridentBackendConfig`, gli amministratori dispongono ora di due metodi unici per gestire i back-end. Questo pone le seguenti domande:

- È possibile creare backend utilizzando `tridentctl` essere gestito con `TridentBackendConfig`?
- È possibile creare backend utilizzando `TridentBackendConfig` essere gestito con `tridentctl`?

Gestire `tridentctl` backend con `TridentBackendConfig`

In questa sezione vengono descritte le procedure necessarie per gestire i backend creati con `tridentctl` Direttamente attraverso l'interfaccia Kubernetes creando `TridentBackendConfig` oggetti.

Questo si applica ai seguenti scenari:

- Backend preesistenti, che non hanno un `TridentBackendConfig` perché sono stati creati con `tridentctl`.
- Nuovi backend creati con `tridentctl`, mentre altri `TridentBackendConfig` esistono oggetti.

In entrambi gli scenari, i backend continueranno a essere presenti, con Trident che pianifica i volumi e li utilizza. Gli amministratori possono scegliere tra due opzioni:

- Continuare a utilizzare `tridentctl` per gestire i back-end creati utilizzando l'it.
- Collegare i backend creati con `tridentctl` a un nuovo `TridentBackendConfig` oggetto. In questo modo, i backend verranno gestiti utilizzando `kubectl` e non `tridentctl`.

Per gestire un backend preesistente utilizzando `kubectl`, sarà necessario creare un `TridentBackendConfig` che si collega al back-end esistente. Ecco una panoramica sul funzionamento di questo sistema:

1. Crea un Kubernetes Secret. Il segreto contiene le credenziali di cui Trident ha bisogno per comunicare con il cluster/servizio di archiviazione.
2. Creare un `TridentBackendConfig` oggetto. Contiene specifiche relative al cluster/servizio di storage e fa riferimento al segreto creato nel passaggio precedente. È necessario specificare parametri di configurazione identici (ad esempio `spec.backendName`, `spec.storagePrefix`, `spec.storageDriverName`e così via). `spec.backendName` deve essere impostato sul nome del backend esistente.

Fase 0: Identificare il backend

Per creare un `TridentBackendConfig` che si collega a un backend esistente, sarà necessario ottenere la configurazione del backend. In questo esempio, supponiamo che sia stato creato un backend utilizzando la seguente definizione JSON:

```
tridentctl get backend ontap-nas-backend -n trident
+----------------------+----------------
+--------------------------------------+--------+---------+
|          NAME        | STORAGE DRIVER |                  UUID
| STATE  | VOLUMES |
+----------------------+----------------
+--------------------------------------+--------+---------+
| ontap-nas-backend   | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |      25 |
+----------------------+----------------
+--------------------------------------+--------+---------+
```

```
cat ontap-nas-backend.json
```

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.10.10.1",
  "dataLIF": "10.10.10.2",
  "backendName": "ontap-nas-backend",
  "svm": "trident_svm",
  "username": "cluster-admin",
  "password": "admin-password",
  "defaults": {
    "spaceReserve": "none",
    "encryption": "false"
  },
  "labels": {
    "store": "nas_store"
  },
  "region": "us_east_1",
  "storage": [
    {
      "labels": {
        "app": "msoffice",
        "cost": "100"
      },
      "zone": "us_east_1a",
      "defaults": {
        "spaceReserve": "volume",
        "encryption": "true",
        "unixPermissions": "0755"
      }
    },
    {
      "labels": {
        "app": "mysqldb",
        "cost": "25"
      },
      "zone": "us_east_1d",
      "defaults": {
        "spaceReserve": "volume",
        "encryption": "false",
        "unixPermissions": "0775"
      }
    }
  ]
}
```

Fase 1: Creare un Kubernetes Secret

Creare un Segreto contenente le credenziali per il backend, come illustrato in questo esempio:

```
cat tbc-ontap-nas-backend-secret.yaml
```

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: ontap-nas-backend-secret
type: Opaque
stringData:
  username: cluster-admin
  password: admin-password
```

```
kubectl create -f tbc-ontap-nas-backend-secret.yaml -n trident
secret/backend-tbc-ontap-san-secret created
```

Fase 2: Creare un `TridentBackendConfig` CR

Il passaggio successivo consiste nella creazione di un `TridentBackendConfig` CR che si associerà automaticamente al preesistente `ontap-nas-backend` (come in questo esempio). Assicurarsi che siano soddisfatti i seguenti requisiti:

- Lo stesso nome backend viene definito in `spec.backendName`.
- I parametri di configurazione sono identici al backend originale.
- I pool virtuali (se presenti) devono mantenere lo stesso ordine del backend originale.
- Le credenziali vengono fornite attraverso un Kubernetes Secret e non in testo normale.

In questo caso, il `TridentBackendConfig` avrà un aspetto simile al seguente:

```
cat backend-tbc-ontap-nas.yaml
```

```yaml
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-ontap-nas-backend
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.10.10.1
  dataLIF: 10.10.10.2
  backendName: ontap-nas-backend
  svm: trident_svm
  credentials:
    name: mysecret
  defaults:
    spaceReserve: none
    encryption: 'false'
  labels:
    store: nas_store
  region: us_east_1
  storage:
  - labels:
      app: msoffice
      cost: '100'
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: 'true'
      unixPermissions: '0755'
  - labels:
      app: mysqldb
      cost: '25'
    zone: us_east_1d
    defaults:
      spaceReserve: volume
      encryption: 'false'
      unixPermissions: '0775'
```

```
kubectl create -f backend-tbc-ontap-nas.yaml -n trident
tridentbackendconfig.trident.netapp.io/tbc-ontap-nas-backend created
```

Fase 3: Verificare lo stato di `TridentBackendConfig` CR

Dopo il `TridentBackendConfig` è stato creato, la sua fase deve essere `Bound`. Deve inoltre riflettere lo stesso nome e UUID del backend esistente.

```
kubectl get tbc tbc-ontap-nas-backend -n trident
NAME                   BACKEND NAME          BACKEND UUID
PHASE   STATUS
tbc-ontap-nas-backend  ontap-nas-backend     52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7   Bound   Success

#confirm that no new backends were created (i.e., TridentBackendConfig did
not end up creating a new backend)
tridentctl get backend -n trident
+---------------------+----------------
+--------------------------------------+--------+---------+
|          NAME       | STORAGE DRIVER |                  UUID
| STATE  | VOLUMES |
+---------------------+----------------
+--------------------------------------+--------+---------+
| ontap-nas-backend   | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |      25 |
+---------------------+----------------
+--------------------------------------+--------+---------+
```

Il back-end verrà ora completamente gestito utilizzando `tbc-ontap-nas-backend` `TridentBackendConfig` oggetto.

Gestire `TridentBackendConfig` backend con `tridentctl`

```
`tridentctl` può essere utilizzato per elencare i backend creati con
`TridentBackendConfig`. Inoltre, gli amministratori possono anche
scegliere di gestire completamente tali backend attraverso `tridentctl`
eliminando `TridentBackendConfig` e assicurandosi `spec.deletionPolicy` è
impostato su `retain`.
```

Fase 0: Identificare il backend

Ad esempio, supponiamo che il seguente backend sia stato creato utilizzando `TridentBackendConfig`:

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                    BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        delete

tridentctl get backend ontap-san-backend -n trident
+-------------------+----------------
+--------------------------------------+--------+---------+
|       NAME        | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+-------------------+----------------
+--------------------------------------+--------+---------+
| ontap-san-backend | ontap-san      | 81abcb27-ea63-49bb-b606-
0a5315ac5f82 | online |      33 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

Dall'output, si vede che `TridentBackendConfig` È stato creato correttamente ed è associato a un backend [osservare l'UUID del backend].

Fase 1: Confermare `deletionPolicy` è impostato su `retain`

Diamo un'occhiata al valore di `deletionPolicy`. Questo deve essere impostato su `retain`. In questo modo, quando si elimina un `TridentBackendConfig` CR, la definizione di backend sarà ancora presente e potrà essere gestita con `tridentctl`.

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                    BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        delete

# Patch value of deletionPolicy to retain
kubectl patch tbc backend-tbc-ontap-san --type=merge -p
'{"spec":{"deletionPolicy":"retain"}}' -n trident
tridentbackendconfig.trident.netapp.io/backend-tbc-ontap-san patched

#Confirm the value of deletionPolicy
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                    BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        retain
```

Non passare alla fase successiva a meno che `deletionPolicy` è impostato su `retain`.

Fase 2: Eliminare `TridentBackendConfig` CR

Il passaggio finale consiste nell'eliminare `TridentBackendConfig` CR. Dopo la conferma di `deletionPolicy` è impostato su `retain`, è possibile procedere con l'eliminazione:

```
kubectl delete tbc backend-tbc-ontap-san -n trident
tridentbackendconfig.trident.netapp.io "backend-tbc-ontap-san" deleted

tridentctl get backend ontap-san-backend -n trident
+-------------------+----------------
+--------------------------------------+--------+---------+
|       NAME        | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+-------------------+----------------
+--------------------------------------+--------+---------+
| ontap-san-backend | ontap-san      | 81abcb27-ea63-49bb-b606-
0a5315ac5f82 | online |      33 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

All'eliminazione dell' `TridentBackendConfig`oggetto, Trident lo rimuove semplicemente senza eliminare effettivamente il backend stesso.

Creare e gestire classi di archiviazione

Creare una classe di storage

Configurare un oggetto Kubernetes StorageClass e creare la classe storage per istruire Trident su come eseguire il provisioning dei volumi.

Configurare un oggetto Kubernetes StorageClass

```
https://kubernetes.io/docs/concepts/storage/storage-classes/["Oggetto
Kubernetes StorageClass"^]Identifica Trident come provisioner utilizzato
per quella classe e istruisce Trident su come effettuare il provisioning
di un volume. Ad esempio:
```

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
mountOptions:
  - nfsvers=3
  - nolock
parameters:
  backendType: "ontap-nas"
  media: "ssd"
allowVolumeExpansion: true
volumeBindingMode: Immediate
```

Per ulteriori informazioni sull'interazione delle classi di archiviazione con i `PersistentVolumeClaim` parametri e per il controllo del provisioning dei volumi da parte di Trident, fare riferimento "Kubernetes e Trident Objects"a.

Creare una classe di storage

Dopo aver creato l'oggetto StorageClass, è possibile creare la classe storage. Campioni di classe di conservazione fornisce alcuni esempi di base che è possibile utilizzare o modificare.

Fasi

1. Si tratta di un oggetto Kubernetes, lo utilizza `kubectl` Per crearlo in Kubernetes.

   ```
   kubectl create -f sample-input/storage-class-basic-csi.yaml
   ```

2. Ora dovresti vedere una classe storage **Basic-csi** in Kubernetes e Trident, e Trident dovrebbe aver scoperto i pool nel back-end.

   ```
   kubectl get sc basic-csi
   ```

```
NAME          PROVISIONER             AGE
basic-csi     csi.trident.netapp.io   15h
```

```
./tridentctl -n trident get storageclass basic-csi -o json
```

```
{
  "items": [
    {
      "Config": {
        "version": "1",
        "name": "basic-csi",
        "attributes": {
          "backendType": "ontap-nas"
        },
        "storagePools": null,
        "additionalStoragePools": null
      },
      "storage": {
        "ontapnas_10.0.0.1": [
          "aggr1",
          "aggr2",
          "aggr3",
          "aggr4"
        ]
      }
    }
  ]
}
```


Campioni di classe di conservazione

Trident fornisce "definizioni semplici delle classi di archiviazione per backend specifici".

In alternativa, è possibile modificare `sample-input/storage-class-csi.yaml.templ` file fornito con il programma di installazione e sostituirlo *`BACKEND_TYPE`* con il nome del driver di storage.

