Documentazione Trident 25.06

Trident

NetApp
October 29, 2025

Sommario

Documentazione Trident 25.06

Note di rilascio

Cosa c'è di nuovo

Le note di rilascio forniscono informazioni sulle nuove funzionalità, sui miglioramenti e sulle correzioni di bug nell'ultima versione di NetApp Trident.

> IL `tridentctl` il binario per Linux fornito nel file zip dell'installer è la versione testata e supportata. Siate consapevoli che il `macos` binario fornito nel `/extras` parte del file zip non è testata né supportata.

Novità della versione 25.06.2

Il riepilogo delle novità fornisce dettagli su miglioramenti, correzioni e deprecazioni per le versioni Trident e Trident Protect.

Trident

Correzioni

- **Kubernetes**: risolto un problema critico per cui venivano rilevati dispositivi iSCSI errati durante il distacco dei volumi dai nodi Kubernetes.

Modifiche nel 25.06.1

Trident

> I clienti che utilizzano SolidFire sono pregati di non effettuare l'aggiornamento alla versione 25.06.1 a causa di un problema noto durante l'annullamento della pubblicazione dei volumi. A breve verrà rilasciata la versione 25.06.2 per risolvere questo problema.

Correzioni

- **Kubernetes**:
  - Risolto un problema per cui gli NQN non venivano controllati prima di essere rimossi dai sottosistemi.
  - Risolto un problema per cui più tentativi di chiudere un dispositivo LUKS causavano errori nello scollegamento dei volumi.
  - Corretto il problema di destage del volume iSCSI quando il percorso del dispositivo è cambiato dalla sua creazione.
  - Clonazione a blocchi di volumi tra classi di archiviazione.
- **OpenShift**: risolto un problema per cui la preparazione del nodo iSCSI non riusciva con OCP 4.19.
- Aumentato il timeout durante la clonazione di un volume utilizzando i backend SolidFire ("Numero 1008").

Modifiche nel 25.06

Trident

Miglioramenti

- **Kubernetes**:
  - Aggiunto supporto per snapshot di gruppi di volumi CSI con `v1beta1` API Kubernetes per snapshot di gruppi di volumi per driver ONTAP-SAN iSCSI. Vedere"Lavorare con gli snapshot del gruppo di volumi" .

    > VolumeGroupSnapshot è una funzionalità beta di Kubernetes con API beta. Kubernetes 1.32 è la versione minima richiesta per VolumeGroupSnapshot.

  - Aggiunto il supporto per ONTAP ASA r2 per NVMe/TCP oltre a iSCSI. Vederelink:"Opzioni ed esempi di configurazione SAN ONTAP" .
  - Aggiunto supporto SMB sicuro per volumi ONTAP-NAS e ONTAP-NAS-Economy. Gli utenti e i gruppi di Active Directory possono ora essere utilizzati con volumi SMB per una maggiore sicurezza. Vedere"Abilita SMB sicuro" .
  - Concorrenza dei nodi Trident migliorata per una maggiore scalabilità nelle operazioni dei nodi per i volumi iSCSI.
  - Aggiunto `--allow-discards` quando si aprono volumi LUKS per consentire i comandi discard/TRIM per il recupero dello spazio.
  - Prestazioni migliorate durante la formattazione di volumi crittografati LUKS.
  - Pulizia LUKS migliorata per dispositivi LUKS non riusciti ma parzialmente formattati.
  - Idempotenza del nodo Trident migliorata per l'attacco e lo scollegamento del volume NVMe.
  - Aggiunto `internalID` campo alla configurazione del volume Trident per il driver ONTAP-SAN-Economy.
  - Aggiunto supporto per la replicazione del volume con SnapMirror per backend NVMe. Vedere"Replicare i volumi utilizzando SnapMirror" .

Miglioramenti sperimentali

> Non utilizzare in ambienti di produzione.

- [Anteprima tecnica] Abilitate le operazioni simultanee del controller Trident tramite `--enable-concurrency` bandiera caratteristica. Ciò consente alle operazioni del controller di funzionare in parallelo, migliorando le prestazioni in ambienti affollati o di grandi dimensioni.

  > Questa funzionalità è sperimentale e attualmente supporta flussi di lavoro paralleli limitati con il driver ONTAP-SAN (protocolli iSCSI e FCP).

- [Anteprima tecnica] Aggiunto il supporto QOS manuale con il driver ANF.

Correzioni

- **Kubernetes**:
  - Risolto un problema con CSI NodeExpandVolume per cui i dispositivi multipath potevano avere dimensioni incongruenti quando i dischi SCSI sottostanti non erano disponibili.
  - Risolto il problema relativo all'errore di pulizia dei criteri di esportazione duplicati per i driver ONTAP-

NAS e ONTAP-NAS-Economy.

  - Corretti i volumi GCNV impostati per impostazione predefinita su NFSv3 quando `nfsMountOptions` non è impostato; ora sono supportati sia i protocolli NFSv3 che NFSv4. Se `nfsMountOptions` non viene fornito, verrà utilizzata la versione NFS predefinita dell'host (NFSv3 o NFSv4).
  - Risolto il problema di distribuzione durante l'installazione Trident tramite Kustomize ("Numero 831").
  - Corretti i criteri di esportazione mancanti per i PVC creati da snapshot ("Numero 1016").
  - Risolto il problema per cui le dimensioni del volume ANF non venivano automaticamente allineate a incrementi di 1 GiB.
  - Risolto il problema durante l'utilizzo di NFSv3 con Bottlerocket.
- Risolto il timeout durante la clonazione di un volume utilizzando i backend SolidFire ("Numero 1008").
- Risolto il problema con i volumi ONTAP-NAS-Economy che si espandevano fino a 300 TB nonostante gli errori di ridimensionamento.
- Risolto il problema per cui le operazioni di suddivisione del clone venivano eseguite in modo sincrono quando si utilizzava l'API REST ONTAP .

Deprecazioni:

- **Kubernetes**: aggiornato il supporto minimo di Kubernetes alla versione 1.27.

Trident protetto

NetApp Trident Protect offre funzionalità avanzate di gestione dei dati delle applicazioni che migliorano la funzionalità e la disponibilità delle applicazioni Kubernetes con stato supportate dai sistemi di storage NetApp ONTAP e dal provisioner di storage NetApp Trident CSI.

Miglioramenti

- Tempi di ripristino migliorati, con la possibilità di eseguire backup completi più frequenti.
- Granularità migliorata della definizione dell'applicazione e ripristino selettivo con filtro Group-Version-Kind (GVK).
- Risincronizzazione efficiente e replica inversa quando si utilizza AppMirrorRelationship (AMR) con NetApp SnapMirror, per evitare la replica PVC completa.
- Aggiunta la possibilità di utilizzare EKS Pod Identity per creare bucket AppVault, eliminando la necessità di specificare un segreto con le credenziali del bucket per i cluster EKS.
- Aggiunta la possibilità di saltare il ripristino di etichette e annotazioni nello spazio dei nomi di ripristino, se necessario.
- AppMirrorRelationship (AMR) ora verificherà l'espansione del PVC di origine ed eseguirà l'espansione appropriata sul PVC di destinazione, se necessario.

Correzioni

- Risolto il bug per cui i valori di annotazione degli snapshot precedenti venivano applicati agli snapshot più recenti. Ora tutte le annotazioni degli snapshot vengono applicate correttamente.
- Definito un segreto per la crittografia del data mover (Kopia/Restic) per impostazione predefinita, se non definito.
- Aggiunti messaggi di convalida e di errore migliorati per la creazione di appvault S3.
- AppMirrorRelationship (AMR) ora replica solo i PV nello stato Bound, per evitare tentativi falliti.

- Risolto il problema per cui venivano visualizzati errori durante l'acquisizione di AppVaultContent su un AppVault con un numero elevato di backup.
- Gli snapshot VMSnapshot di KubeVirt vengono esclusi dalle operazioni di ripristino e failover per evitare errori.
- Risolto il problema con Kopia per cui gli snapshot venivano rimossi prematuramente perché la pianificazione di conservazione predefinita di Kopia sovrascriveva quanto impostato dall'utente nella pianificazione.

Modifiche nel 25.02.1

Trident

Correzioni

- **Kubernetes**:
  - Risolto un problema nell'operatore trident in cui i nomi e le versioni delle immagini sidecar venivano compilati in modo errato quando si utilizzava un registro di immagini non predefinito ("Numero 983").
  - Risolto il problema per cui le sessioni multipath non riuscivano a ripristinarsi durante un failover ONTAP ("Numero 961").

Modifiche nel 25.02

A partire da Trident 25.02, il riepilogo delle novità fornisce dettagli su miglioramenti, correzioni e deprecazioni per le versioni Trident e Trident Protect.

Trident

Miglioramenti

- **Kubernetes**:
  - Aggiunto supporto per ONTAP ASA r2 per iSCSI.
  - Aggiunto supporto per il distacco forzato per volumi ONTAP-NAS durante scenari di arresto del nodo non regolare. I nuovi volumi ONTAP-NAS ora utilizzeranno criteri di esportazione per volume gestiti da Trident. Fornito un percorso di aggiornamento per i volumi esistenti per passare al nuovo modello di policy di esportazione in caso di annullamento della pubblicazione senza influire sui carichi di lavoro attivi.
  - Aggiunta l'annotazione cloneFromSnapshot.
  - Aggiunto supporto per la clonazione di volumi tra namespace.
  - Miglioramento delle risoluzioni di scansione auto-riparante iSCSI per avviare nuove scansioni in base all'host esatto, al canale, alla destinazione e all'ID LUN.
  - Aggiunto supporto per Kubernetes 1.32.
- **OpenShift**:
  - Aggiunto supporto per la preparazione automatica dei nodi iSCSI per RHCOS su cluster ROSA.
  - Aggiunto supporto per OpenShift Virtualization per driver ONTAP .
- Aggiunto il supporto Fibre Channel sul driver ONTAP-SAN.
- Aggiunto il supporto NVMe LUKS.

- Passato all'immagine di prova per tutte le immagini di base.
- Aggiunto rilevamento e registrazione dello stato della connessione iSCSI quando le sessioni iSCSI dovrebbero essere registrate ma non lo sono ("Numero 961").
- Aggiunto supporto per volumi SMB con driver google-cloud-netapp-volumes.
- Aggiunto supporto per consentire ai volumi ONTAP di saltare la coda di ripristino durante l'eliminazione.
- Aggiunto supporto per sovrascrivere le immagini predefinite utilizzando SHA anziché tag.
- Aggiunto il flag image-pull-secrets al programma di installazione tridentctl.

Correzioni

- **Kubernetes**:
  - Corretti gli indirizzi IP dei nodi mancanti dalle policy di esportazione automatica ("Numero 965").
  - Corretti i problemi di passaggio prematuro delle policy di esportazione automatica alla policy per volume per ONTAP-NAS-Economy.
  - Credenziali di configurazione del backend fisse per supportare tutte le partizioni AWS ARN disponibili ("Numero 913").
  - Aggiunta opzione per disabilitare la riconciliazione del configuratore automatico nell'operatore Trident ("Numero 924").
  - Aggiunto securityContext per il contenitore csi-resizer ("Numero 976").

Trident protetto

NetApp Trident Protect offre funzionalità avanzate di gestione dei dati delle applicazioni che migliorano la funzionalità e la disponibilità delle applicazioni Kubernetes con stato supportate dai sistemi di storage NetApp ONTAP e dal provisioner di storage NetApp Trident CSI.

Miglioramenti

- Aggiunto supporto per backup e ripristino per VM di virtualizzazione KubeVirt/OpenShift per l'archiviazione volumeMode: File e volumeMode: Block (dispositivo raw). Questo supporto è compatibile con tutti i driver Trident e migliora le funzionalità di protezione esistenti durante la replica dello storage utilizzando NetApp SnapMirror con Trident Protect.
- Aggiunta la possibilità di controllare il comportamento di blocco a livello di applicazione per gli ambienti Kubevirt.
- Aggiunto supporto per la configurazione delle connessioni proxy AutoSupport .
- Aggiunta la possibilità di definire un segreto per la crittografia del data mover (Kopia / Restic).
- Aggiunta la possibilità di eseguire manualmente un hook di esecuzione.
- Aggiunta la possibilità di configurare i vincoli del contesto di sicurezza (SCC) durante l'installazione Trident Protect.
- Aggiunto supporto per la configurazione di nodeSelector durante l'installazione Trident Protect.
- Aggiunto supporto per proxy di uscita HTTP/HTTPS per oggetti AppVault.
- ResourceFilter esteso per consentire l'esclusione delle risorse con ambito cluster.
- Aggiunto supporto per il token di sessione AWS nelle credenziali S3 AppVault.
- Aggiunto supporto per la raccolta di risorse dopo gli hook di esecuzione pre-snapshot.

Correzioni

- Migliorata la gestione dei volumi temporanei per saltare la coda di ripristino del volume ONTAP .
- Le annotazioni SCC sono ora ripristinate ai valori originali.
- Efficienza di ripristino migliorata con supporto per operazioni parallele.
- Supporto migliorato per i timeout dell'hook di esecuzione per applicazioni di grandi dimensioni.

Modifiche nel 24.10.1

Miglioramenti

- **Kubernetes**: Aggiunto supporto per Kubernetes 1.32.
- Aggiunto rilevamento e registrazione dello stato della connessione iSCSI quando le sessioni iSCSI dovrebbero essere registrate ma non lo sono ("Numero 961").

Correzioni

- Corretti gli indirizzi IP dei nodi mancanti dalle policy di esportazione automatica ("Numero 965").
- Corretti i problemi di passaggio prematuro delle policy di esportazione automatica alla policy per volume per ONTAP-NAS-Economy.
- Aggiornate le dipendenze Trident e Trident-ASUP per risolvere CVE-2024-45337 e CVE-2024-45310.
- Rimossi i disconnessioni per i portali non-CHAP intermittenti non funzionanti durante l'auto-riparazione iSCSI ("Numero 961").

Modifiche nel 24.10

Miglioramenti

- Il driver Google Cloud NetApp Volumes è ora generalmente disponibile per i volumi NFS e supporta il provisioning basato sulla zona.
- GCP Workload Identity verrà utilizzato come Cloud Identity per Google Cloud NetApp Volumes con GKE.
- Aggiunto `formatOptions` parametro di configurazione per i driver ONTAP-SAN e ONTAP-SAN-Economy per consentire agli utenti di specificare le opzioni di formato LUN.
- Riduzione delle dimensioni minime del volume di Azure NetApp Files a 50 GiB. La nuova dimensione minima di Azure dovrebbe essere disponibile al pubblico a partire da novembre.
- Aggiunto `denyNewVolumePools` parametro di configurazione per limitare i driver ONTAP-NAS-Economy e ONTAP-SAN-Economy ai pool Flexvol preesistenti.
- Aggiunto il rilevamento per l'aggiunta, la rimozione o la ridenominazione degli aggregati dall'SVM su tutti i driver ONTAP .
- Aggiunti 18 MiB di overhead ai LUN LUKS per garantire che le dimensioni PVC segnalate siano utilizzabili.
- Migliorata la gestione degli errori di fase e di non fase dei nodi ONTAP-SAN e ONTAP-SAN-Economy per consentire alla funzione di non fase di rimuovere i dispositivi dopo una fase non riuscita.
- Aggiunto un generatore di ruoli personalizzato che consente ai clienti di creare un ruolo minimalista per Trident in ONTAP.
- Aggiunta ulteriore registrazione per la risoluzione dei problemi `lsscsi` ("Numero 792").

Kubernetes

- Aggiunte nuove funzionalità Trident per i flussi di lavoro nativi di Kubernetes:
  - Protezione dei dati
  - Migrazione dei dati
  - Ripristino dopo un disastro
  - Mobilità delle applicazioni

    "Scopri di più su Trident Protect".
- Aggiunta una nuova bandiera `--k8s-api-qps` agli installatori per impostare il valore QPS utilizzato da Trident per comunicare con il server API Kubernetes.
- Aggiunto `--node-prep` flag per gli installatori per la gestione automatica delle dipendenze del protocollo di archiviazione sui nodi del cluster Kubernetes. Compatibilità testata e verificata con il protocollo di archiviazione iSCSI Amazon Linux 2023
- Aggiunto supporto per il distacco forzato per volumi ONTAP-NAS-Economy durante scenari di arresto del nodo non regolare.
- I nuovi volumi ONTAP-NAS-Economy NFS utilizzeranno criteri di esportazione per qtree quando si utilizzano `autoExportPolicy` opzione backend. I Qtree verranno mappati solo alle policy di esportazione restrittive dei nodi al momento della pubblicazione, per migliorare il controllo degli accessi e la sicurezza. I qtree esistenti passeranno al nuovo modello di policy di esportazione quando Trident annullerà la pubblicazione del volume da tutti i nodi, senza influire sui carichi di lavoro attivi.
- Aggiunto supporto per Kubernetes 1.31.

Miglioramenti sperimentali

- Aggiunta anteprima tecnica per il supporto Fibre Channel sul driver ONTAP-SAN.

Correzioni

- **Kubernetes**:
  - Corretto il webhook di ammissione del Rancher che impediva l'installazione Trident Helm ("Numero 839").
  - Chiave di affinità fissa nei valori della tabella del timone ("Numero 898").
  - Corretto: tridentControllerPluginNodeSelector/tridentNodePluginNodeSelector non funzionerà con il valore "true" ("Numero 899").
  - Eliminati gli snapshot effimeri creati durante la clonazione ("Numero 901").
- Aggiunto il supporto per Windows Server 2019.
- Corretto `go mod tidy` nel repository Trident ("Numero 767").

Deprecazioni

- **Kubernetes:**
  - Aggiornato il supporto minimo di Kubernetes alla versione 1.25.
  - Rimosso il supporto per la politica di sicurezza POD.

Rebranding del prodotto

A partire dalla versione 24.10, Astra Trident è stato rinominato Trident (Netapp Trident). Questo rebranding non influisce sulle funzionalità, sulle piattaforme supportate o sull'interoperabilità di Trident.

Modifiche nel 24.06

Miglioramenti

- **IMPORTANTE**: Il `limitVolumeSize` Il parametro ora limita le dimensioni qtree/LUN nei driver economici ONTAP . Usa il nuovo `limitVolumePoolSize` parametro per controllare le dimensioni Flexvol in quei driver. ("Numero 341").
- Aggiunta la possibilità per l'auto-riparazione iSCSI di avviare scansioni SCSI tramite ID LUN esatto se sono in uso igroup obsoleti ("Numero 883").
- Aggiunto il supporto per le operazioni di clonazione e ridimensionamento del volume, che possono essere consentite anche quando il backend è in modalità sospesa.
- Aggiunta la possibilità di propagare ai pod dei nodi Trident le impostazioni di registro configurate dall'utente per il controller Trident .
- Aggiunto il supporto in Trident per utilizzare REST per impostazione predefinita anziché ONTAPI (ZAPI) per le versioni ONTAP 9.15.1 e successive.
- Aggiunto supporto per nomi di volumi personalizzati e metadati sui backend di archiviazione ONTAP per i nuovi volumi persistenti.
- Migliorato il `azure-netapp-files` Driver (ANF) per abilitare automaticamente la directory snapshot per impostazione predefinita quando le opzioni di montaggio NFS sono impostate per utilizzare NFS versione 4.x.
- Aggiunto il supporto Bottlerocket per i volumi NFS.
- Aggiunto supporto per l'anteprima tecnica per Google Cloud NetApp Volumes.

Kubernetes

- Aggiunto supporto per Kubernetes 1.30.
- Aggiunta la possibilità per Trident DaemonSet di pulire le cavalcature zombie e i file di tracciamento residui all'avvio ("Numero 883").
- Aggiunta annotazione PVC `trident.netapp.io/luksEncryption` per importare dinamicamente volumi LUKS ("Numero 849").
- Aggiunta la consapevolezza della topologia al driver ANF.
- Aggiunto supporto per i nodi di Windows Server 2022.

Correzioni

- Risolti i problemi di installazione Trident dovuti a transazioni obsolete.
- Corretto tridentctl per ignorare i messaggi di avviso da Kubernetes ("Numero 892").
- Controller Trident modificato `SecurityContextConstraint` priorità a `0` ("Numero 887").
- I driver ONTAP ora accettano dimensioni di volume inferiori a 20 MiB ("Problema[#885").
- Corretto Trident per impedire la riduzione dei volumi FlexVol durante l'operazione di ridimensionamento per il driver ONTAP-SAN.

- Risolto il problema di importazione del volume ANF con NFS v4.1.

Modifiche nel 24.02

Miglioramenti

- Aggiunto supporto per Cloud Identity.
  - AKS con ANF - Azure Workload Identity verrà utilizzato come identità cloud.
  - EKS con FSxN: il ruolo AWS IAM verrà utilizzato come identità cloud.
- Aggiunto supporto per installare Trident come componente aggiuntivo sul cluster EKS dalla console EKS.
- Aggiunta la possibilità di configurare e disabilitare l'auto-riparazione iSCSI ("Numero 864").
- Aggiunta la personalità Amazon FSx ai driver ONTAP per abilitare l'integrazione con AWS IAM e SecretsManager e per consentire a Trident di eliminare i volumi FSx con backup ("Numero 453").

Kubernetes

- Aggiunto supporto per Kubernetes 1.29.

Correzioni

- Messaggi di avviso ACP corretti, quando ACP non è abilitato ("Numero 866").
- Aggiunto un ritardo di 10 secondi prima di eseguire una divisione del clone durante l'eliminazione dello snapshot per i driver ONTAP , quando un clone è associato allo snapshot.

Deprecazioni

- Rimosso il framework di attestazioni in-toto dai manifest delle immagini multipiattaforma.

Modifiche nel 23.10

Correzioni

- Espansione del volume fissa se una nuova dimensione richiesta è inferiore alla dimensione totale del volume per i driver di archiviazione ontap-nas e ontap-nas-flexgroup ("Numero 834").
- Dimensione del volume fissa per visualizzare solo la dimensione utilizzabile del volume durante l'importazione per i driver di archiviazione ontap-nas e ontap-nas-flexgroup ("Numero 722").
- Corretta conversione del nome FlexVol per ONTAP-NAS-Economy.
- Risolto il problema di inizializzazione Trident su un nodo Windows quando il nodo viene riavviato.

Miglioramenti

Kubernetes

Aggiunto supporto per Kubernetes 1.28.

Trident

- Aggiunto supporto per l'utilizzo di Azure Managed Identities (AMI) con il driver di archiviazione azure-netapp-files.

- Aggiunto supporto per NVMe su TCP per il driver ONTAP-SAN.
- Aggiunta la possibilità di mettere in pausa il provisioning di un volume quando il backend è impostato sullo stato sospeso dall'utente ("Numero 558").

Modifiche nel 23.07.1

Kubernetes: Corretta l'eliminazione del daemonset per supportare gli aggiornamenti senza tempi di inattività ("Numero 740").

Modifiche nel 23.07

Correzioni

Kubernetes

- Aggiornamento Trident corretto per ignorare i vecchi pod bloccati nello stato di terminazione ("Numero 740").
- Aggiunta tolleranza alla definizione "transient-trident-version-pod" ("Numero 795").

Trident

- Richieste ONTAPI (ZAPI) corrette per garantire che i numeri di serie LUN vengano interrogati durante l'ottenimento degli attributi LUN per identificare e correggere i dispositivi iSCSI fantasma durante le operazioni di Node Staging.
- Corretta la gestione degli errori nel codice del driver di archiviazione ("Numero 816").
- Risolto il problema di ridimensionamento della quota quando si utilizzano driver ONTAP con use-rest=true.
- Corretta la creazione del clone LUN in ontap-san-economy.
- Ripristina il campo delle informazioni di pubblicazione da `rawDevicePath` A `devicePath` ; aggiunta logica per popolare e recuperare (in alcuni casi) `devicePath` campo.

Miglioramenti

Kubernetes

- Aggiunto supporto per l'importazione di snapshot pre-provisionati.
- Permessi Linux di distribuzione e daemonset ridotti al minimo ("Numero 817").

Trident

- Non viene più segnalato il campo di stato per volumi e snapshot "online".
- Aggiorna lo stato del backend se il backend ONTAP è offline ("Numeri #801" ,"#543").
- Il numero di serie LUN viene sempre recuperato e pubblicato durante il flusso di lavoro ControllerVolumePublish.
- Aggiunta logica aggiuntiva per verificare il numero di serie e le dimensioni del dispositivo multipath iSCSI.
- Verifica aggiuntiva per i volumi iSCSI per garantire che il dispositivo multipath corretto non sia in fase di staging.

Miglioramento sperimentale

Aggiunto il supporto dell'anteprima tecnica per NVMe su TCP per il driver ONTAP-SAN.

Documentazione

Sono stati apportati numerosi miglioramenti organizzativi e di formattazione.

Deprecazioni

Kubernetes

- Rimosso il supporto per gli snapshot v1beta1.
- Rimosso il supporto per volumi e classi di archiviazione pre-CSI.
- Aggiornato il supporto minimo di Kubernetes alla versione 1.22.

Modifiche nel 23.04

> La forzatura del distacco del volume per i volumi ONTAP-SAN-* è supportata solo con le versioni di Kubernetes con la funzionalità di arresto del nodo non regolare abilitata. La disconnessione forzata deve essere abilitata al momento dell'installazione utilizzando `--enable-force-detach` Flag di installazione Trident .

Correzioni

- Corretto l'operatore Trident per utilizzare IPv6 localhost per l'installazione quando specificato nelle specifiche.
- Corretti i permessi del ruolo cluster dell'operatore Trident per essere sincronizzati con i permessi del bundle ("Numero 799").
- Risolto il problema relativo all'associazione di volumi di blocchi grezzi su più nodi in modalità RWX.
- Corretto il supporto per la clonazione FlexGroup e l'importazione di volumi per volumi SMB.
- Risolto il problema per cui il controller Trident non poteva spegnersi immediatamente ("Numero 811").
- Aggiunta correzione per elencare tutti i nomi igroup associati a una LUN specificata fornita con i driver ontap-san-*.
- Aggiunta una correzione per consentire l'esecuzione dei processi esterni fino al completamento.
- Corretto errore di compilazione per l'architettura s390 ("Numero 537").
- Corretto livello di registrazione errato durante le operazioni di montaggio del volume ("Numero 781").
- Corretto errore di asserzione del tipo potenziale ("Numero 802").

Miglioramenti

- Kubernetes:
  - Aggiunto supporto per Kubernetes 1.27.
  - Aggiunto supporto per l'importazione di volumi LUKS.
  - Aggiunto supporto per la modalità di accesso PVC ReadWriteOncePod.
  - Aggiunto supporto per il distacco forzato per volumi ONTAP-SAN-* durante scenari di arresto del nodo non regolare.

    - Tutti i volumi ONTAP-SAN-* ora utilizzeranno igroup per nodo. Per migliorare la nostra sicurezza, le LUN verranno mappate solo sugli igroup quando vengono pubblicate attivamente su tali nodi. I volumi esistenti verranno opportunisticamente convertiti al nuovo schema igroup quando Trident determinerà che è sicuro farlo senza influire sui carichi di lavoro attivi ("Numero 758").
    - Miglioramento della sicurezza Trident mediante la pulizia degli igroup gestiti da Trident inutilizzati dai backend ONTAP-SAN-*.
- Aggiunto il supporto per volumi SMB con Amazon FSx ai driver di archiviazione ontap-nas-economy e ontap-nas-flexgroup.
- Aggiunto supporto per condivisioni SMB con i driver di archiviazione ontap-nas, ontap-nas-economy e ontap-nas-flexgroup.
- Aggiunto supporto per i nodi arm64 ("Numero 732").
- Procedura di spegnimento Trident migliorata disattivando prima i server API ("Numero 811").
- Aggiunto il supporto per la compilazione multipiattaforma per host Windows e arm64 a Makefile; vedere BUILD.md.

Deprecazioni

Kubernetes: Gli igroup con ambito backend non verranno più creati durante la configurazione dei driver ontap-san e ontap-san-economy ("Numero 758").

Modifiche nel 23.01.1

Correzioni

- Corretto l'operatore Trident per utilizzare IPv6 localhost per l'installazione quando specificato nelle specifiche.
- Corretti i permessi del ruolo del cluster Trident Operator per essere sincronizzati con i permessi del bundle"Numero 799" .
- Aggiunta una correzione per consentire l'esecuzione dei processi esterni fino al completamento.
- Risolto il problema relativo all'associazione di volumi di blocchi grezzi su più nodi in modalità RWX.
- Corretto il supporto per la clonazione FlexGroup e l'importazione di volumi per volumi SMB.

Modifiche nel 23.01

Kubernetes 1.27 è ora supportato in Trident. Aggiornare Trident prima di aggiornare Kubernetes.

Correzioni

- Kubernetes: aggiunte opzioni per escludere la creazione di Pod Security Policy per correggere le installazioni Trident tramite Helm ("Numeri #783, #794").

Miglioramenti

Kubernetes

- Aggiunto supporto per Kubernetes 1.26.
- Miglioramento dell'utilizzo complessivo delle risorse Trident RBAC ("Numero 757").

- Aggiunta automazione per rilevare e correggere sessioni iSCSI interrotte o obsolete sui nodi host.
- Aggiunto supporto per l'espansione dei volumi crittografati LUKS.
- Kubernetes: aggiunto il supporto per la rotazione delle credenziali per i volumi crittografati LUKS.

Trident

- Aggiunto il supporto per volumi SMB con Amazon FSx for NetApp ONTAP al driver di archiviazione ontap-nas.
- Aggiunto supporto per le autorizzazioni NTFS quando si utilizzano volumi SMB.
- Aggiunto supporto per pool di archiviazione per volumi GCP con livello di servizio CVS.
- Aggiunto supporto per l'uso facoltativo di flexgroupAggregateList durante la creazione di FlexGroup con il driver di archiviazione ontap-nas-flexgroup.
- Prestazioni migliorate per il driver di archiviazione ontap-nas-economy durante la gestione di più volumi FlexVol
- Aggiornamenti dataLIF abilitati per tutti i driver di archiviazione NAS ONTAP .
- Aggiornata la convenzione di denominazione Trident Deployment e DaemonSet per riflettere il sistema operativo del nodo host.

Deprecazioni

- Kubernetes: aggiornato il supporto minimo di Kubernetes alla versione 1.21.
- I DataLIF non dovrebbero più essere specificati durante la configurazione `ontap-san` O `ontap-san-economy` conducenti.

Modifiche nel 22.10

È necessario leggere le seguenti informazioni fondamentali prima di effettuare l'aggiornamento a Trident 22.10.

Informazioni critiche su Trident 22.10

- Kubernetes 1.25 è ora supportato in Trident. È necessario aggiornare Trident alla versione 22.10 prima di eseguire l'aggiornamento a Kubernetes 1.25.
- Trident ora impone rigorosamente l'uso della configurazione multipathing negli ambienti SAN, con un valore consigliato di `find_multipaths: no` nel file multipath.conf.

  Utilizzo di una configurazione non multipathing o utilizzo di `find_multipaths: yes` O `find_multipaths: smart` il valore nel file multipath.conf causerà errori di montaggio. Trident ha raccomandato l'uso di `find_multipaths: no` dalla versione 21.07.

Correzioni

- Risolto il problema specifico del backend ONTAP creato utilizzando `credentials` il campo non è riuscito a essere online durante l'aggiornamento 22.07.0 ("Numero 759").
- **Docker:** Risolto un problema che impediva l'avvio del plug-in del volume Docker in alcuni ambienti ("Numero 548" E"Numero 760").
- Risolto il problema SLM specifico dei backend SAN ONTAP per garantire che venga pubblicato solo un sottoinsieme di dataLIF appartenenti ai nodi di reporting.

- Risolto il problema di prestazioni per cui si verificavano scansioni non necessarie per iSCSI LUN durante il collegamento di un volume.
- Sono stati rimossi i nuovi tentativi granulari nel flusso di lavoro iSCSI Trident per evitare errori e ridurre gli intervalli di nuovi tentativi esterni.
- Risolto il problema per cui veniva restituito un errore durante lo svuotamento di un dispositivo iSCSI quando il dispositivo multipath corrispondente era già stato svuotato.

Miglioramenti

- Kubernetes:
  - Aggiunto supporto per Kubernetes 1.25. È necessario aggiornare Trident alla versione 22.10 prima di eseguire l'aggiornamento a Kubernetes 1.25.
  - Aggiunti ServiceAccount, ClusterRole e ClusterRoleBinding separati per Trident Deployment e DaemonSet per consentire futuri miglioramenti delle autorizzazioni.
  - Aggiunto supporto per"condivisione del volume tra più namespace" .
- Tutto Trident `ontap-*` i driver di archiviazione ora funzionano con l'API REST ONTAP .
- Aggiunto il nuovo operatore yaml(`bundle_post_1_25.yaml`) senza un `PodSecurityPolicy` per supportare Kubernetes 1.25.
- Aggiunto"supporto per volumi crittografati LUKS" per `ontap-san` E `ontap-san-economy` driver di archiviazione.
- Aggiunto supporto per i nodi di Windows Server 2019.
- Aggiunto"supporto per volumi SMB su nodi Windows" attraverso il `azure-netapp-files` driver di archiviazione.
- Il rilevamento automatico del passaggio MetroCluster a driver ONTAP è ora disponibile a livello generale.

Deprecazioni

- **Kubernetes:** Aggiornato il supporto minimo di Kubernetes alla versione 1.20.
- Rimosso il driver Astra Data Store (ADS).
- Rimosso il supporto per `yes` E `smart` opzioni per `find_multipaths` durante la configurazione del multipathing del nodo worker per iSCSI.

Modifiche nel 22.07

Correzioni

Kubernetes

- Risolto il problema relativo alla gestione dei valori booleani e numerici per il selettore di nodi durante la configurazione Trident con Helm o l'operatore Trident . ("Problema GitHub n. 700")
- Risolto il problema nella gestione degli errori provenienti da percorsi non CHAP, in modo che kubelet riprovi in caso di errore. "Problema GitHub n. 736")

Miglioramenti

- Passaggio da k8s.gcr.io a registry.k8s.io come registro predefinito per le immagini CSI

- I volumi ONTAP-SAN ora utilizzeranno igroup per nodo e mapperanno le LUN sugli igroup solo quando vengono pubblicate attivamente su tali nodi, per migliorare la nostra sicurezza. I volumi esistenti verranno opportunamente convertiti al nuovo schema igroup quando Trident stabilirà che è sicuro farlo senza influire sui carichi di lavoro attivi.
- È stato incluso un ResourceQuota con le installazioni Trident per garantire che Trident DaemonSet venga pianificato quando il consumo di PriorityClass è limitato per impostazione predefinita.
- Aggiunto il supporto per le funzionalità di rete al driver Azure NetApp Files . ("Problema GitHub n. 717")
- Aggiunta l'anteprima tecnica del rilevamento automatico del passaggio MetroCluster ai driver ONTAP . ("Problema GitHub n. 228")

Deprecazioni

- **Kubernetes:** Aggiornato il supporto minimo di Kubernetes alla versione 1.19.
- La configurazione del backend non consente più più tipi di autenticazione in una singola configurazione.

Traslochi

- Il driver AWS CVS (obsoleto dalla versione 22.04) è stato rimosso.
- Kubernetes
  - Rimossa la funzionalità SYS_ADMIN non necessaria dai pod dei nodi.
  - Riduce nodeprep a semplici informazioni sull'host e alla scoperta di servizi attivi per confermare con la massima efficacia che i servizi NFS/iSCSI sono disponibili sui nodi worker.

Documentazione

Un nuovo"Standard di sicurezza del pod" È stata aggiunta la sezione (PSS) che descrive in dettaglio le autorizzazioni abilitate da Trident durante l'installazione.

Modifiche nel 22.04

NetApp migliora e potenzia costantemente i suoi prodotti e servizi. Ecco alcune delle ultime funzionalità di Trident. Per le versioni precedenti, fare riferimento a "Versioni precedenti della documentazione" .

> Se si esegue l'aggiornamento da una versione precedente Trident e si utilizza Azure NetApp Files,`location` Il parametro config è ora un campo singleton obbligatorio.