```
./tridentctl -n trident get backend
+-------------+----------------+-------------------------------------
+--------+---------+
|    NAME     | STORAGE DRIVER |                 UUID                |
STATE  | VOLUMES |
+-------------+----------------+-------------------------------------
+--------+---------+
| nas-backend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       0 |
+-------------+----------------+-------------------------------------
+--------+---------+

cp sample-input/storage-class-csi.yaml.templ sample-input/storage-class-
basic-csi.yaml

# Modify __BACKEND_TYPE__ with the storage driver field above (e.g.,
ontap-nas)
vi sample-input/storage-class-basic-csi.yaml
```

Gestire le classi di storage

È possibile visualizzare le classi di storage esistenti, impostare una classe di storage predefinita, identificare il backend della classe di storage ed eliminare le classi di storage.

Visualizzare le classi di storage esistenti

- Per visualizzare le classi di storage Kubernetes esistenti, eseguire il seguente comando:

```
kubectl get storageclass
```

- Per visualizzare i dettagli della classe storage Kubernetes, eseguire il seguente comando:

```
kubectl get storageclass <storage-class> -o json
```

- Per visualizzare le classi di archiviazione sincronizzata di Trident, eseguire il comando seguente:

```
tridentctl get storageclass
```

- Per visualizzare i dettagli della classe di archiviazione sincronizzata di Trident, eseguire il comando seguente:

```
tridentctl get storageclass <storage-class> -o json
```

Impostare una classe di storage predefinita

Kubernetes 1.6 ha aggiunto la possibilità di impostare una classe di storage predefinita. Si tratta della classe di storage che verrà utilizzata per eseguire il provisioning di un volume persistente se un utente non ne specifica uno in un PVC (Persistent Volume Claim).

- Definire una classe di storage predefinita impostando l'annotazione `storageclass.kubernetes.io/is-default-class` a true nella definizione della classe di storage. In base alla specifica, qualsiasi altro valore o assenza di annotazione viene interpretato come falso.
- È possibile configurare una classe di storage esistente come classe di storage predefinita utilizzando il seguente comando:

```
kubectl patch storageclass <storage-class-name> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

- Allo stesso modo, è possibile rimuovere l'annotazione predefinita della classe di storage utilizzando il seguente comando:

```
kubectl patch storageclass <storage-class-name> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"false"}}}'
```

Nel bundle del programma di installazione di Trident sono presenti anche alcuni esempi che includono questa annotazione.

> Nel cluster deve essere presente una sola classe di archiviazione predefinita alla volta. Kubernetes non impedisce tecnicamente di averne più di una, ma si comporta come se non ci fosse alcuna classe di storage predefinita.

Identificare il backend per una classe di storage

Questo è un esempio del tipo di domande a cui è possibile rispondere con il JSON che `tridentctl` emette per gli oggetti back-end Trident. In questo modo viene utilizzata l' `jq`utilità, che potrebbe essere necessario installare per prima.

```
tridentctl get storageclass -o json | jq  '[.items[] | {storageClass:
.Config.name, backends: [.storage]|unique}]'
```

Eliminare una classe di storage

Per eliminare una classe di storage da Kubernetes, eseguire il seguente comando:

```
kubectl delete storageclass <storage-class>
```

`<storage-class>` deve essere sostituito con la classe di storage.

Tutti i volumi persistenti creati tramite questa classe di storage non verranno toccati e Trident continuerà a

gestirli.

> Trident applica uno spazio vuoto `fsType` ai volumi che crea. Per i backend iSCSI, si consiglia di applicare `parameters.fsType` in StorageClass. È necessario eliminare gli StorageClasses esistenti e ricrearli con `parameters.fsType` specificato.

Provisioning e gestione dei volumi

Provisioning di un volume

Creare un PersistentVolumeClaim (PVC) che utilizzi Kubernetes StorageClass configurato per richiedere l'accesso al PV. È quindi possibile montare il PV su un pod.

Panoramica

Un "*PersistentVolumeClaim*" (PVC) è una richiesta di accesso a PersistentVolume sul cluster.

Il PVC può essere configurato per richiedere la memorizzazione di una determinata dimensione o modalità di accesso. Utilizzando StorageClass associato, l'amministratore del cluster può controllare più delle dimensioni di PersistentVolume e della modalità di accesso, ad esempio le prestazioni o il livello di servizio.

Dopo aver creato il PVC, è possibile montare il volume in un pod.

Creare il PVC

Fasi

1. Creare il PVC.

```
kubectl create -f pvc.yaml
```

2. Verificare lo stato del PVC.

```
kubectl get pvc
```

```
NAME        STATUS  VOLUME     CAPACITY   ACCESS MODES  STORAGECLASS AGE
pvc-storage Bound   pv-name    1Gi        RWO                        5m
```

1. Montare il volume in un pod.

```
kubectl create -f pv-pod.yaml
```

> È possibile monitorare l'avanzamento utilizzando `kubectl get pod --watch`.

2. Verificare che il volume sia montato su `/my/mount/path`.

```
kubectl exec -it task-pv-pod -- df -h /my/mount/path
```

3. A questo punto è possibile eliminare il pod. L'applicazione Pod non esisterà più, ma il volume rimarrà.

```
kubectl delete pod pv-pod
```

Manifesti campione

Manifesti di campioni PersistentVolumeClaim

Questi esempi mostrano le opzioni di configurazione di base del PVC.

PVC con accesso RWO

Questo esempio mostra un PVC di base con accesso RWO associato a un nome StorageClass `basic-csi`.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-storage
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: basic-csi
```

PVC con NVMe/TCP

Questo esempio mostra un PVC di base per NVMe/TCP con accesso RWO associato a una StorageClass denominata `protection-gold`.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: pvc-san-nvme
spec:
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: protection-gold
```

Campioni manifesti pod

Questi esempi mostrano le configurazioni di base per collegare il PVC a un pod.

Configurazione di base

```
kind: Pod
apiVersion: v1
metadata:
  name: pv-pod
spec:
  volumes:
    - name: storage
      persistentVolumeClaim:
       claimName: pvc-storage
  containers:
    - name: pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: storage
```

Configurazione NVMe/TCP di base

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-nginx
spec:
  volumes:
    - name: basic-pvc
      persistentVolumeClaim:
        claimName: pvc-san-nvme
  containers:
    - name: task-pv-container
      image: nginx
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: basic-pvc
```

Per ulteriori informazioni sull'interazione delle classi di archiviazione con i `PersistentVolumeClaim` parametri e per il controllo del provisioning dei volumi da parte di Trident, fare riferimento "Kubernetes e Trident Objects"a.

Espandere i volumi

Trident offre agli utenti Kubernetes la possibilità di espandere i propri volumi dopo che sono stati creati. Trova informazioni sulle configurazioni necessarie per espandere i volumi iSCSI, NFS e FC.

Espandere un volume iSCSI

È possibile espandere un volume persistente iSCSI (PV) utilizzando il provisioning CSI.

> L'espansione del volume iSCSI è supportata da `ontap-san`, `ontap-san-economy`, `solidfire-san` Driver e richiede Kubernetes 1.16 e versioni successive.

Fase 1: Configurare StorageClass per supportare l'espansione dei volumi

Modificare la definizione StorageClass per impostare `allowVolumeExpansion` campo a. `true`.

```
cat storageclass-ontapsan.yaml
```

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-san
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
allowVolumeExpansion: True
```

Per un StorageClass già esistente, modificarlo per includere `allowVolumeExpansion` parametro.

Fase 2: Creare un PVC con la StorageClass creata

Modificare la definizione PVC e aggiornare `spec.resources.requests.storage` per riflettere le nuove dimensioni desiderate, che devono essere superiori alle dimensioni originali.

```
cat pvc-ontapsan.yaml
```

```yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: san-pvc
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-san
```

Trident crea un volume persistente (PV) e lo associa a questa dichiarazione di volume persistente (PVC).

```
kubectl get pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi
RWO            ontap-san      8s

kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM             STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi        RWO
Delete           Bound    default/san-pvc   ontap-san               10s
```

Fase 3: Definire un pod che colleghi il PVC

Collegare il PV a un pod affinché venga ridimensionato. Esistono due scenari quando si ridimensiona un PV iSCSI:

- Se il PV è collegato a un pod, Trident espande il volume sul backend dello storage, esegue una nuova scansione del dispositivo e ridimensiona il file system.
- Quando si tenta di ridimensionare un PV non collegato, Trident espande il volume sul backend dello storage. Dopo aver associato il PVC a un pod, Trident esegue nuovamente la scansione del dispositivo e ridimensiona il file system. Kubernetes aggiorna quindi le dimensioni del PVC dopo il completamento dell'operazione di espansione.

In questo esempio, viene creato un pod che utilizza `san-pvc`.

```
 kubectl get pod
NAME         READY   STATUS    RESTARTS   AGE
ubuntu-pod   1/1     Running   0          65s

 kubectl describe pvc san-pvc
Name:          san-pvc
Namespace:     default
StorageClass:  ontap-san
Status:        Bound
Volume:        pvc-8a814d62-bd58-4253-b0d1-82f2885db671
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
               volume.beta.kubernetes.io/storage-provisioner:
csi.trident.netapp.io
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      1Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Mounted By:    ubuntu-pod
```

Fase 4: Espandere il PV

Per ridimensionare il PV creato da 1 Gi a 2 Gi, modificare la definizione PVC e aggiornare `spec.resources.requests.storage` A 2 Gi.

```
kubectl edit pvc san-pvc
```

```
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: "2019-10-10T17:32:29Z"
  finalizers:
  - kubernetes.io/pvc-protection
  name: san-pvc
  namespace: default
  resourceVersion: "16609"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/san-pvc
  uid: 8a814d62-bd58-4253-b0d1-82f2885db671
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
 # ...
```

Fase 5: Convalidare l’espansione

È possibile convalidare il corretto funzionamento dell’espansione controllando le dimensioni del PVC, del PV e del volume Trident:

```
kubectl get pvc san-pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi
RWO            ontap-san      11m
kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM             STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi        RWO
Delete           Bound    default/san-pvc   ontap-san               12m
tridentctl get volumes -n trident
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
| pvc-8a814d62-bd58-4253-b0d1-82f2885db671 | 2.0 GiB | ontap-san     |
block    | a9b7bfff-0505-4e31-b6c5-59f492e02d33 | online | true    |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
```

Espandere un volume FC

È possibile espandere un volume persistente FC (PV) utilizzando il provisioner CSI.

> L'espansione del volume FC è supportata dal `ontap-san` driver e richiede Kubernetes 1,16 e versioni successive.

Fase 1: Configurare StorageClass per supportare l'espansione dei volumi

Modificare la definizione StorageClass per impostare `allowVolumeExpansion` campo a. `true`.

```
cat storageclass-ontapsan.yaml
```

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-san
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
allowVolumeExpansion: True
```

Per un StorageClass già esistente, modificarlo per includere `allowVolumeExpansion` parametro.

Fase 2: Creare un PVC con la StorageClass creata

Modificare la definizione PVC e aggiornare `spec.resources.requests.storage` per riflettere le nuove dimensioni desiderate, che devono essere superiori alle dimensioni originali.

```
cat pvc-ontapsan.yaml
```

```yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: san-pvc
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-san
```

Trident crea un volume persistente (PV) e lo associa a questa dichiarazione di volume persistente (PVC).

```
kubectl get pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi
RWO            ontap-san      8s

kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM             STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi        RWO
Delete           Bound    default/san-pvc   ontap-san               10s
```

Fase 3: Definire un pod che colleghi il PVC

Collegare il PV a un pod affinché venga ridimensionato. Quando si ridimensiona un FV FC, esistono due scenari:

- Se il PV è collegato a un pod, Trident espande il volume sul backend dello storage, esegue una nuova scansione del dispositivo e ridimensiona il file system.
- Quando si tenta di ridimensionare un PV non collegato, Trident espande il volume sul backend dello storage. Dopo aver associato il PVC a un pod, Trident esegue nuovamente la scansione del dispositivo e ridimensiona il file system. Kubernetes aggiorna quindi le dimensioni del PVC dopo il completamento

dell'operazione di espansione.

In questo esempio, viene creato un pod che utilizza `san-pvc`.