Correzioni

- Analisi migliorata dei nomi degli iniziatori iSCSI. ("Problema GitHub n. 681")
- Risolto il problema per cui i parametri della classe di archiviazione CSI non erano consentiti. ("Problema GitHub n. 598")
- Corretta la dichiarazione di chiave duplicata in Trident CRD. ("Problema GitHub n. 671")
- Corretti i log CSI Snapshot imprecisi. ("Problema GitHub n. 629"))
- Risolto il problema relativo all'annullamento della pubblicazione dei volumi sui nodi eliminati. ("Problema GitHub n. 691")
- Aggiunta la gestione delle incongruenze del file system sui dispositivi a blocchi. ("Problema GitHub n. 656")

- Risolto il problema relativo all'estrazione delle immagini di supporto automatico durante l'impostazione del `imageRegistry` flag durante l'installazione. ("Problema GitHub n. 715")
- Risolto il problema per cui il driver Azure NetApp Files non riusciva a clonare un volume con più regole di esportazione.

Miglioramenti

- Le connessioni in entrata agli endpoint sicuri di Trident ora richiedono almeno TLS 1.3. ("Problema GitHub n. 698")
- Trident ora aggiunge intestazioni HSTS alle risposte provenienti dai suoi endpoint sicuri.
- Trident ora tenta di abilitare automaticamente la funzionalità di autorizzazioni Unix Azure NetApp Files .
- **Kubernetes**: il daemonset Trident ora viene eseguito con la classe di priorità critica del nodo di sistema. ("Problema GitHub n. 694")

Traslochi

Il driver E-Series (disattivato dalla versione 20.07) è stato rimosso.

Modifiche nel 22.01.1

Correzioni

- Risolto il problema relativo all'annullamento della pubblicazione dei volumi sui nodi eliminati. ("Problema GitHub n. 691")
- Risolto il problema di panico durante l'accesso ai campi nil per lo spazio aggregato nelle risposte API ONTAP .

Modifiche nel 22.01.0

Correzioni

- **Kubernetes:** Aumenta il tempo di ripetizione del backoff della registrazione dei nodi per cluster di grandi dimensioni.
- Risolto il problema per cui il driver azure-netapp-files poteva essere confuso da più risorse con lo stesso nome.
- I DataLIF ONTAP SAN IPv6 ora funzionano se specificati tra parentesi.
- Risolto il problema per cui il tentativo di importare un volume già importato restituiva EOF, lasciando PVC nello stato in sospeso. ("Problema GitHub n. 489")
- Risolto il problema per cui le prestazioni Trident rallentano quando vengono creati più di 32 snapshot su un volume SolidFire .
- Sostituito SHA-1 con SHA-256 nella creazione del certificato SSL.
- Corretto il driver Azure NetApp Files per consentire nomi di risorse duplicati e limitare le operazioni a un'unica posizione.
- Corretto il driver Azure NetApp Files per consentire nomi di risorse duplicati e limitare le operazioni a un'unica posizione.

Miglioramenti

- Miglioramenti di Kubernetes:
  - Aggiunto supporto per Kubernetes 1.23.
  - Aggiungere opzioni di pianificazione per i pod Trident quando installati tramite Trident Operator o Helm. ("Problema GitHub n. 651")
- Consenti volumi tra regioni diverse nel driver GCP. ("Problema GitHub n. 633")
- Aggiunto il supporto per l'opzione 'unixPermissions' ai volumi Azure NetApp Files . ("Problema GitHub n. 666")

Deprecazioni

L'interfaccia REST Trident può ascoltare e servire solo agli indirizzi 127.0.0.1 o [::1]

Modifiche nel 21.10.1

La versione v21.10.0 presenta un problema che può portare il controller Trident in uno stato CrashLoopBackOff quando un nodo viene rimosso e poi aggiunto nuovamente al cluster Kubernetes. Questo problema è stato risolto nella versione 21.10.1 (problema GitHub 669).

Correzioni

- Risolto il problema che causava una potenziale condizione di competizione durante l'importazione di un volume su un backend CVS GCP, con conseguente errore di importazione.
- È stato risolto un problema che poteva portare il controller Trident in uno stato CrashLoopBackOff quando un nodo veniva rimosso e poi aggiunto nuovamente al cluster Kubernetes (problema GitHub 669).
- Risolto il problema per cui le SVM non venivano più rilevate se non veniva specificato alcun nome SVM (problema GitHub 612).

Modifiche nel 21.10.0

Correzioni

- Risolto il problema per cui i cloni dei volumi XFS non potevano essere montati sullo stesso nodo del volume di origine (problema GitHub 514).
- Risolto il problema per cui Trident registrava un errore fatale durante l'arresto (problema GitHub 597).
- Correzioni relative a Kubernetes:
  - Restituisce lo spazio utilizzato di un volume come restoreSize minimo quando si creano snapshot con `ontap-nas` E `ontap-nas-flexgroup` driver (problema GitHub 645).
  - Risolto il problema per cui `Failed to expand filesystem` è stato registrato un errore dopo il ridimensionamento del volume (problema GitHub 560).
  - Risolto il problema per cui un pod poteva rimanere bloccato `Terminating` stato (problema GitHub 572).
  - Risolto il caso in cui un `ontap-san-economy` FlexVol potrebbe essere pieno di LUN snapshot (problema GitHub 533).
  - Risolto il problema dell'installer YAML personalizzato con un'immagine diversa (problema GitHub 613).

- Corretto il calcolo delle dimensioni dello snapshot (problema GitHub 611).
- Risolto il problema per cui tutti gli installatori Trident potevano identificare Kubernetes semplice come OpenShift (problema GitHub 639).
- Corretto l'operatore Trident per interrompere la riconciliazione se il server API Kubernetes non è raggiungibile (problema GitHub 599).

Miglioramenti

- Aggiunto supporto per `unixPermissions` opzione per i volumi GCP-CVS Performance.
- Aggiunto supporto per volumi CVS ottimizzati in scala in GCP nell'intervallo da 600 GiB a 1 TiB.
- Miglioramenti relativi a Kubernetes:
  - Aggiunto supporto per Kubernetes 1.22.
  - Abilitato l'operatore Trident e il grafico Helm per funzionare con Kubernetes 1.22 (problema GitHub 628).
  - Aggiunta l'immagine dell'operatore a `tridentctl` comando images (problema GitHub 570).

Miglioramenti sperimentali

- Aggiunto supporto per la replica del volume in `ontap-san` autista.
- Aggiunto il supporto REST **anteprima tecnica** per `ontap-nas-flexgroup` , `ontap-san` , E `ontap-nas-economy` conducenti.

Problemi noti

I problemi noti identificano i problemi che potrebbero impedirti di utilizzare il prodotto correttamente.

- Quando si aggiorna un cluster Kubernetes da 1.24 a 1.25 o versione successiva su cui è installato Trident , è necessario aggiornare values.yaml per impostare `excludePodSecurityPolicy` A `true` o aggiungere `--set excludePodSecurityPolicy=true` al `helm upgrade` comando prima di poter aggiornare il cluster.
- Trident ora impone un vuoto `fsType` (`fsType=""`) per i volumi che non hanno il `fsType` specificato nella loro StorageClass. Quando si lavora con Kubernetes 1.17 o versioni successive, Trident supporta la fornitura di un vuoto `fsType` per volumi NFS. Per i volumi iSCSI, è necessario impostare `fsType` sul tuo StorageClass quando imponi un `fsGroup` utilizzando un contesto di sicurezza.
- Quando si utilizza un backend su più istanze Trident , ogni file di configurazione del backend dovrebbe avere un diverso `storagePrefix` valore per i backend ONTAP o utilizzare un valore diverso `TenantName` per i backend SolidFire . Trident non riesce a rilevare i volumi creati da altre istanze di Trident . Il tentativo di creare un volume esistente sui backend ONTAP o SolidFire riesce, perché Trident tratta la creazione del volume come un'operazione idempotente. Se `storagePrefix` O `TenantName` non differiscono, potrebbero verificarsi conflitti di nomi per i volumi creati sullo stesso backend.
- Durante l'installazione Trident (utilizzando `tridentctl` o l'operatore Trident) e utilizzando `tridentctl` per gestire Trident, dovresti assicurarti che `KUBECONFIG` la variabile d'ambiente è impostata. Ciò è necessario per indicare il cluster Kubernetes che `tridentctl` dovrebbe lavorare contro. Quando si lavora con più ambienti Kubernetes, è necessario assicurarsi che `KUBECONFIG` il file è di provenienza accurata.
- Per eseguire il recupero dello spazio online per i PV iSCSI, il sistema operativo sottostante sul nodo worker potrebbe richiedere che le opzioni di montaggio vengano passate al volume. Ciò è vero per le istanze RHEL/Red Hat Enterprise Linux CoreOS (RHCOS), che richiedono `discard` "opzione di montaggio" ;

assicurati che l'opzione di montaggio scarto sia inclusa nel tuo[`StorageClass` ^] per supportare lo scarto dei blocchi online.

- Se si dispone di più di un'istanza di Trident per cluster Kubernetes, Trident non può comunicare con altre istanze e non può rilevare altri volumi da esso creati, il che comporta un comportamento imprevisto e non corretto se più di un'istanza viene eseguita all'interno di un cluster. Dovrebbe esserci una sola istanza di Trident per cluster Kubernetes.
- Se basato su Trident `StorageClass` gli oggetti vengono eliminati da Kubernetes mentre Trident è offline, Trident non rimuove le classi di archiviazione corrispondenti dal suo database quando torna online. Dovresti eliminare queste classi di archiviazione utilizzando `tridentctl` o l'API REST.
- Se un utente elimina un PV fornito da Trident prima di eliminare il PVC corrispondente, Trident non elimina automaticamente il volume di supporto. Dovresti rimuovere il volume tramite `tridentctl` o l'API REST.
- ONTAP non può fornire contemporaneamente più di un FlexGroup alla volta, a meno che il set di aggregati non sia univoco per ogni richiesta di provisioning.
- Quando si utilizza Trident su IPv6, è necessario specificare `managementLIF` E `dataLIF` nella definizione del backend tra parentesi quadre. Per esempio, `[fd20:8b1e:b258:2000:f816:3eff:feec:0]` .

> Non puoi specificare `dataLIF` su un backend ONTAP SAN. Trident rileva tutti i LIF iSCSI disponibili e li utilizza per stabilire la sessione multipath.

- Se si utilizza il `solidfire-san` driver con OpenShift 4.5, assicurarsi che i nodi worker sottostanti utilizzino MD5 come algoritmo di autenticazione CHAP. Con Element 12.7 sono disponibili gli algoritmi CHAP sicuri conformi allo standard FIPS SHA1, SHA-256 e SHA3-256.

Trova maggiori informazioni

- "Trident GitHub"
- "Blog Trident"

Versioni precedenti della documentazione

Se non si utilizza Trident 25.06, la documentazione per le versioni precedenti è disponibile in base a"Ciclo di vita del supporto Trident" .

- "Trident 25.02"
- "Trident 24.10"
- "Trident 24.06"
- "Trident 24.02"
- "Trident 23.10"
- "Trident 23.07"
- "Trident 23.04"
- "Trident 23.01"
- "Trident 22.10"

Problemi noti

I problemi noti identificano i problemi che potrebbero impedirti di utilizzare correttamente questa versione del prodotto.

I seguenti problemi noti interessano la versione corrente:

Il ripristino dei backup Restic di file di grandi dimensioni potrebbe non riuscire

Quando si ripristinano file da 30 GB o più da un backup Amazon S3 effettuato tramite Restic, l'operazione di ripristino potrebbe non riuscire. Come soluzione alternativa, esegui il backup dei dati utilizzando Kopia come strumento di spostamento dati (Kopia è lo strumento di spostamento dati predefinito per i backup). Fare riferimento a "Proteggi le applicazioni utilizzando Trident Protect" per istruzioni.

Iniziare

Scopri di più su Trident

Scopri di più su Trident

Trident è un progetto open source completamente supportato e gestito da NetApp. È stato progettato per aiutarti a soddisfare le esigenze di persistenza delle tue applicazioni containerizzate utilizzando interfacce standard del settore, come Container Storage Interface (CSI).

Che cos'è Trident?

Netapp Trident consente l'utilizzo e la gestione delle risorse di storage su tutte le piattaforme di storage NetApp più diffuse, nel cloud pubblico o in sede, inclusi i cluster ONTAP in sede (AFF, FAS e ASA), ONTAP Select, Cloud Volumes ONTAP, Element software (NetApp HCI, SolidFire), Azure NetApp Files, Amazon FSx for NetApp ONTAP e Cloud Volumes Service su Google Cloud.

Trident è un orchestratore di storage dinamico conforme a Container Storage Interface (CSI) che si integra in modo nativo con"Kubernetes" . Trident viene eseguito come un singolo Controller Pod più un Node Pod su ciascun nodo worker nel cluster. Fare riferimento a "Architettura Trident" per i dettagli.

Trident fornisce anche l'integrazione diretta con l'ecosistema Docker per le piattaforme di storage NetApp . Il NetApp Docker Volume Plugin (nDVP) supporta il provisioning e la gestione delle risorse di storage dalla piattaforma di storage agli host Docker. Fare riferimento a "Distribuisci Trident per Docker" per i dettagli.

Se è la prima volta che utilizzi Kubernetes, dovresti familiarizzare con"Concetti e strumenti di Kubernetes" .

Integrazione di Kubernetes con i prodotti NetApp

Il portfolio di prodotti di storage NetApp si integra con molti aspetti di un cluster Kubernetes, offrendo funzionalità avanzate di gestione dei dati che migliorano la funzionalità, le capacità, le prestazioni e la disponibilità della distribuzione Kubernetes.

Amazon FSx for NetApp ONTAP

"Amazon FSx for NetApp ONTAP"è un servizio AWS completamente gestito che consente di avviare ed eseguire file system basati sul sistema operativo di storage NetApp ONTAP .

Azure NetApp Files

"Azure NetApp Files"è un servizio di condivisione file Azure di livello aziendale, basato su NetApp. Puoi eseguire i carichi di lavoro basati su file più impegnativi in Azure in modo nativo, con le prestazioni e la gestione avanzata dei dati che ti aspetti da NetApp.

Cloud Volumes ONTAP

"Cloud Volumes ONTAP"è un dispositivo di archiviazione esclusivamente software che esegue il software di gestione dati ONTAP nel cloud.

Google Cloud NetApp Volumes

"Google Cloud NetApp Volumes"è un servizio di archiviazione file completamente gestito su Google Cloud che fornisce archiviazione file di livello aziendale e ad alte prestazioni.

Software Element

"Elemento"consente all'amministratore dello storage di consolidare i carichi di lavoro garantendo le prestazioni e consentendo un ingombro di storage semplificato e ottimizzato.

NetApp HCI

"NetApp HCI"semplifica la gestione e la scalabilità del data center automatizzando le attività di routine e consentendo agli amministratori dell'infrastruttura di concentrarsi su funzioni più importanti.

Trident può fornire e gestire dispositivi di storage per applicazioni containerizzate direttamente sulla piattaforma di storage NetApp HCI sottostante.

NetApp ONTAP

"NetApp ONTAP"è il sistema operativo di storage unificato multiprotocollo NetApp che fornisce funzionalità avanzate di gestione dei dati per qualsiasi applicazione.

I sistemi ONTAP hanno configurazioni all-flash, ibride o all-HDD e offrono molti modelli di distribuzione diversi: cluster FAS, AFA e ASA on-premise, ONTAP Select e Cloud Volumes ONTAP. Trident supporta questi modelli di distribuzione ONTAP .

Architettura Trident

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

Comprensione dei pod controller e dei pod node

Trident si schiera come un singoloPod di controllo Trident e uno o piùPod del nodo Trident sul cluster Kubernetes e utilizza i contenitori laterali CSI Kubernetes standard per semplificare la distribuzione dei plugin CSI. "Contenitori Sidecar Kubernetes CSI" sono gestiti dalla community Kubernetes Storage.

Kubernetes"selettori di nodo" E"tolleranze e contaminazioni" vengono utilizzati per vincolare un pod all'esecuzione su un nodo specifico o preferito. Durante l'installazione Trident è possibile configurare i selettori dei nodi e le tolleranze per i controller e i pod dei nodi.

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

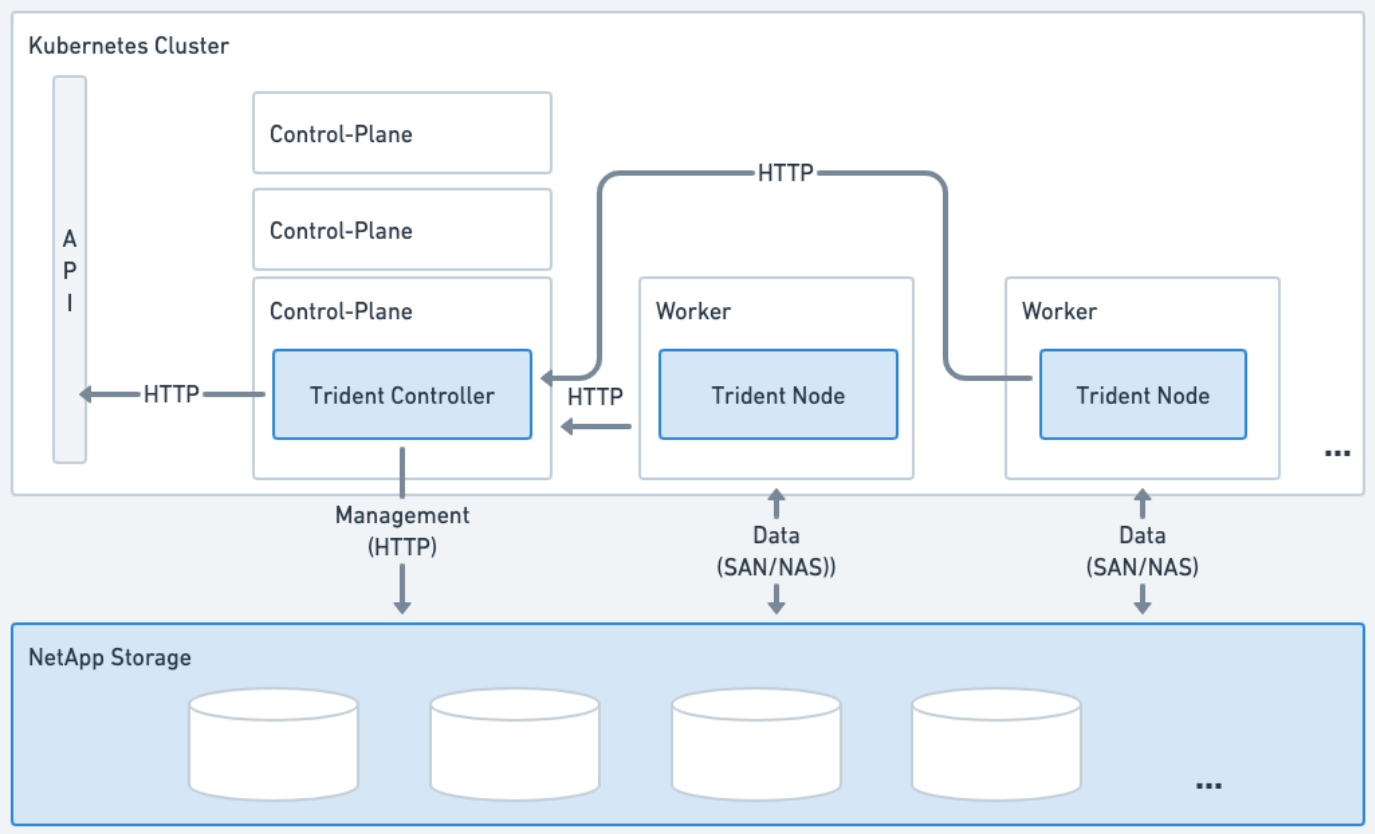


Figura 1. Trident distribuito sul cluster Kubernetes

Pod di controllo Trident

Il Trident Controller Pod è un singolo Pod che esegue il plugin CSI Controller.

- Responsabile del provisioning e della gestione dei volumi nello storage NetApp
- Gestito da una distribuzione Kubernetes
- Può essere eseguito sul piano di controllo o sui nodi worker, a seconda dei parametri di installazione.

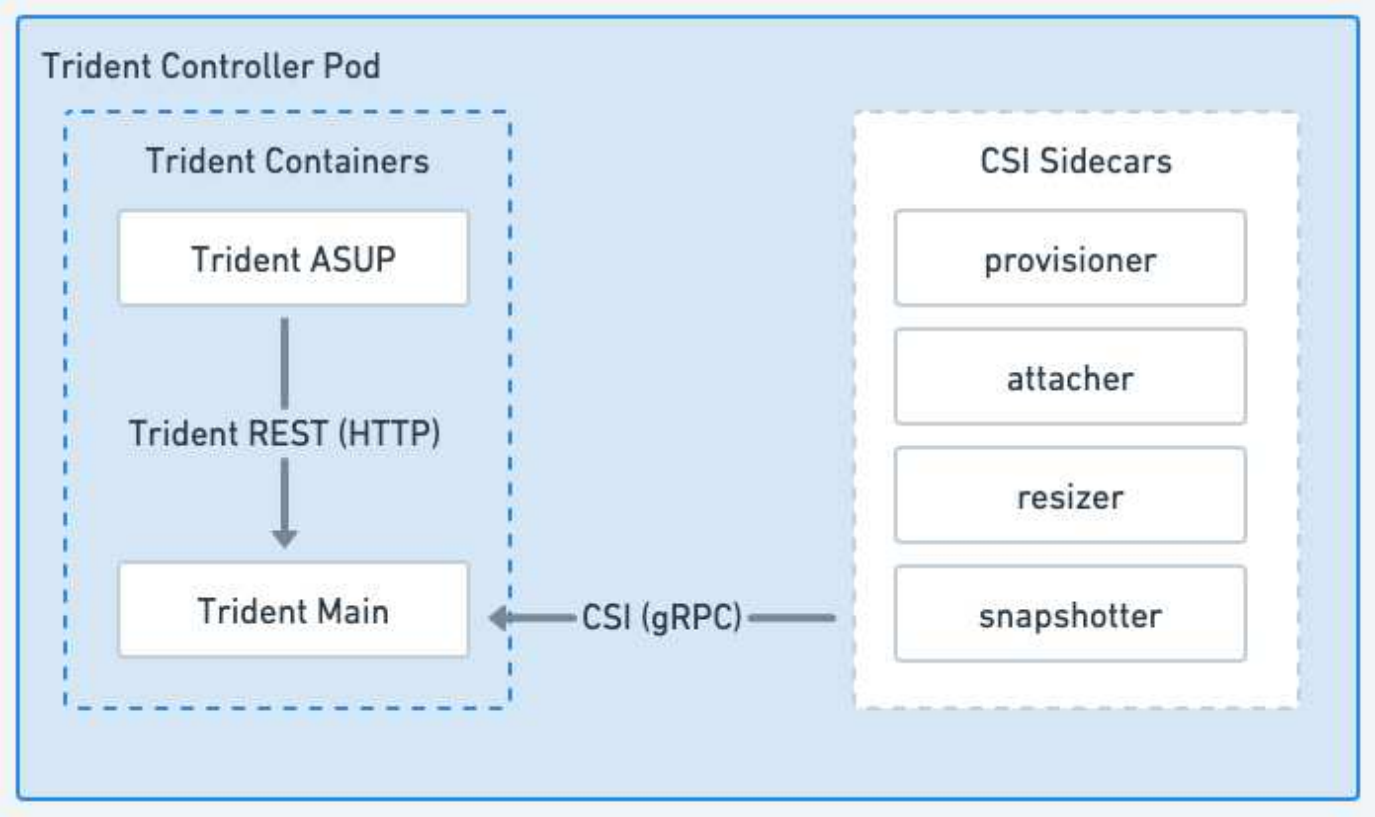


Figura 2. Diagramma del pod di controllo Trident

Pod del nodo Trident

I Trident Node Pod sono Pod privilegiati che eseguono il plugin CSI Node.

- Responsabile del montaggio e dello smontaggio dello storage per i Pod in esecuzione sull'host
- Gestito da un Kubernetes DaemonSet
- Deve essere eseguito su qualsiasi nodo che monterà lo storage NetApp

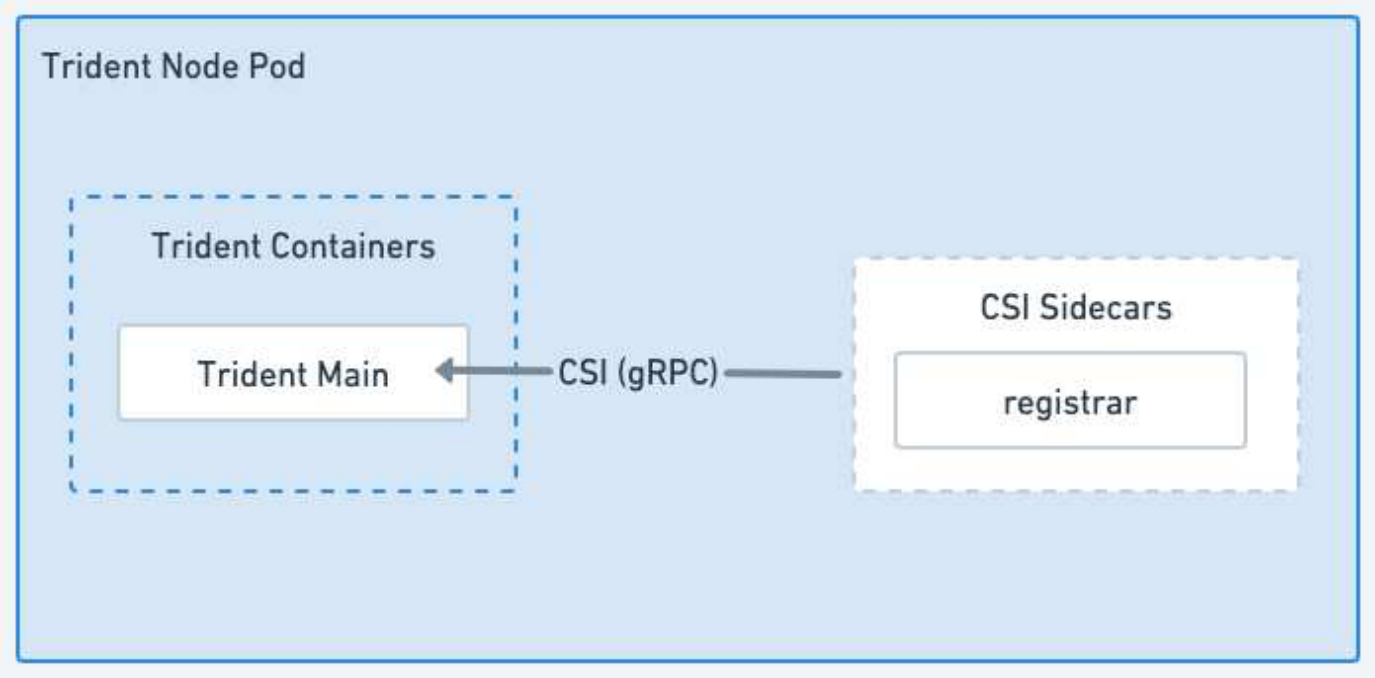


Figura 3. Diagramma del pod del nodo Trident

Architetture cluster Kubernetes supportate

Trident è supportato dalle seguenti architetture Kubernetes:

Architetture dei cluster Kubernetes	Supportato	Installazione predefinita
Master singolo, calcolo	SÌ	SÌ
Master multiplo, calcolo	SÌ	SÌ
Maestro, `etcd` , calcolare	SÌ	SÌ
Master, infrastruttura, calcolo	SÌ	SÌ

Concetti

Approvvigionamento

Il provisioning in Trident si articola in due fasi principali. La prima fase associa una classe di archiviazione al set di pool di archiviazione back-end idonei e si verifica come necessaria preparazione prima del provisioning. La seconda fase comprende la creazione del volume vero e proprio e richiede la scelta di un pool di archiviazione tra quelli associati alla classe di archiviazione del volume in sospeso.

Associazione della classe di archiviazione

L'associazione di pool di archiviazione back-end con una classe di archiviazione si basa sia sugli attributi richiesti della classe di archiviazione che sui suoi `storagePools` , `additionalStoragePools` , E `excludeStoragePools` liste. Quando si crea una classe di archiviazione, Trident confronta gli attributi e i

pool offerti da ciascuno dei suoi backend con quelli richiesti dalla classe di archiviazione. Se gli attributi e il nome di un pool di archiviazione corrispondono a tutti gli attributi e ai nomi dei pool richiesti, Trident aggiunge tale pool di archiviazione al set di pool di archiviazione idonei per quella classe di archiviazione. Inoltre, Trident aggiunge tutti i pool di archiviazione elencati in `additionalStoragePools` elenco a tale set, anche se i loro attributi non soddisfano tutti o alcuni degli attributi richiesti dalla classe di archiviazione. Dovresti usare il `excludeStoragePools` elenco per sovrascrivere e rimuovere i pool di archiviazione dall'uso per una classe di archiviazione. Trident esegue un processo simile ogni volta che si aggiunge un nuovo backend, verificando se i suoi pool di archiviazione soddisfano quelli delle classi di archiviazione esistenti e rimuovendo quelli contrassegnati come esclusi.

Creazione del volume

Trident utilizza quindi le associazioni tra classi di archiviazione e pool di archiviazione per determinare dove effettuare il provisioning dei volumi. Quando si crea un volume, Trident ottiene innanzitutto il set di pool di archiviazione per la classe di archiviazione di quel volume e, se si specifica un protocollo per il volume, Trident rimuove i pool di archiviazione che non possono fornire il protocollo richiesto (ad esempio, un backend NetApp HCI/ SolidFire non può fornire un volume basato su file mentre un backend ONTAP NAS non può fornire un volume basato su blocchi). Trident randomizza l'ordine di questo set risultante, per facilitare una distribuzione uniforme dei volumi, e quindi lo esegue iterativamente, tentando di effettuare il provisioning del volume su ciascun pool di archiviazione a turno. Se riesce in una delle due, ritorna correttamente, registrando tutti gli errori riscontrati nel processo. Trident restituisce un errore **solo** se non riesce a eseguire il provisioning su **tutti** i pool di archiviazione disponibili per la classe di archiviazione e il protocollo richiesti.

Istantanee del volume

Scopri di più su come Trident gestisce la creazione di snapshot di volume per i suoi driver.

Scopri di più sulla creazione di snapshot del volume

- Per il `ontap-nas` , `ontap-san` , `gcp-cvs` , E `azure-netapp-files` driver, ogni Persistent Volume (PV) viene mappato su un FlexVol volume. Di conseguenza, gli snapshot del volume vengono creati come snapshot NetApp . La tecnologia snapshot NetApp offre maggiore stabilità, scalabilità, recuperabilità e prestazioni rispetto alle tecnologie snapshot della concorrenza. Queste copie snapshot sono estremamente efficienti sia in termini di tempo necessario per crearle sia di spazio di archiviazione.
- Per il `ontap-nas-flexgroup` driver, ogni Persistent Volume (PV) viene mappato su un FlexGroup. Di conseguenza, gli snapshot del volume vengono creati come snapshot NetApp FlexGroup . La tecnologia snapshot NetApp offre maggiore stabilità, scalabilità, recuperabilità e prestazioni rispetto alle tecnologie snapshot della concorrenza. Queste copie snapshot sono estremamente efficienti sia in termini di tempo necessario per crearle sia di spazio di archiviazione.
- Per il `ontap-san-economy` driver, i PV vengono mappati sui LUN creati sui volumi FlexVol condivisi. I VolumeSnapshot dei PV vengono ottenuti eseguendo FlexClone del LUN associato. La tecnologia ONTAP FlexClone consente di creare copie anche dei set di dati più grandi in modo quasi istantaneo. Le copie condividono blocchi di dati con i loro genitori, senza consumare spazio di archiviazione, se non quello necessario per i metadati.
- Per il `solidfire-san` driver, ogni PV viene mappato su una LUN creata sul software NetApp Element /cluster NetApp HCI . I VolumeSnapshot sono rappresentati da snapshot Element della LUN sottostante. Questi snapshot sono copie puntuali e occupano solo una piccola quantità di risorse e spazio di sistema.
- Quando si lavora con il `ontap-nas` E `ontap-san` driver, gli snapshot ONTAP sono copie puntuali del FlexVol e occupano spazio sul FlexVol stesso. Ciò può comportare una riduzione nel tempo della quantità di spazio scrivibile nel volume man mano che vengono creati/pianificati gli snapshot. Un modo semplice per risolvere questo problema è aumentare il volume ridimensionandolo tramite Kubernetes. Un'altra

opzione è quella di eliminare gli snapshot che non sono più necessari. Quando un VolumeSnapshot creato tramite Kubernetes viene eliminato, Trident eliminerà lo snapshot ONTAP associato. Anche gli snapshot ONTAP che non sono stati creati tramite Kubernetes possono essere eliminati.

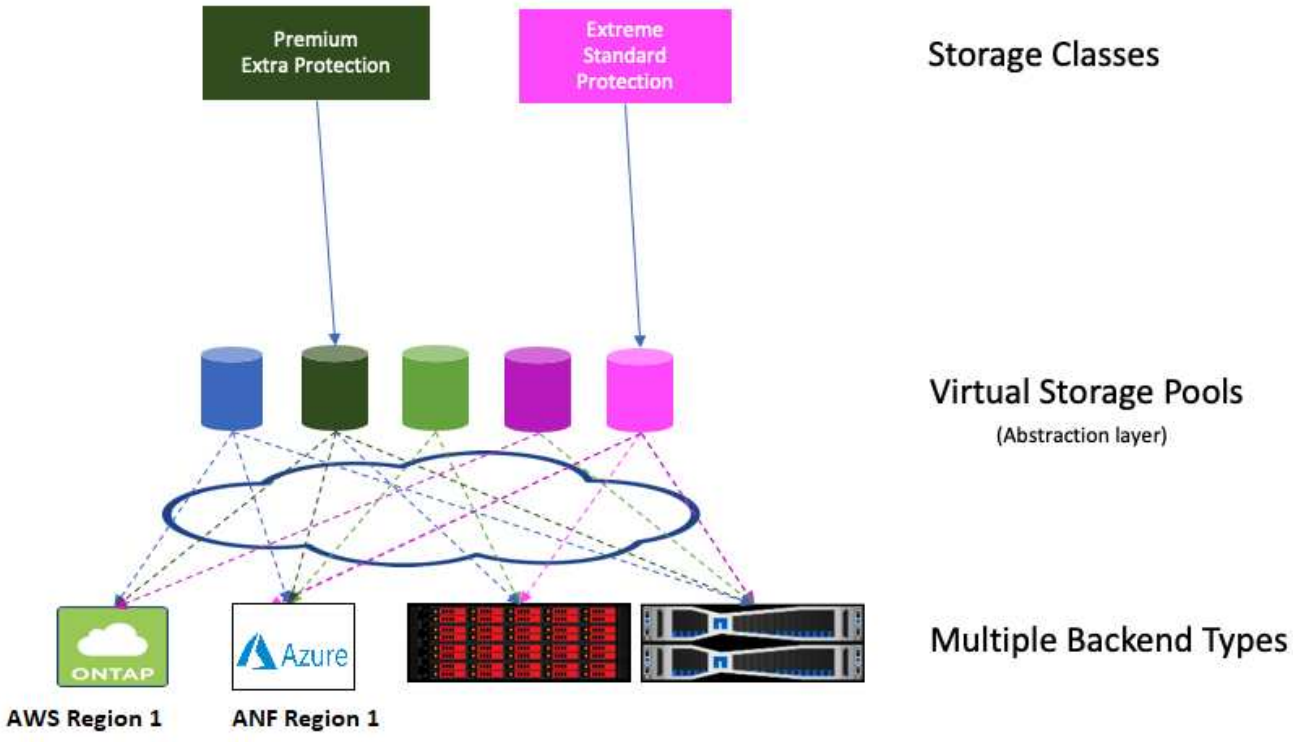
Con Trident, puoi usare VolumeSnapshots per creare nuovi PV da essi. La creazione di PV da questi snapshot viene eseguita utilizzando la tecnologia FlexClone per i backend ONTAP e CVS supportati. Quando si crea un PV da uno snapshot, il volume di supporto è un FlexClone del volume padre dello snapshot. IL `solidfire-san` il driver utilizza cloni di volume del software Element per creare PV da snapshot. Qui viene creato un clone dallo snapshot dell'Elemento.

Pool virtuali

I pool virtuali forniscono un livello di astrazione tra i backend di archiviazione Trident e Kubernetes `StorageClasses` . Consentono a un amministratore di definire aspetti quali posizione, prestazioni e protezione per ciascun backend in un modo comune e indipendente dal backend, senza dover effettuare una `StorageClass` specificare quale backend fisico, pool di backend o tipo di backend utilizzare per soddisfare i criteri desiderati.