```
 kubectl get pod
NAME         READY   STATUS    RESTARTS   AGE
ubuntu-pod   1/1     Running   0          65s

 kubectl describe pvc san-pvc
Name:          san-pvc
Namespace:     default
StorageClass:  ontap-san
Status:        Bound
Volume:        pvc-8a814d62-bd58-4253-b0d1-82f2885db671
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
               volume.beta.kubernetes.io/storage-provisioner:
csi.trident.netapp.io
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      1Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Mounted By:    ubuntu-pod
```

Fase 4: Espandere il PV

Per ridimensionare il PV creato da 1 Gi a 2 Gi, modificare la definizione PVC e aggiornare `spec.resources.requests.storage` A 2 Gi.

```
kubectl edit pvc san-pvc
```

```
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: "2019-10-10T17:32:29Z"
  finalizers:
  - kubernetes.io/pvc-protection
  name: san-pvc
  namespace: default
  resourceVersion: "16609"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/san-pvc
  uid: 8a814d62-bd58-4253-b0d1-82f2885db671
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
 # ...
```

Fase 5: Convalidare l’espansione

È possibile convalidare il corretto funzionamento dell’espansione controllando le dimensioni del PVC, del PV e del volume Trident:

```
kubectl get pvc san-pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi
RWO            ontap-san      11m
kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM             STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi        RWO
Delete           Bound    default/san-pvc   ontap-san               12m
tridentctl get volumes -n trident
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
| pvc-8a814d62-bd58-4253-b0d1-82f2885db671 | 2.0 GiB | ontap-san     |
block    | a9b7bfff-0505-4e31-b6c5-59f492e02d33 | online | true    |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
```

Espandere un volume NFS

Trident supporta l'espansione del volume per i PVS NFS forniti su `ontap-nas`, , `ontap-nas-economy` `ontap-nas-flexgroup` , `gcp-cvs` e `azure-netapp-files` backend.

Fase 1: Configurare StorageClass per supportare l'espansione dei volumi

Per ridimensionare un PV NFS, l'amministratore deve prima configurare la classe di storage per consentire l'espansione del volume impostando `allowVolumeExpansion` campo a. `true`:

```
cat storageclass-ontapnas.yaml
```

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontapnas
provisioner: csi.trident.netapp.io
parameters:
  backendType: ontap-nas
allowVolumeExpansion: true
```

Se è già stata creata una classe di storage senza questa opzione, è possibile modificare semplicemente la

classe di storage esistente utilizzando `kubectl edit storageclass` per consentire l'espansione del volume.

Fase 2: Creare un PVC con la StorageClass creata

```
cat pvc-ontapnas.yaml
```

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: ontapnas20mb
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 20Mi
  storageClassName: ontapnas
```

Trident dovrebbe creare un PV NFS da 20 MiB per questo PVC:

```
kubectl get pvc
NAME           STATUS   VOLUME
CAPACITY     ACCESS MODES   STORAGECLASS    AGE
ontapnas20mb   Bound    pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   20Mi
RWO            ontapnas        9s

kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM                  STORAGECLASS    REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   20Mi       RWO
Delete           Bound    default/ontapnas20mb   ontapnas
2m42s
```

Fase 3: Espandere il PV

Per ridimensionare il PV da 20 MiB appena creato a 1 GiB, modificare il PVC e impostare `spec.resources.requests.storage` a 1 GiB:

```
kubectl edit pvc ontapnas20mb
```

```
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: 2018-08-21T18:26:44Z
  finalizers:
  - kubernetes.io/pvc-protection
  name: ontapnas20mb
  namespace: default
  resourceVersion: "1958015"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/ontapnas20mb
  uid: c1bd7fa5-a56f-11e8-b8d7-fa163e59eaab
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
# ...
```

Fase 4: Convalidare l'espansione

È possibile convalidare il ridimensionamento corretto controllando le dimensioni del PVC, PV e del volume Trident:

```
kubectl get pvc ontapnas20mb
NAME           STATUS   VOLUME
CAPACITY   ACCESS MODES   STORAGECLASS    AGE
ontapnas20mb   Bound    pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   1Gi
RWO            ontapnas        4m44s

kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM                  STORAGECLASS    REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   1Gi        RWO
Delete           Bound    default/ontapnas20mb   ontapnas
5m35s

tridentctl get volume pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 -n trident
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
| pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 | 1.0 GiB | ontapnas      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | true    |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
```

Importa volumi

È possibile importare volumi di storage esistenti come PV Kubernetes utilizzando `tridentctl import`.

Panoramica e considerazioni

È possibile importare un volume in Trident in:

- Containerizzare un'applicazione e riutilizzare il set di dati esistente
- Utilizzare un clone di un set di dati per un'applicazione temporanea
- Ricostruire un cluster Kubernetes guasto
- Migrazione dei dati delle applicazioni durante il disaster recovery

Considerazioni

Prima di importare un volume, esaminare le seguenti considerazioni.

- Trident può importare solo volumi ONTAP di tipo RW (lettura-scrittura). I volumi di tipo DP (data Protection) sono volumi di destinazione SnapMirror. Interrompere la relazione di mirroring prima di importare il volume in Trident.

- Si consiglia di importare volumi senza connessioni attive. Per importare un volume utilizzato attivamente, clonare il volume ed eseguire l'importazione.

  > Ciò è particolarmente importante per i volumi a blocchi, in quanto Kubernetes non sarebbe a conoscenza della connessione precedente e potrebbe facilmente collegare un volume attivo a un pod. Ciò può causare il danneggiamento dei dati.

- Sebbene `StorageClass` debba essere specificato su un PVC, Trident non utilizza questo parametro durante l'importazione. Le classi di storage vengono utilizzate durante la creazione del volume per selezionare i pool disponibili in base alle caratteristiche dello storage. Poiché il volume esiste già, durante l'importazione non è richiesta alcuna selezione del pool. Pertanto, l'importazione non avrà esito negativo anche se il volume esiste in un backend o in un pool che non corrisponde alla classe di storage specificata nel PVC.
- La dimensione del volume esistente viene determinata e impostata nel PVC. Una volta importato il volume dal driver di storage, il PV viene creato con un ClaimRef sul PVC.
  - La policy di recupero viene inizialmente impostata su `retain` Nel PV. Dopo che Kubernetes ha eseguito il binding con PVC e PV, la policy di recupero viene aggiornata in modo da corrispondere alla policy di recupero della classe di storage.
  - Se il criterio di recupero della classe di storage è `delete`, Il volume di storage viene cancellato quando il PV viene cancellato.
- Per impostazione predefinita, Trident gestisce il PVC e rinomina FlexVol volume e LUN del backend. È possibile passare il `--no-manage` flag per importare un volume non gestito. Se si utilizza `--no-manage`, Trident non esegue alcuna operazione aggiuntiva sul PVC o sul PV per il ciclo di vita degli oggetti. Il volume di storage non viene cancellato quando il PV viene cancellato e vengono ignorate anche altre operazioni come il clone del volume e il ridimensionamento del volume.

  > Questa opzione è utile se si desidera utilizzare Kubernetes per carichi di lavoro containerizzati, ma altrimenti si desidera gestire il ciclo di vita del volume di storage al di fuori di Kubernetes.

- Al PVC e al PV viene aggiunta un'annotazione che serve a doppio scopo per indicare che il volume è stato importato e se il PVC e il PV sono gestiti. Questa annotazione non deve essere modificata o rimossa.

Importare un volume

È possibile utilizzare `tridentctl import` per importare un volume.

Fasi

1. Creare il file PVC (Persistent Volume Claim) (ad esempio, `pvc.yaml`) Che verrà utilizzato per creare il PVC. Il file PVC deve includere `name`, `namespace`, `accessModes`, e. `storageClassName`. In alternativa, è possibile specificare `unixPermissions` Nella definizione di PVC.

   Di seguito viene riportato un esempio di specifica minima:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: my_claim
  namespace: my_namespace
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: my_storage_class
```

> Non includere parametri aggiuntivi come il nome PV o le dimensioni del volume. Questo può causare l'errore del comando di importazione.

2. Utilizzare il `tridentctl import` comando per specificare il nome del backend Trident contenente il volume e il nome che identifica in modo univoco il volume sullo storage (ad esempio: ONTAP FlexVol, Element Volume, Cloud Volumes Service path). L' `-f`argomento è necessario per specificare il percorso del file PVC.

```
tridentctl import volume <backendName> <volumeName> -f <path-to-pvc-
file>
```

Esempi

Consultare i seguenti esempi di importazione di volumi per i driver supportati.

NAS ONTAP e NAS FlexGroup ONTAP

Trident supporta l'importazione dei volumi utilizzando `ontap-nas` driver e `ontap-nas-flexgroup` .

> - Il `ontap-nas-economy` il driver non può importare e gestire qtree.
> - Il `ontap-nas` e. `ontap-nas-flexgroup` i driver non consentono nomi di volumi duplicati.

Ogni volume creato con il `ontap-nas` driver è un FlexVol volume nel cluster ONTAP. L'importazione dei volumi FlexVol con il `ontap-nas` driver funziona allo stesso modo. È possibile importare come PVC i volumi FlexVol già presenti in un cluster ONTAP `ontap-nas`. Analogamente, i FlexGroup vol possono essere importati come `ontap-nas-flexgroup` PVC.

Esempi di NAS ONTAP

Di seguito viene illustrato un esempio di importazione di un volume gestito e di un volume non gestito.

Volume gestito

Nell'esempio seguente viene importato un volume denominato `managed_volume` su un backend denominato `ontap_nas`:

```
tridentctl import volume ontap_nas managed_volume -f <path-to-pvc-file>

+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
| pvc-bf5ad463-afbb-11e9-8d9f-5254004dfdb7 | 1.0 GiB | standard      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | true    |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
```

Volume non gestito

Quando si utilizza l' `--no-manage`argomento, Trident non rinomina il volume.

L'esempio seguente importa `unmanaged_volume` su `ontap_nas` back-end:

```
tridentctl import volume nas_blog unmanaged_volume -f <path-to-pvc-
file> --no-manage

+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
| pvc-df07d542-afbc-11e9-8d9f-5254004dfdb7 | 1.0 GiB | standard      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | false   |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
```

ONTAP SAN

Trident supporta l'importazione dei volumi utilizzando `ontap-san` driver e `ontap-san-economy` .

Trident può importare volumi FlexVol SAN di ONTAP che contengono una singola LUN. Questa operazione è coerente con il `ontap-san` driver, che crea una FlexVol volume per ogni PVC e un LUN all'interno della FlexVol volume. Trident importa il FlexVol volume e lo associa alla definizione PVC.

Esempi DI SAN ONTAP

Di seguito viene illustrato un esempio di importazione di un volume gestito e di un volume non gestito.

Volume gestito

Per i volumi gestiti, Trident rinomina FlexVol volume nel `pvc-<uuid>` formato e il LUN all'interno di FlexVol volume in `lun0`.

Nell'esempio seguente viene importato il `ontap-san-managed` FlexVol volume presente sul `ontap_san_default` backend:

```
tridentctl import volume ontapsan_san_default ontap-san-managed -f pvc-
basic-import.yaml -n trident -d

+------------------------------------------+--------+---------------
+----------+--------------------------------------+--------+---------+
|                   NAME                   |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+--------+---------------
+----------+--------------------------------------+--------+---------+
| pvc-d6ee4f54-4e40-4454-92fd-d00fc228d74a | 20 MiB | basic         |
block    | cd394786-ddd5-4470-adc3-10c5ce4ca757 | online | true    |
+------------------------------------------+--------+---------------
+----------+--------------------------------------+--------+---------+
```

Volume non gestito

L'esempio seguente importa `unmanaged_example_volume` su `ontap_san` back-end:

```
tridentctl import volume -n trident san_blog unmanaged_example_volume
-f pvc-import.yaml --no-manage
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
| pvc-1fc999c9-ce8c-459c-82e4-ed4380a4b228 | 1.0 GiB | san-blog      |
block    | e3275890-7d80-4af6-90cc-c7a0759f555a | online | false   |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
```

Se si dispone DI LUN mappati a igroups che condividono un IQN con un nodo Kubernetes IQN, come mostrato nell'esempio seguente, viene visualizzato l'errore: `LUN already mapped to initiator(s) in this group`. Per importare il volume, è necessario rimuovere l'iniziatore o annullare la mappatura del LUN.

```
Vserver   Igroup       Protocol OS Type  Initiators
--------- ------------ -------- -------- ------------------------------------
svm0      k8s-nodename.example.com-fe5d36f2-cded-4f38-9eb0-c7719fc2f9f3
                       iscsi    linux    iqn.1994-05.com.redhat:4c2e1cf35e0
svm0      unmanaged-example-igroup
                       mixed    linux    iqn.1994-05.com.redhat:4c2e1cf35e0
```

Elemento

Trident supporta il software NetApp Element e l'importazione di volumi NetApp HCI utilizzando il `solidfire-san` driver.

> Il driver Element supporta nomi di volumi duplicati. Tuttavia, Trident restituisce un errore se sono presenti nomi di volume duplicati. Come soluzione alternativa, clonare il volume, fornire un nome di volume univoco e importare il volume clonato.

Esempio di elemento

Nell'esempio seguente viene importato un `element-managed` volume sul back-end `element_default`.

```
tridentctl import volume element_default element-managed -f pvc-basic-
import.yaml -n trident -d

+------------------------------------------+--------+---------------
+----------+--------------------------------------+--------+---------+
|                   NAME                   |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+--------+---------------
+----------+--------------------------------------+--------+---------+
| pvc-970ce1ca-2096-4ecd-8545-ac7edc24a8fe | 10 GiB | basic-element |
block    | d3ba047a-ea0b-43f9-9c42-e38e58301c49 | online | true    |
+------------------------------------------+--------+---------------
+----------+--------------------------------------+--------+---------+
```

Piattaforma Google Cloud

Trident supporta l'importazione di volumi utilizzando il `gcp-cvs` driver.

> Per importare un volume supportato da NetApp Cloud Volumes Service in Google Cloud Platform, identificare il volume in base al relativo percorso. Il percorso del volume è la parte del percorso di esportazione del volume dopo `:/`. Ad esempio, se il percorso di esportazione è `10.0.0.1:/adroit-jolly-swift`, il percorso del volume è `adroit-jolly-swift`.

Esempio di piattaforma Google Cloud

Nell'esempio seguente viene importato un `gcp-cvs` volume sul back-end `gcpcvs_YEppr` con il percorso del volume di `adroit-jolly-swift`.

```
tridentctl import volume gcpcvs_YEppr adroit-jolly-swift -f <path-to-pvc-
file> -n trident

+------------------------------------------+--------+---------------
+----------+--------------------------------------+--------+---------+
|                   NAME                   |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+--------+---------------
+----------+--------------------------------------+--------+---------+
| pvc-a46ccab7-44aa-4433-94b1-e47fc8c0fa55 | 93 GiB | gcp-storage   | file
| e1a6e65b-299e-4568-ad05-4f0a105c888f | online | true    |
+------------------------------------------+--------+---------------
+----------+--------------------------------------+--------+---------+
```

Azure NetApp Files

Trident supporta l'importazione di volumi utilizzando il `azure-netapp-files` driver.

> Per importare un volume Azure NetApp Files, identificare il volume in base al relativo percorso. Il percorso del volume è la parte del percorso di esportazione del volume dopo `:/`. Ad esempio, se il percorso di montaggio è `10.0.0.2:/importvol1`, il percorso del volume è `importvol1`.

Esempio di Azure NetApp Files

Nell'esempio seguente viene importato un `azure-netapp-files` volume sul back-end `azurenetappfiles_40517` con il percorso del volume `importvol1`.

```
tridentctl import volume azurenetappfiles_40517 importvol1 -f <path-to-
pvc-file> -n trident

+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
| pvc-0ee95d60-fd5c-448d-b505-b72901b3a4ab | 100 GiB | anf-storage   |
file     | 1c01274f-d94b-44a3-98a3-04c953c9a51e | online | true    |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
```

Google Cloud NetApp Volumes

Trident supporta l'importazione di volumi utilizzando il `google-cloud-netapp-volumes` driver.

Esempio di Google Cloud NetApp Volumes

Nell'esempio seguente viene importato un `google-cloud-netapp-volumes` volume sul backend `backend-tbc-gcnv1` con il volume `testvoleasiaeast1`.

```
tridentctl import volume backend-tbc-gcnv1 "testvoleasiaeast1" -f < path-
to-pvc> -n trident

+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS
| PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
| pvc-a69cda19-218c-4ca9-a941-aea05dd13dc0 |  10 GiB | gcnv-nfs-sc-
identity | file     | 8c18cdf1-0770-4bc0-bcc5-c6295fe6d837 | online | true
|
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

Nell'esempio seguente viene importato un `google-cloud-netapp-volumes` volume quando nella stessa regione sono presenti due volumi:

```
tridentctl import volume backend-tbc-gcnv1
"projects/123456789100/locations/asia-east1-a/volumes/testvoleasiaeast1"
-f <path-to-pvc> -n trident

+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS
| PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
| pvc-a69cda19-218c-4ca9-a941-aea05dd13dc0 |  10 GiB | gcnv-nfs-sc-
identity | file     | 8c18cdf1-0770-4bc0-bcc5-c6295fe6d837 | online | true
|
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

Personalizzare i nomi e le etichette dei volumi

Con Trident, è possibile assegnare nomi e etichette significativi ai volumi creati. Questo ti aiuta a identificare e mappare facilmente i volumi alle rispettive risorse Kubernetes (PVC). È inoltre possibile definire modelli di backend per la creazione di nomi di volumi personalizzati ed etichette personalizzate; i volumi creati, importati o clonati aderiranno ai modelli.

Prima di iniziare

Nomi di volumi ed etichette personalizzabili supportano:

1. Operazioni di creazione, importazione e cloning del volume.
2. Nel caso del driver ontap-nas-Economy, solo il nome del volume Qtree soddisfa il modello del nome.
3. Nel caso del driver ontap-san-Economy, solo il nome LUN è conforme al modello del nome.

Limitazioni

1. I nomi dei volumi personalizzabili sono compatibili solo con i driver ONTAP on-premise.
2. I nomi dei volumi personalizzabili non si applicano ai volumi esistenti.

Comportamenti chiave dei nomi di volume personalizzabili

1. Se si verifica un errore a causa di una sintassi non valida in un modello di nome, la creazione del backend non riesce. Tuttavia, se l'applicazione modello non riesce, il volume verrà denominato in base alla convenzione di denominazione esistente.
2. Il prefisso di archiviazione non è applicabile quando un volume viene nominato utilizzando un modello di

nome dalla configurazione backend. Qualsiasi valore di prefisso desiderato può essere aggiunto direttamente al modello.

Esempi di configurazione backend con modello di nome ed etichette

I modelli con nomi personalizzati possono essere definiti a livello di root e/o pool.

Esempio di livello root

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "ontap-nfs-backend",
  "managementLIF": "<ip address>",
  "svm": "svm0",
  "username": "<admin>",
  "password": "<password>",
  "defaults": {
    "nameTemplate":
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.volume.Requ
estName}}"
  },
  "labels": {
    "cluster": "ClusterA",
    "PVC": "{{.volume.Namespace}}_{{.volume.RequestName}}"
  }
}
```

Esempio di livello pool

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "ontap-nfs-backend",
  "managementLIF": "<ip address>",
  "svm": "svm0",
  "username": "<admin>",
  "password": "<password>",
  "useREST": true,
  "storage": [
    {
      "labels": {
        "labelname": "label1",
        "name": "{{ .volume.Name }}"
      },
      "defaults": {
        "nameTemplate": "pool01_{{ .volume.Name }}_{{ .labels.cluster
}}_{{ .volume.Namespace }}_{{ .volume.RequestName }}"
      }
    },
    {
      "labels": {
        "cluster": "label2",
        "name": "{{ .volume.Name }}"
      },
      "defaults": {
        "nameTemplate": "pool02_{{ .volume.Name }}_{{ .labels.cluster
}}_{{ .volume.Namespace }}_{{ .volume.RequestName }}"
      }
    }
  ]
}
```

Esempi di modelli di nome

Esempio 1:

```
"nameTemplate": "{{ .config.StoragePrefix }}_{{ .volume.Name }}_{{
.config.BackendName }}"
```

Esempio 2:

```
"nameTemplate": "pool_{{ .config.StoragePrefix }}_{{ .volume.Name }}_{{
slice .volume.RequestName 1 5 }}""
```

Punti da considerare

1. Nel caso di importazioni di volumi, le etichette vengono aggiornate solo se il volume esistente presenta etichette in un formato specifico. Ad esempio: `{"provisioning":{"Cluster":"ClusterA", "PVC": "pvcname"}}`.
2. Nel caso di importazioni di volumi gestiti, il nome del volume segue il modello di nome definito al livello principale nella definizione di backend.
3. Trident non supporta l'uso di un operatore di sezione con il prefisso di memorizzazione.
4. Se i modelli non generano nomi di volume univoci, Trident aggiungerà alcuni caratteri casuali per creare nomi di volume univoci.
5. Se il nome personalizzato per un volume economico NAS supera i 64 caratteri di lunghezza, Trident denominerà i volumi in base alla convenzione di denominazione esistente. Per tutti gli altri driver ONTAP, se il nome del volume supera il limite del nome, il processo di creazione del volume non riesce.

Condividere un volume NFS tra spazi dei nomi

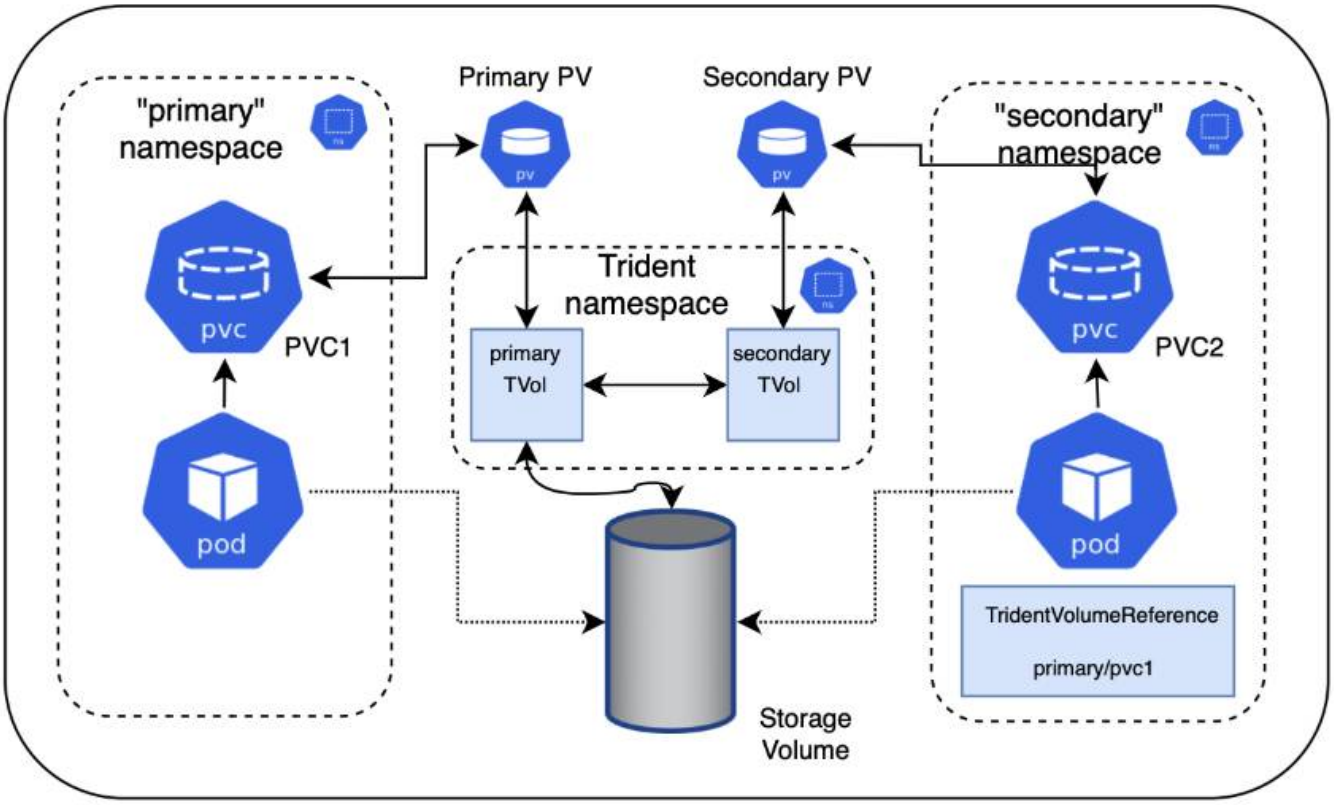
Utilizzando Trident, è possibile creare un volume in un namespace primario e condividerlo in uno o più namespace secondari.