Scopri di più sui pool virtuali

L'amministratore dell'archiviazione può definire pool virtuali su qualsiasi backend Trident in un file di definizione JSON o YAML.



Qualsiasi aspetto specificato al di fuori dell'elenco dei pool virtuali è globale per il backend e verrà applicato a tutti i pool virtuali, mentre ogni pool virtuale potrebbe specificare uno o più aspetti individualmente (sovrascrivendo qualsiasi aspetto globale del backend).

- Quando si definiscono pool virtuali, non tentare di riorganizzare l'ordine dei pool virtuali esistenti in una definizione backend.
- Sconsigliamo di modificare gli attributi di un pool virtuale esistente. Per apportare modifiche, è necessario definire un nuovo pool virtuale.

La maggior parte degli aspetti sono specificati in termini specifici del backend. Fondamentalmente, i valori di aspetto non sono esposti all'esterno del driver del backend e non sono disponibili per la corrispondenza in `StorageClasses` L'amministratore definisce invece una o più etichette per ogni pool virtuale. Ogni etichetta è una coppia chiave:valore e le etichette potrebbero essere comuni a backend univoci. Come gli aspetti, le etichette possono essere specificate per pool o globalmente nel backend. A differenza degli aspetti, che hanno nomi e valori predefiniti, l'amministratore ha piena discrezionalità nel definire chiavi e valori delle etichette in base alle necessità. Per comodità, gli amministratori di storage possono definire etichette per pool virtuale e raggruppare i volumi in base all'etichetta.

Le etichette del pool virtuale possono essere definite utilizzando questi caratteri:

- lettere maiuscole `A-Z`
- lettere minuscole `a-z`
- numeri `0-9`
- sottolineature _
- trattini -

UN `StorageClass` identifica quale pool virtuale utilizzare facendo riferimento alle etichette all'interno di un parametro selettore. I selettori di pool virtuali supportano i seguenti operatori:

Operatore	Esempio	Il valore dell'etichetta di un pool deve:
`=`	prestazioni=premium	Incontro
`!=`	prestazioni!=estreme	Non corrisponde
`in`	posizione in (est, ovest)	Essere nell'insieme dei valori
`notin`	prestazioni notin (argento, bronzo)	Non essere nell'insieme dei valori
`<key>`	protezione	Esiste con qualsiasi valore
`!<key>`	!protezione	Non esistere

Gruppi di accesso al volume

Scopri di più su come Trident utilizza "gruppi di accesso al volume" .

Ignorare questa sezione se si utilizza CHAP, opzione consigliata per semplificare la gestione ed evitare il limite di scalabilità descritto di seguito. Inoltre, se si utilizza Trident in modalità CSI, è possibile ignorare questa sezione. Trident utilizza CHAP quando installato come provisioner CSI avanzato.

Scopri di più sui gruppi di accesso al volume

Trident può utilizzare gruppi di accesso al volume per controllare l'accesso ai volumi di cui si occupa. Se CHAP è disabilitato, si aspetta di trovare un gruppo di accesso chiamato `trident` a meno che non si specifichino

uno o più ID di gruppo di accesso nella configurazione.

Sebbene Trident associ nuovi volumi ai gruppi di accesso configurati, non crea né gestisce in altro modo i gruppi di accesso stessi. I gruppi di accesso devono esistere prima che il backend di archiviazione venga aggiunto a Trident e devono contenere gli IQN iSCSI da ogni nodo nel cluster Kubernetes che potrebbe potenzialmente montare i volumi forniti da quel backend. Nella maggior parte delle installazioni, ciò include ogni nodo worker nel cluster.

Per i cluster Kubernetes con più di 64 nodi, è consigliabile utilizzare più gruppi di accesso. Ogni gruppo di accesso può contenere fino a 64 IQN e ogni volume può appartenere a quattro gruppi di accesso. Con un massimo di quattro gruppi di accesso configurati, qualsiasi nodo in un cluster di dimensioni fino a 256 nodi sarà in grado di accedere a qualsiasi volume. Per i limiti più recenti sui gruppi di accesso al volume, fare riferimento a "Qui" .

Se stai modificando la configurazione da una che utilizza quella predefinita `trident` gruppo di accesso a uno che ne utilizza anche altri, includere l'ID per il `trident` gruppo di accesso nell'elenco.

Avvio rapido per Trident

Puoi installare Trident e iniziare a gestire le risorse di archiviazione in pochi passaggi. Prima di iniziare, rivedere"Requisiti Trident" .

Per Docker, fare riferimento a"Trident per Docker" .

1 Preparare il nodo worker

Tutti i nodi worker nel cluster Kubernetes devono essere in grado di montare i volumi forniti per i pod.

"Preparare il nodo worker"

2 Installa Trident

Trident offre diversi metodi e modalità di installazione ottimizzati per una varietà di ambienti e organizzazioni.

"Installa Trident"

3 Crea un backend

Un backend definisce la relazione tra Trident e un sistema di archiviazione. Indica a Trident come comunicare con quel sistema di archiviazione e come Trident deve effettuare il provisioning dei volumi da esso.

"Configurare un backend"per il tuo sistema di archiviazione

4 Creare una classe di archiviazione Kubernetes

L'oggetto StorageClass di Kubernetes specifica Trident come provisioner e consente di creare una classe di archiviazione per eseguire il provisioning di volumi con attributi personalizzabili. Trident crea una classe di archiviazione corrispondente per gli oggetti Kubernetes che specificano il provisioner Trident .

"Creare una classe di archiviazione"

5 **Fornire un volume**

Un *PersistentVolume* (PV) è una risorsa di archiviazione fisica fornita dall'amministratore del cluster su un cluster Kubernetes. *PersistentVolumeClaim* (PVC) è una richiesta di accesso al PersistentVolume sul cluster.

Creare un PersistentVolume (PV) e un PersistentVolumeClaim (PVC) che utilizzi la StorageClass Kubernetes configurata per richiedere l'accesso al PV. È quindi possibile montare il fotovoltaico su un pod.

"Fornire un volume"

Cosa succederà ora?

Ora puoi aggiungere backend aggiuntivi, gestire classi di archiviazione, gestire backend ed eseguire operazioni sui volumi.

Requisiti

Prima di installare Trident dovresti rivedere questi requisiti generali di sistema. I backend specifici potrebbero avere requisiti aggiuntivi.

Informazioni critiche su Trident

È necessario leggere le seguenti informazioni critiche su Trident.

Informazioni critiche su Trident

- Kubernetes 1.34 è ora supportato in Trident. Aggiornare Trident prima di aggiornare Kubernetes.
- Trident impone rigorosamente l'uso della configurazione multipathing negli ambienti SAN, con un valore consigliato di `find_multipaths: no` nel file multipath.conf.

  Utilizzo di una configurazione non multipathing o utilizzo di `find_multipaths: yes` O `find_multipaths: smart` il valore nel file multipath.conf causerà errori di montaggio. Trident ha raccomandato l'uso di `find_multipaths: no` dalla versione 21.07.

Frontend supportati (orchestratori)

Trident supporta più motori di container e orchestratori, tra cui:

- Anthos On-Prem (VMware) e Anthos su bare metal 1.16
- Kubernetes 1.27 - 1.34
- OpenShift 4.12, 4.14 - 4.19 (se si prevede di utilizzare la preparazione del nodo iSCSI con OpenShift 4.19, la versione minima supportata Trident è 25.06.1.)

Trident continua a supportare le versioni precedenti di OpenShift in linea con"Ciclo di vita della versione Red Hat Extended Update Support (EUS)" , anche se si basano su versioni di Kubernetes che non sono più ufficialmente supportate a monte. In questi casi, durante l'installazione Trident , puoi tranquillamente ignorare eventuali messaggi di avviso relativi alla versione di Kubernetes.

- Rancher Kubernetes Engine 2 (RKE2) v1.27.x - 1.34.x

Sebbene Trident sia supportato su Rancher Kubernetes Engine 2 (RKE2) versioni 1.27.x - 1.34.x, Trident è attualmente qualificato solo su RKE2 v1.28.5+rke2r1.

Trident funziona anche con una serie di altre offerte Kubernetes completamente gestite e autogestite, tra cui Google Kubernetes Engine (GKE), Amazon Elastic Kubernetes Services (EKS), Azure Kubernetes Service (AKS), Mirantis Kubernetes Engine (MKE) e VMWare Tanzu Portfolio.

Trident e ONTAP possono essere utilizzati come provider di storage per"KubeVirt" .

Prima di aggiornare un cluster Kubernetes dalla versione 1.25 alla versione 1.26 o successiva su cui è installato Trident , fare riferimento a"Aggiornare un'installazione di Helm" .

Backend supportati (archiviazione)

Per utilizzare Trident, è necessario uno o più dei seguenti backend supportati:

- Amazon FSx for NetApp ONTAP
- Azure NetApp Files
- Cloud Volumes ONTAP
- Google Cloud NetApp Volumes
- Array NetApp All SAN (ASA)
- Versioni di cluster FAS, AFF, Select o ASA r2 (iSCSI e NVMe/TCP) in locale con supporto limitato di NetApp. Vedere"Supporto della versione software" .
- Software NetApp HCI/Element 11 o superiore

Supporto Trident per KubeVirt e OpenShift Virtualization

Driver di archiviazione supportati:

Trident supporta i seguenti driver ONTAP per KubeVirt e OpenShift Virtualization:

- ontap-nas
- ontap-nas-economy
- ontap-san (iSCSI, FCP, NVMe su TCP)
- ontap-san-economy (solo iSCSI)

Punti da considerare:

- Aggiorna la classe di archiviazione per avere `fsType` parametro (ad esempio: `fsType: "ext4"`) nell'ambiente di virtualizzazione OpenShift. Se necessario, impostare la modalità volume per bloccare esplicitamente utilizzando `volumeMode=Block` parametro nel `dataVolumeTemplates` per notificare a

CDI di creare volumi di dati a blocchi.

- *Modalità di accesso RWX per driver di archiviazione a blocchi*: i driver ontap-san (iSCSI, NVMe/TCP, FC) e ontap-san-economy (iSCSI) sono supportati solo con "volumeMode: Block" (dispositivo raw). Per questi conducenti, il `fstype` Il parametro non può essere utilizzato perché i volumi sono forniti in modalità dispositivo raw.
- Per i flussi di lavoro di migrazione in tempo reale in cui è richiesta la modalità di accesso RWX, sono supportate le seguenti combinazioni:
  - NFS + `volumeMode=Filesystem`
  - iSCSI + `volumeMode=Block` (dispositivo grezzo)
  - NVMe/TCP + `volumeMode=Block` (dispositivo grezzo)
  - FC + `volumeMode=Block` (dispositivo grezzo)

Requisiti delle funzionalità

La tabella seguente riassume le funzionalità disponibili con questa versione di Trident e le versioni di Kubernetes supportate.

Caratteristica	Versione di Kubernetes	Sono richiesti cancelli di funzionalità?
Trident	1,27 - 1,34	NO
Istantanee del volume	1,27 - 1,34	NO
PVC da istantanee di volume	1,27 - 1,34	NO
Ridimensionamento PV iSCSI	1,27 - 1,34	NO
ONTAP CHAP bidirezionale	1,27 - 1,34	NO
Politiche di esportazione dinamiche	1,27 - 1,34	NO
Operatore Trident	1,27 - 1,34	NO
Topologia CSI	1,27 - 1,34	NO

Sistemi operativi host testati

Sebbene Trident non supporti ufficialmente sistemi operativi specifici, è noto che i seguenti funzionano:

- Versioni di Red Hat Enterprise Linux CoreOS (RHCOS) supportate da OpenShift Container Platform (AMD64 e ARM64)
- RHEL 8+ (AMD64 e ARM64)

> NVMe/TCP richiede RHEL 9 o versione successiva.

- Ubuntu 22.04 o successivo (AMD64 e ARM64)
- Windows Server 2022

Per impostazione predefinita, Trident viene eseguito in un contenitore e, pertanto, può essere eseguito su qualsiasi worker Linux. Tuttavia, tali lavoratori devono essere in grado di montare i volumi forniti Trident utilizzando il client NFS standard o l'iniziatore iSCSI, a seconda dei backend utilizzati.

IL `tridentctl` L'utilità funziona anche su una qualsiasi di queste distribuzioni di Linux.

Configurazione host

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

"Preparare il nodo worker"

Configurazione del sistema di archiviazione

Trident potrebbe richiedere modifiche al sistema di archiviazione prima che una configurazione backend possa utilizzarlo.

"Configurare i backend"

Porte Trident

Trident necessita di accesso a porte specifiche per la comunicazione.

"Porte Trident"

Immagini dei container e versioni corrispondenti di Kubernetes

Per le installazioni con air gap, l'elenco seguente è un riferimento delle immagini dei contenitori necessarie per installare Trident. Utilizzare il `tridentctl images` comando per verificare l'elenco delle immagini contenitore necessarie.

Immagini del contenitore richieste per Trident 25.06.2

Versioni di Kubernetes	Immagine del contenitore
v1.27.0, v1.28.0, v1.29.0, v1.30.0, v1.31.0, v1.32.0, v1.33.0, v1.34.0	• docker.io/netapp/trident:25.06.2 • docker.io/netapp/trident-autosupport:25.06 • registry.k8s.io/sig-storage/csi-provisioner:v5.2.0 • registry.k8s.io/sig-storage/csi-attacher:v4.8.1 • registry.k8s.io/sig-storage/csi-resizer:v1.13.2 • registry.k8s.io/sig-storage/csi-snapshotter:v8.2.1 • registry.k8s.io/sig-storage/csi-node-driver-registrar:v2.13.0 • docker.io/netapp/trident-operator:25.06.2 (facoltativo)

Immagini del contenitore richieste per Trident 25.06

Versioni di Kubernetes	Immagine del contenitore
v1.27.0, v1.28.0, v1.29.0, v1.30.0, v1.31.0, v1.32.0, v1.33.0, v1.34.0	• docker.io/netapp/trident:25.06.0 • docker.io/netapp/trident-autosupport:25.06 • registry.k8s.io/sig-storage/csi-provisioner:v5.2.0 • registry.k8s.io/sig-storage/csi-attacher:v4.8.1 • registry.k8s.io/sig-storage/csi-resizer:v1.13.2 • registry.k8s.io/sig-storage/csi-snapshotter:v8.2.1 • registry.k8s.io/sig-storage/csi-node-driver-registrar:v2.13.0 • docker.io/netapp/trident-operator:25.06.0 (facoltativo)

Installa Trident

Installa utilizzando l’operatore Trident

Installare utilizzando tridentctl

Installa utilizzando l’operatore certificato OpenShift

Usa Trident

Preparare il nodo worker

Tutti i nodi worker nel cluster Kubernetes devono essere in grado di montare i volumi forniti per i pod. Per preparare i nodi worker, 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. Nelle versioni recenti di Red Hat Enterprise Linux CoreOS (RHCOS) gli strumenti sono installati per impostazione predefinita.

Strumenti NFS

"Installa gli strumenti NFS"se stai utilizzando: `ontap-nas` , `ontap-nas-economy` , `ontap-nas-flexgroup` , `azure-netapp-files` , `gcp-cvs` .

strumenti iSCSI

"Installare gli strumenti iSCSI"se stai utilizzando: `ontap-san` , `ontap-san-economy` , `solidfire-san` .

Strumenti NVMe

"Installa gli strumenti NVMe"se stai usando `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 e FC-NVMe" per ulteriori informazioni sulla configurazione degli host SAN FC e FC-NVMe.

"Installa gli strumenti FC"se stai usando `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. * L'auto-riparazione iSCSI non è applicabile a SCSI su FC.

Rilevamento del servizio nodo

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

> La scoperta del servizio nodo identifica i servizi rilevati ma non garantisce che siano configurati correttamente. Al contrario, l'assenza di un servizio scoperto non garantisce che il montaggio del volume fallirà.

Rivedi gli eventi

Trident crea eventi affinché il nodo identifichi i servizi rilevati. Per rivedere questi eventi, eseguire:

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

Esamina i servizi scoperti

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

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

Volumi NFS

Installare gli strumenti NFS utilizzando i comandi per il sistema operativo in uso. 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 worker dopo aver installato gli strumenti NFS per evitare errori durante il collegamento dei volumi ai container.

volumi iSCSI

Trident può stabilire automaticamente una sessione iSCSI, analizzare le LUN e rilevare dispositivi multipath, formattarli e montarli su un pod.

Capacità di auto-riparazione iSCSI

Per i sistemi ONTAP , Trident esegue l'auto-riparazione iSCSI ogni cinque minuti per:

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

I registri delle attività di auto-guarigione si trovano in `trident-main` contenitore sul rispettivo pod Daemonset. Per visualizzare i registri, è necessario aver impostato `debug` su "true" durante l'installazione Trident .

Le funzionalità di auto-riparazione iSCSI Trident possono aiutare a prevenire:

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

> Ad esempio, se i segreti CHAP venissero ruotati sul controller di archiviazione e la rete perdesse la connettività, i vecchi segreti CHAP (*obsoleti*) potrebbero persistere. La funzione di auto-riparazione è in grado di riconoscere questo problema e di ristabilire automaticamente la sessione per applicare i segreti CHAP aggiornati.

- Sessioni iSCSI mancanti
- LUN mancanti

Punti da considerare prima di aggiornare Trident

- Se vengono utilizzati solo igroup per nodo (introdotti nella versione 23.04+), la funzione di auto-riparazione iSCSI avvierà nuove scansioni SCSI per tutti i dispositivi nel bus SCSI.
- Se vengono utilizzati solo igroup con ambito backend (obsoleti a partire dalla versione 23.04), la funzione di auto-riparazione iSCSI avvierà nuove scansioni SCSI per individuare gli ID LUN esatti nel bus SCSI.
- Se viene utilizzato un mix di igroup per nodo e igroup con ambito backend, la funzione di auto-riparazione iSCSI avvierà nuove scansioni SCSI per gli ID LUN esatti nel bus SCSI.

Installare gli strumenti iSCSI

Installare gli strumenti iSCSI utilizzando i comandi del sistema operativo in uso.

Prima di iniziare

- Ogni nodo nel 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 e Element OS 12.5 o precedente, assicurarsi che l'algoritmo di autenticazione CHAP sia impostato su MD5 in `/etc/iscsi/iscsid.conf` Con Element 12.7 sono disponibili gli algoritmi CHAP sicuri conformi a FIPS SHA1, SHA-256 e SHA3-256.

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

- Quando si utilizzano nodi worker che eseguono RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) con PV iSCSI, specificare `discard` mountOption in StorageClass per eseguire il recupero dello spazio in linea. Fare riferimento a "Documentazione Red Hat" .
- Assicurati di aver aggiornato all'ultima versione del `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. Imposta la scansione su manuale:

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

4. Abilita multipathing:

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

> Garantire `/etc/multipath.conf` contiene `find_multipaths no` Sotto `defaults` .

5. Assicurare che `iscsid` E `multipathd` stanno correndo:

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

6. Abilita e avvia `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 bionic) o 2.0.874-7.1ubuntu6.1 o successiva (per focal):

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

3. Imposta la scansione su manuale:

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

4. Abilita 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
```

> Garantire `/etc/multipath.conf` contiene `find_multipaths no` Sotto `defaults` .

5. Assicurare 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 scoprire le porte di destinazione con `iscsiadm` prima di iniziare `open-iscsi` per l'avvio del demone iSCSI. In alternativa, puoi modificare il `iscsi` servizio da avviare `iscsid` automaticamente.

Configurare o disabilitare l'auto-riparazione iSCSI

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

- **Intervallo di auto-riparazione iSCSI**: determina la frequenza con cui viene richiamata l'auto-riparazione iSCSI (predefinito: 5 minuti). È possibile configurarlo in modo che venga eseguito più frequentemente impostando un numero più piccolo o meno frequentemente impostando un numero più grande.

Impostando l'intervallo di auto-riparazione iSCSI su 0, l'auto-riparazione iSCSI viene interrotta completamente. Non consigliamo di disabilitare la funzione di auto-riparazione iSCSI; dovrebbe essere disabilitata solo in determinati scenari, quando la funzione di auto-riparazione iSCSI non funziona come previsto o per scopi di debug.

- **Tempo di attesa per l'auto-riparazione iSCSI**: determina la durata di attesa dell'auto-riparazione iSCSI prima di disconnettersi da una sessione non integra e tentare di accedere nuovamente (impostazione predefinita: 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 poi venga effettuato un tentativo di accesso successivo, oppure su un numero inferiore per disconnettersi e accedere prima.

Timone

Per configurare o modificare le impostazioni di auto-riparazione iSCSI, passare il `iscsiSelfHealingInterval` E `iscsiSelfHealingWaitTime` parametri durante l'installazione o l'aggiornamento di Helm.

Nell'esempio seguente l'intervallo di auto-riparazione iSCSI viene impostato su 3 minuti e il tempo di attesa per l'auto-riparazione 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 auto-riparazione iSCSI, passare il `iscsi-self-healing-interval` E `iscsi-self-healing-wait-time` parametri durante l'installazione o l'aggiornamento di tridentctl.

Nell'esempio seguente l'intervallo di auto-riparazione iSCSI viene impostato su 3 minuti e il tempo di attesa per l'auto-riparazione 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 per il 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, potrebbe essere necessario aggiornare la versione del kernel del nodo a una 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
```

Verifica l'installazione

Dopo l'installazione, verifica che ogni nodo nel cluster Kubernetes abbia un NQN univoco utilizzando il comando:

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

Trident modifica il `ctrl_device_tmo` valore per garantire che NVMe non rinunci al percorso in caso di problemi. Non modificare questa impostazione.

SCSI su volumi FC

Ora è possibile utilizzare il protocollo Fibre Channel (FC) con Trident per fornire e gestire le risorse di storage sul sistema ONTAP .

Prerequisiti

Configurare le impostazioni di rete e nodo richieste per FC.

Impostazioni di rete

1. Ottieni il WWPN delle interfacce di destinazione. Fare riferimento a "mostra interfaccia di rete" per maggiori informazioni.
2. Ottieni il WWPN per le interfacce sull'iniziatore (host).

   Fare riferimento alle utilità del sistema operativo host corrispondenti.

3. Configurare la suddivisione in zone sullo switch FC utilizzando i WWPN dell'host e della destinazione.

   Per informazioni, fare riferimento alla documentazione del rispettivo fornitore dello switch.

   Per maggiori dettagli, fare riferimento alla seguente documentazione ONTAP :

   - "Panoramica sulla zonizzazione Fibre Channel e FCoE"

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

Installa gli strumenti FC

Installa gli strumenti FC utilizzando i comandi per il tuo sistema operativo.

- Quando si utilizzano nodi worker che eseguono RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) con FC PV, specificare `discard` mountOption in StorageClass per eseguire il recupero dello spazio in linea. Fare riferimento a "Documentazione Red Hat" .

RHEL 8+

1. Installare i seguenti pacchetti di sistema:

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

2. Abilita multipathing:

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

> Garantire `/etc/multipath.conf` contiene `find_multipaths no` Sotto `defaults`.

3. Assicurare che `multipathd` è 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. Abilita 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
```

> Garantire `/etc/multipath.conf` contiene `find_multipaths no` Sotto `defaults`.

3. Assicurare che `multipath-tools` è abilitato 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 archiviazione. Indica a Trident come comunicare con quel sistema di archiviazione e come Trident deve effettuare il provisioning dei volumi da esso.

Trident offre automaticamente pool di archiviazione da backend che corrispondono ai requisiti definiti da una classe di archiviazione. Scopri come configurare il backend per il tuo sistema di archiviazione.

- "Configurare un backend di Azure NetApp Files"
- "Configurare un backend Google Cloud NetApp Volumes"
- "Configurare un Cloud Volumes Service per il backend di Google Cloud Platform"
- "Configurare un backend NetApp HCI o SolidFire"
- "Configurare un backend con driver ONTAP o Cloud Volumes ONTAP NAS"
- "Configurare un backend con driver ONTAP o Cloud Volumes ONTAP SAN"
- "Utilizzare Trident con Amazon FSx for NetApp ONTAP"

Azure NetApp Files

Configurare un backend di Azure NetApp Files

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

Dettagli del driver di Azure NetApp Files

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

Autista	Protocollo	Modalità volume	Modalità di accesso supportate	Sistemi di file supportati
`azure-netapp-files`	NFS SMB	File system	RWO, ROX, RWX, RWOP	`nfs`, `smb`

Considerazioni

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

Identità gestite per AKS

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

- Un cluster Kubernetes distribuito tramite AKS
- Identità gestite configurate sul cluster AKS Kubernetes
- Trident installato che include il `cloudProvider` specificare `"Azure"` .

Operatore Trident

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

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

Timone

L'esempio seguente installa i set Trident `cloudProvider` ad 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>

L'esempio seguente installa Trident e imposta il `cloudProvider` bandiera a `Azure` :

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

Identità cloud per AKS

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

Per sfruttare i vantaggi dell'identità cloud in Azure, è necessario disporre di:

- Un cluster Kubernetes distribuito tramite AKS

- Identità del carico di lavoro e oidc-issuer configurati sul cluster AKS Kubernetes
- Trident installato che include il `cloudProvider` specificare `"Azure"` E `cloudIdentity` specificando l'identità del carico di lavoro

Operatore Trident

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

Per 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'"
```

L'esempio seguente installa Trident e imposta `cloudProvider` ad Azure utilizzando la variabile di ambiente `$CP` e imposta il `cloudIdentity` utilizzando la variabile d'ambiente `$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"
```

L'esempio seguente installa Trident e imposta il `cloud-provider` bandiera a `$CP` , E `cloud-identity` A `$CI` :

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

Prepararsi a configurare un backend di Azure NetApp Files

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

Prerequisiti per i 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 impostare 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" backend, hai bisogno di quanto segue:

- `subscriptionID`, `tenantID` , `clientID` , `location` , E `clientSecret` sono facoltativi quando si utilizzano identità gestite su un cluster AKS.
- `tenantID`, `clientID` , E `clientSecret` sono facoltativi quando si utilizza un'identità cloud su un cluster AKS.

- Un bacino di capacità. Fare riferimento a"Microsoft: creare un pool di capacità per Azure NetApp Files" .
- Una subnet delegata ad Azure NetApp Files. Fare riferimento a"Microsoft: delegare una subnet ad Azure NetApp Files" .
- `subscriptionID`da una sottoscrizione Azure con Azure NetApp Files abilitato.
- `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'app deve utilizzare:
  - Il ruolo di Proprietario o Collaboratore"predefinito da Azure" .
  - UN"ruolo di collaboratore personalizzato" a livello di abbonamento(`assignableScopes`) con le seguenti autorizzazioni limitate solo a quanto richiesto Trident . Dopo aver creato il ruolo personalizzato,"assegnare il ruolo utilizzando il portale di Azure" .

Ruolo di 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": []
        }
      ]
    }
}
```

- L'azzurro `location` che contiene almeno uno "sottorete delegata" . A partire dal Trident 22.01, il `location` Il parametro è un campo obbligatorio al livello superiore del file di configurazione del backend. I valori di posizione specificati nei pool virtuali vengono ignorati.
- Per usare `Cloud Identity` , ottenere il `client ID` da un "identità gestita assegnata dall'utente" e specificare quell'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 ad Azure NetApp Files. Fare riferimento a"Microsoft: creare e gestire connessioni Active Directory per Azure NetApp Files" .
- Un cluster Kubernetes con un nodo controller Linux e almeno un nodo worker Windows che esegue Windows Server 2022. Trident supporta volumi SMB montati su pod in esecuzione solo su nodi Windows.
- Almeno un segreto Trident contenente le credenziali di Active Directory, in modo che Azure NetApp Files possa autenticarsi in Active Directory. Per generare segreto `smbcreds` :

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

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

Opzioni ed esempi di configurazione del backend Azure NetApp Files

Scopri le opzioni di configurazione backend NFS e SMB per Azure NetApp Files e rivedi

gli esempi di configurazione.

Opzioni di configurazione del backend

Trident utilizza la configurazione backend (subnet, rete virtuale, livello di servizio e posizione) per creare volumi Azure NetApp Files su pool di capacità disponibili nella posizione richiesta e che corrispondono 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 di Azure NetApp Files forniscono queste opzioni di configurazione.

Parametro	Descrizione	Predefinito
`version`		Sempre 1
`storageDriverName`	Nome del driver di archiviazione	"file-azure-netapp"
`backendName`	Nome personalizzato o backend di archiviazione	Nome del conducente + "_" + caratteri casuali
`subscriptionID`	ID sottoscrizione della sottoscrizione di Azure. Facoltativo quando le identità gestite sono abilitate su un cluster AKS.	
`tenantID`	L'ID tenant da una registrazione app facoltativa quando le identità gestite o l'identità cloud vengono utilizzate su un cluster AKS.	
`clientID`	ID client da una registrazione app Facoltativo quando si utilizzano identità gestite o identità cloud su un cluster AKS.	
`clientSecret`	Il segreto client di una registrazione app facoltativa quando vengono utilizzate 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. Facoltativo quando le identità gestite sono abilitate su un cluster AKS.	
`resourceGroups`	Elenco dei gruppi di risorse per filtrare le risorse scoperte	"[]" (nessun filtro)
`netappAccounts`	Elenco degli account NetApp per filtrare le risorse rilevate	"[]" (nessun filtro)

Parametro	Descrizione	Predefinito
`capacityPools`	Elenco dei pool di capacità per filtrare le risorse scoperte	"[]" (nessun filtro, casuale)
`virtualNetwork`	Nome di una rete virtuale con una subnet delegata	""
`subnet`	Nome di una subnet delegata a `Microsoft.Netapp/volumes`	""
`networkFeatures`	Insieme di funzionalità VNet per un volume, può essere `Basic` O `Standard` . Le funzionalità di rete non sono disponibili in tutte le regioni e potrebbero dover essere abilitate tramite un abbonamento. Specificando `networkFeatures` quando la funzionalità non è abilitata, il provisioning del volume non riesce.	""
`nfsMountOptions`	Controllo dettagliato delle opzioni di montaggio NFS. Ignorato per i volumi SMB. Per montare volumi utilizzando NFS versione 4.1, includere `nfsvers=4` nell'elenco delle opzioni di montaggio delimitate da virgole per scegliere NFS v4.1. Le opzioni di montaggio impostate in una definizione di classe di archiviazione sostituiscono le opzioni di montaggio impostate nella configurazione del backend.	"nfsvers=3"
`limitVolumeSize`	Il provisioning non riesce se la dimensione del volume richiesto è superiore a questo valore	"" (non applicato di default)
`debugTraceFlags`	Flag di debug da utilizzare durante la risoluzione dei problemi. Esempio, `\{"api": false, "method": true, "discovery": true}` . Non utilizzare questa funzione a meno che non si desideri risolvere un problema e richiedere un dump dettagliato del registro.	null
`nasType`	Configurare la creazione di volumi NFS o SMB. Le opzioni sono `nfs` , `smb` o nullo. Impostando il valore su null, i volumi NFS vengono impostati di default.	`nfs`

Parametro	Descrizione	Predefinito
`supportedTopologies`	Rappresenta un elenco di regioni e zone supportate da questo backend. Per maggiori 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 in MiB/sec. Supportato solo per pool di capacità QoS manuali. **Anteprima tecnica per Trident 25.06**	`4 MiB/sec`

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

Autorizzazioni e risorse richieste

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

I valori per `resourceGroups` , `netappAccounts` , `capacityPools` , `virtualNetwork` , E `subnet` possono essere specificati utilizzando nomi brevi o completamente qualificati. Nella maggior parte delle situazioni si consiglia di utilizzare nomi completi, poiché 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 delle risorse rilevate a quelle disponibili per questo backend di archiviazione e possono essere specificati in qualsiasi combinazione. I nomi completi seguono questo formato:

Tipo	Formato
Gruppo di risorse	<gruppo di risorse>
Conto NetApp	<gruppo di risorse>/<account NetApp>
Capacità di pool	<gruppo di risorse>/<account NetApp>/<pool di capacità>
Rete virtuale	<gruppo di risorse>/<rete virtuale>
Sottorete	<gruppo di risorse>/<rete virtuale>/<sottorete>

Provisioning del volume

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

Parametro	Descrizione	Predefinito
`exportRule`	Regole di esportazione per i nuovi volumi. `exportRule` deve essere un elenco separato da virgole di qualsiasi combinazione di indirizzi IPv4 o subnet IPv4 in 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	"100G"
`unixPermissions`	I permessi Unix dei nuovi volumi (4 cifre ottali). Ignorato per i volumi SMB.	"" (funzione di anteprima, richiede l'inserimento nella whitelist nell'abbonamento)

Configurazioni di esempio

Gli esempi seguenti mostrano 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 minima assoluta del backend. Con questa configurazione, Trident rileva tutti gli account NetApp , i pool di capacità e le subnet delegate ad Azure NetApp Files nella posizione configurata e posiziona casualmente i nuovi volumi su uno di questi pool e subnet. Perché `nasType` viene omesso, il `nfs` si applica l'impostazione predefinita e il backend provvederà al provisioning dei volumi NFS.

Questa configurazione è ideale quando si inizia a utilizzare Azure NetApp Files e si provano le cose, ma in pratica si vorrà fornire un ambito aggiuntivo per i volumi di cui si esegue il provisioning.

```
---
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 del backend omette `subscriptionID` , `tenantID` , `clientID` , E `clientSecret` , che sono facoltativi 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 del backend omette `tenantID` , `clientID` , E `clientSecret` , che sono facoltativi 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 del pool di capacità

Questa configurazione del backend posiziona i volumi in Azure `eastus` posizione in un `Ultra` capacità del pool. Trident rileva automaticamente tutte le subnet delegate ad Azure NetApp Files in quella posizione e posiziona casualmente un nuovo volume su una di esse.

```
---
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 del backend riduce ulteriormente l'ambito del posizionamento del volume a una singola subnet e modifica anche 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 del pool virtuale

Questa configurazione backend definisce più pool di archiviazione in un singolo file. Questa funzionalità è utile quando si hanno più pool di capacità che supportano diversi livelli di servizio e si desidera creare classi di archiviazione in Kubernetes che li rappresentino. 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 semplifica il provisioning dei volumi per carichi di lavoro in base alle regioni e alle zone di disponibilità. IL `supportedTopologies` Il blocco in questa configurazione backend viene utilizzato per fornire un elenco di regioni e zone per backend. I valori di regione e zona specificati qui devono corrispondere ai valori di regione e zona delle etichette su ciascun 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 regioni e delle zone fornite in un backend, Trident crea volumi nella regione e nella zona menzionate. Per maggiori 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 archiviazione

Il seguente `StorageClass` le definizioni si riferiscono ai pool di archiviazione sopra indicati.

Definizioni di esempio utilizzando `parameter.selector` campo

Utilizzando `parameter.selector` puoi specificare per ciascuno `StorageClass` il pool virtuale utilizzato per ospitare un volume. Il volume avrà gli aspetti definiti nel pool scelto.

```
---
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

Utilizzando `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

```
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 ogni namespace

```
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

```
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` O `nasType: null` filtri per pool NFS.

Crea il backend

Dopo aver creato il file di configurazione del backend, eseguire il seguente comando:

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

Se la creazione del backend fallisce, c'è qualcosa che non va nella configurazione del backend. È possibile visualizzare i registri 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 backend per Trident. È possibile collegare volumi NFS e SMB utilizzando un backend Google Cloud NetApp Volumes .

Dettagli del driver Google Cloud NetApp Volumes

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

Autista	Protocollo	Modalità volume	Modalità di accesso supportate	Sistemi di file supportati
`google-cloud-netapp-volumes`	NFS SMB	File system	RWO, ROX, RWX, RWOP	`nfs`, `smb`

Identità cloud per GKE

Cloud Identity consente ai pod Kubernetes di accedere alle risorse di Google Cloud autenticandosi come identità del carico di lavoro anziché fornire credenziali Google Cloud esplicite.

Per sfruttare i vantaggi dell'identità cloud in Google Cloud, è necessario disporre di:

- Un cluster Kubernetes distribuito tramite GKE.
- Identità del carico di lavoro configurata sul cluster GKE e GKE MetaData Server configurato sui pool di nodi.
- Un account di servizio GCP con il ruolo Google Cloud NetApp Volumes Admin (roles/netapp.admin) o un

ruolo personalizzato.