Caratteristiche

TridentVolumeReference CR consente di condividere in modo sicuro i volumi NFS ReadWriteMany (RWX) in uno o più namespace Kubernetes. Questa soluzione nativa di Kubernetes offre i seguenti vantaggi:

- Diversi livelli di controllo degli accessi per garantire la sicurezza
- Funziona con tutti i driver di volume NFS Trident
- Nessuna dipendenza da tridentctl o da altre funzionalità Kubernetes non native

Questo diagramma illustra la condivisione del volume NFS tra due spazi dei nomi Kubernetes.



Avvio rapido

Puoi configurare la condivisione dei volumi NFS in pochi passaggi.

1. **Configurare il PVC di origine per la condivisione del volume**

Il proprietario dello spazio dei nomi di origine concede il permesso di accedere ai dati nel PVC di origine.

2. **Concedere il permesso di creare una CR nello spazio dei nomi di destinazione**

L'amministratore del cluster concede l'autorizzazione al proprietario dello spazio dei nomi di destinazione per creare la CR di TridentVolumeReference.

3. **Creare TridentVolumeReference nello spazio dei nomi di destinazione**

Il proprietario dello spazio dei nomi di destinazione crea la CR di TridentVolumeReference per fare riferimento al PVC di origine.

4. **Creare il PVC subordinato nello spazio dei nomi di destinazione**

Il proprietario dello spazio dei nomi di destinazione crea il PVC subordinato per utilizzare l'origine dati dal PVC di origine.

Configurare gli spazi dei nomi di origine e di destinazione

Per garantire la sicurezza, la condivisione di spazi dei nomi incrociati richiede la collaborazione e l'azione del proprietario dello spazio dei nomi di origine, dell'amministratore del cluster e del proprietario dello spazio dei nomi di destinazione. Il ruolo dell'utente viene designato in ogni fase.

Fasi

1. **Source namespace owner:** Crea il PVC (`pvc1`) nello spazio dei nomi di origine che concede l'autorizzazione per la condivisione con lo spazio dei nomi di destinazione (`namespace2`) utilizzando `shareToNamespace` annotazione.

```yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1
  namespace: namespace1
  annotations:
    trident.netapp.io/shareToNamespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

   Trident crea il PV e il suo volume di storage NFS di back-end.

   - È possibile condividere il PVC con più spazi dei nomi utilizzando un elenco delimitato da virgole. Ad esempio, `trident.netapp.io/shareToNamespace: namespace2,namespace3,namespace4`.
   - È possibile condividere con tutti gli spazi dei nomi utilizzando `*`. Ad esempio, `trident.netapp.io/shareToNamespace: *`
   - È possibile aggiornare il PVC per includere `shareToNamespace` annotazione in qualsiasi momento.

2. **Amministratore del cluster:** assicurarsi che sia presente il corretto RBAC per concedere l'autorizzazione al proprietario dello spazio dei nomi di destinazione per creare il CR TridentVolumeReference nello spazio dei nomi di destinazione.
3. **Destination namespace owner:** creare una CR di TridentVolumeReference nello spazio dei nomi di destinazione che si riferisce allo spazio dei nomi di origine `pvc1`.

```yaml
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. **Proprietario dello spazio dei nomi di destinazione:** Crea un PVC (`pvc2`) nello spazio dei nomi di destinazione (`namespace2`) utilizzando `shareFromPVC` Annotazione per indicare il PVC di origine.

```yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  annotations:
    trident.netapp.io/shareFromPVC: namespace1/pvc1
  name: pvc2
  namespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

La dimensione del PVC di destinazione deve essere inferiore o uguale al PVC di origine.

Risultati

Trident legge l' `shareFromPVC`annotazione sul PVC di destinazione e crea il PV di destinazione come volume subordinato senza una risorsa di storage propria che punta al PV di origine e condivide la risorsa di storage PV di origine. Il PVC e il PV di destinazione appaiono associati come normali.

Eliminare un volume condiviso

È possibile eliminare un volume condiviso tra più spazi dei nomi. Trident rimuoverà l'accesso al volume sul namespace di origine e manterrà l'accesso agli altri namespace che condividono il volume. Quando tutti gli spazi dei nomi che fanno riferimento al volume vengono rimossi, Trident elimina il volume.

Utilizzare `tridentctl get` per eseguire query sui volumi subordinati

Utilizzando il[`tridentctl` è possibile eseguire `get` comando per ottenere volumi subordinati. Per ulteriori informazioni, fare riferimento al `tridentctl` comandi e opzioni.

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

Allarmi:

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

Limitazioni

- Trident non può impedire la scrittura degli spazi dei nomi di destinazione nel volume condiviso. È necessario utilizzare il blocco dei file o altri processi per impedire la sovrascrittura dei dati dei volumi condivisi.
- Non è possibile revocare l'accesso al PVC di origine rimuovendo `shareToNamespace` oppure `shareFromNamespace` annotazioni o eliminazione di `TridentVolumeReference` CR. Per revocare l'accesso, è necessario eliminare il PVC subordinato.
- Snapshot, cloni e mirroring non sono possibili sui volumi subordinati.

Per ulteriori informazioni

Per ulteriori informazioni sull'accesso ai volumi tra spazi dei nomi:

- Visitare il sito "Condivisione di volumi tra spazi dei nomi: Dai il benvenuto all'accesso a volumi tra spazi dei nomi".
- Guarda la demo su "NetAppTV".

Clona i volumi tra namespace

Utilizzando Trident, puoi creare nuovi volumi utilizzando volumi esistenti o volumesnapshot da un namespace diverso all'interno dello stesso cluster Kubernetes.

Prerequisiti

Prima di clonare i volumi, verificare che i backend di origine e di destinazione siano dello stesso tipo e abbiano la stessa classe di storage.

> La clonazione tra spazi dei nomi è supportata solo per `ontap-san` E `ontap-nas` driver di archiviazione. I cloni di sola lettura non sono supportati.

Avvio rapido

Il cloning dei volumi può essere configurato in pochi passaggi.

1. **Configurare il PVC di origine per clonare il volume**

Il proprietario dello spazio dei nomi di origine concede il permesso di accedere ai dati nel PVC di origine.

2 **Concedere il permesso di creare una CR nello spazio dei nomi di destinazione**

L'amministratore del cluster concede l'autorizzazione al proprietario dello spazio dei nomi di destinazione per creare la CR di TridentVolumeReference.

3 **Creare TridentVolumeReference nello spazio dei nomi di destinazione**

Il proprietario dello spazio dei nomi di destinazione crea la CR di TridentVolumeReference per fare riferimento al PVC di origine.

4 **Creare il PVC clone nello spazio dei nomi di destinazione**

Il proprietario dello spazio dei nomi di destinazione crea PVC per clonare il PVC dallo spazio dei nomi di origine.

Configurare gli spazi dei nomi di origine e di destinazione

Per garantire la sicurezza, il cloning dei volumi negli spazi dei nomi richiede collaborazione e azione da parte del proprietario dello spazio dei nomi di origine, dell'amministratore del cluster e del proprietario dello spazio dei nomi di destinazione. Il ruolo dell'utente viene designato in ogni fase.

Fasi

1. **Proprietario dello spazio dei nomi di origine:** creare il PVC (`pvc1`nello spazio dei (`namespace1`nomi di origine) che concede il permesso di condividere con lo spazio dei nomi di destinazione (`namespace2`) utilizzando l' `cloneToNamespace`annotazione.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1
  namespace: namespace1
  annotations:
    trident.netapp.io/cloneToNamespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

Trident crea il PV e il suo volume di storage di backend.

- È possibile condividere il PVC con più spazi dei nomi utilizzando un elenco delimitato da virgole. Ad esempio, `trident.netapp.io/cloneToNamespace: namespace2,namespace3,namespace4`.
- È possibile condividere tutti gli spazi dei nomi utilizzando `*`. Ad esempio, `trident.netapp.io/cloneToNamespace: *`
- È possibile aggiornare il PVC per includere l' `cloneToNamespace`annotazione in qualsiasi momento.

2. **Amministratore del cluster:** assicurarsi che sia presente il corretto RBAC per concedere l'autorizzazione al proprietario dello spazio dei nomi di destinazione per creare il CR TridentVolumeReference nello spazio dei nomi di destinazione(`namespace2`).
3. **Destination namespace owner:** creare una CR di TridentVolumeReference nello spazio dei nomi di destinazione che si riferisce allo spazio dei nomi di origine `pvc1`.

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. **Proprietario dello spazio dei nomi di destinazione:** creare un PVC (`namespace2`)(`pvc2` nello spazio dei nomi di destinazione utilizzando la `cloneFromPVC` o `cloneFromSnapshot`, e `cloneFromNamespace` le annotazioni per designare il PVC di origine.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  annotations:
    trident.netapp.io/cloneFromPVC: pvc1
    trident.netapp.io/cloneFromNamespace: namespace1
  name: pvc2
  namespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

Limitazioni

- Per i PVC forniti utilizzando driver ONTAP-nas-Economy, i cloni di sola lettura non sono supportati.

Replica dei volumi con SnapMirror

Trident supporta le relazioni di mirroring tra un volume di origine su un cluster e il volume di destinazione sul cluster in peering per la replica dei dati per il disaster recovery. È possibile utilizzare una definizione di risorsa personalizzata (CRD) con namespace, denominata Trident Mirror Relationship (TMR), per eseguire le seguenti operazioni:

- Creare relazioni di mirroring tra volumi (PVC)
- Rimuovere le relazioni di mirroring tra volumi
- Interrompere le relazioni di mirroring
- Promozione del volume secondario in condizioni di disastro (failover)
- Eseguire la transizione senza perdita di dati delle applicazioni da cluster a cluster (durante failover o migrazioni pianificati)

Prerequisiti per la replica

Prima di iniziare, verificare che siano soddisfatti i seguenti prerequisiti:

Cluster ONTAP

- **Trident**: Trident versione 22,10 o successiva deve esistere su entrambi i cluster Kubernetes di origine e di destinazione che utilizzano ONTAP come backend.
- **Licenze**: Le licenze asincrone di ONTAP SnapMirror che utilizzano il bundle di protezione dati devono essere attivate sia sul cluster ONTAP di origine che su quello di destinazione. Per ulteriori informazioni, fare riferimento "Panoramica sulle licenze SnapMirror in ONTAP" a.

  A partire da ONTAP 9.10.1, tutte le licenze vengono fornite come file di licenza NetApp (NLF), che è un singolo file che abilita più funzioni. Per ulteriori informazioni, fare riferimento "Licenze incluse con ONTAP ONE" a.

  > È supportata solo la protezione asincrona SnapMirror.

Peering

- **Cluster e SVM**: I backend dello storage ONTAP devono essere peering. Per ulteriori informazioni, fare riferimento "Panoramica del peering di cluster e SVM" a.

  > Assicurati che i nomi delle SVM utilizzati nella relazione di replica tra due cluster ONTAP siano univoci.

- **Trident e SVM**: Le SVM remote in peering devono essere disponibili per Trident nel cluster di destinazione.

Driver supportati

NetApp Trident supporta la replicazione dei volumi con la tecnologia NetApp SnapMirror utilizzando classi di archiviazione supportate dai seguenti driver: **`ontap-nas`** : **NFS** `ontap-san` : iSCSI **`ontap-san`** : **FC** `ontap-san` : NVMe/TCP (richiede almeno la versione ONTAP 9.15.1)

La replicazione dei volumi tramite SnapMirror non è supportata per i sistemi ASA r2. Per informazioni sui sistemi ASA r2, vedere "Informazioni sui sistemi di storage ASA R2" .

Creare un PVC specchiato

Seguire questi passaggi e utilizzare gli esempi CRD per creare una relazione di mirroring tra volumi primari e secondari.

Fasi

1. Eseguire i seguenti passaggi sul cluster Kubernetes primario:
   a. Creare un oggetto StorageClass con il `trident.netapp.io/replication: true` parametro.

      Esempio

      ```
      apiVersion: storage.k8s.io/v1
      kind: StorageClass
      metadata:
        name: csi-nas
      provisioner: csi.trident.netapp.io
      parameters:
        backendType: "ontap-nas"
        fsType: "nfs"
        trident.netapp.io/replication: "true"
      ```

   b. Crea un PVC con StorageClass creato in precedenza.

      Esempio

      ```
      kind: PersistentVolumeClaim
      apiVersion: v1
      metadata:
        name: csi-nas
      spec:
        accessModes:
        - ReadWriteMany
        resources:
          requests:
            storage: 1Gi
        storageClassName: csi-nas
      ```

   c. Creare una CR MirrorRelationship con informazioni locali.

Esempio

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: promoted
  volumeMappings:
  - localPVCName: csi-nas
```