- Trident installato che include cloudProvider per specificare "GCP" e cloudIdentity che specifica il nuovo account di servizio GCP. Di seguito è riportato un esempio.

Operatore Trident

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

Per 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'"
```

L'esempio seguente installa Trident e imposta `cloudProvider` a GCP utilizzando la variabile di ambiente `$CP` e imposta il `cloudIdentity` utilizzando la variabile d'ambiente `$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'"
```

L'esempio seguente installa Trident e imposta il `cloud-provider` bandiera a `$CP` , E `cloud-identity` A `$ANNOTATION` :

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

Preparati a configurare un backend Google Cloud NetApp Volumes

Prima di poter configurare il backend di Google Cloud NetApp Volumes , è necessario assicurarsi che siano soddisfatti i seguenti requisiti.

Prerequisiti per i volumi NFS

Se si utilizza Google Cloud NetApp Volumes per la prima volta o in una nuova posizione, è necessaria una configurazione iniziale per impostare Google Cloud NetApp Volumes e creare un volume NFS. Fare riferimento a"Prima di iniziare" .

Prima di configurare il backend 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 a"Google Cloud NetApp Volumes" .
- Numero di progetto del tuo account Google Cloud. Fare riferimento a"Identificazione dei progetti" .
- Un account di servizio Google Cloud con NetApp Volumes Admin(`roles/netapp.admin`) ruolo. Fare riferimento a"Ruoli e autorizzazioni di gestione dell'identità e dell'accesso" .
- File chiave API per il tuo account GCNV. Fare riferimento a"Crea una chiave dell'account di servizio"
- Un pool di stoccaggio. Fare riferimento a"Panoramica dei pool di archiviazione" .

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 del backend Google Cloud NetApp Volumes

Scopri le opzioni di configurazione backend per Google Cloud NetApp Volumes e rivedi gli esempi di configurazione.

Opzioni di configurazione del backend

Ogni backend esegue il provisioning dei volumi in una singola regione 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 archiviazione	Il valore di `storageDriverName` deve essere specificato come "google-cloud-netapp-volumes".
`backendName`	(Facoltativo) Nome personalizzato del backend di archiviazione	Nome del driver + "_" + parte della chiave API

Parametro	Descrizione	Predefinito
`storagePools`	Parametro facoltativo utilizzato per specificare i pool di archiviazione per la creazione del volume.	
`projectNumber`	Numero di progetto dell'account Google Cloud. Il valore si trova nella home page del portale Google Cloud.	
`location`	Posizione di Google Cloud in cui Trident crea i volumi GCNV. Quando si creano cluster Kubernetes interregionali, i volumi creati in un `location` può essere utilizzato nei carichi di lavoro pianificati sui nodi in più regioni di Google Cloud. Il traffico interregionale comporta un costo aggiuntivo.	
`apiKey`	Chiave API per l'account del servizio Google Cloud con `netapp.admin` ruolo. Include il contenuto in formato JSON del file della chiave privata di un account di servizio Google Cloud (copiato letteralmente nel file di configurazione del backend). IL `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 la dimensione del volume richiesto è superiore a questo valore.	"" (non applicato di default)
`serviceLevel`	Il livello di servizio di un pool di archiviazione e dei suoi volumi. I valori sono `flex` , `standard` , `premium` , O `extreme` .	
`labels`	Insieme di etichette arbitrarie formattate in JSON da applicare ai volumi	""
`network`	Rete Google Cloud utilizzata per i volumi GCNV.	
`debugTraceFlags`	Flag di debug da utilizzare durante la risoluzione dei problemi. Esempio, `{"api":false, "method":true}` . Non utilizzare questa funzione a meno che non si desideri risolvere un problema e richiedere un dump dettagliato del registro.	null
`nasType`	Configurare la creazione di volumi NFS o SMB. Le opzioni sono `nfs` , `smb` o nullo. Impostando il valore su null, i volumi NFS vengono impostati di default.	`nfs`

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

Opzioni di provisioning del volume

È possibile controllare il provisioning del volume predefinito in `defaults` sezione 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 al `.snapshot` elenco	"true" per NFSv4 "false" per NFSv3
`snapshotReserve`	Percentuale di volume riservata agli snapshot	"" (accetta il valore predefinito 0)
`unixPermissions`	I permessi Unix dei nuovi volumi (4 cifre ottali).	""

Configurazioni di esempio

Gli esempi seguenti mostrano 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 minima assoluta del backend. Con questa configurazione, Trident rileva tutti i pool di archiviazione delegati a Google Cloud NetApp Volumes nella posizione configurata e posiziona casualmente i nuovi volumi su uno di questi pool. Perché `nasType` viene omesso, il `nfs` si applica l'impostazione predefinita e il backend provvederà al provisioning dei volumi NFS.

Questa configurazione è ideale quando si inizia a utilizzare Google Cloud NetApp Volumes e si provano le funzionalità, ma in pratica sarà molto probabilmente necessario fornire un ambito aggiuntivo per i volumi di cui si esegue il provisioning.

```
---
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 filtro StoragePools

```
---
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 del pool virtuale

Questa configurazione backend definisce più pool virtuali in un singolo file. I pool virtuali sono definiti nel `storage` sezione. Sono utili quando si hanno più pool di archiviazione che supportano diversi livelli di servizio e si desidera creare classi di archiviazione in Kubernetes che li rappresentino. Per differenziare i pool vengono utilizzate etichette virtuali. Ad esempio, nell'esempio seguente `performance` etichetta e `serviceLevel` Il tipo viene utilizzato per differenziare i pool virtuali.

È anche possibile impostare alcuni valori predefiniti da applicare a tutti i pool virtuali e sovrascrivere i valori predefiniti per i singoli pool virtuali. Nell'esempio seguente, `snapshotReserve` E `exportRule` servono come valori predefiniti per tutti i pool virtuali.

Per maggiori 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 semplifica il provisioning dei volumi per carichi di lavoro in base alle regioni e alle zone di disponibilità. IL `supportedTopologies` Il blocco in questa configurazione backend viene utilizzato per fornire un elenco di regioni e zone per backend. I valori di regione e zona specificati qui devono corrispondere ai valori di regione e zona delle etichette su ciascun 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 regioni e delle zone fornite in un backend, Trident crea volumi nella regione e nella zona menzionate. Per maggiori 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
```

Cosa succederà ora?

Dopo aver creato il file di configurazione del backend, eseguire il seguente comando:

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

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

```
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 fallisce, c'è qualcosa che non va nella configurazione del backend. Puoi descrivere il backend usando il `kubectl get tridentbackendconfig <backend-name>` comando o visualizzare i registri per determinare la causa eseguendo il seguente comando:

```
tridentctl logs
```

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

Definizioni delle classi di archiviazione

Quello che segue è un esempio di base `StorageClass` definizione che si riferisce al backend di cui 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:

Utilizzando `parameter.selector` puoi specificare per ciascuno `StorageClass` IL"piscina virtuale" che viene utilizzato per ospitare un volume. Il volume avrà gli aspetti definiti nel pool scelto.

```
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 maggiori dettagli sulle classi di archiviazione, fare riferimento a"Creare una classe di archiviazione" .

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 Active Directory richieste. Per il segreto della fase del nodo è possibile utilizzare qualsiasi utente/password di Active Directory con qualsiasi/nessuna autorizzazione.

Configurazione di base sullo spazio dei nomi predefinito

```
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 ogni namespace

```
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

```
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` O `nasType: null` filtri per pool NFS.

Esempio di definizione di 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 vincolato, 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 backend per la tua installazione Trident utilizzando le configurazioni di esempio fornite.

Dettagli del driver Google Cloud

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

Autista	Protocollo	Modalità volume	Modalità di accesso supportate	Sistemi di file supportati
`gcp-cvs`	NFS	File system	RWO, ROX, RWX, RWOP	`nfs`

Scopri di più sul 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 prestazioni è particolarmente adatto ai carichi di lavoro di produzione che danno valore alle prestazioni. Il tipo di servizio CVS-Performance è un'opzione hardware che supporta volumi con una dimensione minima di 100 GiB. Puoi scegliere uno dei"tre livelli di servizio" :
  - `standard`
  - `premium`
  - `extreme`
- **CVS**: Il tipo di servizio CVS fornisce un'elevata disponibilità zonale con livelli di prestazioni da limitati a moderati. Il tipo di servizio CVS è un'opzione software che utilizza pool di archiviazione per supportare volumi piccoli fino a 1 GiB. Il pool di archiviazione può contenere fino a 50 volumi, tutti i quali condividono la capacità e le prestazioni del pool. Puoi scegliere uno dei"due livelli di servizio" :
  - `standardsw`
  - `zoneredundantstandardsw`

Cosa ti servirà

Per configurare e utilizzare il "Cloud Volumes Service per Google Cloud" backend, hai bisogno di quanto segue:

- Un account Google Cloud configurato con il servizio NetApp Cloud Volumes Service
- Numero di progetto del tuo account Google Cloud
- Account di servizio Google Cloud con `netappcloudvolumes.admin` ruolo
- File chiave API per il tuo account Cloud Volumes Service

Opzioni di configurazione del backend

Ogni backend esegue il provisioning dei volumi in una singola regione 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 archiviazione	"gcp-cvs"
`backendName`	Nome personalizzato o backend di archiviazione	Nome del driver + "_" + parte della chiave API
`storageClass`	Parametro facoltativo utilizzato per specificare il tipo di servizio CVS. Utilizzo `software` per selezionare il tipo di servizio CVS. In caso contrario, Trident presuppone il tipo di servizio CVS-Performance(`hardware`).	
`storagePools`	Solo tipo di servizio CVS. Parametro facoltativo utilizzato per specificare i pool di archiviazione per la creazione del volume.	
`projectNumber`	Numero di progetto dell'account Google Cloud. Il valore si trova nella home page del portale Google Cloud.	

Parametro	Descrizione	Predefinito
`hostProjectNumber`	Obbligatorio se si utilizza una rete VPC condivisa. In questo scenario, `projectNumber` è il progetto di servizio, e `hostProjectNumber` è il progetto ospitante.	
`apiRegion`	La regione di Google Cloud in cui Trident crea i volumi Cloud Volumes Service . Quando si creano cluster Kubernetes interregionali, i volumi creati in un `apiRegion` può essere utilizzato nei carichi di lavoro pianificati sui nodi in più regioni di 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 del file della chiave privata di un account di servizio Google Cloud (copiato letteralmente nel file di configurazione del backend).	
`proxyURL`	URL proxy se è necessario un server proxy per 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 la dimensione del volume richiesto è superiore a questo valore.	"" (non applicato di default)
`serviceLevel`	Il 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` O `zoneredundantstandardsw` .	L'impostazione predefinita di CVS-Performance è "standard". L'impostazione predefinita di CVS è "standardsw".
`network`	Rete Google Cloud utilizzata per i volumi Cloud Volumes Service .	"predefinito"
`debugTraceFlags`	Flag di debug da utilizzare durante la risoluzione dei problemi. Esempio, `\{"api":false, "method":true}` . Non utilizzare questa funzione a meno che non si desideri risolvere un problema e richiedere un dump dettagliato del registro.	null

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

Opzioni di provisioning del volume

È possibile controllare il provisioning del volume predefinito in `defaults` sezione 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 in notazione CIDR.	"0.0.0.0/0"
`snapshotDir`	Accesso al `.snapshot` elenco	"falso"
`snapshotReserve`	Percentuale di volume riservata agli snapshot	"" (accetta il valore predefinito CVS pari a 0)
`size`	Le dimensioni dei nuovi volumi. Il requisito minimo per le prestazioni CVS è 100 GiB. Il minimo CVS è 1 GiB.	Il tipo di servizio CVS-Performance è impostato per impostazione predefinita su "100GiB". Il tipo di servizio CVS non imposta un valore predefinito ma richiede almeno 1 GiB.

Esempi di tipi di servizio CVS-Performance

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

Esempio 1: Configurazione minima

Questa è la configurazione minima del backend 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

Questo esempio illustra le opzioni di configurazione del backend, tra cui 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 campione utilizza `storage` per configurare pool virtuali e `StorageClasses` che rimandano ad essi. Fare riferimento aDefinizioni delle classi di archiviazione per vedere come sono state definite le classi di archiviazione.

Qui vengono impostati valori predefiniti specifici per tutti i pool virtuali, che impostano il `snapshotReserve` al 5% e il `exportRule` a 0.0.0.0/0. I pool virtuali sono definiti nel `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 archiviazione

Le seguenti definizioni StorageClass si applicano all'esempio di configurazione del pool virtuale. Utilizzando `parameters.selector` , è possibile specificare per ogni StorageClass il pool virtuale utilizzato per ospitare un volume. Il volume avrà gli aspetti definiti nel pool scelto.

Esempio di classe di archiviazione

```
---
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
```

```
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`) corrisponde al primo pool virtuale. Questa è l'unica piscina che offre prestazioni estreme con una riserva istantanea del 10%.
- L'ultima StorageClass(`cvs-extra-protection`) richiama qualsiasi pool di archiviazione che fornisca una riserva di snapshot del 10%. Trident decide quale pool virtuale selezionare e garantisce che venga soddisfatto il requisito di riserva degli snapshot.

Esempi di tipi di servizio CVS

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

Esempio 1: Configurazione minima

Questa è la configurazione minima del backend utilizzando `storageClass` per specificare il tipo di servizio CVS e quello 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 archiviazione

Questa configurazione di backend di esempio utilizza `storagePools` per configurare un pool di archiviazione.

```
---
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
```

Cosa succederà ora?

Dopo aver creato il file di configurazione del backend, eseguire il seguente comando:

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

Se la creazione del backend fallisce, c'è qualcosa che non va nella configurazione del backend. È possibile visualizzare i registri 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

Scopri come creare e utilizzare un backend Element con la tua installazione Trident .

Dettagli del driver dell'elemento

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

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

Autista	Protocollo	Modalità Volume	Modalità di accesso supportate	Sistemi di file supportati
`solidfire-san`	iSCSI	Bloccare	RWO, ROX, RWX, RWOP	Nessun file system. Dispositivo a blocchi grezzi.
`solidfire-san`	iSCSI	File system	RWO, RWOP	`xfs`, `ext3` , `ext4`

Prima di iniziare

Prima di creare un backend Element, avrai bisogno di quanto segue.

- Un sistema di archiviazione 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 worker di Kubernetes devono avere installati gli strumenti iSCSI appropriati. Fare riferimento a"informazioni sulla preparazione del nodo worker" .

Opzioni di configurazione del backend

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

Parametro	Descrizione	Predefinito
`version`		Sempre 1
`storageDriverName`	Nome del driver di archiviazione	Sempre "solidfire-san"
`backendName`	Nome personalizzato o backend di archiviazione	"solidfire_" + indirizzo IP di archiviazione (iSCSI)

Parametro	Descrizione	Predefinito
`Endpoint`	MVIP per il cluster SolidFire con credenziali tenant	
`SVIP`	Indirizzo IP e porta di archiviazione (iSCSI)	
`labels`	Insieme di etichette arbitrarie in formato JSON da applicare ai volumi.	""
`TenantName`	Nome del 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 CHAP.	VERO
`AccessGroups`	Elenco degli ID dei gruppi di accesso da utilizzare	Trova l'ID di un gruppo di accesso denominato "trident"
`Types`	Specifiche QoS	
`limitVolumeSize`	Il provisioning non riesce se la dimensione del volume richiesto è superiore a questo valore	"" (non applicato di default)
`debugTraceFlags`	Flag di debug da utilizzare durante la risoluzione dei problemi. Esempio, {"api":false, "method":true}	null

Non usare `debugTraceFlags` a meno che non si stia risolvendo un problema e si necessiti di un dump di registro dettagliato.

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

Questo esempio mostra un file backend che utilizza l'autenticazione CHAP e modella tre tipi di volume con garanzie QoS specifiche. Molto probabilmente definiresti quindi classi di archiviazione per consumare ciascuna di queste utilizzando `IOPS` parametro della classe di archiviazione.

```
---
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 della classe di backend e di archiviazione per `solidfire-san` autista con pool virtuali

Questo esempio mostra il file di definizione del backend configurato con pool virtuali insieme alle StorageClass che vi fanno riferimento.

Trident copia le etichette presenti su un pool di archiviazione nella LUN di archiviazione back-end durante il provisioning. Per comodità, gli amministratori di storage possono definire etichette per pool virtuale e raggruppare i volumi in base all'etichetta.

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

```
---
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 StorageClass fanno riferimento ai pool virtuali sopra indicati. Utilizzando il `parameters.selector` campo, ogni StorageClass richiama quale pool virtuale può essere utilizzato per

ospitare un volume. Il volume avrà gli aspetti definiti nel pool virtuale scelto.

Il primo StorageClass(`solidfire-gold-four`) verrà mappato sul primo pool virtuale. Questa è l'unica piscina che offre prestazioni d'oro con un `Volume Type QoS` d'oro. L'ultima StorageClass(`solidfire-silver`) richiama qualsiasi pool di archiviazione che offra prestazioni Silver. Trident deciderà quale pool virtuale selezionare e garantirà che i requisiti di archiviazione siano soddisfatti.

```
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

---
```

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

Trova maggiori informazioni

- "Gruppi di accesso al volume"

Driver ONTAP SAN

Panoramica del driver ONTAP SAN

Scopri come configurare un backend ONTAP con i driver ONTAP e Cloud Volumes ONTAP SAN.

Dettagli del driver ONTAP SAN

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

Autista	Protocollo	Modalità volume	Modalità di accesso supportate	Sistemi di file supportati
`ontap-san`	iSCSI SCSI su FC	Bloccare	RWO, ROX, RWX, RWOP	Nessun file system; dispositivo a blocchi raw
`ontap-san`	iSCSI SCSI su FC	File system	RWO, RWOP ROX e RWX non sono disponibili nella modalità volume Filesystem.	`xfs`, `ext3` , `ext4`
`ontap-san`	NVMe/TCP Fare riferimento aConsiderazioni aggiuntive per NVMe/TCP .	Bloccare	RWO, ROX, RWX, RWOP	Nessun file system; dispositivo a blocchi raw

Autista	Protocollo	Modalità volume	Modalità di accesso supportate	Sistemi di file supportati
`ontap-san`	NVMe/TCP Fare riferimento aConsiderazioni aggiuntive per NVMe/TCP.	File system	RWO, RWOP ROX e RWX non sono disponibili nella modalità volume Filesystem.	`xfs`, `ext3` , `ext4`
`ontap-san-economy`	iSCSI	Bloccare	RWO, ROX, RWX, RWOP	Nessun file system; dispositivo a blocchi raw
`ontap-san-economy`	iSCSI	File system	RWO, RWOP ROX e RWX non sono disponibili nella modalità volume Filesystem.	`xfs`, `ext3` , `ext4`

- Utilizzo `ontap-san-economy` solo se si prevede che il conteggio dell'utilizzo del volume persistente sia superiore a"limiti di volume ONTAP supportati" .
- Utilizzo `ontap-nas-economy` solo se si prevede che il conteggio dell'utilizzo del volume persistente sia superiore a"limiti di volume ONTAP supportati" e il `ontap-san-economy` il driver non può essere utilizzato.
- Non usare usare `ontap-nas-economy` se prevedi la necessità di protezione dei dati, ripristino di emergenza o mobilità.
- NetApp non consiglia di utilizzare Flexvol autogrow in tutti i driver ONTAP , ad eccezione di ontap-san. Come soluzione alternativa, Trident supporta l'uso della riserva snapshot e ridimensiona di conseguenza i volumi Flexvol.

Permessi utente

Trident prevede di essere eseguito come amministratore ONTAP o SVM, in genere utilizzando `admin` utente del cluster o un `vsadmin` Utente SVM o un utente con un nome diverso che ha lo stesso ruolo. Per le distribuzioni Amazon FSx for NetApp ONTAP , Trident prevede di essere eseguito come amministratore ONTAP o SVM, utilizzando il cluster `fsxadmin` utente o un `vsadmin` Utente SVM o un utente con un nome diverso che ha lo stesso ruolo. IL `fsxadmin` L'utente è un sostituto limitato dell'utente amministratore del cluster.

Se usi il `limitAggregateUsage` parametro, sono richieste le autorizzazioni di amministratore del cluster. Quando si utilizza Amazon FSx for NetApp ONTAP con Trident, il `limitAggregateUsage` il parametro non funzionerà con il `vsadmin` E `fsxadmin` account utente. Se si specifica questo parametro, l'operazione di configurazione non riuscirà.

Sebbene sia possibile creare un ruolo più restrittivo all'interno di ONTAP utilizzabile da un conducente Trident , non lo consigliamo. La maggior parte delle nuove versioni di Trident richiameranno API aggiuntive di cui

bisognerebbe tenere conto, rendendo gli aggiornamenti difficili e soggetti a errori.

Considerazioni aggiuntive per NVMe/TCP

Trident supporta il protocollo NVMe (Non-Volatile Memory Express) utilizzando `ontap-san` autista incluso:

- 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
- Multipathing nativo NVMe
- Arresto regolare o non regolare dei nodi K8s (24.06)

Trident non supporta:

- DH-HMAC-CHAP supportato nativamente da NVMe
- Multipathing del mappatore di dispositivi (DM)
- Crittografia LUKS

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

Prepararsi a configurare il backend con i driver ONTAP SAN

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

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 archiviazione. Nei sistemi ASA r2, al posto degli aggregati vengono utilizzate zone di disponibilità dello storage. Fare riferimento a"Questo" Articolo della Knowledge Base su come assegnare aggregati alle SVM nei sistemi ASA r2.

Ricorda che puoi anche eseguire più di un driver e creare classi di archiviazione che puntano all'uno o all'altro. Ad esempio, potresti configurare un `san-dev` classe che utilizza il `ontap-san` autista e un `san-default` classe che utilizza il `ontap-san-economy` uno.

Tutti i nodi worker di Kubernetes devono avere installati gli strumenti iSCSI appropriati. Fare riferimento a "Preparare il nodo worker" per i dettagli.

Autenticare il backend ONTAP

Trident offre due modalità di autenticazione di un backend ONTAP .

- Basato su 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` O `vsadmin` per garantire la massima compatibilità con le versioni ONTAP .

- Basato su certificato: Trident può anche comunicare con un cluster ONTAP utilizzando un certificato installato sul backend. Qui, la definizione del backend deve contenere 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 metodi basati su credenziali a metodi basati su certificati. Tuttavia, è supportato solo un metodo di autenticazione alla volta. Per passare a un metodo di autenticazione diverso, è necessario rimuovere il metodo esistente dalla configurazione del backend.

> Se si tenta di fornire **sia credenziali che certificati**, la creazione del backend fallirà e verrà visualizzato un errore che indica che nel file di configurazione è stato fornito più di un metodo di autenticazione.

Abilita l'autenticazione basata sulle credenziali

Trident richiede le credenziali di un amministratore con ambito SVM/cluster per comunicare con il backend ONTAP . Si consiglia di utilizzare ruoli standard predefiniti come `admin` O `vsadmin` . Ciò garantisce la compatibilità futura con le future versioni ONTAP che potrebbero esporre API di funzionalità da utilizzare nelle future versioni Trident . È possibile creare e utilizzare un ruolo di accesso di sicurezza personalizzato con Trident, ma non è consigliato.

Un esempio di definizione del backend sarà simile a questo:

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"
}
```

Tieni presente che la definizione del backend è l'unico posto in cui le credenziali vengono archiviate in testo normale. Dopo aver 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. Si tratta pertanto di un'operazione riservata esclusivamente all'amministratore, che deve essere eseguita dall'amministratore di Kubernetes/archiviazione.

Abilita l'autenticazione basata sul certificato

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

- clientCertificate: valore codificato in Base64 del certificato client.
- 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. Questo può essere ignorato se non viene utilizzata alcuna CA attendibile.

Un flusso di lavoro tipico prevede i seguenti passaggi.

Passi

1. Genera un certificato client e una chiave. Durante la generazione, impostare il nome comune (CN) sull'utente ONTAP con cui effettuare l'autenticazione.

```
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 . Questa operazione potrebbe essere già gestita dall'amministratore dell'archiviazione. Ignora 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 client e la chiave (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. Conferma che il ruolo di accesso alla sicurezza ONTAP supporta `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 l'autenticazione utilizzando il certificato generato. Sostituire < ONTAP Management LIF> e <nome vserver> con l'IP Management LIF e il 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 il certificato, la chiave e il 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...Vaaalllluuuueeee",
"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 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Aggiorna i metodi di autenticazione o ruota le credenziali

È possibile aggiornare un backend esistente per utilizzare un metodo di autenticazione diverso o per ruotare le credenziali. Funziona in entrambi i modi: i backend che utilizzano nome utente/password possono essere aggiornati per utilizzare certificati; i backend che utilizzano certificati possono essere aggiornati per utilizzare nome utente/password. Per fare ciò, 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 richiesti per eseguire `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 dell'archiviazione deve prima aggiornare la password per l'utente su ONTAP. Segue un aggiornamento del backend. 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 eliminato dal cluster ONTAP .

L'aggiornamento di un backend non interrompe l'accesso ai volumi già creati, né influisce sulle connessioni ai volumi effettuate in seguito. Un aggiornamento del backend riuscito indica che Trident può comunicare con il backend ONTAP e gestire le future operazioni sui volumi.

Crea un ruolo ONTAP personalizzato per Trident

È possibile creare un ruolo cluster ONTAP con privilegi minimi in modo da non dover utilizzare il ruolo di

amministratore ONTAP per eseguire 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.

Fare riferimento a"Generatore di ruoli personalizzati Trident" per ulteriori informazioni sulla creazione di ruoli personalizzati Trident .

Utilizzo di ONTAP CLI

1. Crea un nuovo ruolo utilizzando il seguente comando:

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

2. Crea 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. Assegna 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

Eseguire i seguenti passaggi in ONTAP System Manager:

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 SVM, selezionare **Archiviazione > VM di archiviazione > `required SVM` > Impostazioni > Utenti e ruoli**.

   b. Selezionare l'icona della freccia (→) accanto a **Utenti e ruoli**.
   c. Selezionare **+Aggiungi** in **Ruoli**.
   d. Definisci le regole per il ruolo e clicca su **Salva**.
2. **Assegnare il ruolo all'utente Trident *: + Eseguire i seguenti passaggi nella pagina *Utenti e ruoli**:
   a. Selezionare Aggiungi icona **+** in **Utenti**.
   b. Selezionare il nome utente richiesto e selezionare un ruolo nel menu a discesa per **Ruolo**.
   c. Fare clic su **Salva**.

Per maggiori informazioni consultare le seguenti pagine:

- "Ruoli personalizzati per l'amministrazione di ONTAP"O"Definisci ruoli personalizzati"
- "Lavorare con ruoli e utenti"

Autenticare le connessioni con CHAP bidirezionale

Trident può autenticare le sessioni iSCSI con CHAP bidirezionale per `ontap-san` E `ontap-san-economy` conducenti. Ciò richiede l'abilitazione del `useCHAP` opzione nella definizione del backend. Quando impostato su `true` Trident configura la sicurezza dell'iniziatore predefinito dell'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` Il parametro è un'opzione booleana che può essere configurata solo una volta. Per impostazione predefinita è impostato su falso. Dopo averlo impostato su true, non è possibile impostarlo su false.

Inoltre `useCHAP=true` , IL `chapInitiatorSecret` , `chapTargetInitiatorSecret` , `chapTargetUsername` , E `chapUsername` i campi devono essere inclusi nella definizione del backend. I segreti possono essere modificati dopo la creazione di un backend eseguendo `tridentctl update` .

Come funziona

Impostando `useCHAP` su true, l'amministratore dell'archiviazione indica a Trident di configurare CHAP sul backend di archiviazione. Ciò include quanto segue:

- Impostazione di CHAP sull'SVM:
  - Se il tipo di sicurezza dell'iniziatore predefinito dell'SVM è nessuno (impostato per impostazione predefinita) **e** non ci sono LUN preesistenti già presenti nel volume, Trident imposterà il tipo di sicurezza predefinito su `CHAP` e procedere alla configurazione dell'iniziatore CHAP e del nome utente e dei segreti di destinazione.
  - Se l'SVM contiene LUN, Trident non abiliterà CHAP sull'SVM. Ciò garantisce che l'accesso ai LUN già presenti sulla SVM non sia 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).

Dopo aver creato il backend, Trident crea un corrispondente `tridentbackend` CRD e memorizza i segreti CHAP e i nomi utente come segreti Kubernetes. Tutti i PV 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. Ciò richiederà l'aggiornamento dei segreti CHAP e l'utilizzo di `tridentctl update` comando per riflettere questi cambiamenti.

> Quando si aggiornano i segreti CHAP per un backend, è necessario utilizzare `tridentctl` per aggiornare il backend. Non aggiornare le credenziali sul cluster di archiviazione utilizzando ONTAP CLI o ONTAP System Manager poiché Trident non sarà in grado di rilevare 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 saranno interessate e continueranno a rimanere attive se le credenziali vengono aggiornate da Trident sull'SVM. Le nuove connessioni utilizzano le credenziali aggiornate, mentre le connessioni esistenti continuano a rimanere attive. Scollegando e ricollegando i vecchi PV, questi utilizzeranno le credenziali aggiornate.

Opzioni ed esempi di configurazione SAN ONTAP

Scopri come creare e utilizzare i driver ONTAP SAN con la tua installazione Trident .

Questa sezione fornisce esempi di configurazione del backend e dettagli per il mapping dei backend alle StorageClass.

"Sistemi ASA r2"differiscono dagli altri sistemi ONTAP (ASA, AFF e FAS) nell'implementazione del loro livello di archiviazione. Tali variazioni incidono 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` Il 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 è ASA r2. Quando selezioni `ontap-san` come il `storageDriverName` , Trident rileva automaticamente il ASA r2 o il tradizionale sistema ONTAP . Alcuni parametri di configurazione del backend non sono applicabili ai sistemi ASA r2, come indicato nella tabella seguente.

Opzioni di configurazione del backend

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

Parametro	Descrizione	Predefinito
`version`		Sempre 1
`storageDriverName`	Nome del driver di archiviazione	`ontap-san`O `ontap-san-economy`
`backendName`	Nome personalizzato o backend di archiviazione	Nome del driver + "_" + dataLIF
`managementLIF`	Indirizzo IP di un cluster o di un LIF di gestione SVM. È possibile specificare un nome di dominio completo (FQDN). Può essere impostato per utilizzare 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 un passaggio senza interruzioni a MetroCluster , vedereEsempio MetroCluster . > Se si utilizzano le credenziali "vsadmin", `managementLIF` deve essere quello dell'SVM; se si utilizzano le credenziali "admin", `managementLIF` deve essere quello del cluster.	"10.0.0.1", "[2001:1234:abcd::fefe]"

Parametro	Descrizione	Predefinito
`dataLIF`	Indirizzo IP del protocollo LIF. Può essere impostato per utilizzare 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.** Usi Trident"Mappa LUN selettiva ONTAP" per scoprire gli iSCSI LIF necessari per stabilire una sessione multi-percorso. Viene generato un avviso se `dataLIF` è definito esplicitamente. **Omettere per Metrocluster.** Vedi ilEsempio MetroCluster .	Derivato dall'SVM
`svm`	Macchina virtuale di archiviazione da utilizzare **Ometti per Metrocluster.** Vedi ilEsempio MetroCluster .	Derivato se un SVM `managementLIF` è specificato
`useCHAP`	Utilizzare CHAP per autenticare iSCSI per i driver ONTAP SAN [Booleano]. Impostato su `true` affinché Trident configuri e utilizzi CHAP bidirezionale come autenticazione predefinita per l'SVM fornito nel backend. Fare riferimento a "Prepararsi a configurare il backend con i driver ONTAP SAN" per i dettagli. **Non supportato per FCP o NVMe/TCP.**	`false`
`chapInitiatorSecret`	Segreto dell'iniziatore CHAP. Obbligatorio se `useCHAP=true`	""
`labels`	Insieme di etichette arbitrarie formattate in JSON da applicare ai volumi	""
`chapTargetInitiatorSecret`	Segreto dell'iniziatore del target CHAP. Obbligatorio se `useCHAP=true`	""
`chapUsername`	Nome utente in entrata. Obbligatorio se `useCHAP=true`	""
`chapTargetUsername`	Nome utente di destinazione. Obbligatorio se `useCHAP=true`	""
`clientCertificate`	Valore codificato in Base64 del certificato client. 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 sulle credenziali. Per l'autenticazione di Active Directory, vedere "Autenticare Trident su un SVM backend utilizzando le credenziali di Active Directory".	""

Parametro	Descrizione	Predefinito
`password`	Password necessaria per comunicare con il cluster ONTAP . Utilizzato per l'autenticazione basata sulle credenziali. Per l'autenticazione di Active Directory, vedere "Autenticare Trident su un SVM backend utilizzando le credenziali di Active Directory".	""
`svm`	Macchina virtuale di archiviazione da utilizzare	Derivato se un SVM `managementLIF` è specificato
`storagePrefix`	Prefisso utilizzato durante il provisioning di nuovi volumi nell'SVM. Non può essere modificato in seguito. Per aggiornare questo parametro, sarà necessario creare un nuovo backend.	`trident`
`aggregate`	Aggregato per il provisioning (facoltativo; se impostato, deve essere assegnato all'SVM). Per il `ontap-nas-flexgroup` driver, questa opzione viene ignorata. Se non assegnato, è possibile utilizzare uno qualsiasi degli aggregati disponibili per eseguire il provisioning di un volume FlexGroup . Quando l'aggregato viene aggiornato in SVM, viene aggiornato automaticamente in Trident interrogando SVM senza dover riavviare Trident Controller. Dopo aver configurato un aggregato specifico in Trident per il provisioning dei volumi, se l'aggregato viene rinominato o spostato fuori dall'SVM, il backend passerà allo stato di errore in Trident durante l'interrogazione dell'aggregato SVM. Per riportare online il backend, è necessario modificare l'aggregato con uno presente sull'SVM oppure rimuoverlo del tutto. **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 for NetApp ONTAP , non specificare `limitAggregateUsage` . Il fornito `fsxadmin` E `vsadmin` non contengono le autorizzazioni richieste per recuperare l'utilizzo aggregato e limitarlo tramite Trident. **Non specificare per i sistemi ASA r2**.	"" (non applicato di default)
`limitVolumeSize`	Il provisioning non riesce se la dimensione del volume richiesto è superiore a questo valore. Limita inoltre la dimensione massima dei volumi gestiti per le LUN.	"" (non applicato di default)
`lunsPerFlexvol`	Numero massimo di LUN per Flexvol, deve essere compreso nell'intervallo [50, 200]	`100`

Parametro	Descrizione	Predefinito
`debugTraceFlags`	Flag di debug da utilizzare durante la risoluzione dei problemi. Esempio: {"api":false, "method":true} Non utilizzare a meno che non si stia risolvendo un problema e si necessiti di un dump di registro dettagliato.	`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 successivo, altrimenti `false`.
`sanType`	Utilizzare per selezionare `iscsi` per iSCSI, `nvme` per NVMe/TCP o `fcp` per SCSI su Fibre Channel (FC).	`iscsi`se vuoto

Parametro	Descrizione	Predefinito
`formatOptions`	Utilizzo `formatOptions` per specificare gli argomenti della riga di comando per `mkfs` comando, che verrà applicato ogni volta che un volume viene formattato. Ciò consente di formattare il volume in base alle proprie preferenze. Assicurarsi di specificare formatOptions in modo simile a quello delle opzioni del comando mkfs, escludendo il percorso del dispositivo. Esempio: "-E nodiscard" **Supportato per `ontap-san` E `ontap-san-economy` driver con protocollo iSCSI. Inoltre, supportato per i sistemi ASA r2 quando si utilizzano i protocolli iSCSI e NVMe/TCP.**	
`limitVolumePoolSize`	Dimensione massima FlexVol richiedibile quando si utilizzano LUN nel backend ontap-san-economy.	"" (non applicato di default)
`denyNewVolumePools`	Limita `ontap-san-economy` backend dalla creazione di nuovi volumi FlexVol per contenere i loro LUN. Per il provisioning di nuovi PV vengono utilizzati solo i Flexvol preesistenti.	

Consigli per l'utilizzo di formatOptions

Trident consiglia la seguente opzione per velocizzare il processo di formattazione:

-E nodiscard:

- Mantieni, non tentare di scartare i blocchi in fase di mkfs (scartare i blocchi inizialmente è utile su dispositivi a stato solido e storage sparse/thin-provisioned). Sostituisce l'opzione obsoleta "-K" ed è applicabile a tutti i file system (xfs, ext3 ed ext4).

Autenticare Trident su un SVM backend utilizzando le credenziali di Active Directory

È possibile configurare Trident per l'autenticazione a un SVM backend utilizzando le credenziali di Active Directory (AD). Prima che un account AD possa accedere all'SVM, è necessario configurare l'accesso del controller di dominio AD al cluster o all'SVM. Per l'amministrazione del cluster con un account AD, è necessario creare un tunnel di dominio. Fare riferimento a "Configurare l'accesso al controller di dominio Active Directory in ONTAP" per i dettagli.

passi

1. Configurare le impostazioni del Domain Name System (DNS) per un SVM backend:

```
vserver services dns create -vserver <svm_name> -dns-servers
<dns_server_ip1>,<dns_server_ip2>
```

2. Eseguire il seguente comando per creare un account computer per l'SVM in Active Directory:

```
vserver active-directory create -vserver DataSVM -account-name ADSERVER1
-domain demo.netapp.com
```

3. Utilizzare questo comando per creare un utente o un gruppo AD per gestire il cluster o SVM

```
security login create -vserver <svm_name> -user-or-group-name
<ad_user_or_group> -application <application> -authentication-method domain
-role vsadmin
```

4. Nel file di configurazione del backend Trident , impostare `username` E `password` parametri rispettivamente per il nome utente o gruppo AD e la password.

Opzioni di configurazione del backend per il provisioning dei volumi

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

Parametro	Descrizione	Predefinito
`spaceAllocation`	Assegnazione 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`	Criterio di snapshot da utilizzare. **Impostato su `none` per sistemi ASA r2**.	"nessuno"
`qosPolicy`	Gruppo di criteri QoS da assegnare ai volumi creati. Scegliere tra qosPolicy o adaptiveQosPolicy per ogni pool di archiviazione/backend. Per utilizzare i gruppi di policy QoS con Trident è necessario ONTAP 9.8 o versione successiva. È necessario utilizzare un gruppo di policy QoS non condiviso e assicurarsi che il gruppo di policy venga applicato individualmente a ciascun componente. Un gruppo di policy QoS condiviso impone il limite massimo per la produttività totale di tutti i carichi di lavoro.	""
`adaptiveQosPolicy`	Gruppo di policy QoS adattivo da assegnare ai volumi creati. Scegli uno tra qosPolicy o adaptiveQosPolicy per pool di archiviazione/backend	""
`snapshotReserve`	Percentuale di volume riservata agli snapshot. **Non specificare per i sistemi ASA r2**.	"0" se `snapshotPolicy` è "nessuno", altrimenti ""
`splitOnClone`	Dividere un clone dal suo genitore al momento della creazione	"falso"
`encryption`	Abilita NetApp Volume Encryption (NVE) sul nuovo volume; l'impostazione predefinita è `false` . Per utilizzare questa opzione, NVE deve essere concesso in licenza e abilitato sul cluster. Se NAE è abilitato sul backend, qualsiasi volume fornito in Trident sarà abilitato per NAE. Per maggiori informazioni, fare riferimento a:"Come funziona Trident con NVE e NAE" .	"false" **Se specificato, impostare su `true` per sistemi ASA r2**.

Parametro	Descrizione	Predefinito
`luksEncryption`	Abilita la crittografia LUKS. Fare riferimento a"Utilizzare Linux Unified Key Setup (LUKS)" .	"" **Impostato su `false` per sistemi ASA r2**.
`tieringPolicy`	Criterio di suddivisione in livelli per utilizzare "nessuno" **Non specificare per i sistemi ASA r2**.	
`nameTemplate`	Modello per creare nomi di volume personalizzati.	""

Esempi di provisioning del volume

Ecco un esempio con i valori predefiniti definiti:

```
---
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 `ontap-san` driver, Trident aggiunge un ulteriore 10 percento di capacità al FlexVol per ospitare i metadati LUN. La LUN verrà fornita con le dimensioni esatte richieste dall'utente nel PVC. Trident aggiunge il 10 percento al FlexVol (mostrato come dimensione disponibile in ONTAP). Gli utenti riceveranno ora la quantità di capacità utilizzabile richiesta. Questa modifica impedisce inoltre che i 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 la dimensione 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. IL `volume show` il comando dovrebbe mostrare risultati simili a questo esempio:

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

Esempi di configurazione minima

Gli esempi seguenti mostrano 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, NetApp consiglia di specificare i nomi DNS per i LIF anziché gli indirizzi IP.