Trident recupera le informazioni interne per il volume e lo stato di protezione dei dati (DP) corrente del volume, quindi compila il campo di stato di MirrorRelationship.

d. Procurarsi il TridentMirrorRelationship CR per ottenere il nome interno e la SVM del PVC.

```
kubectl get tmr csi-nas
```

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
  generation: 1
spec:
  state: promoted
  volumeMappings:
  - localPVCName: csi-nas
status:
  conditions:
  - state: promoted
    localVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
    localPVCName: csi-nas
    observedGeneration: 1
```

2. Eseguire i seguenti passaggi sul cluster Kubernetes secondario:

   a. Creare una classe StorageClass con il parametro trident.netapp.io/replication: true.

Esempio

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-nas
provisioner: csi.trident.netapp.io
parameters:
  trident.netapp.io/replication: true
```

b. Creare una CR MirrorRelationship con informazioni sulla destinazione e sulla sorgente.

Esempio

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: established
  volumeMappings:
  - localPVCName: csi-nas
    remoteVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
```

Trident creerà una relazione SnapMirror con il nome del criterio di relazione configurato (o predefinito per ONTAP) e la inizializzerà.

c. Crea un PVC con StorageClass creato in precedenza per agire come secondario (destinazione SnapMirror).

Esempio

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
  annotations:
    trident.netapp.io/mirrorRelationship: csi-nas
spec:
  accessModes:
  - ReadWriteMany
resources:
  requests:
    storage: 1Gi
storageClassName: csi-nas
```

Trident verificherà la presenza del CRD TridentMirrorRelationship e non riuscirà a creare il volume se la relazione non esiste. Se la relazione esiste, Trident garantirà il posizionamento del nuovo FlexVol volume in una SVM a cui viene eseguito il peering con la SVM remota definita nella MirrorRelationship.

Stati di replica dei volumi

Una relazione mirror Trident (TMR) è un CRD che rappresenta un'estremità di una relazione di replica tra PVC. Il TMR di destinazione ha uno stato che indica a Trident lo stato desiderato. Il TMR di destinazione ha i seguenti stati:

- **Stabilito**: Il PVC locale è il volume di destinazione di una relazione speculare, e questa è una nuova relazione.
- **Promosso**: Il PVC locale è ReadWrite e montabile, senza alcuna relazione speculare attualmente in vigore.
- **Ristabilito**: Il PVC locale è il volume di destinazione di una relazione speculare ed era anche precedentemente in quella relazione speculare.
  - Lo stato ristabilito deve essere utilizzato se il volume di destinazione era in una relazione con il volume di origine perché sovrascrive il contenuto del volume di destinazione.
  - Se il volume non era precedentemente in relazione con l'origine, lo stato ristabilito non riuscirà.

Promozione del PVC secondario durante un failover non pianificato

Eseguire il seguente passaggio sul cluster Kubernetes secondario:

- Aggiornare il campo *spec.state* di TridentMirrorRelationship a `promoted`.

Promozione del PVC secondario durante un failover pianificato

Durante un failover pianificato (migrazione), eseguire le seguenti operazioni per promuovere il PVC secondario:

Fasi

1. Sul cluster Kubernetes primario, creare una snapshot del PVC e attendere la creazione dello snapshot.
2. Sul cluster Kubernetes primario, creare SnapshotInfo CR per ottenere dettagli interni.

   Esempio

   ```
   kind: SnapshotInfo
   apiVersion: trident.netapp.io/v1
   metadata:
     name: csi-nas
   spec:
     snapshot-name: csi-nas-snapshot
   ```

3. Nel cluster Kubernetes secondario, aggiornare il campo *spec.state* del *TridentMirrorRelationship* CR a *Promoted* e *spec.promotedSnapshotHandle* come nome interno dello snapshot.
4. Sul cluster Kubernetes secondario, confermare lo stato (campo status.state) di TridentMirrorRelationship a promosso.

Ripristinare una relazione di mirroring dopo un failover

Prima di ripristinare una relazione di specchiatura, scegliere il lato che si desidera creare come nuovo primario.

Fasi

1. Nel cluster Kubernetes secondario, verificare che i valori per il campo *spec.remoteVolumeHandle* in TridentMirrorRelationship siano aggiornati.
2. Sul cluster Kubernetes secondario, aggiornare il campo *spec.mirror* di TridentMirrorRelationship a `reestablished`.

Operazioni supplementari

Trident supporta le seguenti operazioni sui volumi primario e secondario:

Replicare il PVC primario in un nuovo PVC secondario

Assicurarsi di disporre già di un PVC primario e di un PVC secondario.

Fasi

1. Eliminare i CRD PersistentVolumeClaim e TridentMirrorRelationship dal cluster (destinazione) secondario stabilito.
2. Eliminare il CRD TridentMirrorRelationship dal cluster primario (origine).
3. Creare un nuovo CRD TridentMirrorRelationship nel cluster primario (di origine) per il nuovo PVC secondario (di destinazione) che si desidera stabilire.

Ridimensionare un PVC specchiato, primario o secondario

Il PVC può essere ridimensionato normalmente, ONTAP espanderà automaticamente qualsiasi flevxols di destinazione se la quantità di dati supera le dimensioni correnti.

Rimuovere la replica da un PVC

Per rimuovere la replica, eseguire una delle seguenti operazioni sul volume secondario corrente:

- Eliminare MirrorRelationship sul PVC secondario. Questo interrompe la relazione di replica.
- In alternativa, aggiornare il campo spec.state a *Promoted*.

Eliminazione di un PVC (precedentemente specchiato)

Trident verifica la presenza di PVC replicati e rilascia il rapporto di replica prima di tentare di eliminare il volume.

Eliminare una TMR

L'eliminazione di una TMR su un lato di una relazione specchiata fa sì che la TMR rimanente passi allo stato *promosso* prima che Trident completi l'eliminazione. Se la TMR selezionata per l'eliminazione è già nello stato *promosso*, non esiste alcuna relazione di mirroring e la TMR verrà rimossa e Trident promuoverà il PVC locale in *ReadWrite*. Questa eliminazione rilascia i metadati SnapMirror per il volume locale in ONTAP. Se in futuro questo volume viene utilizzato in una relazione di mirroring, deve utilizzare un nuovo TMR con uno stato di replica del volume *stabilito* quando si crea la nuova relazione di mirroring.

Aggiorna relazioni mirror quando ONTAP è online

Le relazioni speculari possono essere aggiornate in qualsiasi momento dopo che sono state stabilite. È possibile utilizzare i `state: promoted` campi o `state: reestablished` per aggiornare le relazioni. Quando si trasferisce un volume di destinazione a un volume ReadWrite regolare, è possibile utilizzare *PromotedSnapshotHandle* per specificare uno snapshot specifico su cui ripristinare il volume corrente.

Aggiorna relazioni di mirroring quando ONTAP non è in linea

Puoi utilizzare un CRD per eseguire un update del SnapMirror senza che Trident disponga di connettività diretta al cluster ONTAP. Fare riferimento al seguente formato di esempio di TridentActionMirrorUpdate:

Esempio

```
apiVersion: trident.netapp.io/v1
kind: TridentActionMirrorUpdate
metadata:
  name: update-mirror-b
spec:
  snapshotHandle: "pvc-1234/snapshot-1234"
  tridentMirrorRelationshipName: mirror-b
```

`status.state` Riflette lo stato del CRD TridentActionMirrorUpdate. Può assumere un valore da *riuscito*, *in corso* o *non riuscito*.

Utilizzare la topologia CSI

Trident può creare e collegare in modo selettivo i volumi ai nodi presenti in un cluster Kubernetes utilizzando "Funzionalità topologia CSI" .

Panoramica

Utilizzando la funzionalità topologia CSI, l'accesso ai volumi può essere limitato a un sottoinsieme di nodi, in base alle aree geografiche e alle zone di disponibilità. I provider di cloud oggi consentono agli amministratori di Kubernetes di generare nodi basati su zone. I nodi possono essere collocati in diverse zone di disponibilità all'interno di una regione o in diverse regioni. Per facilitare il provisioning dei volumi per i carichi di lavoro in un'architettura multi-zona, Trident utilizza la topologia CSI.

> Scopri di più sulla funzionalità topologia CSI "qui".

Kubernetes offre due esclusive modalità di binding del volume:

- Con `VolumeBindingMode` impostato su `Immediate`, Trident crea il volume senza alcuna conoscenza della topologia. Il binding dei volumi e il provisioning dinamico vengono gestiti quando viene creato il PVC. Questa è l'impostazione predefinita `VolumeBindingMode` ed è adatta per i cluster che non applicano vincoli di topologia. I volumi persistenti vengono creati senza alcuna dipendenza dai requisiti di pianificazione del pod richiedente.
- Con `VolumeBindingMode` impostare su `WaitForFirstConsumer`, La creazione e il binding di un volume persistente per un PVC viene ritardata fino a quando un pod che utilizza il PVC viene pianificato e creato. In questo modo, i volumi vengono creati per soddisfare i vincoli di pianificazione imposti dai requisiti di topologia.

Il `WaitForFirstConsumer` la modalità di binding non richiede etichette di topologia. Questo può essere utilizzato indipendentemente dalla funzionalità topologia CSI.

Di cosa hai bisogno

Per utilizzare la topologia CSI, è necessario disporre di quanto segue:

- Un cluster Kubernetes che esegue un "Versione Kubernetes supportata"

```
kubectl version
Client Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:50:19Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:41:49Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
```

- I nodi nel cluster devono avere etichette che introducano la conoscenza della topologia (`topology.kubernetes.io/region`e `topology.kubernetes.io/zone`). Queste etichette **devono essere presenti sui nodi nel cluster** prima che Trident venga installato affinché Trident sia in grado di riconoscere la topologia.

```
kubectl get nodes -o=jsonpath='{range .items[*]}[{.metadata.name},
{.metadata.labels}]{"\n"}{end}' | grep --color "topology.kubernetes.io"
[node1,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node1","kubernetes.io/
os":"linux","node-
role.kubernetes.io/master":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-a"}]
[node2,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node2","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-b"}]
[node3,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node3","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-c"}]
```

Fase 1: Creazione di un backend compatibile con la topologia

I backend di storage Trident possono essere progettati per eseguire il provisioning selettivo dei volumi in base alle zone di disponibilità. Ogni backend può portare un blocco opzionale `supportedTopologies` che rappresenta un elenco di zone e regioni supportate. Per StorageClasses che utilizzano tale backend, un volume viene creato solo se richiesto da un'applicazione pianificata in una regione/zona supportata.

Ecco un esempio di definizione di backend:

YAML

```yaml
---
version: 1
storageDriverName: ontap-san
backendName: san-backend-us-east1
managementLIF: 192.168.27.5
svm: iscsi_svm
username: admin
password: password
supportedTopologies:
  - topology.kubernetes.io/region: us-east1
    topology.kubernetes.io/zone: us-east1-a
  - topology.kubernetes.io/region: us-east1
    topology.kubernetes.io/zone: us-east1-b
```

JSON

```json
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "san-backend-us-east1",
  "managementLIF": "192.168.27.5",
  "svm": "iscsi_svm",
  "username": "admin",
  "password": "password",
  "supportedTopologies": [
    {
      "topology.kubernetes.io/region": "us-east1",
      "topology.kubernetes.io/zone": "us-east1-a"
    },
    {
      "topology.kubernetes.io/region": "us-east1",
      "topology.kubernetes.io/zone": "us-east1-b"
    }
  ]
}
```

`supportedTopologies` viene utilizzato per fornire un elenco di aree e zone per backend. Queste regioni e zone rappresentano l'elenco dei valori consentiti che possono essere forniti in una StorageClass. Per StorageClasses che contengono un sottoinsieme delle regioni e delle zone fornite in un backend, Trident crea un volume sul backend.

È possibile definire `supportedTopologies` anche per pool di storage. Vedere il seguente esempio:

```yaml
---
version: 1
storageDriverName: ontap-nas
backendName: nas-backend-us-central1
managementLIF: 172.16.238.5
svm: nfs_svm
username: admin
password: password
supportedTopologies:
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-a
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-b
storage:
  - labels:
      workload: production
    supportedTopologies:
      - topology.kubernetes.io/region: us-central1
        topology.kubernetes.io/zone: us-central1-a
  - labels:
      workload: dev
    supportedTopologies:
      - topology.kubernetes.io/region: us-central1
        topology.kubernetes.io/zone: us-central1-b
```

In questo esempio, il `region` e. `zone` le etichette indicano la posizione del pool di storage. `topology.kubernetes.io/region` e. `topology.kubernetes.io/zone` stabilire da dove possono essere consumati i pool di storage.

Fase 2: Definire StorageClasses che siano compatibili con la topologia

In base alle etichette della topologia fornite ai nodi del cluster, è possibile definire StorageClasses in modo da contenere informazioni sulla topologia. In questo modo verranno determinati i pool di storage che fungono da candidati per le richieste PVC effettuate e il sottoinsieme di nodi che possono utilizzare i volumi forniti da Trident.

Vedere il seguente esempio:

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata: null
name: netapp-san-us-east1
provisioner: csi.trident.netapp.io
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
  - matchLabelExpressions: null
  - key: topology.kubernetes.io/zone
    values:
      - us-east1-a
      - us-east1-b
  - key: topology.kubernetes.io/region
    values:
      - us-east1
parameters:
  fsType: ext4
```

Nella definizione StorageClass fornita sopra, `volumeBindingMode` è impostato su `WaitForFirstConsumer`. I PVC richiesti con questa classe di storage non verranno utilizzati fino a quando non saranno referenziati in un pod. E, `allowedTopologies` fornisce le zone e la regione da utilizzare. `netapp-san-us-east1`StorageClass crea PVC sul `san-backend-us-east1` backend definito sopra.

Fase 3: Creare e utilizzare un PVC

Con StorageClass creato e mappato a un backend, è ora possibile creare PVC.

Vedere l'esempio `spec` sotto:

```yaml
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata: null
name: pvc-san
spec: null
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: netapp-san-us-east1
```

La creazione di un PVC utilizzando questo manifesto comporta quanto segue:

```
kubectl create -f pvc.yaml
persistentvolumeclaim/pvc-san created
kubectl get pvc
NAME      STATUS    VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS
AGE
pvc-san   Pending                                      netapp-san-us-east1
2s
kubectl describe pvc
Name:          pvc-san
Namespace:     default
StorageClass:  netapp-san-us-east1
Status:        Pending
Volume:
Labels:        <none>
Annotations:   <none>
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:
Access Modes:
VolumeMode:    Filesystem
Mounted By:    <none>
Events:
  Type    Reason                Age   From                         Message
  ----    ------                ----  ----                         -------
  Normal  WaitForFirstConsumer  6s    persistentvolume-controller  waiting
for first consumer to be created before binding
```

Affinché Trident crei un volume e lo leghi al PVC, utilizza il PVC in un pod. Vedere il seguente esempio:

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: app-pod-1
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: topology.kubernetes.io/region
            operator: In
            values:
            - us-east1
      preferredDuringSchedulingIgnoredDuringExecution:
      - weight: 1
        preference:
          matchExpressions:
          - key: topology.kubernetes.io/zone
            operator: In
            values:
            - us-east1-a
            - us-east1-b
  securityContext:
    runAsUser: 1000
    runAsGroup: 3000
    fsGroup: 2000
  volumes:
  - name: vol1
    persistentVolumeClaim:
      claimName: pvc-san
  containers:
  - name: sec-ctx-demo
    image: busybox
    command: [ "sh", "-c", "sleep 1h" ]
    volumeMounts:
    - name: vol1
      mountPath: /data/demo
    securityContext:
      allowPrivilegeEscalation: false
```

Questo podSpec indica a Kubernetes di pianificare il pod sui nodi presenti in `us-east1` e scegliere tra i nodi presenti in `us-east1-a` oppure `us-east1-b` zone.

Vedere il seguente output:

```
kubectl get pods -o wide
NAME        READY   STATUS    RESTARTS   AGE   IP               NODE
NOMINATED NODE   READINESS GATES
app-pod-1   1/1     Running   0          19s   192.168.25.131   node2
<none>           <none>
kubectl get pvc -o wide
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS          AGE   VOLUMEMODE
pvc-san   Bound    pvc-ecb1e1a0-840c-463b-8b65-b3d033e2e62b   300Mi
RWO            netapp-san-us-east1   48s   Filesystem
```

Aggiorna i back-end da includere `supportedTopologies`

I backend preesistenti possono essere aggiornati per includere un elenco di `supportedTopologies` utilizzo di `tridentctl backend update`. Ciò non influisce sui volumi già sottoposti a provisioning e verrà utilizzato solo per i PVC successivi.

Trova ulteriori informazioni

- "Gestire le risorse per i container"
- "NodeSelector"
- "Affinità e anti-affinità"
- "Contamini e pedaggi"

Lavorare con le istantanee

Le snapshot del volume di Kubernetes dei volumi persistenti (PVS) consentono copie point-in-time dei volumi. Puoi creare una snapshot di un volume creato utilizzando Trident, importare uno snapshot creato al di fuori di Trident, creare un nuovo volume da una snapshot esistente e recuperare i dati del volume da snapshot.

Panoramica

L'istantanea del volume è supportata da `ontap-nas` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy` , `solidfire-san` , `gcp-cvs` , `azure-netapp-files` , E `google-cloud-netapp-volumes` conducenti.

Prima di iniziare

Per utilizzare gli snapshot, è necessario disporre di un controller snapshot esterno e di CRD (Custom Resource Definitions). Questa è la responsabilità del Kubernetes orchestrator (ad esempio: Kubeadm, GKE, OpenShift).

Se la distribuzione Kubernetes non include il controller di snapshot e i CRD, fare riferimento a. Implementare un controller per lo snapshot dei volumi.

> Non creare un controller di snapshot se si creano snapshot di volumi on-demand in un ambiente GKE. GKE utilizza un controller di snapshot integrato e nascosto.

Creare un'istantanea del volume

Fasi

1. Creare un `VolumeSnapshotClass`. Per ulteriori informazioni, fare riferimento a. "VolumeSnapshotClass".
   - `driver`Indica il driver Trident CSI.
   - `deletionPolicy` può essere `Delete` oppure `Retain`. Quando è impostato su `Retain`, lo snapshot fisico sottostante sul cluster di storage viene conservato anche quando `VolumeSnapshot` oggetto eliminato.

     Esempio

     ```
     cat snap-sc.yaml
     ```

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

2. Creare un'istantanea di un PVC esistente.

   Esempi

   - Questo esempio crea un'istantanea di un PVC esistente.

     ```
     cat snap.yaml
     ```

     ```yaml
     apiVersion: snapshot.storage.k8s.io/v1
     kind: VolumeSnapshot
     metadata:
       name: pvc1-snap
     spec:
       volumeSnapshotClassName: csi-snapclass
       source:
         persistentVolumeClaimName: pvc1
     ```

   - Questo esempio crea un oggetto snapshot di volume per un PVC denominato `pvc1` e il nome dello snapshot è impostato su `pvc1-snap`. Un'istantanea VolumeSnapshot è analoga a un PVC ed è associata a un `VolumeSnapshotContent` oggetto che rappresenta lo snapshot effettivo.

```
kubectl create -f snap.yaml
volumesnapshot.snapshot.storage.k8s.io/pvc1-snap created

kubectl get volumesnapshots
NAME                   AGE
pvc1-snap              50s
```

- È possibile identificare `VolumeSnapshotContent` oggetto per `pvc1-snap` VolumeSnapshot descrivendolo. Il `Snapshot Content Name` Identifica l'oggetto VolumeSnapshotContent che fornisce questa snapshot. Il `Ready To Use` Parametro indica che l'istantanea può essere utilizzata per creare un nuovo PVC.

```
kubectl describe volumesnapshots pvc1-snap
Name:         pvc1-snap
Namespace:    default
...
Spec:
  Snapshot Class Name:    pvc1-snap
  Snapshot Content Name:  snapcontent-e8d8a0ca-9826-11e9-9807-
525400f3f660
  Source:
    API Group:
    Kind:       PersistentVolumeClaim
    Name:       pvc1
Status:
  Creation Time:  2019-06-26T15:27:29Z
  Ready To Use:   true
  Restore Size:   3Gi
...
```

Creare un PVC da uno snapshot di volume

È possibile utilizzare `dataSource` Per creare un PVC utilizzando un VolumeSnapshot denominato `<pvc-name>` come origine dei dati. Una volta creato, il PVC può essere collegato a un pod e utilizzato come qualsiasi altro PVC.

> Il PVC verrà creato nello stesso backend del volume di origine. Fare riferimento a. "KB: La creazione di un PVC da uno snapshot PVC Trident non può essere creata in un backend alternativo".

Nell'esempio seguente viene creato il PVC utilizzando `pvc1-snap` come origine dei dati.

```
cat pvc-from-snap.yaml
```

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: golden
  resources:
    requests:
      storage: 3Gi
  dataSource:
    name: pvc1-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

Importare uno snapshot di volume

Trident supporta l' "Processo Snapshot con pre-provisioning di Kubernetes" per consentire all'amministratore del cluster di creare un `VolumeSnapshotContent` oggetto e importare snapshot creati all'esterno di Trident.

Prima di iniziare

Trident deve aver creato o importato il volume principale dello snapshot.

Fasi

1. **Cluster admin:** creare un `VolumeSnapshotContent` oggetto che fa riferimento allo snapshot backend. Viene avviato il flusso di lavoro dello snapshot in Trident.

   - Specificare il nome dell'istantanea backend in `annotations` come `trident.netapp.io/internalSnapshotName: <"backend-snapshot-name">`.
   - Specificare `<name-of-parent-volume-in-trident>/<volume-snapshot-content-name>` in `snapshotHandle`. si tratta delle uniche informazioni fornite a Trident dallo snap-over esterno nella `ListSnapshots` chiamata.

     > Il `<volumeSnapshotContentName>` Impossibile corrispondere sempre al nome dell'istantanea backend a causa di vincoli di denominazione CR.