Esempio ONTAP SAN

Questa è una configurazione di base che utilizza il `ontap-san` autista.

```
---
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 il passaggio e il ritorno durante"Replica e ripristino SVM" .

Per un passaggio e un ritorno senza interruzioni, specificare l'SVM utilizzando `managementLIF` e omettere il `svm` parametri. Per esempio:

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

Esempio di economia ONTAP SAN

```
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) vengono popolati in `backend.json` e prendere rispettivamente i valori codificati in 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` impostato su `true` .

Esempio ONTAP SAN 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 CHAP economico ONTAP SAN

```
---
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

È necessario disporre di una SVM configurata con NVMe sul backend ONTAP . Questa è 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 SCSI su FC (FCP)

È necessario disporre di un SVM configurato con FC sul backend ONTAP . Questa è una 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 del 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 formatOptions 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 backend di esempio, vengono impostati valori predefiniti specifici per tutti i pool di archiviazione, ad esempio `spaceReserve` a nessuno, `spaceAllocation` a falso, 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 sul FlexVol volume. Trident copia tutte le etichette presenti su un pool virtuale nel volume di archiviazione al momento del provisioning. Per comodità, gli amministratori di storage possono definire etichette per pool virtuale e raggruppare i volumi in base all'etichetta.

In questi esempi, alcuni dei pool di archiviazione impostano i propri `spaceReserve` , `spaceAllocation` , E `encryption` valori e alcuni pool sovrascrivono i valori predefiniti.

Esempio ONTAP SAN

```
---
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 ONTAP SAN

```
---
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"
```

Mappa i backend su StorageClasses

Le seguenti definizioni StorageClass fanno riferimento aEsempi di backend con pool virtuali . Utilizzando il `parameters.selector` campo, ogni StorageClass richiama quali pool virtuali possono essere utilizzati per ospitare un volume. Il volume avrà gli aspetti definiti nel pool virtuale scelto.

- IL `protection-gold` StorageClass verrà mappato sul primo pool virtuale nel `ontap-san` backend. Questa è l'unica piscina 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 verrà mappato sul secondo e terzo pool virtuale in `ontap-san` backend. 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 verrà mappato sul terzo pool virtuale in `ontap-san-economy` backend. Questo è l'unico pool che offre la configurazione del pool di archiviazione per l'app 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 verrà mappato sul secondo pool virtuale in `ontap-san` backend. Questo è l'unico pool che offre protezione di livello Silver e 20.000 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 verrà mappato sul terzo pool virtuale in `ontap-san` backend e il quarto pool virtuale nel `ontap-san-economy` backend. 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"
```

- IL `my-test-app-sc` StorageClass verrà mappato su `testAPP` piscina virtuale nel `ontap-san` autista con `sanType: nvme` . Questa è l'unica piscina che offre `testApp` .

```
---
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 selezionare e garantirà che i requisiti di archiviazione siano soddisfatti.

Driver NAS ONTAP

Panoramica del driver NAS ONTAP

Scopri come configurare un backend ONTAP con i driver ONTAP e Cloud Volumes ONTAP NAS.

Dettagli del driver ONTAP NAS

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

Autista	Protocollo	Modalità volume	Modalità di accesso supportate	Sistemi di file supportati
`ontap-nas`	NFS SMB	File system	RWO, ROX, RWX, RWOP	"", `nfs` , `smb`
`ontap-nas-economy`	NFS SMB	File system	RWO, ROX, RWX, RWOP	"", `nfs` , `smb`
`ontap-nas-flexgroup`	NFS SMB	File system	RWO, ROX, RWX, RWOP	"", `nfs` , `smb`

- Utilizzo `ontap-san-economy` solo se si prevede che il conteggio dell'utilizzo del volume persistente sia superiore a"limiti di volume ONTAP supportati" .
- Utilizzo `ontap-nas-economy` solo se si prevede che il conteggio dell'utilizzo del volume persistente sia superiore a"limiti di volume ONTAP supportati" e il `ontap-san-economy` il driver non può essere utilizzato.
- Non usare usare `ontap-nas-economy` se prevedi la necessità di protezione dei dati, ripristino di emergenza o mobilità.
- NetApp non consiglia di utilizzare Flexvol autogrow in tutti i driver ONTAP , ad eccezione di ontap-san. Come soluzione alternativa, Trident supporta l'uso della riserva snapshot e ridimensiona di conseguenza i volumi Flexvol.

Permessi utente

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

Per le distribuzioni Amazon FSx for NetApp ONTAP , Trident prevede di essere eseguito come amministratore ONTAP o SVM, utilizzando il cluster `fsxadmin` utente o un `vsadmin` Utente SVM o un utente con un nome diverso che ha lo stesso ruolo. IL `fsxadmin` L'utente è un sostituto limitato dell'utente amministratore del cluster.

Se usi il `limitAggregateUsage` parametro, sono richieste le autorizzazioni di amministratore del cluster. Quando si utilizza Amazon FSx for NetApp ONTAP con Trident, il `limitAggregateUsage` il parametro non funzionerà con il `vsadmin` E `fsxadmin` account utente. Se si specifica questo parametro, l'operazione di configurazione non riuscirà.

Sebbene sia possibile creare un ruolo più restrittivo all'interno di ONTAP utilizzabile da un conducente Trident , non lo consigliamo. La maggior parte delle nuove versioni di Trident richiameranno API aggiuntive di cui bisognerebbe tenere conto, rendendo gli aggiornamenti difficili e soggetti a errori.

Prepararsi a configurare un backend con i driver ONTAP NAS

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

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 archiviazione che puntano all'uno o all'altro. Ad esempio, è possibile configurare una classe Gold che utilizza `ontap-nas` driver e una classe Bronze che utilizza il `ontap-nas-economy` uno.
- Tutti i nodi worker di Kubernetes devono avere installati gli strumenti NFS appropriati. Fare riferimento a"Qui" per maggiori dettagli.
- Trident supporta volumi SMB montati su pod in esecuzione solo su nodi Windows. Fare riferimento a Prepararsi al provisioning dei volumi SMB per i dettagli.

Autenticare il backend ONTAP

Trident offre due modalità di autenticazione di un backend ONTAP .

- Basato su 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` O `vsadmin` per garantire la massima compatibilità con le versioni ONTAP .
- Basato su certificato: questa modalità richiede un certificato installato sul backend affinché Trident possa comunicare con un cluster ONTAP . Qui, la definizione del backend deve contenere 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 metodi basati su credenziali a metodi basati su certificati. Tuttavia, è supportato solo un metodo di autenticazione alla volta. Per passare a un metodo di autenticazione diverso, è necessario rimuovere il metodo esistente dalla configurazione del backend.

Se si tenta di fornire **sia credenziali che certificati**, la creazione del backend fallirà e verrà visualizzato un errore che indica che nel file di configurazione è stato fornito più di un metodo di autenticazione.

Abilita l'autenticazione basata sulle credenziali

Trident richiede le credenziali di un amministratore con ambito SVM/cluster per comunicare con il backend ONTAP . Si consiglia di utilizzare ruoli standard predefiniti come `admin` O `vsadmin` . Ciò garantisce la compatibilità futura con le future versioni ONTAP che potrebbero esporre API di funzionalità da utilizzare nelle future versioni Trident . È possibile creare e utilizzare un ruolo di accesso di sicurezza personalizzato con Trident, ma non è consigliato.

Un esempio di definizione del backend sarà simile a questo:

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"
    }
}
```

Tieni presente che la definizione del backend è l'unico posto in cui le credenziali vengono archiviate in testo normale. Dopo aver creato il backend, i nomi utente e le password vengono codificati con Base64 e memorizzati come segreti Kubernetes. La creazione/aggiornamento di un backend è l'unico passaggio che richiede la conoscenza delle credenziali. Si tratta pertanto di un'operazione riservata esclusivamente all'amministratore, che deve essere eseguita dall'amministratore di Kubernetes/archiviazione.

Abilita l'autenticazione basata su certificato

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

- clientCertificate: valore codificato in Base64 del certificato client.
- 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. Questo può essere ignorato se non viene utilizzata alcuna CA attendibile.

Un flusso di lavoro tipico prevede i seguenti passaggi.

Passi

1. Genera un certificato client e una chiave. Durante la generazione, impostare il nome comune (CN) sull'utente ONTAP con cui effettuare l'autenticazione.

```
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 . Questa operazione potrebbe essere già gestita dall'amministratore dell'archiviazione. Ignora 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 client e la chiave (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. Conferma che il ruolo di accesso alla sicurezza ONTAP supporta `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 l'autenticazione utilizzando il certificato generato. Sostituire < ONTAP Management LIF> e <nome vserver> con l'IP Management LIF e il nome SVM. È necessario assicurarsi che la politica di servizio del 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 il certificato, la chiave e il 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...Vaaalllluuuueeee",
"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 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Aggiorna i metodi di autenticazione o ruota le credenziali

È possibile aggiornare un backend esistente per utilizzare un metodo di autenticazione diverso o per ruotare le credenziali. Funziona in entrambi i modi: i backend che utilizzano nome utente/password possono essere aggiornati per utilizzare certificati; i backend che utilizzano certificati possono essere aggiornati per utilizzare nome utente/password. Per fare ciò, 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 richiesti per eseguire `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 dell'archiviazione deve prima aggiornare la password per l'utente su ONTAP. Segue un aggiornamento del backend. 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 eliminato dal cluster ONTAP .

L'aggiornamento di un backend non interrompe l'accesso ai volumi già creati, né influisce sulle connessioni ai volumi effettuate in seguito. Un aggiornamento del backend riuscito indica che Trident può comunicare con il backend ONTAP e gestire le future operazioni sui volumi.

Crea un ruolo ONTAP personalizzato per Trident

È possibile creare un ruolo cluster ONTAP con privilegi minimi in modo da non dover utilizzare il ruolo di amministratore ONTAP per eseguire 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.

Fare riferimento a"Generatore di ruoli personalizzati Trident" per ulteriori informazioni sulla creazione di ruoli personalizzati Trident .

Utilizzo di ONTAP CLI

1. Crea un nuovo ruolo utilizzando il seguente comando:

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

2. Crea 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. Assegna 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

Eseguire i seguenti passaggi in ONTAP System Manager:

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 SVM, selezionare **Archiviazione > VM di archiviazione > `required SVM` > Impostazioni > Utenti e ruoli**.

   b. Selezionare l'icona della freccia (→) accanto a **Utenti e ruoli**.
   c. Selezionare **+Aggiungi** in **Ruoli**.
   d. Definisci le regole per il ruolo e clicca su **Salva**.
2. **Assegnare il ruolo all'utente Trident *: + Eseguire i seguenti passaggi nella pagina *Utenti e ruoli**:
   a. Selezionare Aggiungi icona **+** in **Utenti**.
   b. Selezionare il nome utente richiesto e selezionare un ruolo nel menu a discesa per **Ruolo**.
   c. Fare clic su **Salva**.

Per maggiori informazioni consultare le seguenti pagine:

- "Ruoli personalizzati per l'amministrazione di ONTAP"O"Definisci ruoli personalizzati"
- "Lavorare con ruoli e utenti"

Gestire le policy di esportazione NFS

Trident utilizza criteri di esportazione NFS per controllare l'accesso ai volumi di cui si occupa.

Trident offre due opzioni quando si lavora con le politiche di esportazione:

- Trident può gestire dinamicamente la politica di esportazione autonomamente; in questa modalità di

funzionamento, l'amministratore dell'archiviazione specifica un elenco di blocchi CIDR che rappresentano indirizzi IP ammissibili. Trident aggiunge automaticamente alla policy di esportazione gli IP dei nodi applicabili che rientrano in questi intervalli al momento della pubblicazione. In alternativa, se non vengono specificati CIDR, tutti gli IP unicast con ambito globale trovati sul nodo su cui viene pubblicato il volume verranno aggiunti alla policy di esportazione.

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

Gestire dinamicamente le politiche di esportazione

Trident offre la possibilità di gestire dinamicamente le policy di esportazione per i backend ONTAP . Ciò consente all'amministratore dell'archiviazione di specificare uno spazio di indirizzamento consentito per gli IP dei nodi worker, anziché definire manualmente regole esplicite. Semplifica notevolmente la gestione delle policy di esportazione: le modifiche alle policy di esportazione non richiedono più un intervento manuale sul cluster di storage. Inoltre, ciò consente di limitare l'accesso al cluster di archiviazione solo ai nodi worker che montano volumi e hanno IP compresi nell'intervallo specificato, supportando una gestione automatizzata e dettagliata.

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

Esempio

Ci sono due opzioni di configurazione che devono essere utilizzate. Ecco un esempio di definizione del 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 funzionalità, è necessario assicurarsi che la giunzione radice nella SVM disponga di una policy di esportazione creata in precedenza con una regola di esportazione che consenta il blocco CIDR del nodo (ad esempio la policy di esportazione predefinita). Seguire sempre le best practice consigliate NetApp per dedicare una SVM a Trident.

Ecco una spiegazione del funzionamento di questa funzionalità utilizzando l'esempio sopra riportato:

- `autoExportPolicy`è impostato su `true`. Ciò indica che Trident crea una policy di esportazione per ogni volume fornito con questo backend per il `svm1` SVM e gestire l'aggiunta e l'eliminazione delle regole utilizzando `autoexportCIDRs` blocchi di indirizzi. Finché un volume non viene collegato a un nodo, il volume utilizza una policy di esportazione vuota, senza regole, per impedire l'accesso indesiderato a tale volume. Quando un volume viene pubblicato su 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 alla policy di esportazione utilizzata dal FlexVol volume padre
  - Per esempio:
    - UUID backend 403b5326-8482-40db-96d0-d83fb3f4daec
    - `autoExportPolicy`impostato su `true`
    - prefisso di archiviazione `trident`
    - Codice UUID PVC a79bcf5f-7b6d-4a40-9876-e2551f159c1c
    - qtree denominato trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c crea una politica di esportazione per FlexVol denominata `trident-403b5326-8482-40db96d0-d83fb3f4daec`, una politica di esportazione per il qtree denominato `trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c` e una politica di esportazione vuota denominata `trident_empty` sull'SVM. Le regole per la policy di esportazione FlexVol saranno un superset di tutte le regole contenute nelle policy di esportazione qtree. La policy di esportazione vuota verrà riutilizzata da tutti i volumi non collegati.
- `autoExportCIDRs`contiene un elenco di blocchi di indirizzi. Questo campo è facoltativo e il suo valore predefinito è ["0.0.0.0/0", "::/0"]. Se non definito, Trident aggiunge tutti gli indirizzi unicast con ambito globale trovati sui nodi worker con pubblicazioni.

In questo esempio, il `192.168.0.0/24` è fornito lo spazio di indirizzamento. Ciò indica che gli IP dei nodi Kubernetes che rientrano in questo intervallo di indirizzi con pubblicazioni verranno aggiunti alla policy di esportazione creata Trident . Quando Trident registra un nodo su cui è in esecuzione, recupera gli indirizzi IP del nodo e li confronta con i blocchi di indirizzi forniti in `autoExportCIDRs` Al momento della pubblicazione, dopo aver filtrato gli IP, Trident crea le regole della policy di esportazione per gli IP client del nodo su cui sta pubblicando.

Puoi aggiornare `autoExportPolicy` E `autoExportCIDRs` per i backend dopo averli creati. È possibile aggiungere nuovi CIDR per un backend gestito automaticamente oppure eliminare i CIDR esistenti. Prestare attenzione quando si eliminano i CIDR per assicurarsi che le connessioni esistenti non vengano interrotte. Puoi anche scegliere di disabilitare `autoExportPolicy` per un backend e ripiegare su una policy di esportazione creata manualmente. Ciò richiederà l'impostazione del `exportPolicy` parametro nella configurazione del backend.

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

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

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

> L'eliminazione di un backend con criteri di esportazione gestiti automaticamente eliminerà il criterio di esportazione creato dinamicamente. Se il backend viene ricreato, viene trattato come un nuovo backend e comporterà 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 la politica di esportazione per i backend che gestisce per riflettere questa modifica dell'IP.

Prepararsi al provisioning dei volumi SMB

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

> È necessario configurare entrambi i protocolli NFS e SMB/CIFS sull'SVM per creare un `ontap-nas-economy` Volume SMB per cluster ONTAP on-premise. La mancata configurazione di uno di questi protocolli causerà il fallimento della creazione del volume SMB.

`autoExportPolicy`non è supportato per i volumi SMB.

Prima di iniziare

Prima di poter effettuare il provisioning dei volumi SMB, è necessario disporre di quanto segue.

- Un cluster Kubernetes con un nodo controller Linux e almeno un nodo worker Windows che esegue Windows Server 2022. Trident supporta volumi SMB montati su pod in esecuzione solo su 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'
```

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

Passi

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

   Le condivisioni SMB sono necessarie per Amazon FSx per ONTAP.

   È possibile creare le condivisioni amministrative SMB in uno dei due modi seguenti: utilizzando"Console di gestione Microsoft" Snap-in Cartelle condivise o tramite ONTAP CLI. Per creare le condivisioni SMB utilizzando ONTAP CLI:

   a. Se necessario, creare la struttura del percorso della 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 fallisce.

   b. Crea una condivisione SMB associata all'SVM specificato:

```
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"Crea una condivisione SMB" per maggiori dettagli.

2. Durante la creazione del backend, è necessario configurare quanto segue per specificare i volumi SMB. Per tutte le opzioni di configurazione del backend FSx for ONTAP , fare riferimento a"Opzioni di configurazione ed esempi di FSx per ONTAP" .

Parametro	Descrizione	Esempio
`smbShare`	È possibile specificare uno dei seguenti elementi: il nome di una condivisione SMB creata tramite Microsoft Management Console o ONTAP CLI; un nome per consentire a Trident di creare la condivisione SMB; oppure è possibile lasciare il parametro vuoto per impedire l'accesso comune alla condivisione 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 nullo, il valore predefinito è `nfs` .	`smb`
`securityStyle`	Stile di sicurezza per i nuovi volumi. **Deve essere impostato su `ntfs` O `mixed` per volumi SMB.**	`ntfs`O `mixed` per volumi SMB
`unixPermissions`	Modalità per 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 è abilitato SMB sicuro, è possibile fornire un accesso controllato alle condivisioni SMB per gli utenti e i 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 abiliti SMB sicuro. Gli utenti non validi non verranno aggiunti all'ACL.
- Se si forniscono autorizzazioni diverse allo stesso utente AD nel backend, nella classe di archiviazione e nel PVC, la priorità delle autorizzazioni sarà: PVC, classe di archiviazione e quindi backend.
- Secure SMB è supportato per `ontap-nas` importazioni di volumi gestiti e non applicabile alle importazioni di volumi non gestiti.

Passi

1. Specificare adAdminUser in TridentBackendConfig come mostrato nell'esempio seguente:

```
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.

   Aggiungi 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 lo stesso del nome utente specificato nel `smbcreds` segreto. Puoi scegliere una delle seguenti opzioni per `smbShareAdUserPermission` : `full_control` , `change` , O `read` . L'autorizzazione predefinita è `full_control` .

```
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 ONTAP NAS con l'installazione Trident . Questa sezione fornisce esempi di configurazione del backend e dettagli per il mapping dei backend alle StorageClass.

Opzioni di configurazione del backend

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

Parametro	Descrizione	Predefinito
`version`		Sempre 1
`storageDriverName`	Nome del driver di archiviazione	`ontap-nas`, `ontap-nas-economy` , O `ontap-nas-flexgroup`
`backendName`	Nome personalizzato o backend di archiviazione	Nome del driver + "_" + dataLIF
`managementLIF`	Indirizzo IP di un cluster o di un LIF di gestione SVM. È possibile specificare un nome di dominio completo (FQDN). Può essere impostato per utilizzare 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 un passaggio senza interruzioni a MetroCluster , vedereEsempio 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 specificato, Trident recupera i dataLIF dall'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 dataLIF. Può essere modificato dopo l'impostazione iniziale. Fare riferimento a . Può essere impostato per utilizzare 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.** Vedi ilEsempio MetroCluster .	Indirizzo specificato o derivato da SVM, se non specificato (non consigliato)
`svm`	Macchina virtuale di archiviazione da utilizzare **Ometti per Metrocluster.** Vedi ilEsempio MetroCluster .	Derivato se un SVM `managementLIF` è specificato
`autoExportPolicy`	Abilita la creazione e l'aggiornamento automatici dei criteri di esportazione [Booleano]. Utilizzando il `autoExportPolicy` E `autoExportCIDRs` opzioni, Trident può gestire automaticamente le policy di esportazione.	falso
`autoExportCIDRs`	Elenco di CIDR per filtrare gli IP dei nodi di Kubernetes quando `autoExportPolicy` è abilitato. Utilizzando il `autoExportPolicy` E `autoExportCIDRs` opzioni, Trident può gestire automaticamente le policy di esportazione.	["0.0.0.0/0", "::/0"]`
`labels`	Insieme di etichette arbitrarie formattate in JSON da applicare ai volumi	""
`clientCertificate`	Valore codificato in Base64 del certificato client. 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 connettersi al cluster/SVM. Utilizzato per l'autenticazione basata sulle credenziali. Per l'autenticazione di Active Directory, vedere "Autenticare Trident su un SVM backend utilizzando le credenziali di Active Directory".	
`password`	Password per connettersi al cluster/SVM. Utilizzato per l'autenticazione basata sulle credenziali. Per l'autenticazione di Active Directory, vedere "Autenticare Trident su un SVM backend utilizzando le credenziali di Active Directory".	

Parametro	Descrizione	Predefinito
`storagePrefix`	Prefisso utilizzato durante il provisioning di nuovi volumi nell'SVM. Non può essere aggiornato dopo averlo impostato Quando si utilizza ontap-nas-economy e un prefisso storage di 24 o più caratteri, i qtree non avranno il prefisso storage incorporato, sebbene sarà presente nel nome del volume.	"tridente"
`aggregate`	Aggregato per il provisioning (facoltativo; se impostato, deve essere assegnato all'SVM). Per il `ontap-nas-flexgroup` driver, questa opzione viene ignorata. Se non assegnato, è possibile utilizzare uno qualsiasi degli aggregati disponibili per eseguire il provisioning di un volume FlexGroup . Quando l'aggregato viene aggiornato in SVM, viene aggiornato automaticamente in Trident interrogando SVM senza dover riavviare Trident Controller. Dopo aver configurato un aggregato specifico in Trident per il provisioning dei volumi, se l'aggregato viene rinominato o spostato fuori dall'SVM, il backend passerà allo stato di errore in Trident durante l'interrogazione dell'aggregato SVM. Per riportare online il backend, è necessario modificare l'aggregato con uno presente sull'SVM oppure rimuoverlo del tutto.	""
`limitAggregateUsage`	Il provisioning non riesce se l'utilizzo è superiore a questa percentuale. **Non si applica ad Amazon FSx per ONTAP**.	"" (non applicato di default)

Parametro	Descrizione	Predefinito
flexgroupAggregateList	Elenco degli aggregati per il provisioning (facoltativo; se impostato, deve essere assegnato all'SVM). Tutti gli aggregati assegnati all'SVM vengono utilizzati per effettuare il provisioning di un volume FlexGroup . Supportato per il driver di archiviazione **ontap-nas-flexgroup**. Quando l'elenco aggregato viene aggiornato in SVM, l'elenco viene aggiornato automaticamente in Trident interrogando SVM senza dover riavviare Trident Controller. Dopo aver configurato un elenco di aggregati specifico in Trident per il provisioning dei volumi, se l'elenco di aggregati viene rinominato o spostato fuori da SVM, il backend passerà allo stato di errore in Trident durante l'interrogazione dell'aggregato SVM. Per riportare online il backend, è necessario modificare l'elenco aggregato con uno presente sull'SVM oppure rimuoverlo del tutto.	""
`limitVolumeSize`	Il provisioning non riesce se la dimensione del volume richiesto è superiore a questo valore. Limita inoltre la dimensione massima dei volumi che gestisce per i qtree e `qtreesPerFlexvol` l'opzione consente di personalizzare il numero massimo di qtree per FlexVol volume	"" (non applicato di default)
`debugTraceFlags`	Flag di debug da utilizzare durante la risoluzione dei problemi. Esempio, {"api":false, "method":true} Non utilizzare `debugTraceFlags` a meno che non si stia risolvendo un problema e si necessiti di un dump di registro dettagliato.	null
`nasType`	Configurare la creazione di volumi NFS o SMB. Le opzioni sono `nfs` , `smb` o nullo. Impostando il valore su null, i volumi NFS vengono impostati di default.	`nfs`
`nfsMountOptions`	Elenco separato da virgole delle opzioni di montaggio NFS. Le opzioni di montaggio per i volumi persistenti di Kubernetes sono normalmente specificate nelle classi di archiviazione, ma se non vengono specificate opzioni di montaggio in una classe di archiviazione, Trident utilizzerà le opzioni di montaggio specificate nel file di configurazione del backend di archiviazione. Se non vengono specificate opzioni di montaggio nella classe di archiviazione o nel file di configurazione, Trident non imposterà alcuna opzione di montaggio su un volume persistente associato.	""

Parametro	Descrizione	Predefinito
`qtreesPerFlexvol`	Il numero massimo di Qtree per FlexVol deve essere compreso nell'intervallo [50, 300]	"200"
`smbShare`	È possibile specificare uno dei seguenti elementi: il nome di una condivisione SMB creata tramite Microsoft Management Console o ONTAP CLI; un nome per consentire a Trident di creare la condivisione SMB; oppure è possibile lasciare il parametro vuoto per impedire l'accesso comune alla condivisione 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 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).	`true`per ONTAP 9.15.1 o successivo, altrimenti `false` .
`limitVolumePoolSize`	Dimensione massima FlexVol richiedibile quando si utilizzano Qtrees nel backend ontap-nas-economy.	"" (non applicato di default)
`denyNewVolumePools`	Limita `ontap-nas-economy` backend dalla creazione di nuovi volumi FlexVol per contenere i loro Qtree. Per il provisioning di nuovi PV 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 del backend per il provisioning dei volumi

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

Parametro	Descrizione	Predefinito
`spaceAllocation`	Assegnazione dello spazio per Qtrees	"VERO"

Parametro	Descrizione	Predefinito
`spaceReserve`	Modalità di prenotazione dello spazio; "nessuno" (sottile) o "volume" (spesso)	"nessuno"
`snapshotPolicy`	Criterio di snapshot da utilizzare	"nessuno"
`qosPolicy`	Gruppo di criteri QoS da assegnare ai volumi creati. Scegli uno tra qosPolicy o adaptiveQosPolicy per pool di archiviazione/backend	""
`adaptiveQosPolicy`	Gruppo di policy QoS adattivo da assegnare ai volumi creati. Scegliere tra qosPolicy o adaptiveQosPolicy per ogni pool di archiviazione/backend. Non supportato da ontap-nas-economy.	""
`snapshotReserve`	Percentuale di volume riservata agli snapshot	"0" se `snapshotPolicy` è "nessuno", altrimenti ""
`splitOnClone`	Dividere un clone dal suo genitore al momento della creazione	"falso"
`encryption`	Abilita NetApp Volume Encryption (NVE) sul nuovo volume; l'impostazione predefinita è `false` . Per utilizzare questa opzione, NVE deve essere concesso in licenza e abilitato sul cluster. Se NAE è abilitato sul backend, qualsiasi volume fornito in Trident sarà abilitato per NAE. Per maggiori informazioni, fare riferimento a:"Come funziona Trident con NVE e NAE" .	"falso"
`tieringPolicy`	Criterio di tiering per utilizzare "nessuno"	
`unixPermissions`	Modalità per nuovi volumi	"777" per i volumi NFS; vuoto (non applicabile) per i volumi SMB
`snapshotDir`	Controlla l'accesso al `.snapshot` elenco	"true" per NFSv4 "false" per NFSv3
`exportPolicy`	Politica di esportazione da utilizzare	"predefinito"
`securityStyle`	Stile di sicurezza per i nuovi volumi. Supporti NFS `mixed` E `unix` stili di sicurezza. Supporti SMB `mixed` E `ntfs` stili di sicurezza.	L'impostazione predefinita di NFS è `unix` . L'impostazione predefinita di SMB è `ntfs` .
`nameTemplate`	Modello per creare nomi di volume personalizzati.	""

Per utilizzare i gruppi di policy QoS con Trident è necessario ONTAP 9.8 o versione successiva. Dovresti utilizzare un gruppo di policy QoS non condiviso e assicurarti che il gruppo di policy venga applicato individualmente a ciascun componente. Un gruppo di policy QoS condiviso impone il limite massimo per la produttività totale di tutti i carichi di lavoro.

Esempi di provisioning del volume

Ecco un esempio con i valori predefiniti definiti:

```
---
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 FlexVol sia dimensionato correttamente con la percentuale 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 ha richiesto è 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 IL `volume show` il comando dovrebbe mostrare risultati simili a questo esempio:

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 produceva 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 minima

Gli esempi seguenti mostrano 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 i LIF anziché gli indirizzi IP.

Esempio di economia NAS ONTAP

```
---
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 ONTAP NAS Flexgroup

```
---
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 il passaggio e il ritorno durante"Replica e ripristino SVM" .

Per un passaggio e un ritorno senza interruzioni, specificare l'SVM utilizzando `managementLIF` e omettere il `dataLIF` E `svm` parametri. Per 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

Questo è un esempio minimo di configurazione backend. `clientCertificate` , `clientPrivateKey` , E `trustedCACertificate` (facoltativo, se si utilizza una CA attendibile) vengono popolati in `backend.json` e prendere rispettivamente i valori codificati in 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

Questo esempio mostra come è possibile istruire Trident a utilizzare criteri di esportazione dinamici per creare e gestire automaticamente i criteri di esportazione. Funziona allo stesso modo per il `ontap-nas-economy` E `ontap-nas-flexgroup` conducenti.

```
---
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 Amazon FSx per ONTAP che utilizza 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 del 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 backend di esempio mostrati di seguito, vengono impostati valori predefiniti specifici per tutti i pool di archiviazione, ad esempio `spaceReserve` a nessuno, `spaceAllocation` a falso, 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 per `ontap-nas` o FlexGroup per `ontap-nas-flexgroup` . Trident copia tutte le etichette presenti su un pool virtuale nel volume di archiviazione al momento del provisioning. Per comodità, gli amministratori di storage possono definire etichette per pool virtuale e raggruppare i volumi in base all'etichetta.

In questi esempi, alcuni dei pool di archiviazione impostano i propri `spaceReserve` , `spaceAllocation` , E `encryption` valori e alcuni pool sovrascrivono i valori predefiniti.

Esempio ONTAP NAS

```
---
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 ONTAP NAS FlexGroup

```
---
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

```
---
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"
```

Mappa i backend su StorageClasses

Le seguenti definizioni StorageClass fanno riferimento aEsempi di backend con pool virtuali . Utilizzando il `parameters.selector` campo, ogni StorageClass richiama quali pool virtuali possono essere utilizzati per ospitare un volume. Il volume avrà gli aspetti definiti nel pool virtuale scelto.

- IL `protection-gold` StorageClass verrà mappato sul primo e sul secondo pool virtuale in `ontap-nas-flexgroup` backend. Queste sono le uniche piscine 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 verrà mappato sul terzo e quarto pool virtuale in `ontap-nas-flexgroup` backend. 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 verrà mappato sul quarto pool virtuale nel `ontap-nas` backend. Questo è l'unico pool che offre la configurazione del pool di archiviazione per l'app 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 verrà mappato sul terzo pool virtuale in `ontap-nas-flexgroup` backend. Questo è l'unico pool che offre protezione di livello Silver e 20.000 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 verrà mappato sul terzo pool virtuale in `ontap-nas` backend e il secondo pool virtuale nel `ontap-nas-economy` backend. 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 selezionare e garantirà che i requisiti di archiviazione siano soddisfatti.

Aggiornamento `dataLIF` dopo la configurazione iniziale

È possibile modificare il dataLIF dopo la configurazione iniziale eseguendo il comando seguente per fornire il nuovo file JSON backend con il dataLIF aggiornato.

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

Se i PVC sono collegati a uno o più pod, è necessario disattivare tutti i pod corrispondenti e quindi riattivarli affinché il nuovo dataLIF abbia effetto.

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

```
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

```
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. Secure SMB 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 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 PVC con più utenti AD

```
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 for NetApp ONTAP

Utilizzare Trident con Amazon FSx for NetApp ONTAP

"Amazon FSx for NetApp ONTAP"è un servizio AWS completamente gestito che consente ai clienti di avviare ed eseguire file system basati sul sistema operativo di storage NetApp ONTAP . FSx for ONTAP ti consente di sfruttare le funzionalità, le prestazioni e le capacità amministrative 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 for NetApp ONTAP con Trident per garantire che i cluster Kubernetes in esecuzione in Amazon Elastic Kubernetes Service (EKS) possano fornire volumi persistenti a blocchi e file supportati da ONTAP.

Un file system è la risorsa principale in Amazon FSx, analogamente a un cluster ONTAP in locale. All'interno di ogni SVM è possibile creare uno o più volumi, ovvero contenitori di dati in cui vengono archiviati i file e le cartelle nel file system. Con Amazon FSx for NetApp ONTAP verrà fornito come file system gestito nel cloud. Il

nuovo tipo di file system si chiama * NetApp ONTAP*.

Utilizzando Trident con Amazon FSx for NetApp ONTAP, puoi garantire che i cluster Kubernetes in esecuzione in Amazon Elastic Kubernetes Service (EKS) possano fornire volumi persistenti di file e blocchi supportati da ONTAP.

Requisiti

Inoltre"Requisiti Trident" Per integrare FSx per ONTAP con Trident, è necessario:

- Un cluster Amazon EKS esistente o un cluster Kubernetes autogestito con `kubectl` installato.
- Un file system Amazon FSx for NetApp ONTAP esistente e una macchina virtuale di storage (SVM) raggiungibile dai nodi worker del cluster.
- Nodi worker preparati per"NFS o iSCSI" .

> Assicurati di seguire i passaggi di preparazione del nodo richiesti per Amazon Linux e Ubuntu "Immagini della macchina Amazon" (AMI) a seconda del tipo di AMI EKS.

Considerazioni

- Volumi SMB:
  - I volumi SMB sono supportati utilizzando `ontap-nas` solo conducente.
  - I volumi SMB non sono supportati dal componente aggiuntivo Trident EKS.
  - Trident supporta volumi SMB montati su pod in esecuzione solo su nodi Windows. Fare riferimento a "Prepararsi al provisioning dei volumi SMB" per i dettagli.
- Prima di Trident 24.02, i volumi creati sui file system Amazon FSx con backup automatici abilitati non potevano essere eliminati da Trident. Per evitare questo problema in Trident 24.02 o versioni successive, specificare `fsxFilesystemID` , AWS `apiRegion` , AWS `apikey` e AWS `secretKey` nel file di configurazione backend per AWS FSx per ONTAP.

> Se si specifica un ruolo IAM per Trident, è possibile omettere di specificare il `apiRegion` , `apiKey` , E `secretKey` campi a Trident in modo esplicito. Per maggiori informazioni, fare riferimento a"Opzioni di configurazione ed esempi di FSx per 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 usare il `fsxadmin` utente per il tuo file system o il `vsadmin` utente configurato per il tuo SVM.

> Trident si aspetta di essere gestito come un `vsadmin` Utente SVM o come utente con un nome diverso che ha lo stesso ruolo. Amazon FSx for NetApp ONTAP ha un `fsxadmin` utente che è una sostituzione limitata ONTAP `admin` utente del cluster. Consigliamo vivamente di utilizzare `vsadmin` con Trident.

- Basato su certificato: Trident comunicherà con l'SVM sul file system FSx utilizzando un certificato installato sull'SVM.

Per i dettagli sull'abilitazione dell'autenticazione, fare riferimento all'autenticazione per il tipo di driver:

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

Immagini macchina Amazon (AMI) testate

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

AMI	NAS	NAS-economia	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 l'AMI desiderata non è elencata qui, non significa che non sia supportata; significa semplicemente che non è stata testata. Questo elenco serve come guida per gli AMI di cui è noto il funzionamento.

Test eseguiti con:

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

Test eseguiti:

- Crea: Classe di archiviazione, pvc, pod
- Elimina: pod, pvc (normale, qtree/lun – economy, NAS con backup AWS)

Trova maggiori informazioni

- "Documentazione Amazon FSx for NetApp ONTAP"
- "Post del blog su Amazon FSx for NetApp ONTAP"

Crea un ruolo IAM e un segreto AWS

È possibile configurare i pod Kubernetes per accedere alle risorse AWS autenticandosi come ruolo AWS IAM anziché fornire credenziali AWS esplicite.

> Per eseguire l'autenticazione tramite un ruolo AWS IAM, è necessario disporre di un cluster Kubernetes distribuito tramite EKS.

Crea il segreto di AWS Secrets Manager

Poiché Trident emetterà API su un server virtuale FSx per gestire l'archiviazione per te, avrà bisogno delle credenziali per farlo. Il modo sicuro per trasmettere tali credenziali è tramite un segreto AWS Secrets Manager. Pertanto, se non ne hai già uno, dovrai creare un segreto AWS Secrets Manager che contenga le credenziali per l'account vsadmin.

Questo esempio crea un segreto AWS Secrets Manager per archiviare le credenziali Trident CSI:

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

Crea policy IAM

Per funzionare correttamente, Trident necessita anche delle autorizzazioni AWS. Pertanto, è necessario creare una policy che fornisca a Trident le autorizzazioni di cui ha bisogno.

Gli esempi seguenti creano una policy IAM utilizzando l'AWS CLI:

```
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"
```

Esempio di JSON della policy:

```
{
  "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 identità Pod o ruolo IAM per l'associazione dell'account di servizio (IRSA)

È possibile configurare un account di servizio Kubernetes affinché assuma 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 Amazon EKS Pod Identity consentono 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:

Creare trust-relationship.json per consentire al Service Principal EKS di assumere questo ruolo per Pod Identity. 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-relationship.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>** - ID del tuo account AWS
- **<oidc_provider>** - L'OIDC del 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, associare 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>
```

Verifica che il fornitore OICD sia associato:

Verifica che il tuo provider OIDC sia associato al tuo cluster. Puoi verificarlo usando questo 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 l'esempio seguente 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
```

Installa Trident

Trident semplifica la gestione dello storage Amazon FSx for NetApp ONTAP in Kubernetes, consentendo a sviluppatori e amministratori di concentrarsi sulla distribuzione delle applicazioni.

Puoi installare Trident utilizzando uno dei seguenti metodi:

- Timone
- Componente aggiuntivo EKS

Se si desidera utilizzare la funzionalità snapshot, installare il componente aggiuntivo CSI Snapshot Controller. Fare riferimento a"Abilita la funzionalità snapshot per i volumi CSI" per maggiori informazioni.

Installa 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
```

Puoi usare il `helm list` comando per rivedere i dettagli dell'installazione quali 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 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
```

Puoi usare il `helm list` comando per rivedere i dettagli dell'installazione quali 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 intendi utilizzare iSCSI, assicurati che iSCSI sia abilitato sul tuo computer client. Se si utilizza il sistema operativo AL2023 Worker node, è possibile 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}
```

Installa Trident tramite il componente aggiuntivo EKS

Il componente aggiuntivo Trident EKS include le patch di sicurezza più recenti, le correzioni di bug ed è convalidato da AWS per funzionare con Amazon EKS. Il componente aggiuntivo EKS consente di garantire costantemente la sicurezza e la stabilità dei cluster Amazon EKS e di ridurre la quantità di lavoro necessaria 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 aggiuntivo
- Autorizzazioni AWS per il marketplace AWS:
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- Tipo 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 for NetApp ONTAP esistente

Abilita 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 navigazione a sinistra, seleziona **Cluster**.
3. Selezionare il nome del cluster per il quale si desidera configurare il componente aggiuntivo NetApp Trident CSI.
4. Seleziona **Componenti aggiuntivi** e poi **Ottieni altri componenti aggiuntivi**.
5. Per selezionare il componente aggiuntivo, seguire 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 delle impostazioni **Configura componenti aggiuntivi selezionati**, procedi come segue:

   Salta questi passaggi se utilizzi l'associazione Pod Identity.

   a. Seleziona la **Versione** che desideri utilizzare.
   b. Se si utilizza l'autenticazione IRSA, assicurarsi di impostare i valori di configurazione disponibili nelle impostazioni di configurazione facoltative:
      - Seleziona la **Versione** che desideri 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 Sostituisci per il metodo di risoluzione dei conflitti, una o più impostazioni del componente aggiuntivo esistente possono essere sovrascritte con le impostazioni del componente aggiuntivo Amazon EKS. Se non si abilita questa opzione e si verifica un conflitto con le impostazioni esistenti, l'operazione fallisce. 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 impostazioni che devi gestire autonomamente.
7. Selezionare **Avanti**.
8. Nella pagina **Revisiona e aggiungi**, seleziona **Crea**.

   Una volta completata l'installazione del componente aggiuntivo, verrà visualizzato il componente aggiuntivo installato.

Interfaccia a riga di comando 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
```

Aggiorna il componente aggiuntivo Trident EKS

Console di gestione

1. Apri la console Amazon EKS https://console.aws.amazon.com/eks/home#/clusters .
2. Nel riquadro di navigazione a sinistra, seleziona **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* e quindi **Modifica**.
6. Nella pagina **Configura Trident di NetApp**, procedere come segue:
   a. Seleziona la **Versione** che desideri utilizzare.
   b. Espandi le **Impostazioni di configurazione facoltative** e modificale secondo necessità.
   c. Seleziona **Salva modifiche**.

Interfaccia a riga di comando AWS

L'esempio seguente aggiorna 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

- Controlla la versione corrente del tuo componente aggiuntivo FSxN Trident CSI. Sostituire `my-cluster` con il nome del tuo 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 il `--force` opzione e una qualsiasi delle impostazioni del componente aggiuntivo Amazon EKS è in conflitto con le impostazioni esistenti, l'aggiornamento del componente aggiuntivo Amazon EKS non riesce e viene visualizzato un messaggio di errore che consente di risolvere il conflitto. Prima di specificare questa opzione, assicurati che il componente aggiuntivo Amazon EKS non gestisca le impostazioni che devi gestire, perché tali impostazioni vengono sovrascritte con questa opzione. Per ulteriori informazioni sulle altre opzioni per questa impostazione, vedere"Componenti aggiuntivi" . Per ulteriori informazioni sulla gestione sul campo di Amazon EKS Kubernetes, vedere"Gestione del campo Kubernetes" .

Disinstallare/rimuovere il componente aggiuntivo Trident EKS

Per rimuovere un componente aggiuntivo Amazon EKS sono disponibili due opzioni:

- **Conserva il software aggiuntivo sul tuo cluster**: questa opzione rimuove la gestione di qualsiasi impostazione da parte di Amazon EKS. Rimuove inoltre la possibilità per Amazon EKS di notificare gli aggiornamenti e di aggiornare automaticamente il componente aggiuntivo Amazon EKS dopo aver avviato un aggiornamento. Tuttavia, mantiene il software aggiuntivo sul cluster. Questa opzione rende il componente aggiuntivo un'installazione autogestita, anziché un componente aggiuntivo Amazon EKS. Con questa opzione non ci saranno tempi di inattività per il componente aggiuntivo. Conservare il `--preserve` opzione nel comando per preservare il componente aggiuntivo.
- **Rimuovere completamente il software aggiuntivo dal cluster**: NetApp consiglia di rimuovere il componente aggiuntivo Amazon EKS dal cluster solo se nel cluster non sono presenti risorse che dipendono da esso. Rimuovere il `--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 navigazione a sinistra, seleziona **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** e quindi selezionare * Trident by NetApp*.*
5. Seleziona **Rimuovi**.
6. Nella finestra di dialogo **Rimuovi conferma netapp_trident-operator**, procedere come segue:
   a. Se desideri che Amazon EKS interrompa la gestione delle impostazioni per il componente aggiuntivo, seleziona **Conserva nel cluster**. Eseguire questa operazione se si desidera mantenere il software aggiuntivo sul cluster, in modo da poter gestire autonomamente tutte le impostazioni dell'add-on.
   b. Inserisci **netapp_trident-operator**.
   c. Seleziona **Rimuovi**.

Interfaccia a riga di comando AWS

Sostituire `my-cluster` con il nome del tuo cluster, quindi 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 l'SVM da cui ottenerlo e come autenticarsi. L'esempio seguente mostra come definire l'archiviazione basata su NAS e utilizzare un segreto AWS per archiviare le credenziali nell'SVM che si desidera utilizzare:

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

```
{
  "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):

- Crea la configurazione del backend Trident (TBC) dal file yaml ed esegui il seguente comando:

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

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

- Convalida che la configurazione del backend 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

È possibile integrare Trident con Amazon FSx for NetApp ONTAP utilizzando i seguenti driver:

- `ontap-san`: Ogni PV fornito è una LUN all'interno del proprio volume Amazon FSx for NetApp ONTAP . Consigliato per l'archiviazione a blocchi.
- `ontap-nas`: Ogni PV fornito è un volume Amazon FSx for NetApp ONTAP completo. Consigliato per NFS e SMB.
- `ontap-san-economy`: Ogni PV fornito è una LUN con un numero configurabile di LUN per volume Amazon FSx for NetApp ONTAP .
- `ontap-nas-economy`: Ogni PV fornito è un qtree, con un numero configurabile di qtree per volume Amazon FSx for NetApp ONTAP .
- `ontap-nas-flexgroup`: Ogni PV fornito è un volume Amazon FSx for NetApp ONTAP FlexGroup completo.

Per i dettagli del driver, 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 del backend ed esempi

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

Parametro	Descrizione	Esempio
`version`		Sempre 1
`storageDriverName`	Nome del driver di archiviazione	`ontap-nas`, `ontap-nas-economy` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy`
`backendName`	Nome personalizzato o backend di archiviazione	Nome del driver + "_" + dataLIF
`managementLIF`	Indirizzo IP di un cluster o di un LIF di gestione SVM. È possibile specificare un nome di dominio completo (FQDN). Può essere impostato per utilizzare 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 il `aws` campo, non è necessario fornire il `managementLIF` perché Trident recupera l'SVM `managementLIF` informazioni da AWS. Quindi, è necessario fornire le credenziali per un utente sotto l'SVM (ad esempio: vsadmin) e l'utente deve avere il `vsadmin` ruolo.	"10.0.0.1", "[2001:1234:abcd::fefe]"

Parametro	Descrizione	Esempio
`dataLIF`	Indirizzo IP del protocollo LIF. * Driver ONTAP NAS*: NetApp consiglia di specificare dataLIF. Se non specificato, Trident recupera i dataLIF dall'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 dataLIF. Può essere modificato dopo l'impostazione iniziale. Fare riferimento a . * Driver ONTAP SAN*: Non specificare per iSCSI. Trident utilizza ONTAP Selective LUN Map per scoprire gli iSCI LIF necessari per stabilire una sessione multi-percorso. Se dataLIF è definito in modo esplicito, viene generato un avviso. Può essere impostato per utilizzare 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 il `autoExportPolicy` E `autoExportCIDRs` opzioni, Trident può gestire automaticamente le policy di esportazione.	`false`
`autoExportCIDRs`	Elenco di CIDR per filtrare gli IP dei nodi di Kubernetes quando `autoExportPolicy` è abilitato. Utilizzando il `autoExportPolicy` E `autoExportCIDRs` opzioni, Trident può gestire automaticamente le policy di esportazione.	"["0.0.0.0/0", "::/0"]"
`labels`	Insieme di etichette arbitrarie formattate in JSON da applicare ai volumi	""
`clientCertificate`	Valore codificato in Base64 del certificato client. 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 connettersi al cluster o alla SVM. Utilizzato per l'autenticazione basata sulle credenziali. Ad esempio, vsadmin.	
`password`	Password per connettersi al cluster o alla SVM. Utilizzato per l'autenticazione basata sulle credenziali.	
`svm`	Macchina virtuale di archiviazione da utilizzare	Derivato se è specificato un managementLIF SVM.
`storagePrefix`	Prefisso utilizzato durante il provisioning di nuovi volumi nell'SVM. Non può essere modificato dopo la creazione. Per aggiornare questo parametro, sarà necessario creare un nuovo backend.	`trident`
`limitAggregateUsage`	**Non specificare per Amazon FSx for NetApp ONTAP.** Il fornito `fsxadmin` E `vsadmin` non contengono le autorizzazioni richieste per recuperare l'utilizzo aggregato e limitarlo tramite Trident.	Non utilizzare.
`limitVolumeSize`	Il provisioning non riesce se la dimensione del volume richiesto è superiore a questo valore. Limita inoltre la dimensione massima dei volumi che gestisce per qtree e LUN e `qtreesPerFlexvol` l'opzione consente di personalizzare il numero massimo di qtree per FlexVol volume	"" (non applicato di default)
`lunsPerFlexvol`	Il numero massimo di LUN per volume Flexvol deve essere compreso nell'intervallo [50, 200]. Solo SAN.	"100"

Parametro	Descrizione	Esempio
`debugTraceFlags`	Flag di debug da utilizzare durante la risoluzione dei problemi. Esempio, {"api":false, "method":true} Non utilizzare `debugTraceFlags` a meno che non si stia risolvendo un problema e si necessiti di un dump di registro dettagliato.	null
`nfsMountOptions`	Elenco separato da virgole delle opzioni di montaggio NFS. Le opzioni di montaggio per i volumi persistenti di Kubernetes sono normalmente specificate nelle classi di archiviazione, ma se non vengono specificate opzioni di montaggio in una classe di archiviazione, Trident utilizzerà le opzioni di montaggio specificate nel file di configurazione del backend di archiviazione. Se non vengono specificate opzioni di montaggio nella classe di archiviazione 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 volumi SMB.** Impostando il valore su null, i volumi NFS vengono impostati di default.	`nfs`
`qtreesPerFlexvol`	Numero massimo di Qtree per FlexVol volume, deve essere compreso nell'intervallo [50, 300]	`"200"`
`smbShare`	È possibile specificare uno dei seguenti elementi: il nome di una condivisione SMB creata tramite Microsoft Management Console o ONTAP CLI oppure un nome che consenta a Trident di creare la condivisione SMB. Questo parametro è obbligatorio per i backend Amazon FSx for ONTAP .	`smb-share`

Parametro	Descrizione	Esempio
`useREST`	Parametro booleano per utilizzare le API REST ONTAP . Quando impostato su `true` Trident utilizzerà le API REST ONTAP 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 `ontap` applicazione. Ciò è soddisfatto dal predefinito `vsadmin` E `cluster-admin` ruoli.	`false`
`aws`	È possibile specificare quanto segue nel file di configurazione per AWS FSx per ONTAP: - `fsxFilesystemID` : Specificare l'ID del file system AWS FSx. - `apiRegion` : Nome della regione API AWS. - `apikey` : Chiave API AWS. - `secretKey` : Chiave segreta AWS.	"" "" ""
`credentials`	Specificare le credenziali FSx SVM da archiviare in AWS Secrets Manager. - `name` : Amazon Resource Name (ARN) del segreto, che contiene le credenziali di SVM. - `type` : Impostato su `awsarn` . Fare riferimento a"Crea un segreto AWS Secrets Manager" per maggiori informazioni.	

Opzioni di configurazione del backend per il provisioning dei volumi

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

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

Parametro	Descrizione	Predefinito
`qosPolicy`	Gruppo di criteri QoS da assegnare ai volumi creati. Scegliere tra qosPolicy o adaptiveQosPolicy per ogni pool di archiviazione o backend. Per utilizzare i gruppi di policy QoS con Trident è necessario ONTAP 9.8 o versione successiva. È necessario utilizzare un gruppo di policy QoS non condiviso e assicurarsi che il gruppo di policy venga applicato individualmente a ciascun componente. Un gruppo di policy QoS condiviso impone il limite massimo per la produttività totale di tutti i carichi di lavoro.	""
`adaptiveQosPolicy`	Gruppo di policy QoS adattivo da assegnare ai volumi creati. Scegliere tra qosPolicy o adaptiveQosPolicy per ogni pool di archiviazione o backend. Non supportato da ontap-nas-economy.	""
`snapshotReserve`	Percentuale di volume riservata agli snapshot "0"	Se `snapshotPolicy` È `none` , `else` ""
`splitOnClone`	Dividere un clone dal suo genitore al momento della creazione	`false`
`encryption`	Abilita NetApp Volume Encryption (NVE) sul nuovo volume; l'impostazione predefinita è `false` . Per utilizzare questa opzione, NVE deve essere concesso in licenza e abilitato sul cluster. Se NAE è abilitato sul backend, qualsiasi volume fornito in Trident sarà abilitato per NAE. Per maggiori informazioni, fare riferimento a:"Come funziona Trident con NVE e NAE" .	`false`
`luksEncryption`	Abilita la crittografia LUKS. Fare riferimento a"Utilizzare Linux Unified Key Setup (LUKS)" . Solo SAN.	""
`tieringPolicy`	Criterio di tiering da utilizzare `none`	
`unixPermissions`	Modalità per nuovi volumi. **Lasciare vuoto per i volumi SMB.**	""

Parametro	Descrizione	Predefinito
`securityStyle`	Stile di sicurezza per i nuovi volumi. Supporti NFS `mixed` E `unix` stili di sicurezza. Supporti SMB `mixed` E `ntfs` stili di sicurezza.	L'impostazione predefinita di NFS è `unix` . L'impostazione predefinita di SMB è `ntfs` .

Prepararsi al provisioning dei volumi SMB

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

Prima di iniziare

Prima di poter effettuare il provisioning dei volumi SMB utilizzando `ontap-nas` conducente, devi avere quanto segue.

- Un cluster Kubernetes con un nodo controller Linux e almeno un nodo worker Windows che esegue Windows Server 2019. Trident supporta volumi SMB montati su pod in esecuzione solo su 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'
```

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

Passi

1. Crea condivisioni SMB. È possibile creare le condivisioni amministrative SMB in uno dei due modi seguenti: utilizzando"Console di gestione Microsoft" Snap-in Cartelle condivise o tramite ONTAP CLI. Per creare le condivisioni SMB utilizzando ONTAP CLI:
   a. Se necessario, creare la struttura del percorso della 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 fallisce.

   b. Crea una condivisione SMB associata all'SVM specificato:

```
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"Crea una condivisione SMB" per maggiori dettagli.

2. Durante la creazione del backend, è necessario configurare quanto segue per specificare i volumi SMB. Per tutte le opzioni di configurazione del backend FSx for ONTAP , fare riferimento a"Opzioni di configurazione ed esempi di FSx per ONTAP" .

Parametro	Descrizione	Esempio
`smbShare`	È possibile specificare uno dei seguenti elementi: il nome di una condivisione SMB creata tramite Microsoft Management Console o ONTAP CLI oppure un nome che consenta a Trident di creare la condivisione SMB. Questo parametro è obbligatorio per i backend Amazon FSx for ONTAP .	`smb-share`
`nasType`	**Deve essere impostato su `smb` .** Se nullo, il valore predefinito è `nfs` .	`smb`
`securityStyle`	Stile di sicurezza per i nuovi volumi. **Deve essere impostato su `ntfs` O `mixed` per volumi SMB.**	`ntfs`O `mixed` per volumi SMB
`unixPermissions`	Modalità per nuovi volumi. **Deve essere lasciato vuoto per i volumi SMB.**	""

Configurare una classe di archiviazione e PVC

Configurare un oggetto StorageClass di Kubernetes e creare la classe di archiviazione per indicare a Trident come effettuare il provisioning dei volumi. Creare un PersistentVolumeClaim (PVC) che utilizzi la StorageClass Kubernetes configurata per richiedere l'accesso al PV. È quindi possibile montare il fotovoltaico su un pod.

Creare una classe di archiviazione

Configurare un oggetto StorageClass di Kubernetes

IL "Oggetto StorageClass di Kubernetes" 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 effettuare il provisioning dei volumi NFSv3 su AWS Bottlerocket, aggiungere i requisiti `mountOptions` alla classe di archiviazione:

```
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
```

Fare riferimento a"Oggetti Kubernetes e Trident" per i dettagli su come le classi di archiviazione interagiscono con `PersistentVolumeClaim` e parametri per controllare il modo in cui Trident approvvigiona i volumi.

Creare una classe di archiviazione

Passi

1. Questo è un oggetto Kubernetes, quindi usa `kubectl` per crearlo in Kubernetes.

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

2. Ora dovresti vedere una classe di archiviazione **basic-csi** sia in Kubernetes che in Trident e Trident dovrebbe aver rilevato i pool sul backend.

```
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 al PersistentVolume sul cluster.

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

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

Esempi di manifesti

Manifesti di esempio 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 una StorageClass denominata `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

Passi

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
```

Fare riferimento a"Oggetti Kubernetes e Trident" per i dettagli su come le classi di archiviazione interagiscono con `PersistentVolumeClaim` e parametri per controllare il modo in cui Trident approvvigiona i volumi.

Attributi Trident

Questi parametri determinano quali pool di archiviazione gestiti da Trident devono essere utilizzati per fornire volumi di un determinato tipo.

Attributo	Tipo	Valori	Offerta	Richiesta	Supportato da
media[1]	corda	hdd, ibrido, ssd	Il pool contiene supporti di questo tipo; ibrido significa entrambi	Tipo di supporto specificato	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san
provisioningType	corda	sottile, spesso	Il pool supporta questo metodo di provisioning	Metodo di provisioning specificato	spesso: tutto ontap; sottile: tutto ontap e solidfire-san
tipo backend	corda	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san, gcp-cvs, azure-netapp-files, ontap-san-economy	Pool appartiene a questo tipo di backend	Backend specificato	Tutti i conducenti
istantanee	bool	vero, falso	Il pool supporta volumi con snapshot	Volume con snapshot abilitati	ontap-nas, ontap-san, solidfire-san, gcp-cvs
cloni	bool	vero, falso	Il pool supporta la clonazione dei volumi	Volume con cloni abilitati	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 abilitata	ontap-nas, ontap-nas-economy, ontap-nas-flexgroups, ontap-san
IOPS	interno	intero positivo	Pool è in grado di garantire IOPS in questo intervallo	Volume garantito per questi IOPS	solidfire-san

[1]: Non supportato dai sistemi ONTAP Select

Distribuisci l'applicazione di esempio

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

Passi

1. Montare il volume in un pod.

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

Questi esempi mostrano le configurazioni di base per fissare 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
```

Puoi monitorare i progressi 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
```

Ora puoi 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 semplifica la gestione dello storage Amazon FSx for NetApp ONTAP in Kubernetes, consentendo a sviluppatori e amministratori di concentrarsi sulla distribuzione delle applicazioni. Il componente aggiuntivo NetApp Trident EKS include le patch di sicurezza più recenti, le correzioni di bug ed è convalidato da AWS per funzionare con Amazon EKS. Il componente aggiuntivo EKS consente di garantire costantemente la sicurezza e la stabilità dei cluster Amazon EKS e di ridurre la quantità di lavoro necessaria per installare, configurare e aggiornare i componenti aggiuntivi.

Prerequisiti

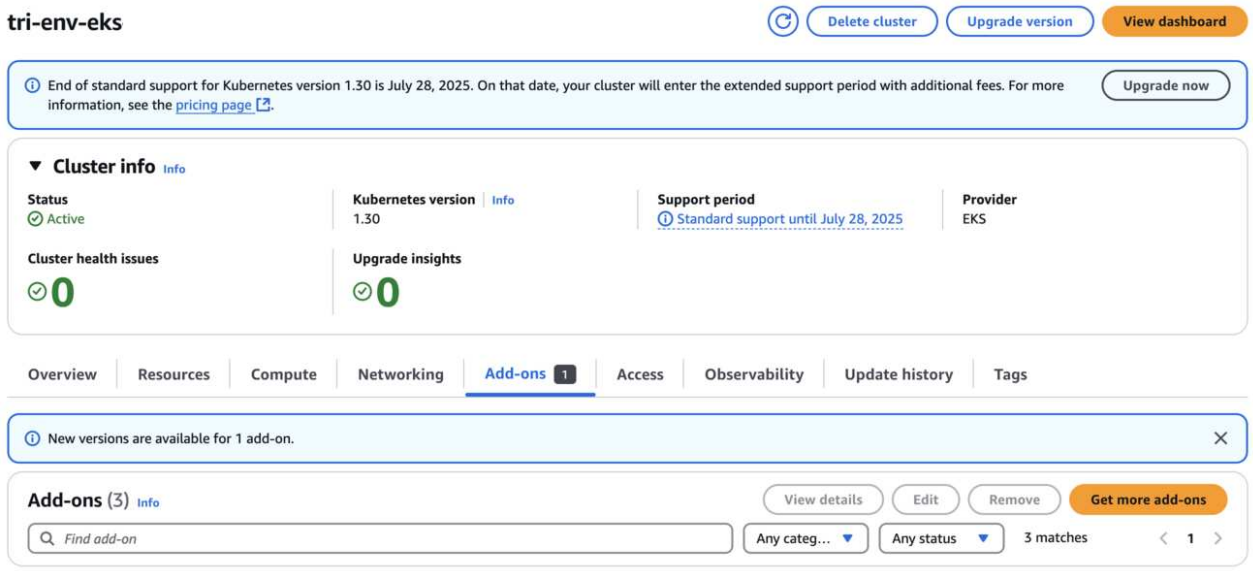
Prima di configurare il componente aggiuntivo Trident per AWS EKS, assicurati di disporre di quanto segue:

- Un account cluster Amazon EKS con autorizzazioni per lavorare con i componenti aggiuntivi. Fare riferimento a"Componenti aggiuntivi Amazon EKS" .
- Autorizzazioni AWS per il marketplace AWS:
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- Tipo 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 for NetApp ONTAP esistente

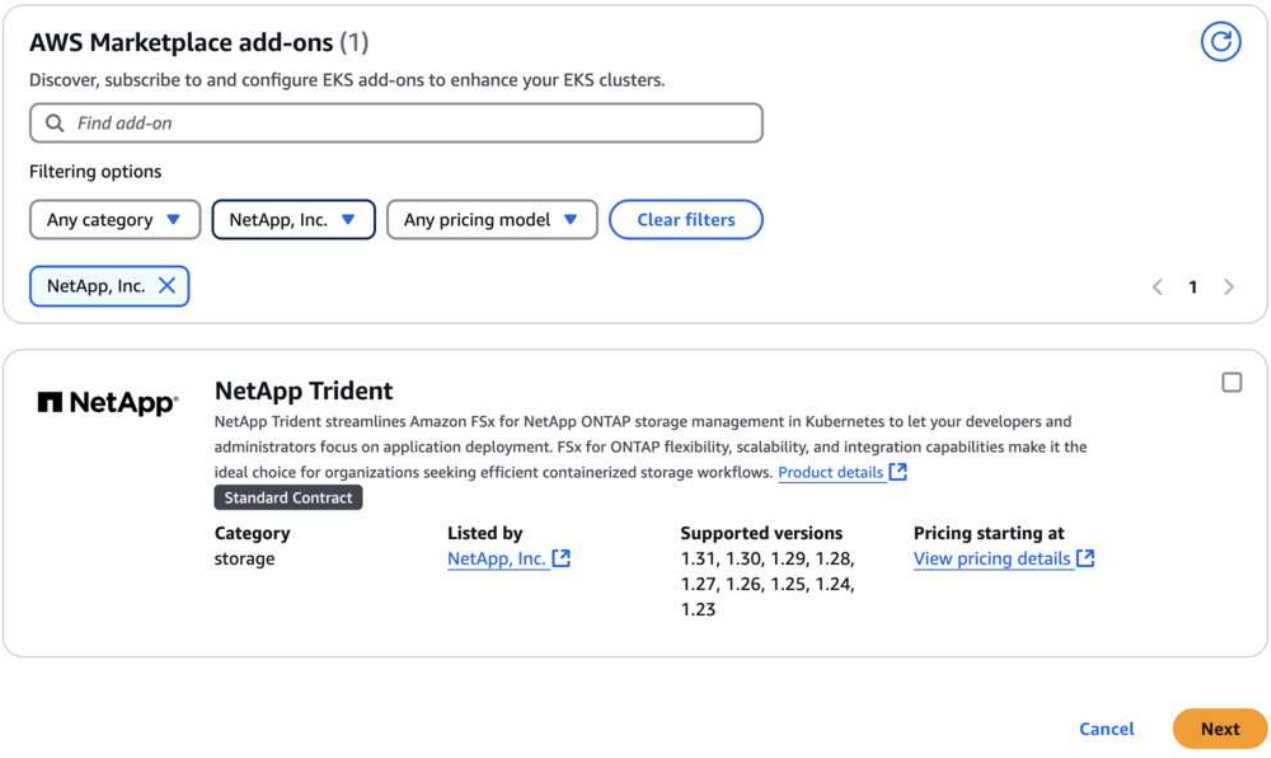
Passi

1. Assicurati di creare il ruolo IAM e il segreto AWS per consentire ai pod EKS di accedere alle risorse AWS. Per le istruzioni, vedere"Crea un ruolo IAM e un segreto AWS" .

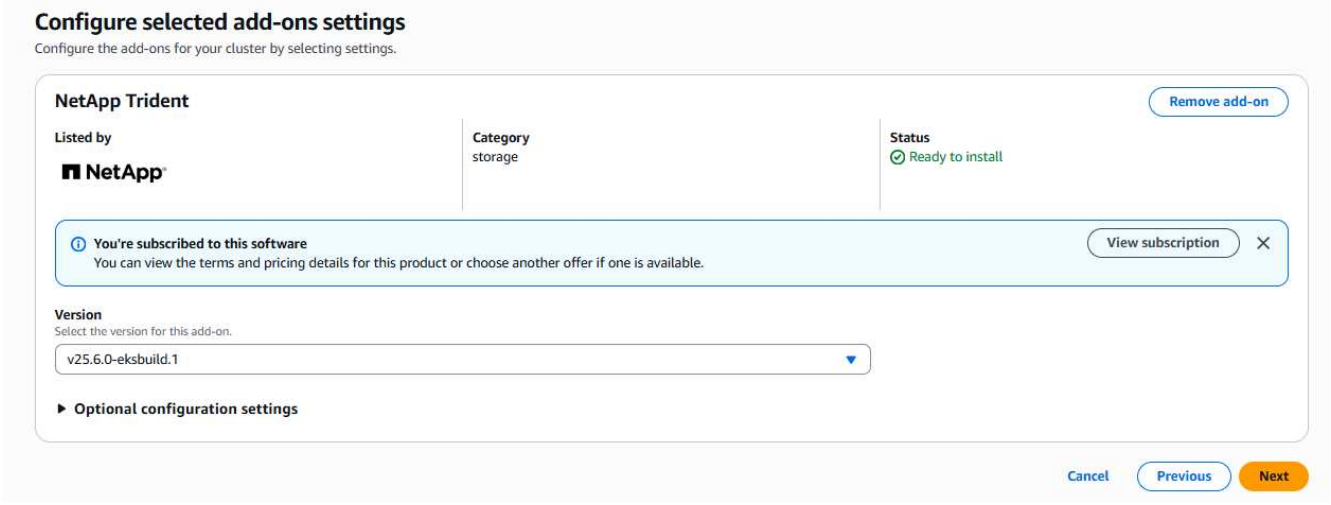
2. Nel cluster EKS Kubernetes, vai alla scheda **Componenti aggiuntivi**.



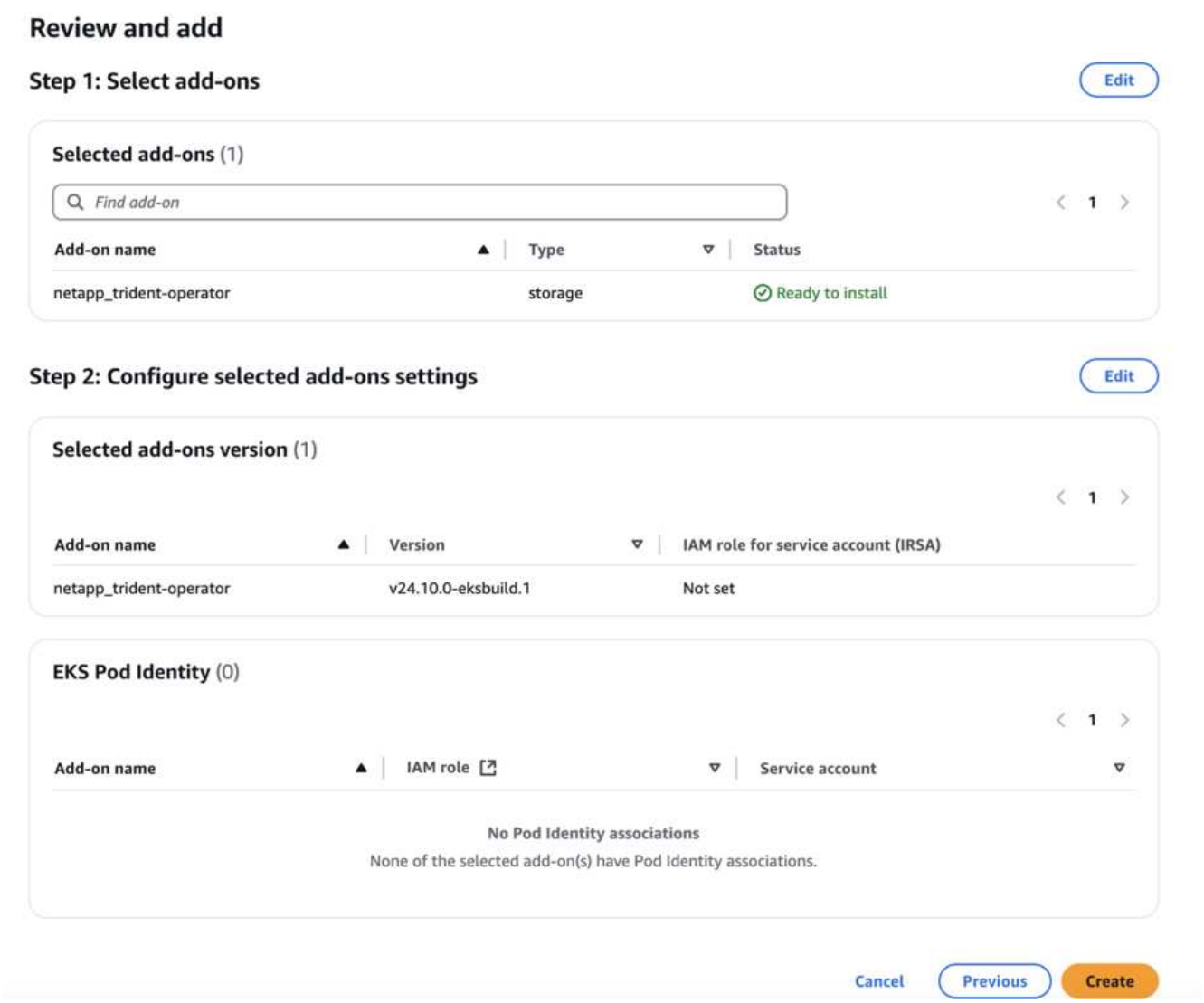
3. Vai su **Componenti aggiuntivi di AWS Marketplace** e scegli la categoria *archiviazione*.



4. Individua * NetApp Trident* e seleziona la casella di controllo per il componente aggiuntivo Trident , quindi fai clic su **Avanti**.
5. Scegli la versione desiderata del componente aggiuntivo.

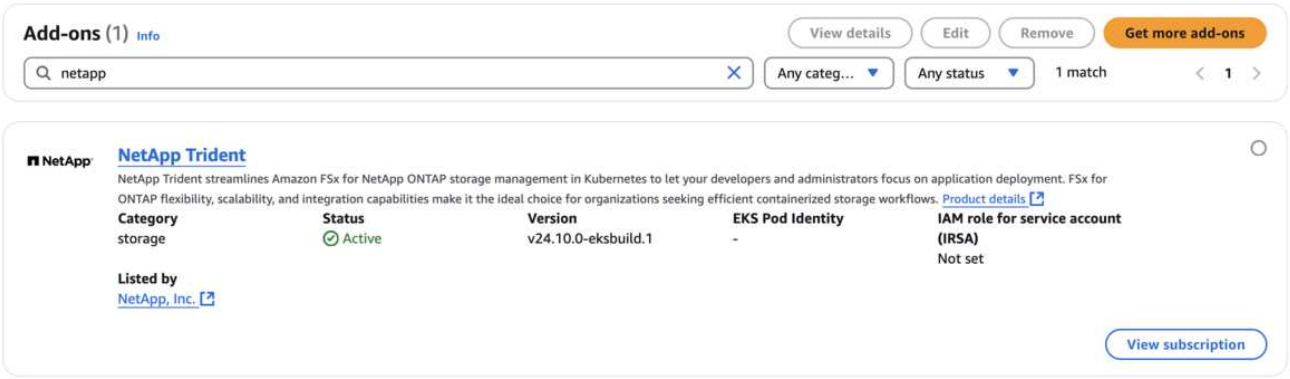


6. Configurare le impostazioni aggiuntive richieste.



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

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



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

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

11. Continuare l'installazione e configurare il backend di archiviazione. Per informazioni, vedere"Configurare il backend di archiviazione" .

Installa/disinstalla il componente aggiuntivo Trident EKS tramite CLI

Installare il componente aggiuntivo NetApp Trident EKS tramite 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 tramite 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 archiviazione. Indica a Trident come comunicare con quel sistema di archiviazione e come Trident deve effettuare il provisioning dei volumi da esso. Dopo aver installato Trident , il passo successivo è creare un backend. IL `TridentBackendConfig` La definizione di risorse personalizzate (CRD) consente di creare e gestire i backend Trident direttamente tramite l'interfaccia Kubernetes. Puoi farlo usando `kubectl` o lo strumento CLI equivalente per la tua distribuzione Kubernetes.

`TridentBackendConfig`

`TridentBackendConfig` (`tbc`, `tbconfig` , `tbackendconfig`) è un frontend, un CRD con namespace che consente di gestire i backend Trident utilizzando `kubectl` . Gli amministratori di Kubernetes e storage possono ora creare e gestire i backend direttamente tramite la CLI di Kubernetes senza richiedere un'utilità

della riga di comando dedicata(`tridentctl`).

Dopo la creazione di un `TridentBackendConfig` oggetto, accade quanto segue:

- Trident crea automaticamente un backend in base alla configurazione fornita. Questo è rappresentato internamente come un `TridentBackend` (`tbe` , `tridentbackend`) CR.
- IL `TridentBackendConfig` è unicamente legato a un `TridentBackend` che è stato creato da Trident.

Ogni `TridentBackendConfig` mantiene una mappatura uno a uno con un `TridentBackend` La prima è l'interfaccia fornita all'utente per progettare e configurare i backend; la seconda è il modo in cui Trident rappresenta l'oggetto backend effettivo.

> `TridentBackend`I CR vengono creati automaticamente da Trident. **Non dovresti** modificarli. Se vuoi apportare aggiornamenti ai backend, fallo modificando il `TridentBackendConfig` oggetto.

Vedere il seguente esempio per il formato del `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
```