     Esempio

     Nell'esempio seguente viene creato un `VolumeSnapshotContent` oggetto che fa riferimento allo snapshot backend `snap-01`.

```yaml
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotContent
metadata:
  name: import-snap-content
  annotations:
    trident.netapp.io/internalSnapshotName: "snap-01"  # This is the
name of the snapshot on the backend
spec:
  deletionPolicy: Retain
  driver: csi.trident.netapp.io
  source:
    snapshotHandle: pvc-f71223b5-23b9-4235-bbfe-e269ac7b84b0/import-
snap-content # <import PV name or source PV name>/<volume-snapshot-
content-name>
  volumeSnapshotRef:
    name: import-snap
    namespace: default
```

2. **Cluster admin:** creare il `VolumeSnapshot` CR che fa riferimento a. `VolumeSnapshotContent` oggetto. In questo modo viene richiesto l'accesso per l'utilizzo di `VolumeSnapshot` in un determinato namespace.

   Esempio

   Nell'esempio seguente viene creato un `VolumeSnapshot` CR con nome `import-snap` questo fa riferimento al `VolumeSnapshotContent` con nome `import-snap-content`.

```yaml
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: import-snap
spec:
  # volumeSnapshotClassName: csi-snapclass (not required for pre-
provisioned or imported snapshots)
  source:
    volumeSnapshotContentName: import-snap-content
```

3. **Elaborazione interna (nessuna azione richiesta):** lo snapshot esterno riconosce il nuovo creato ed esegue `ListSnapshots` la `VolumeSnapshotContent` chiamata. Trident crea la `TridentSnapshot`.
   - Lo snapshot esterno imposta `VolumeSnapshotContent` a. `readyToUse` e a. `VolumeSnapshot` a. `true`.
   - Trident ritorna `readyToUse=true`.
4. **Qualsiasi utente:** creare un `PersistentVolumeClaim` per fare riferimento al nuovo `VolumeSnapshot`, dove il `spec.dataSource` (o. `spec.dataSourceRef`) è il `VolumeSnapshot` nome.

   Esempio

Nell'esempio seguente viene creato un PVC che fa riferimento a. `VolumeSnapshot` con nome `import-snap`.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: simple-sc
  resources:
    requests:
      storage: 1Gi
  dataSource:
    name: import-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

Ripristinare i dati del volume utilizzando le snapshot

La directory Snapshot è nascosta per impostazione predefinita per facilitare la massima compatibilità dei volumi con cui viene eseguito il provisioning mediante `ontap-nas` e. `ontap-nas-economy` driver. Attivare il `.snapshot` directory per ripristinare i dati direttamente dalle snapshot.

Utilizzare la CLI ONTAP per il ripristino dello snapshot del volume per ripristinare uno stato di un volume registrato in uno snapshot precedente.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

> Quando si ripristina una copia snapshot, la configurazione del volume esistente viene sovrascritta. Le modifiche apportate ai dati del volume dopo la creazione della copia snapshot andranno perse.

Ripristino del volume in-place da uno snapshot

Trident consente il ripristino rapido e in-place del volume da uno snapshot utilizzando il `TridentActionSnapshotRestore` CR (TASR). Questo CR funziona come un'azione imperativa di Kubernetes e non persiste al termine dell'operazione.

Trident supporta il ripristino delle istantanee su `ontap-san` , `ontap-san-economy`, `ontap-nas`, , `ontap-nas-flexgroup`, `azure-netapp-files` `gcp-cvs`, `google-cloud-netapp-volumes` e `solidfire-san` driver.

Prima di iniziare

È necessario disporre di un PVC associato e di uno snapshot del volume disponibile.

- Verificare che lo stato del PVC sia limitato.

```
kubectl get pvc
```

- Verificare che lo snapshot del volume sia pronto per l'uso.

```
kubectl get vs
```

Fasi

1. Creare TASR CR. In questo esempio viene creata una CR per PVC `pvc1` e snapshot volume `pvc1-snapshot`.

   Il TASR CR deve trovarsi in uno spazio dei nomi in cui esistono PVC e VS.

```
cat tasr-pvc1-snapshot.yaml
```

```yaml
apiVersion: trident.netapp.io/v1
kind: TridentActionSnapshotRestore
metadata:
  name: trident-snap
  namespace: trident
spec:
  pvcName: pvc1
  volumeSnapshotName: pvc1-snapshot
```

2. Applicare la CR per eseguire il ripristino dall'istantanea. Nell'esempio riportato di seguito vengono ripristinati gli snapshot `pvc1`.

```
kubectl create -f tasr-pvc1-snapshot.yaml
```

```
tridentactionsnapshotrestore.trident.netapp.io/trident-snap created
```

Risultati

Trident ripristina i dati dalla snapshot. È possibile verificare lo stato di ripristino dello snapshot:

```
kubectl get tasr -o yaml
```

```
apiVersion: trident.netapp.io/v1
items:
- apiVersion: trident.netapp.io/v1
  kind: TridentActionSnapshotRestore
  metadata:
    creationTimestamp: "2023-04-14T00:20:33Z"
    generation: 3
    name: trident-snap
    namespace: trident
    resourceVersion: "3453847"
    uid: <uid>
  spec:
    pvcName: pvc1
    volumeSnapshotName: pvc1-snapshot
  status:
    startTime: "2023-04-14T00:20:34Z"
    completionTime: "2023-04-14T00:20:37Z"
    state: Succeeded
kind: List
metadata:
  resourceVersion: ""
```

- Nella maggior parte dei casi, Trident non ritenta automaticamente l'operazione in caso di errore. Sarà necessario eseguire nuovamente l'operazione.
- Gli utenti Kubernetes senza accesso amministrativo potrebbero dover essere autorizzati dall'amministratore a creare una TASR CR nel namespace delle applicazioni.

Eliminare un PV con gli snapshot associati

Quando si elimina un volume persistente con gli snapshot associati, il volume Trident corrispondente viene aggiornato allo "stato di eliminazione". Rimuovere gli snapshot del volume per eliminare il volume Trident.

Implementare un controller per lo snapshot dei volumi

Se la distribuzione Kubernetes non include lo snapshot controller e i CRD, è possibile implementarli come segue.

Fasi

1. Creare CRD snapshot di volume.

```
cat snapshot-setup.sh
```

```
#!/bin/bash
# Create volume snapshot CRDs
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshotclasses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshotcontents.yam
l
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshots.yaml
```

2. Creare il controller di snapshot.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-6.1/deploy/kubernetes/snapshot-
controller/rbac-snapshot-controller.yaml
```

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-6.1/deploy/kubernetes/snapshot-
controller/setup-snapshot-controller.yaml
```

> Se necessario, aprire `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` e aggiornare `namespace` allo spazio dei nomi.

Link correlati

- "Snapshot dei volumi"
- "VolumeSnapshotClass"

Lavorare con gli snapshot del gruppo di volumi

Snapshot del gruppo di volumi Kubernetes di volumi persistenti (PV). NetApp Trident offre la possibilità di creare snapshot di più volumi (un gruppo di snapshot di volume). Questo snapshot del gruppo di volumi rappresenta copie di più volumi acquisite nello stesso momento.

> VolumeGroupSnapshot è una funzionalità beta di Kubernetes con API beta. La versione minima richiesta per VolumeGroupSnapshot è Kubernetes 1.32.

Creare snapshot del gruppo di volumi

L'istantanea del gruppo di volumi è supportata con `ontap-san` driver, solo per protocollo iSCSI, non ancora supportato con Fibre Channel (FCP) né NVMe/TCP. Prima di iniziare

- Assicurati che la versione di Kubernetes sia K8s 1.32 o successiva.
- Per utilizzare gli snapshot, è necessario disporre di un controller snapshot esterno e di CRD (Custom Resource Definitions). Questa è la responsabilità del Kubernetes orchestrator (ad esempio: Kubeadm, GKE, OpenShift).

  Se la distribuzione di Kubernetes non include il controller di snapshot esterno e i CRD, fare riferimento a Implementare un controller per lo snapshot dei volumi .

  > Non creare uno snapshot controller se si creano snapshot di gruppi di volumi su richiesta in un ambiente GKE. GKE utilizza un controller di snapshot integrato e nascosto.

- Nel controller snapshot YAML, imposta `CSIVolumeGroupSnapshot` feature gate su 'true' per garantire che lo snapshot del gruppo di volumi sia abilitato.
- Creare le classi di snapshot del gruppo di volumi richieste prima di creare uno snapshot del gruppo di volumi.
- Assicurarsi che tutti i PVC/volumi siano sullo stesso SVM per poter creare VolumeGroupSnapshot.

Fasi

- Creare una VolumeGroupSnapshotClass prima di creare una VolumeGroupSnapshot. Per ulteriori informazioni, fare riferimento a "Classe VolumeGroupSnapshot".

```yaml
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshotClass
metadata:
  name: csi-group-snap-class
  annotations:
    kubernetes.io/description: "Trident group snapshot class"
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

- Creare PVC con le etichette richieste utilizzando le classi di archiviazione esistenti oppure aggiungere queste etichette ai PVC esistenti.
  1. Definisci la chiave e il valore dell'etichetta in base alle tue esigenze

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1-group-snap
  labels:
    consistentGroupSnapshot: groupA
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Mi
  storageClassName: sc1-1
```

- Crea un VolumeGroupSnapshot con la stessa etichetta (`consistentGroupSnapshot: groupA`) specificato nel PVC.

  Questo esempio crea uno snapshot del gruppo di volumi:

```
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshot
metadata:
  name: "vgs1"
  namespace: trident
spec:
  volumeGroupSnapshotClassName: csi-group-snap-class
  source:
    selector:
      matchLabels:
        consistentGroupSnapshot: groupA
```

Recupera i dati del volume utilizzando uno snapshot di gruppo

È possibile ripristinare singoli volumi persistenti utilizzando i singoli snapshot creati come parte dello snapshot del gruppo di volumi. Non è possibile ripristinare lo snapshot del gruppo di volumi come unità.

Utilizzare la CLI ONTAP per il ripristino dello snapshot del volume per ripristinare uno stato di un volume registrato in uno snapshot precedente.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

> Quando si ripristina una copia snapshot, la configurazione del volume esistente viene sovrascritta. Le modifiche apportate ai dati del volume dopo la creazione della copia snapshot andranno perse.

Ripristino del volume in-place da uno snapshot

Trident consente il ripristino rapido e in-place del volume da uno snapshot utilizzando il `TridentActionSnapshotRestore` CR (TASR). Questo CR funziona come un'azione imperativa di Kubernetes e non persiste al termine dell'operazione.

Per ulteriori informazioni, vedere "Ripristino del volume in-place da uno snapshot".

Elimina un PV con snapshot di gruppo associati

Quando si elimina uno snapshot del volume di gruppo:

- È possibile eliminare i VolumeGroupSnapshot nel loro insieme, non i singoli snapshot del gruppo.
- Se i PersistentVolume vengono eliminati mentre esiste uno snapshot per quel PersistentVolume, Trident sposterà quel volume in uno stato di "eliminazione" perché lo snapshot deve essere rimosso prima che il volume possa essere rimosso in modo sicuro.
- Se è stato creato un clone utilizzando uno snapshot raggruppato e in seguito il gruppo deve essere eliminato, verrà avviata un'operazione di suddivisione su clone e il gruppo non potrà essere eliminato finché la suddivisione non sarà completata.

Implementare un controller per lo snapshot dei volumi

Se la distribuzione Kubernetes non include lo snapshot controller e i CRD, è possibile implementarli come segue.

Fasi

1. Creare CRD snapshot di volume.

```
cat snapshot-setup.sh
```

```
#!/bin/bash
# Create volume snapshot CRDs
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotcl
asses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotco
ntents.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshots.
yaml
```

2. Creare il controller di snapshot.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-8.2/deploy/kubernetes/snapshot-
controller/rbac-snapshot-controller.yaml
```

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-8.2/deploy/kubernetes/snapshot-
controller/setup-snapshot-controller.yaml
```

> Se necessario, aprire `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` e aggiornare `namespace` allo spazio dei nomi.

Link correlati

- "Classe VolumeGroupSnapshot"
- "Snapshot dei volumi"

Informazioni sul copyright

Copyright © 2025 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.