Puoi anche dare un'occhiata agli esempi nel "trident-installer" directory per configurazioni di esempio per la piattaforma/servizio di archiviazione desiderato.

IL `spec` accetta parametri di configurazione specifici del backend. In questo esempio, il backend utilizza il `ontap-san` driver di archiviazione e utilizza i parametri di configurazione qui tabulati. Per l'elenco delle opzioni di configurazione per il driver di archiviazione desiderato, fare riferimento a"informazioni sulla configurazione del backend per il driver di archiviazione" .

IL `spec` la sezione include anche `credentials` E `deletionPolicy` campi, che sono stati recentemente introdotti nel `TridentBackendConfig` CR:

- `credentials`: Questo parametro è un campo obbligatorio e contiene le credenziali utilizzate per l'autenticazione con il sistema/servizio di archiviazione. Viene impostato su un segreto Kubernetes creato dall'utente. Le credenziali non possono essere trasmesse in testo normale e ciò causerebbe un errore.
- `deletionPolicy`: Questo campo definisce cosa dovrebbe accadere quando il `TridentBackendConfig` viene eliminato. Può assumere uno dei due valori possibili:

- `delete`: Ciò comporta l'eliminazione di entrambi `TridentBackendConfig` CR e il backend associato. Questo è il valore predefinito.
- `retain`: Quando un `TridentBackendConfig` CR viene eliminato, la definizione del backend sarà ancora presente e potrà essere gestita con `tridentctl` . Impostazione della politica di eliminazione su `retain` consente agli utenti di eseguire il downgrade a una versione precedente (precedente alla 21.04) e di mantenere i backend creati. Il valore di questo campo può essere aggiornato dopo un `TridentBackendConfig` è creato.

Il nome di un backend viene impostato utilizzando `spec.backendName` . Se non specificato, il nome del backend viene impostato sul nome del `TridentBackendConfig` oggetto (metadati.nome). Si consiglia di impostare esplicitamente i nomi del backend utilizzando `spec.backendName` .

Backend creati con `tridentctl` non hanno un associato `TridentBackendConfig` oggetto. Puoi scegliere di gestire tali backend con `kubectl` creando un `TridentBackendConfig` CR. Bisogna fare attenzione a specificare parametri di configurazione identici (come `spec.backendName` , `spec.storagePrefix` , `spec.storageDriverName` , e così via). Trident collegherà automaticamente il nuovo creato `TridentBackendConfig` con il backend preesistente.

Panoramica dei passaggi

Per creare un nuovo backend utilizzando `kubectl` , dovresti fare quanto segue:

1. Crea un "Segreto di Kubernetes" Il segreto contiene le credenziali di cui Trident ha bisogno per comunicare con il cluster/servizio di archiviazione.
2. Crea un `TridentBackendConfig` oggetto. Contiene informazioni specifiche sul cluster/servizio di archiviazione e fa riferimento al segreto creato nel passaggio precedente.

Dopo aver creato un backend, puoi osservarne lo stato utilizzando `kubectl get tbc <tbc-name> -n <trident-namespace>` e raccogliere ulteriori dettagli.

Passaggio 1: creare un segreto Kubernetes

Crea un segreto che contenga le credenziali di accesso per il backend. Questa caratteristica è unica per ogni servizio/piattaforma di archiviazione. Ecco un esempio:

```
kubectl -n trident create -f backend-tbc-ontap-san-secret.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 Segreto per ogni piattaforma di archiviazione:

Descrizione dei campi segreti della piattaforma di archiviazione	Segreto	Descrizione dei campi
Azure NetApp Files	ID cliente	L'ID client da una registrazione dell'app
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	chiave privata	Chiave privata. Parte della chiave API per l'account di servizio GCP con ruolo di amministratore CVS
Elemento (NetApp HCI/ SolidFire)	Punto finale	MVIP per il cluster SolidFire con credenziali tenant
ONTAP	nome utente	Nome utente per connettersi al cluster/SVM. Utilizzato per l'autenticazione basata sulle credenziali
ONTAP	password	Password per connettersi al cluster/SVM. Utilizzato per l'autenticazione basata sulle credenziali
ONTAP	chiave privata del cliente	Valore codificato in Base64 della chiave privata del client. Utilizzato per l'autenticazione basata su certificato

Descrizione dei campi segreti della piattaforma di archiviazione	Segreto	Descrizione dei campi
ONTAP	chapUsername	Nome utente in entrata. Obbligatorio se useCHAP=true. Per `ontap-san` E `ontap-san-economy`
ONTAP	chapInitiatorSecret	Segreto dell'iniziatore CHAP. Obbligatorio se useCHAP=true. Per `ontap-san` E `ontap-san-economy`
ONTAP	chapTargetUsername	Nome utente di destinazione. Obbligatorio se useCHAP=true. Per `ontap-san` E `ontap-san-economy`
ONTAP	chapTargetInitiatorSecret	Segreto dell'iniziatore del target CHAP. Obbligatorio se useCHAP=true. Per `ontap-san` E `ontap-san-economy`

Il segreto creato in questo passaggio verrà referenziato nel `spec.credentials` campo del `TridentBackendConfig` oggetto che viene creato nel passaggio successivo.

Passaggio 2: creare il `TridentBackendConfig` CR

Ora sei pronto per creare il tuo `TridentBackendConfig` CR. In questo esempio, un backend che utilizza il `ontap-san` il driver viene creato utilizzando il `TridentBackendConfig` oggetto mostrato di seguito:

```
kubectl -n trident create -f backend-tbc-ontap-san.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 del `TridentBackendConfig` CR

Ora che hai creato il `TridentBackendConfig` CR, puoi 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 con successo e associato a `TridentBackendConfig` CR.

La fase può assumere uno dei seguenti valori:

- `Bound`: IL `TridentBackendConfig` CR è associato a un backend e quel backend contiene `configRef` impostato su `TridentBackendConfig` Uid di CR.
- `Unbound`: Rappresentato utilizzando `""` . IL `TridentBackendConfig` l'oggetto non è vincolato a un backend. Tutti i nuovi creati `TridentBackendConfig` I CR si trovano in questa fase per impostazione predefinita. Dopo il cambio di fase, non è più possibile tornare allo stato Unbound.
- `Deleting`: IL `TridentBackendConfig` CR `deletionPolicy` era impostato per essere eliminato. Quando il `TridentBackendConfig` Una volta eliminato il CR, passa allo stato Eliminazione.
  - Se non esistono richieste di volume persistenti (PVC) sul backend, l'eliminazione `TridentBackendConfig` comporterà l'eliminazione del backend Trident e anche del `TridentBackendConfig` CR.
  - Se sul backend sono presenti uno o più PVC, questo passa allo stato di eliminazione. IL `TridentBackendConfig` Successivamente anche CR entra nella fase di eliminazione. Il backend e `TridentBackendConfig` vengono eliminati solo dopo che tutti i PVC sono stati eliminati.
- `Lost`: Il backend associato al `TridentBackendConfig` CR è stato eliminato accidentalmente o deliberatamente e il `TridentBackendConfig` CR ha ancora un riferimento al backend eliminato. IL `TridentBackendConfig` CR può ancora essere eliminato indipendentemente dal `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 il `tridentbackends.trident.netapp.io` Manca il CRD. Potrebbe essere necessario un intervento.

A questo punto, il backend è stato creato correttamente! Ci sono diverse operazioni che possono essere gestite ulteriormente, come ad esempio"aggiornamenti del backend ed eliminazioni del backend" .

(Facoltativo) Passaggio 4: Ottieni maggiori dettagli

Puoi eseguire il seguente comando per ottenere maggiori informazioni sul tuo 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 anche possibile ottenere un dump YAML/JSON di `TridentBackendConfig` .

```
kubectl -n trident get tbc backend-tbc-ontap-san -o 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 che è stato creato in risposta al `TridentBackendConfig` CR. IL `lastOperationStatus` campo rappresenta lo stato dell'ultima operazione del `TridentBackendConfig` CR, che può essere attivato dall'utente (ad esempio, l'utente ha modificato qualcosa in `spec`) o attivato da Trident (ad esempio, durante i riavvii Trident). Può essere un successo o un fallimento. `phase` rappresenta lo stato della relazione tra `TridentBackendConfig` CR e backend. Nell'esempio sopra, `phase` ha il valore Bound, il che significa che il `TridentBackendConfig` CR è associato al backend.

Puoi eseguire il `kubectl -n trident describe tbc <tbc-cr-name>` comando per ottenere i dettagli dei registri eventi.

Non è possibile aggiornare o eliminare un backend che contiene un associato `TridentBackendConfig` oggetto utilizzando `tridentctl` . Per comprendere i passaggi coinvolti nel passaggio tra `tridentctl` E `TridentBackendConfig` ,"vedi qui" .

Gestire i backend

Eseguire la gestione del backend con kubectl

Scopri come eseguire operazioni di gestione del backend utilizzando `kubectl` .

Elimina un backend

Eliminando un `TridentBackendConfig` , istruisci Trident a eliminare/conservare i backend (in base a `deletionPolicy`). Per eliminare un backend, assicurati che `deletionPolicy` è impostato per l'eliminazione. Per eliminare solo il `TridentBackendConfig` , assicurarsi che `deletionPolicy` è impostato per mantenere. Ciò garantisce che il backend sia ancora presente e possa essere gestito utilizzando `tridentctl` .

Esegui il seguente comando:

```
kubectl delete tbc <tbc-name> -n trident
```

Trident non elimina i segreti di Kubernetes che erano in uso da `TridentBackendConfig` . L'utente Kubernetes è responsabile della pulizia dei segreti. Bisogna fare attenzione quando si eliminano i segreti. Dovresti eliminare i segreti solo se non sono utilizzati dai backend.

Visualizza i backend esistenti

Esegui il seguente comando:

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

Puoi anche correre `tridentctl get backend -n trident` O `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` .

Aggiorna un backend

Possono esserci molteplici motivi per aggiornare un backend:

- Le credenziali per il sistema di archiviazione sono cambiate. Per aggiornare le credenziali, il segreto Kubernetes utilizzato in `TridentBackendConfig` l'oggetto deve essere aggiornato. Trident aggiornerà automaticamente il backend con le credenziali più recenti fornite. Eseguire il seguente comando per aggiornare il segreto di Kubernetes:

  ```
  kubectl apply -f <updated-secret-file.yaml> -n trident
  ```

- È necessario aggiornare i parametri (ad esempio il nome ONTAP SVM utilizzato).
  - Puoi aggiornare `TridentBackendConfig` oggetti direttamente tramite Kubernetes utilizzando il seguente comando:

```
kubectl apply -f <updated-backend-file.yaml>
```

- In alternativa, è possibile apportare modifiche a quelle esistenti `TridentBackendConfig` CR utilizzando il seguente comando:

```
kubectl edit tbc <tbc-name> -n trident
```

- Se un aggiornamento del backend fallisce, il backend continua a mantenere la sua ultima configurazione nota. È possibile visualizzare i registri per determinare la causa eseguendo `kubectl get tbc <tbc-name> -o yaml -n trident` O `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 di aggiornamento.

Eseguire la gestione del backend con tridentctl

Scopri come eseguire operazioni di gestione del backend utilizzando `tridentctl` .

Crea un backend

Dopo aver creato un"file di configurazione del backend" , eseguire il seguente comando:

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

Se la creazione del backend fallisce, significa che c'è stato un errore nella configurazione del backend. È possibile visualizzare i registri per determinare la causa eseguendo il seguente comando:

```
tridentctl logs -n trident
```

Dopo aver identificato e corretto il problema con il file di configurazione, puoi semplicemente eseguire il comando `create` comando di nuovo.

Elimina un backend

Per eliminare un backend da Trident, procedere come segue:

1. Recupera il nome del backend:

```
tridentctl get backend -n trident
```

2. Elimina il backend:

```
tridentctl delete backend <backend-name> -n trident
```

Se Trident ha eseguito il provisioning di volumi e snapshot da questo backend che esistono ancora, l'eliminazione del backend impedisce che vengano eseguiti il provisioning di nuovi volumi. Il backend continuerà a esistere nello stato "Eliminazione".

Visualizza i backend esistenti

Per visualizzare i backend noti a Trident , 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
```

Aggiorna un backend

Dopo aver creato un nuovo file di configurazione backend, eseguire il seguente comando:

```
tridentctl update backend <backend-name> -f <backend-file> -n trident
```

Se l'aggiornamento del backend fallisce, c'è stato un problema con la configurazione del backend oppure hai tentato un aggiornamento non valido. È possibile visualizzare i registri per determinare la causa eseguendo il seguente comando:

```
tridentctl logs -n trident
```

Dopo aver identificato e corretto il problema con il file di configurazione, puoi semplicemente eseguire il comando `update` comando di nuovo.

Identificare le classi di archiviazione 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. Questo utilizza il `jq` utilità che devi installare.

```
tridentctl get backend -o json | jq '[.items[] | {backend: .name,
storageClasses: [.storage[].storageClasses]|unique}]'
```

Questo vale anche per i backend creati utilizzando `TridentBackendConfig` .

Spostarsi tra le opzioni di gestione del backend

Scopri i diversi modi di gestire i backend in Trident.

Opzioni per la gestione dei backend

Con l'introduzione di `TridentBackendConfig` , gli amministratori ora hanno due modi unici per gestire i backend. Ciò solleva le seguenti domande:

- I backend possono essere creati utilizzando `tridentctl` essere gestito con `TridentBackendConfig` ?
- I backend possono essere creati utilizzando `TridentBackendConfig` essere gestito utilizzando `tridentctl` ?

Maneggio `tridentctl` backend utilizzando `TridentBackendConfig`

Questa sezione illustra i passaggi necessari per gestire i backend creati utilizzando `tridentctl` direttamente tramite l'interfaccia Kubernetes creando `TridentBackendConfig` oggetti.

Ciò si applicherà ai seguenti scenari:

- Backend preesistenti, che non hanno un `TridentBackendConfig` perché sono stati creati con `tridentctl` .
- Nuovi backend creati con `tridentctl` , mentre altri `TridentBackendConfig` gli oggetti esistono.

In entrambi gli scenari, i backend continueranno a essere presenti, con Trident che pianifica i volumi e opera su di essi. In questo caso gli amministratori hanno due possibilità:

- Continua a usare `tridentctl` per gestire i backend creati utilizzandolo.
- Associa i backend creati utilizzando `tridentctl` a un nuovo `TridentBackendConfig` oggetto. Ciò significherebbe che i backend saranno gestiti utilizzando `kubectl` e non `tridentctl` .

Per gestire un backend preesistente utilizzando `kubectl` , dovrai creare un `TridentBackendConfig` che si collega al backend esistente. Ecco una panoramica di come funziona:

1. Crea un segreto Kubernetes. Il segreto contiene le credenziali di cui Trident ha bisogno per comunicare con il cluster/servizio di archiviazione.
2. Crea un `TridentBackendConfig` oggetto. Contiene informazioni specifiche sul cluster/servizio di archiviazione e fa riferimento al segreto creato nel passaggio precedente. Bisogna fare attenzione a specificare parametri di configurazione identici (come `spec.backendName` , `spec.storagePrefix` , `spec.storageDriverName` , e così via). `spec.backendName` deve essere impostato sul nome del backend esistente.

Passaggio 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"
      }
    }
  ]
}
```

Passaggio 1: creare un segreto Kubernetes

Crea un segreto che contenga le credenziali per il backend, come mostrato in questo esempio:

```
cat tbc-ontap-nas-backend-secret.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
```

Passaggio 2: creare un `TridentBackendConfig` CR

Il passo successivo è creare un `TridentBackendConfig` CR che si collegherà automaticamente al preesistente `ontap-nas-backend` (come in questo esempio). Assicurarsi che siano soddisfatti i seguenti requisiti:

- Lo stesso nome del backend è 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 tramite un segreto Kubernetes e non in testo normale.

In questo caso, il `TridentBackendConfig` sarà simile a questo:

```
cat backend-tbc-ontap-nas.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 del `TridentBackendConfig` CR

Dopo il `TridentBackendConfig` è stato creato, la sua fase deve essere `Bound` . Dovrebbe inoltre riflettere lo stesso nome backend 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 backend sarà ora completamente gestito tramite `tbc-ontap-nas-backend` `TridentBackendConfig` oggetto.

Maneggio `TridentBackendConfig` backend utilizzando `tridentctl`

```
`tridentctl`può essere utilizzato per elencare i backend creati
utilizzando `TridentBackendConfig` .  Inoltre, gli amministratori possono
anche scegliere di gestire completamente tali backend tramite `tridentctl`
eliminando `TridentBackendConfig` e assicurandosi `spec.deletionPolicy` è
impostato su `retain` .
```

Passaggio 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].

Passaggio 1: conferma `deletionPolicy` **è impostato su** `retain`

Diamo un'occhiata al valore di `deletionPolicy` . Questo deve essere impostato su `retain` . Ciò garantisce che quando un `TridentBackendConfig` CR viene eliminato, la definizione del 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 procedere al passaggio successivo a meno che `deletionPolicy` è impostato su `retain` .

Passaggio 2: Eliminare il `TridentBackendConfig` CR

Il passaggio finale è quello di eliminare il `TridentBackendConfig` CR. Dopo aver confermato il `deletionPolicy` è impostato su `retain` , puoi 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 |
+------------------+---------------
+--------------------------------------+--------+---------+
```

Dopo la cancellazione del `TridentBackendConfig` oggetto, Trident lo rimuove semplicemente senza eliminare effettivamente il backend stesso.

Creare e gestire classi di archiviazione

Creare una classe di archiviazione

Configurare un oggetto StorageClass di Kubernetes e creare la classe di archiviazione per indicare a Trident come effettuare il provisioning dei volumi.

Configurare un oggetto StorageClass di Kubernetes

IL "Oggetto StorageClass di Kubernetes" identifica Trident come il provisioner utilizzato per quella classe e indica a Trident come effettuare il provisioning di un volume. Per esempio:

```
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
```

Fare riferimento a"Oggetti Kubernetes e Trident" per i dettagli su come le classi di archiviazione interagiscono con `PersistentVolumeClaim` e parametri per controllare il modo in cui Trident approvvigiona i volumi.

Creare una classe di archiviazione

Dopo aver creato l'oggetto StorageClass, è possibile creare la classe di archiviazione. Campioni di classe di archiviazione fornisce alcuni esempi di base che puoi utilizzare o modificare.

Passi

1. Questo è un oggetto Kubernetes, quindi usa `kubectl` per crearlo in Kubernetes.

```
kubectl create -f sample-input/storage-class-basic-csi.yaml
```

2. Ora dovresti vedere una classe di archiviazione **basic-csi** sia in Kubernetes che in Trident e Trident dovrebbe aver rilevato i pool sul backend.

```
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 archiviazione

Trident fornisce "definizioni di classi di archiviazione semplici per backend specifici" .

In alternativa, puoi modificare `sample-input/storage-class-csi.yaml.templ` file fornito con il programma di installazione e sostituisci *`BACKEND_TYPE`* con il nome del driver di archiviazione.

```
./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 archiviazione

È possibile visualizzare le classi di archiviazione esistenti, impostare una classe di archiviazione predefinita, identificare il backend della classe di archiviazione ed eliminare le classi di archiviazione.

Visualizza le classi di archiviazione esistenti

- Per visualizzare le classi di archiviazione Kubernetes esistenti, eseguire il seguente comando:

```
kubectl get storageclass
```

- Per visualizzare i dettagli della classe di archiviazione Kubernetes, eseguire il seguente comando:

```
kubectl get storageclass <storage-class> -o json
```

- Per visualizzare le classi di archiviazione sincronizzate di Trident, eseguire il seguente comando:

```
tridentctl get storageclass
```

- Per visualizzare i dettagli della classe di archiviazione sincronizzata di Trident, eseguire il seguente comando:

```
tridentctl get storageclass <storage-class> -o json
```

Imposta una classe di archiviazione predefinita

Kubernetes 1.6 ha aggiunto la possibilità di impostare una classe di archiviazione predefinita. Questa è la classe di archiviazione che verrà utilizzata per fornire un volume persistente se un utente non ne specifica uno in una richiesta di volume persistente (PVC).

- Definisci una classe di archiviazione predefinita impostando l'annotazione `storageclass.kubernetes.io/is-default-class` su true nella definizione della classe di archiviazione. Secondo la specifica, qualsiasi altro valore o l'assenza dell'annotazione viene interpretato come falso.
- È possibile configurare una classe di archiviazione esistente come classe di archiviazione 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 della classe di archiviazione predefinita utilizzando il seguente comando:

```
kubectl patch storageclass <storage-class-name> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"false"}}}'
```

Ci sono anche esempi nel pacchetto di installazione Trident che includono questa annotazione.

> Nel cluster dovrebbe essere presente una sola classe di archiviazione predefinita alla volta. Tecnicamente Kubernetes non impedisce di averne più di una, ma si comporterà come se non ci fosse alcuna classe di archiviazione predefinita.

Identificare il backend per una classe di archiviazione

Questo è un esempio del tipo di domande a cui puoi rispondere con il JSON che `tridentctl` output per gli oggetti backend Trident . Questo utilizza il `jq` utilità, che potrebbe essere necessario installare prima.

```
tridentctl get storageclass -o json | jq  '[.items[] | {storageClass:
.Config.name, backends: [.storage]|unique}]'
```

Elimina una classe di archiviazione

Per eliminare una classe di archiviazione da Kubernetes, esegui il seguente comando:

```
kubectl delete storageclass <storage-class>
```

`<storage-class>`dovrebbe essere sostituito con la tua classe di archiviazione.

Tutti i volumi persistenti creati tramite questa classe di archiviazione rimarranno intatti e Trident continuerà a gestirli.

> Trident impone un vuoto `fsType` per i volumi che crea. Per i backend iSCSI, si consiglia di applicare `parameters.fsType` nella StorageClass. Dovresti eliminare le StorageClass esistenti e ricrearle con `parameters.fsType` specificato.

Fornire e gestire i volumi

Fornire un volume

Creare un PersistentVolumeClaim (PVC) che utilizzi la StorageClass Kubernetes configurata per richiedere l'accesso al PV. È quindi possibile montare il fotovoltaico su un pod.

Panoramica

UN "*PersistentVolumeClaim*" (PVC) è una richiesta di accesso al PersistentVolume sul cluster.

Il PVC può essere configurato per richiedere una determinata dimensione di archiviazione o modalità di accesso. Utilizzando la StorageClass associata, l'amministratore del cluster può controllare molto più della dimensione e della modalità di accesso di PersistentVolume, 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

Passi

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
```

Puoi monitorare i progressi 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. Ora puoi eliminare il Pod. L'applicazione Pod non esisterà più, ma il volume rimarrà.

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

Esempi di manifesti

Manifesti di esempio 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 una StorageClass denominata `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 di manifesti del pod

Questi esempi mostrano le configurazioni di base per fissare 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
```

Fare riferimento a"Oggetti Kubernetes e Trident" per i dettagli su come le classi di archiviazione interagiscono con `PersistentVolumeClaim` e parametri per controllare il modo in cui Trident approvvigiona i volumi.

Espandi i volumi

Trident offre agli utenti di Kubernetes la possibilità di espandere i propri volumi dopo averli creati. Trova informazioni sulle configurazioni necessarie per espandere i volumi iSCSI, NFS, SMB, NVMe/TCP e FC.

Espandere un volume iSCSI

È possibile espandere un volume persistente iSCSI (PV) utilizzando il provisioner 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.

Passaggio 1: configurare StorageClass per supportare l'espansione del volume

Modificare la definizione StorageClass per impostare `allowVolumeExpansion` campo a `true` .

```
cat storageclass-ontapsan.yaml
```

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-san
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
allowVolumeExpansion: True
```

Per una StorageClass già esistente, modificarla per includere `allowVolumeExpansion` parametro.

Passaggio 2: creare un PVC con la StorageClass creata

Modifica la definizione PVC e aggiornala `spec.resources.requests.storage` per riflettere la nuova dimensione desiderata, che deve essere maggiore della dimensione originale.

```
cat pvc-ontapsan.yaml
```

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

Trident crea un Persistent Volume (PV) e lo associa a questo Persistent Volume Claim (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 collega il PVC

Collega il PV a un pod per ridimensionarlo. Esistono due scenari quando si ridimensiona un PV iSCSI:

- Se il PV è collegato a un pod, Trident espande il volume sul backend di archiviazione, 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 di archiviazione. Dopo che il PVC è stato associato a un pod, Trident esegue una nuova scansione del dispositivo e ridimensiona il file system. Kubernetes aggiorna quindi la dimensione del PVC una volta completata correttamente l'operazione di espansione.

In questo esempio, viene creato un pod che utilizza il `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: Espandi il PV

Per ridimensionare il PV creato da 1Gi a 2Gi, modificare la definizione PVC e aggiornare `spec.resources.requests.storage` a 2Gi.

```
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 verificare che l'espansione abbia funzionato correttamente 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    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

Espandi un volume FC

È possibile espandere un FC Persistent Volume (PV) utilizzando il provisioner CSI.

> L'espansione del volume FC è supportata da `ontap-san` driver e richiede Kubernetes 1.16 e versioni successive.

Passaggio 1: configurare StorageClass per supportare l'espansione del volume

Modificare la definizione StorageClass per impostare `allowVolumeExpansion` campo a `true` .

```
cat storageclass-ontapsan.yaml
```

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-san
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
allowVolumeExpansion: True
```

Per una StorageClass già esistente, modificarla per includere `allowVolumeExpansion` parametro.

Passaggio 2: creare un PVC con la StorageClass creata

Modifica la definizione PVC e aggiornala `spec.resources.requests.storage` per riflettere la nuova dimensione desiderata, che deve essere maggiore della dimensione originale.

```
cat pvc-ontapsan.yaml
```

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

Trident crea un Persistent Volume (PV) e lo associa a questo Persistent Volume Claim (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 collega il PVC

Collega il PV a un pod per ridimensionarlo. Esistono due scenari quando si ridimensiona un FC PV:

- Se il PV è collegato a un pod, Trident espande il volume sul backend di archiviazione, 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 di archiviazione. Dopo che il PVC è stato associato a un pod, Trident esegue una nuova scansione del dispositivo e ridimensiona il file system. Kubernetes aggiorna quindi la dimensione del PVC una volta completata correttamente l'operazione di espansione.

In questo esempio, viene creato un pod che utilizza il `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: Espandi il PV

Per ridimensionare il PV creato da 1Gi a 2Gi, modificare la definizione PVC e aggiornare `spec.resources.requests.storage` a 2Gi.

```
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 verificare che l'espansione abbia funzionato correttamente 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 PV NFS forniti su `ontap-nas` , `ontap-nas-economy` , `ontap-nas-flexgroup` , `gcp-cvs` , E `azure-netapp-files` backend.

Passaggio 1: configurare StorageClass per supportare l'espansione del volume

Per ridimensionare un PV NFS, l'amministratore deve prima configurare la classe di archiviazione per consentire l'espansione del volume impostando `allowVolumeExpansion` campo a `true` :

```
cat storageclass-ontapnas.yaml
```

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontapnas
provisioner: csi.trident.netapp.io
parameters:
  backendType: ontap-nas
allowVolumeExpansion: true
```

Se hai già creato una classe di archiviazione senza questa opzione, puoi semplicemente modificare la classe

di archiviazione esistente utilizzando `kubectl edit storageclass` per consentire l'espansione del volume.

Passaggio 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: Espandi 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 verificare che il ridimensionamento abbia funzionato correttamente controllando le dimensioni del PVC, del 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    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

Volumi di importazione

È possibile importare volumi di archiviazione esistenti come PV Kubernetes utilizzando `tridentctl import`.

Panoramica e considerazioni

È possibile importare un volume in Trident per:

- Containerizzare un'applicazione e riutilizzare il suo set di dati esistente
- Utilizzare un clone di un set di dati per un'applicazione effimera
- Ricostruisci un cluster Kubernetes non riuscito
- Migrare i dati dell'applicazione durante il ripristino di emergenza

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 (protezione dati) sono volumi di destinazione SnapMirror . Prima di importare il volume in Trident, è necessario interrompere la relazione mirror.

- Suggeriamo di importare volumi senza connessioni attive. Per importare un volume utilizzato attivamente, clonare il volume e quindi eseguire l'importazione.

  Ciò è particolarmente importante per i volumi a blocchi, poiché 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.

- Anche se `StorageClass` deve essere specificato su un PVC, Trident non utilizza questo parametro durante l'importazione. Le classi di archiviazione vengono utilizzate durante la creazione del volume per selezionare tra i pool disponibili in base alle caratteristiche di archiviazione. Poiché il volume esiste già, non è richiesta alcuna selezione del pool durante l'importazione. Pertanto, l'importazione non fallirà anche se il volume esiste su un backend o pool che non corrisponde alla classe di archiviazione specificata nel PVC.
- La dimensione del volume esistente viene determinata e impostata nel PVC. Dopo che il volume è stato importato dal driver di archiviazione, il PV viene creato con un ClaimRef al PVC.
  - La politica di recupero è inizialmente impostata su `retain` nel PV. Dopo che Kubernetes ha associato correttamente PVC e PV, la policy di recupero viene aggiornata per corrispondere alla policy di recupero della classe di archiviazione.
  - Se la politica di recupero della classe di archiviazione è `delete` , il volume di archiviazione verrà eliminato quando il PV verrà eliminato.
- Per impostazione predefinita, Trident gestisce il PVC e rinomina il FlexVol volume e il LUN sul backend. Puoi passare il `--no-manage` flag per importare un volume non gestito. Se usi `--no-manage` , Trident non esegue alcuna operazione aggiuntiva sul PVC o sul PV per l'intero ciclo di vita degli oggetti. Il volume di archiviazione non viene eliminato quando si elimina il PV e anche altre operazioni come la clonazione del volume e il ridimensionamento del volume vengono ignorate.

  Questa opzione è utile se si desidera utilizzare Kubernetes per carichi di lavoro containerizzati ma si desidera gestire il ciclo di vita del volume di archiviazione al di fuori di Kubernetes.

- Viene aggiunta un'annotazione al PVC e al PV che ha il duplice scopo di indicare che il volume è stato importato e se il PVC e il PV sono gestiti. Questa annotazione non deve essere modificata o rimossa.

Importa un volume

Puoi usare `tridentctl import` per importare un volume.

Passi

1. Creare il file Persistent Volume Claim (PVC) (ad esempio, `pvc.yaml`) che verrà utilizzato per creare il PVC. Il file PVC dovrebbe includere `name` , `namespace` , `accessModes` , E `storageClassName` . Facoltativamente, puoi specificare `unixPermissions` nella tua definizione di PVC.

   Di seguito è 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 la dimensione del volume. Ciò può causare il fallimento 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 nello storage (ad esempio: ONTAP FlexVol, Element Volume, percorso Cloud Volumes Service). IL `-f` L'argomento è necessario per specificare il percorso del file PVC.

```
tridentctl import volume <backendName> <volumeName> -f <path-to-pvc-
file>
```

Esempi

Esaminare i seguenti esempi di importazione di volumi per i driver supportati.

ONTAP NAS e ONTAP NAS FlexGroup

Trident supporta l'importazione di volumi utilizzando `ontap-nas` E `ontap-nas-flexgroup` conducenti.

- Trident non supporta l'importazione di volumi utilizzando `ontap-nas-economy` autista.
- IL `ontap-nas` E `ontap-nas-flexgroup` i driver non consentono nomi di volume duplicati.

Ogni volume creato con il `ontap-nas` il driver è un FlexVol volume sul cluster ONTAP . Importazione di volumi FlexVol con `ontap-nas` il driver funziona allo stesso modo. Un volume FlexVol già esistente su un cluster ONTAP può essere importato come `ontap-nas` PVC. Allo stesso modo, i volumi FlexGroup possono essere importati come `ontap-nas-flexgroup` PVC.

Esempi di ONTAP NAS

Di seguito è riportato un esempio di importazione di un volume gestito e di un volume non gestito.

Volume gestito

L'esempio seguente importa 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 il `--no-manage` argomento, Trident non rinomina il volume.

Il seguente esempio importa `unmanaged_volume` sul `ontap_nas` backend:

```
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 di volumi utilizzando `ontap-san` (iSCSI, NVMe/TCP e FC) e `ontap-san-economy` conducenti.

Trident può importare volumi ONTAP SAN FlexVol che contengono un singolo LUN. Ciò è coerente con il `ontap-san` driver, che crea un FlexVol volume per ogni PVC e un LUN all'interno del FlexVol volume. Trident importa il FlexVol volume e lo associa alla definizione PVC. Trident può importare `ontap-san-economy`

volumi che contengono più LUN.

Esempi ONTAP SAN

Di seguito è riportato un esempio di importazione di un volume gestito e di un volume non gestito.

Volume gestito

Per i volumi gestiti, Trident rinomina il FlexVol volume in `pvc-<uuid>` formato e LUN all'interno del FlexVol volume per `lun0` .

L'esempio seguente importa 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

Il seguente esempio importa `unmanaged_example_volume` sul `ontap_san` backend:

```
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 hai LUNS mappati su igroup che condividono un IQN con un IQN del nodo Kubernetes, come mostrato nell'esempio seguente, riceverai l'errore: `LUN already mapped to initiator(s) in this group` .

Per importare il volume sarà 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 `solidfire-san` autista.

> Il driver Element supporta nomi di volume 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

L'esempio seguente importa un `element-managed` volume sul backend `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 `gcp-cvs` autista.

> Per importare un volume supportato dal servizio NetApp Cloud Volumes Service in Google Cloud Platform, identifica il volume tramite il suo percorso. Il percorso del volume è la parte del percorso di esportazione del volume dopo il `:/` . 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 Google Cloud Platform

L'esempio seguente importa un `gcp-cvs` volume sul backend `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 `azure-netapp-files` autista.

> Per importare un volume Azure NetApp Files , identifica il volume tramite il suo percorso. Il percorso del volume è la parte del percorso di esportazione del volume dopo il `:/` . Ad esempio, se il percorso di montaggio è `10.0.0.2:/importvol1` , il percorso del volume è `importvol1` .

Esempio Azure NetApp Files

L'esempio seguente importa un `azure-netapp-files` volume sul backend `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 `google-cloud-netapp-volumes` autista.

Esempio Google Cloud NetApp Volumes

L'esempio seguente importa 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
|
+-----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

L'esempio seguente importa un `google-cloud-netapp-volumes` volume quando due volumi sono presenti nella stessa regione:

```
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
|
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

Personalizza i nomi e le etichette dei volumi

Con Trident puoi assegnare nomi ed etichette significativi ai volumi che crei. Ciò consente di identificare e mappare facilmente i volumi alle rispettive risorse Kubernetes (PVC). È anche possibile definire modelli a livello di backend per creare nomi di volume ed etichette personalizzate; tutti i volumi creati, importati o clonati aderiranno ai modelli.

Prima di iniziare

Supporto per nomi e etichette di volume personalizzabili:

1. Operazioni di creazione, importazione e clonazione del volume.
2. Nel caso del driver ontap-nas-economy, solo il nome del volume Qtree è conforme al modello di nome.
3. Nel caso del driver ontap-san-economy, solo il nome LUN è conforme al modello di nome.

Limitazioni

1. I nomi dei volumi personalizzabili sono compatibili solo con i driver ONTAP on-premises.
2. I nomi dei volumi personalizzabili non si applicano ai volumi esistenti.

Comportamenti chiave dei nomi dei volumi personalizzabili

1. Se si verifica un errore a causa di una sintassi non valida in un modello di nome, la creazione del backend fallisce. Tuttavia, se l'applicazione del modello fallisce, il volume verrà denominato in base alla convenzione di denominazione esistente.
2. Il prefisso di archiviazione non è applicabile quando un volume viene denominato utilizzando un modello di nome dalla configurazione backend. È possibile aggiungere direttamente al modello qualsiasi valore di

prefisso desiderato.

Esempi di configurazione del backend con modello di nome ed etichette

È possibile definire modelli di nomi personalizzati a livello di radice e/o di pool.

Esempio di livello radice

```
{
  "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 della piscina

```
{
  "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 nomi

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 ha etichette in un formato specifico. Per esempio: `{"provisioning":{"Cluster":"ClusterA", "PVC": "pvcname"}}`.
2. Nel caso di importazioni di volumi gestiti, il nome del volume segue il modello di nome definito a livello radice nella definizione del backend.
3. Trident non supporta l'uso di un operatore slice con il prefisso storage.
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, Trident nominerà i volumi in base alla convenzione di denominazione esistente. Per tutti gli altri driver ONTAP , se il nome del volume supera il limite, il processo di creazione del volume fallisce.

Condividere un volume NFS tra gli spazi dei nomi

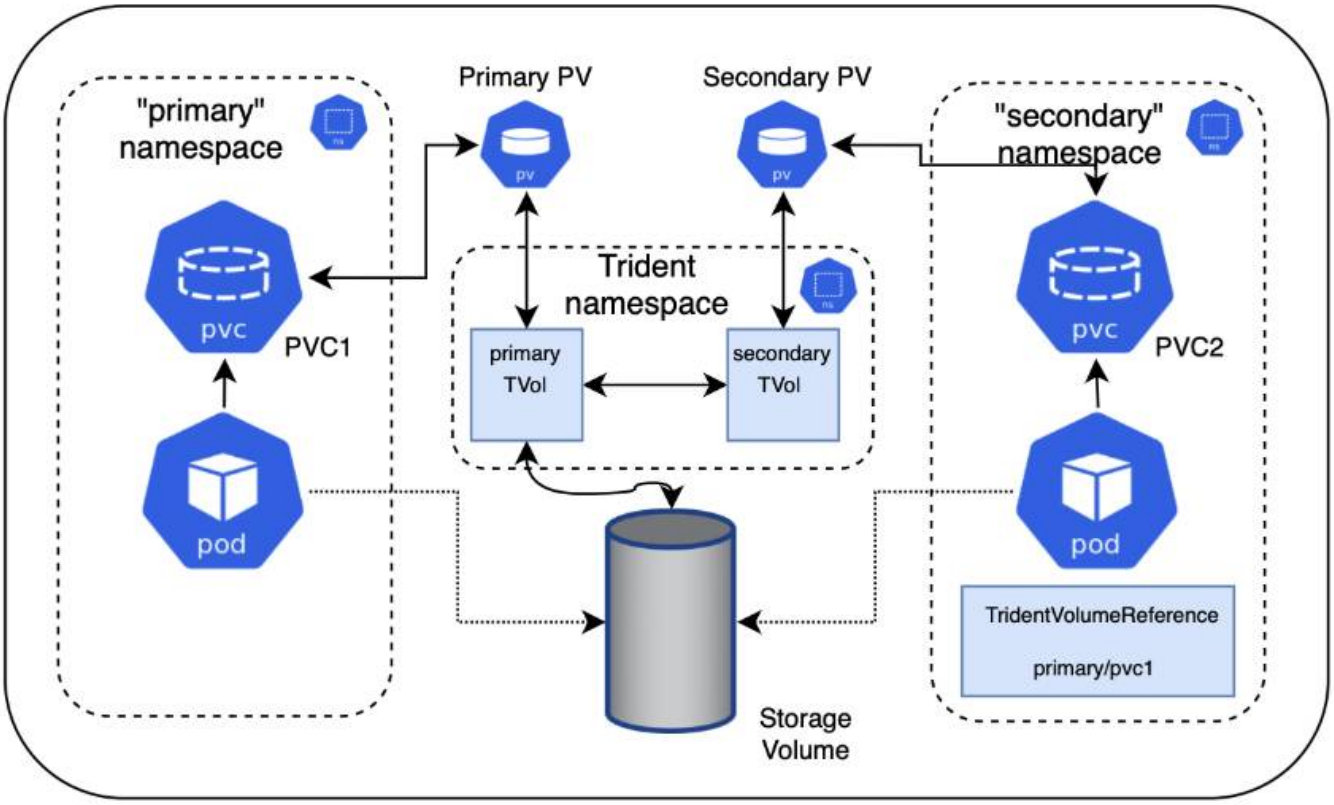
Utilizzando Trident, è possibile creare un volume in uno spazio dei nomi primario e condividerlo in uno o più spazi dei nomi secondari.

Caratteristiche

TridentVolumeReference CR consente di condividere in modo sicuro volumi NFS ReadWriteMany (RWX) su uno o più namespace Kubernetes. Questa soluzione nativa di Kubernetes offre i seguenti vantaggi:

- Più livelli di controllo degli accessi per garantire la sicurezza
- Funziona con tutti i driver di volume Trident NFS
- Nessuna dipendenza da tridentctl o da qualsiasi altra funzionalità non nativa di Kubernetes

Questo diagramma illustra la condivisione del volume NFS tra due namespace Kubernetes.



Avvio rapido

È possibile configurare la condivisione del volume NFS in pochi semplici passaggi.

1 **Configurare il PVC di origine per condividere il volume**

Il proprietario dello spazio dei nomi di origine concede l'autorizzazione ad accedere ai dati nel PVC di origine.

2 **Concedi l'autorizzazione per creare un CR nello spazio dei nomi di destinazione**

L'amministratore del cluster concede l'autorizzazione al proprietario dello spazio dei nomi di destinazione per creare il CR TridentVolumeReference.

3 **Crea TridentVolumeReference nello spazio dei nomi di destinazione**

Il proprietario dello spazio dei nomi di destinazione crea il CR TridentVolumeReference per fare riferimento al PVC di origine.

4 **Crea 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 tra namespace 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.

Passi

1. **Proprietario dello spazio dei nomi di origine:** Crea il PVC(`pvc1`) nello spazio dei nomi di origine che concede l'autorizzazione alla condivisione con lo spazio dei nomi di destinazione(`namespace2`) utilizzando il `shareToNamespace` annotazione.

   ```
   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 archiviazione NFS backend.

   - È possibile condividere il PVC con più namespace utilizzando un elenco delimitato da virgole. Per esempio, `trident.netapp.io/shareToNamespace: namespace2,namespace3,namespace4` .
   - Puoi condividere con tutti gli spazi dei nomi utilizzando `*` . Per esempio, `trident.netapp.io/shareToNamespace: *`
   - È possibile aggiornare il PVC per includere il `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. **Proprietario dello spazio dei nomi di destinazione:** Crea un CR TridentVolumeReference nello spazio dei nomi di destinazione che fa riferimento 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:** Crea un PVC(`pvc2`) nello spazio dei nomi di destinazione(`namespace2`) utilizzando il `shareFromPVC` annotazione per designare il PVC sorgente.

```
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 a quella del PVC di origine.

Risultati

Il Trident legge il `shareFromPVC` annotazione sul PVC di destinazione e crea il PV di destinazione come volume subordinato senza risorse di archiviazione proprie che punta al PV di origine e condivide la risorsa di archiviazione del PV di origine. La destinazione PVC e PV appaiono vincolate normalmente.

Elimina un volume condiviso

È possibile eliminare un volume condiviso tra più namespace. Trident rimuoverà l'accesso al volume nello spazio dei nomi di origine e manterrà l'accesso per gli altri spazi dei nomi che condividono il volume. Quando tutti gli spazi dei nomi che fanno riferimento al volume vengono rimossi, Trident elimina il volume.

Utilizzo `tridentctl get` per interrogare volumi subordinati

Utilizzando il[`tridentctl` utilità, è possibile eseguire il `get` comando per ottenere volumi subordinati. Per maggiori informazioni, fare riferimento al `tridentctl` comandi e opzioni.

```
Usage:
  tridentctl get [option]
```

Bandiere:

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

Limitazioni

- Trident non può impedire agli spazi dei nomi di destinazione di scrivere sul volume condiviso. Per evitare la sovrascrittura dei dati del volume condiviso, è opportuno utilizzare il blocco dei file o altri processi.
- Non è possibile revocare l'accesso al PVC di origine rimuovendo il `shareToNamespace` O `shareFromNamespace` annotazioni o eliminazione del `TridentVolumeReference` CR. Per revocare l'accesso, è necessario eliminare il PVC subordinato.
- Non è possibile eseguire snapshot, cloni e mirroring sui volumi subordinati.

Per maggiori informazioni

Per saperne di più sull'accesso ai volumi tra più namespace:

- Visita"Condivisione di volumi tra namespace: dai il benvenuto all'accesso ai volumi tra namespace" .
- Guarda la demo su"NetAppTV" .

Clonazione di volumi tra spazi dei nomi

Utilizzando Trident, puoi creare nuovi volumi utilizzando volumi esistenti o snapshot di volume da uno spazio dei nomi diverso all'interno dello stesso cluster Kubernetes.

Prerequisiti

Prima di clonare i volumi, assicurarsi che i backend di origine e di destinazione siano dello stesso tipo e abbiano la stessa classe di archiviazione.

> 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

È possibile impostare la clonazione del volume in pochi semplici passaggi.

1. **Configurare il PVC di origine per clonare il volume**

Il proprietario dello spazio dei nomi di origine concede l'autorizzazione ad accedere ai dati nel PVC di origine.

2 Concedi l'autorizzazione per creare un CR nello spazio dei nomi di destinazione

L'amministratore del cluster concede l'autorizzazione al proprietario dello spazio dei nomi di destinazione per creare il CR TridentVolumeReference.

3 Crea TridentVolumeReference nello spazio dei nomi di destinazione

Il proprietario dello spazio dei nomi di destinazione crea il CR TridentVolumeReference per fare riferimento al PVC di origine.

4 Crea 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, la clonazione di volumi tra namespace richiede la collaborazione e l'azione del proprietario del namespace di origine, dell'amministratore del cluster e del proprietario del namespace di destinazione. Il ruolo dell'utente viene designato in ogni fase.

Passi

1. **Proprietario dello spazio dei nomi di origine:** Crea il PVC(`pvc1`) nello spazio dei nomi di origine(`namespace1`) che concede l'autorizzazione alla condivisione con lo spazio dei nomi di destinazione(`namespace2`) utilizzando il `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 archiviazione backend.

- È possibile condividere il PVC con più namespace utilizzando un elenco delimitato da virgole. Per esempio, `trident.netapp.io/cloneToNamespace: namespace2,namespace3,namespace4` .
- Puoi condividere con tutti gli spazi dei nomi utilizzando `*` . Per esempio, `trident.netapp.io/cloneToNamespace: *`
- È possibile aggiornare il PVC per includere il `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. **Proprietario dello spazio dei nomi di destinazione:** Crea un CR TridentVolumeReference nello spazio dei nomi di destinazione che fa riferimento 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:** Crea un PVC(`pvc2`) nello spazio dei nomi di destinazione(`namespace2`) utilizzando il `cloneFromPVC` O `cloneFromSnapshot` , E `cloneFromNamespace` 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 tramite driver ontap-nas-economy, i cloni di sola lettura non sono supportati.

Replicare i volumi utilizzando SnapMirror

Trident supporta relazioni mirror tra un volume di origine su un cluster e il volume di destinazione sul cluster peer per replicare i dati in caso di ripristino di emergenza. È possibile utilizzare una definizione di risorsa personalizzata (CRD) con namespace, denominata Trident Mirror Relationship (TMR), per eseguire le seguenti operazioni:

- Creare relazioni speculari tra volumi (PVC)
- Rimuovi le relazioni speculari tra i volumi
- Rompere le relazioni speculari
- Promuovere il volume secondario durante le condizioni di disastro (failover)
- Eseguire la transizione senza perdite delle applicazioni da un cluster all'altro (durante i failover o le migrazioni pianificate)

Prerequisiti di replicazione

Prima di iniziare, assicurati che siano soddisfatti i seguenti prerequisiti:

Cluster ONTAP

- * Trident*: Trident versione 22.10 o successiva deve essere presente sia sui cluster Kubernetes di origine che su quelli di destinazione che utilizzano ONTAP come backend.
- **Licenze**: le licenze asincrone ONTAP SnapMirror che utilizzano il bundle Data Protection devono essere abilitate sia sul cluster ONTAP di origine che su quello di destinazione. Fare riferimento a "Panoramica delle licenze SnapMirror in ONTAP" per maggiori informazioni.

  A partire da ONTAP 9.10.1, tutte le licenze vengono fornite come file di licenza NetApp (NLF), ovvero un singolo file che abilita più funzionalità. Fare riferimento a"Licenze incluse con ONTAP One" per maggiori informazioni.

  ⓘ È supportata solo la protezione asincrona SnapMirror .

Sbirciando

- **Cluster e SVM**: i backend di archiviazione ONTAP devono essere peered. Fare riferimento a "Panoramica del peering di cluster e SVM" per maggiori informazioni.

  ⓘ Assicurarsi che i nomi SVM utilizzati nella relazione di replica tra due cluster ONTAP siano univoci.

- * Trident e SVM*: le SVM remote peered devono essere disponibili per Trident sul 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 del volume tramite SnapMirror non è supportata per i sistemi ASA r2. Per informazioni sui sistemi ASA r2, vedere"Scopri di più sui sistemi di archiviazione ASA r2" .

Crea un PVC specchiato

Seguire questi passaggi e utilizzare gli esempi CRD per creare una relazione speculare tra volumi primari e secondari.

Passi

1. Eseguire i seguenti passaggi sul cluster Kubernetes primario:
   a. Creare un oggetto StorageClass con `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. Creare 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 corrente di protezione dei dati (DP) del volume, quindi popola il campo di stato di MirrorRelationship.

d. Ottenere il CR TridentMirrorRelationship per ottenere il nome interno e l'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 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. Creare un PVC con StorageClass creato in precedenza che funga da destinazione secondaria (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 il CRD TridentMirrorRelationship e non riuscirà a creare il volume se la relazione non esiste. Se la relazione esiste, Trident assicurerà che il nuovo FlexVol volume venga posizionato su una SVM collegata alla SVM remota definita in MirrorRelationship.

Stati di replica del volume

Una relazione Trident Mirror (TMR) è una CRD che rappresenta un'estremità di una relazione di replicazione tra PVC. Il TMR di destinazione ha uno stato che indica a Trident qual è lo stato desiderato. La TMR di destinazione presenta 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 mirror 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 è mai stato in una relazione con il volume di origine, perché sovrascrive il contenuto del volume di destinazione.
  - Lo stato ripristinato non riuscirà se il volume non era precedentemente in relazione con la sorgente.

Promuovere il PVC secondario durante un failover non pianificato

Eseguire il seguente passaggio sul cluster Kubernetes secondario:

- Aggiorna il campo *spec.state* di TridentMirrorRelationship a `promoted` .

Promuovere PVC secondario durante un failover pianificato

Durante un failover pianificato (migrazione), eseguire i seguenti passaggi per promuovere il PVC secondario:

Passi

1. Sul cluster Kubernetes primario, creare uno snapshot del PVC e attendere che venga creato.
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 CR *TridentMirrorRelationship* in *promoted* e *spec.promotedSnapshotHandle* in modo che sia internalName dello snapshot.
4. Nel cluster Kubernetes secondario, confermare lo stato (campo status.state) di TridentMirrorRelationship su promosso.

Ripristinare una relazione mirror dopo un failover

Prima di ripristinare una relazione speculare, scegli il lato che vuoi rendere primario.

Passi

1. Nel cluster Kubernetes secondario, assicurarsi che i valori per il campo *spec.remoteVolumeHandle* su TridentMirrorRelationship siano aggiornati.
2. Sul cluster Kubernetes secondario, aggiorna il campo *spec.mirror* di TridentMirrorRelationship a `reestablished`.

Operazioni aggiuntive

Trident supporta le seguenti operazioni sui volumi primario e secondario:

Replicare il PVC primario in un nuovo PVC secondario

Assicurati di avere già un PVC primario e un PVC secondario.

Passi

1. Eliminare i CRD PersistentVolumeClaim e TridentMirrorRelationship dal cluster secondario (di destinazione) stabilito.
2. Eliminare il CRD TridentMirrorRelationship dal cluster primario (sorgente).
3. Creare un nuovo CRD TridentMirrorRelationship sul cluster primario (sorgente) per il nuovo PVC secondario (destinazione) che si desidera stabilire.

Ridimensiona un PVC specchiato, primario o secondario

Il PVC può essere ridimensionato normalmente, ONTAP espanderà automaticamente tutti i flevxol di destinazione se la quantità di dati supera la dimensione corrente.

Rimuovere la replicazione da un PVC

Per rimuovere la replica, eseguire una delle seguenti operazioni sul volume secondario corrente:

- Eliminare la MirrorRelationship sul PVC secondario. Ciò interrompe la relazione di replicazione.
- Oppure, aggiorna il campo spec.state in *promoted*.

Elimina un PVC (che era stato precedentemente specchiato)

Trident verifica la presenza di PVC replicati e rilascia la relazione di replicazione prima di tentare di eliminare il volume.

Elimina un TMR

L'eliminazione di un TMR su un lato di una relazione speculare fa sì che il TMR rimanente passi allo stato *promosso* prima che Trident completi l'eliminazione. Se il TMR selezionato per l'eliminazione è già nello stato *promosso*, non esiste alcuna relazione mirror e il TMR verrà rimosso e Trident promuoverà il PVC locale a *ReadWrite*. Questa eliminazione rilascia i metadati SnapMirror per il volume locale in ONTAP. Se in futuro questo volume verrà utilizzato in una relazione mirror, dovrà utilizzare un nuovo TMR con uno stato di replica del volume *stabilito* durante la creazione della nuova relazione mirror.

Aggiorna le relazioni mirror quando ONTAP è online

Le relazioni speculari possono essere aggiornate in qualsiasi momento dopo essere state stabilite. Puoi usare il `state: promoted` O `state: reestablished` campi per aggiornare le relazioni. Quando si promuove un volume di destinazione a un normale volume ReadWrite, è possibile utilizzare *promotedSnapshotHandle* per specificare uno snapshot specifico in cui ripristinare il volume corrente.

Aggiorna le relazioni mirror quando ONTAP è offline

È possibile utilizzare un CRD per eseguire un aggiornamento SnapMirror senza che Trident abbia 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 tra *Riuscito*, *In corso* o *Non riuscito*.

Utilizzare la topologia CSI

Trident può creare e collegare selettivamente volumi ai nodi presenti in un cluster Kubernetes utilizzando "Funzionalità di topologia CSI" .

Panoramica

Utilizzando la funzionalità CSI Topology, l'accesso ai volumi può essere limitato a un sottoinsieme di nodi, in base alle regioni e alle zone di disponibilità. Oggi i provider cloud consentono agli amministratori di Kubernetes di generare nodi basati su zone. I nodi possono essere ubicati in diverse zone di disponibilità all'interno di una regione o tra regioni diverse. Per facilitare il provisioning dei volumi per i carichi di lavoro in un'architettura multizona, Trident utilizza la topologia CSI.

Scopri di più sulla funzionalità CSI Topology "Qui" .

Kubernetes offre due modalità di associazione dei volumi uniche:

- Con `VolumeBindingMode` impostato su `Immediate` Trident crea il volume senza alcuna consapevolezza della topologia. Il binding del volume e il provisioning dinamico vengono gestiti durante la creazione del PVC. Questo è l'impostazione predefinita `VolumeBindingMode` ed è adatto per cluster che non impongono vincoli di topologia. I volumi persistenti vengono creati senza alcuna dipendenza dai requisiti di pianificazione del pod richiedente.
- Con `VolumeBindingMode` impostato su `WaitForFirstConsumer` , la creazione e l'associazione di un volume persistente per un PVC vengono ritardate finché non viene pianificato e creato un pod che utilizza il PVC. In questo modo, i volumi vengono creati per soddisfare i vincoli di pianificazione imposti dai requisiti di topologia.

IL `WaitForFirstConsumer` la modalità di associazione non richiede etichette topologiche. Può essere utilizzato indipendentemente dalla funzionalità CSI Topology.

Cosa ti servirà

Per utilizzare la topologia CSI, è necessario 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 dovrebbero avere etichette che introducono la consapevolezza della topologia(`topology.kubernetes.io/region` E `topology.kubernetes.io/zone`). Queste etichette **dovrebbero essere presenti sui nodi del cluster** prima dell'installazione di Trident affinché Trident sia a conoscenza della 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"}]
```

Passaggio 1: creare un backend consapevole della topologia

I backend di archiviazione Trident possono essere progettati per effettuare il provisioning selettivo dei volumi in base alle zone di disponibilità. Ogni backend può trasportare un optional `supportedTopologies` blocco che rappresenta un elenco di zone e regioni supportate. Per le StorageClass che utilizzano tale backend, un volume verrà creato solo se richiesto da un'applicazione pianificata in una regione/zona supportata.

Ecco un esempio di definizione del 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 regioni e zone per backend. Queste regioni e zone rappresentano l'elenco dei valori consentiti che possono essere forniti in una StorageClass. Per le StorageClass che contengono un sottoinsieme delle regioni e delle zone fornite in un backend, Trident crea un volume sul backend.

Puoi definire `supportedTopologies` anche per pool di archiviazione. Vedere il seguente esempio:

```
---
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 archiviazione. `topology.kubernetes.io/region` E `topology.kubernetes.io/zone` determinano da dove possono essere consumati i pool di stoccaggio.

Passaggio 2: definire StorageClass che riconoscono la topologia

In base alle etichette topologiche fornite ai nodi del cluster, è possibile definire StorageClass in modo che contengano informazioni sulla topologia. Ciò determinerà i pool di archiviazione 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:

```
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 StorageClass non verranno elaborati finché non verranno referenziati in un pod. E, `allowedTopologies` fornisce le zone e la regione da utilizzare. IL `netapp-san-us-east1` StorageClass crea PVC su `san-backend-us-east1` backend definito sopra.

Fase 3: creare e utilizzare un PVC

Dopo aver creato StorageClass e mappato un backend, è ora possibile creare PVC.

Vedi l'esempio `spec` sotto:

```
---
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 produrrebbe 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
```

Per far sì Trident crei un volume e lo leghi al PVC, utilizzare il PVC in un contenitore. Vedere il seguente esempio:

```
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 istruisce Kubernetes a pianificare il pod sui nodi presenti nel `us-east1` regione e scegli tra qualsiasi nodo presente nella `us-east1-a` O `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 backend per includere `supportedTopologies`

I backend preesistenti possono essere aggiornati per includere un elenco di `supportedTopologies` usando `tridentctl backend update` . Ciò non influirà sui volumi già forniti e verrà utilizzato solo per i PVC successivi.

Trova maggiori informazioni

- "Gestire le risorse per i contenitori"
- "Selettore di nodi"
- "Affinità e anti-affinità"
- "Contaminazioni e tolleranze"

Lavorare con gli snapshot

Gli snapshot dei volumi persistenti (PV) di Kubernetes consentono copie point-in-time dei volumi. È possibile creare uno snapshot di un volume creato utilizzando Trident, importare uno snapshot creato al di fuori di Trident, creare un nuovo volume da uno snapshot esistente e recuperare i dati del volume dagli snapshot.

Panoramica

Lo snapshot del volume è supportato 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 lavorare con gli snapshot è necessario disporre di un controller snapshot esterno e di definizioni di risorse personalizzate (CRD). Questa è responsabilità dell'orchestratore di Kubernetes (ad esempio: Kubeadm, GKE, OpenShift).

Se la distribuzione Kubernetes non include il controller snapshot e i CRD, fare riferimento aDistribuisci un controller di snapshot del volume .

> Non creare un controller snapshot se si creano snapshot di volumi on-demand in un ambiente GKE. GKE utilizza un controller snapshot nascosto e integrato.

Crea uno snapshot del volume

Passi

1. Crea un `VolumeSnapshotClass` Per ulteriori informazioni, fare riferimento a"Classe VolumeSnapshot" .
   - IL `driver` indica il pilota Trident CSI.
   - `deletionPolicy`può essere `Delete` O `Retain` . Quando impostato su `Retain` , lo snapshot fisico sottostante sul cluster di archiviazione viene mantenuto anche quando `VolumeSnapshot` l'oggetto viene eliminato.

     Esempio

     ```
     cat snap-sc.yaml
     ```

     ```
     apiVersion: snapshot.storage.k8s.io/v1
     kind: VolumeSnapshotClass
     metadata:
       name: csi-snapclass
     driver: csi.trident.netapp.io
     deletionPolicy: Delete
     ```

2. Crea un'istantanea di un PVC esistente.

   Esempi

   - Questo esempio crea un'istantanea di un PVC esistente.

     ```
     cat snap.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 del volume per un PVC denominato `pvc1` e il nome dello snapshot è impostato su `pvc1-snap` . Un VolumeSnapshot è analogo a un PVC ed è associato a un `VolumeSnapshotContent` oggetto che rappresenta l'istantanea effettiva.

```
kubectl create -f snap.yaml
volumesnapshot.snapshot.storage.k8s.io/pvc1-snap created

kubectl get volumesnapshots
NAME                    AGE
pvc1-snap               50s
```

- Puoi identificare il `VolumeSnapshotContent` oggetto per il `pvc1-snap` VolumeSnapshot descrivendolo. IL `Snapshot Content Name` identifica l'oggetto VolumeSnapshotContent che gestisce questo snapshot. IL `Ready To Use` Il parametro indica che lo snapshot può essere utilizzato 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 del volume

Puoi usare `dataSource` per creare un PVC utilizzando un VolumeSnapshot denominato `<pvc-name>` come fonte dei dati. Una volta creato, il PVC può essere attaccato a un contenitore e utilizzato come qualsiasi altro PVC.

> Il PVC verrà creato nello stesso backend del volume sorgente. Fare riferimento a"KB: La creazione di un PVC da uno snapshot PVC Trident non può essere creata in un backend alternativo" .

L'esempio seguente crea il PVC utilizzando `pvc1-snap` come fonte dei dati.

```
cat pvc-from-snap.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
```

Importa uno snapshot del volume

Trident supporta il"Processo di snapshot pre-provisioning di Kubernetes" per consentire all'amministratore del cluster di creare un `VolumeSnapshotContent` oggetti e importare snapshot creati al di fuori di Trident.

Prima di iniziare

Trident deve aver creato o importato il volume padre dello snapshot.

Passi

1. **Amministratore del cluster:** Crea un `VolumeSnapshotContent` oggetto che fa riferimento allo snapshot del backend. In questo modo si avvia il flusso di lavoro degli snapshot in Trident.
   - Specificare il nome dello snapshot del backend in `annotations` COME `trident.netapp.io/internalSnapshotName: <"backend-snapshot-name">` .
   - Specificare `<name-of-parent-volume-in-trident>/<volume-snapshot-content-name>` In `snapshotHandle` Questa è l'unica informazione fornita a Trident dallo snapshotter esterno nel `ListSnapshots` chiamata.

     IL `<volumeSnapshotContentName>` non può sempre corrispondere al nome dello snapshot backend a causa dei vincoli di denominazione CR.

     Esempio

     L'esempio seguente crea un `VolumeSnapshotContent` oggetto che fa riferimento allo snapshot del backend `snap-01` .

```
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. **Amministratore del cluster:** Crea il `VolumeSnapshot` CR che fa riferimento al `VolumeSnapshotContent` oggetto. Questa richiesta richiede l'accesso per utilizzare il `VolumeSnapshot` in un dato spazio dei nomi.

   Esempio

   L'esempio seguente crea un `VolumeSnapshot` CR nominato `import-snap` che fa riferimento al `VolumeSnapshotContent` nominato `import-snap-content` .

```
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 snapshotter esterno riconosce il nuovo creato `VolumeSnapshotContent` e gestisce il `ListSnapshots` chiamata. Trident crea il `TridentSnapshot` .
   - Lo snapshotter esterno imposta il `VolumeSnapshotContent` A `readyToUse` e il `VolumeSnapshot` A `true` .
   - Il ritorno Trident `readyToUse=true` .
4. **Qualsiasi utente:** Crea un `PersistentVolumeClaim` per fare riferimento al nuovo `VolumeSnapshot` , dove il `spec.dataSource` (O `spec.dataSourceRef`) il nome è il `VolumeSnapshot` nome.

Esempio

L'esempio seguente crea un PVC che fa riferimento a `VolumeSnapshot` nominato `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
```

Recupera i dati del volume utilizzando gli snapshot

La directory snapshot è nascosta per impostazione predefinita per facilitare la massima compatibilità dei volumi forniti tramite `ontap-nas` E `ontap-nas-economy` conducenti. Abilita il `.snapshot` directory per recuperare direttamente i dati dagli snapshot.

Utilizzare il ripristino dello snapshot del volume ONTAP CLI per ripristinare un volume a uno stato 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 loco da uno snapshot

Trident fornisce un rapido ripristino del volume in loco da uno snapshot utilizzando `TridentActionSnapshotRestore` (TASR) CR. Questa CR funziona come un'azione Kubernetes imperativa e non persiste dopo il completamento dell'operazione.

Trident supporta il ripristino degli snapshot su `ontap-san` , `ontap-san-economy` , `ontap-nas` , `ontap-nas-flexgroup` , `azure-netapp-files` , `gcp-cvs` , `google-cloud-netapp-volumes` , E `solidfire-san` conducenti.

Prima di iniziare

È necessario disporre di un PVC associato e di uno snapshot del volume disponibile.

- Verificare che lo stato del PVC sia vincolato.

```
kubectl get pvc
```

- Verificare che lo snapshot del volume sia pronto per l'uso.

```
kubectl get vs
```

Passi

1. Creare il CR TASR. Questo esempio crea un CR per PVC `pvc1` e snapshot del volume `pvc1-snapshot` .

   Il CR TASR deve trovarsi in uno spazio dei nomi in cui sono presenti PVC e VS.

```
cat tasr-pvc1-snapshot.yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentActionSnapshotRestore
metadata:
  name: trident-snap
  namespace: trident
spec:
  pvcName: pvc1
  volumeSnapshotName: pvc1-snapshot
```

2. Applicare il CR per ripristinare dallo snapshot. Questo esempio ripristina da uno snapshot `pvc1` .

```
kubectl create -f tasr-pvc1-snapshot.yaml
```

```
tridentactionsnapshotrestore.trident.netapp.io/trident-snap created
```

Risultati

Trident ripristina i dati dallo 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 riproverà automaticamente l'operazione in caso di errore. Sarà necessario eseguire nuovamente l'operazione.
- Gli utenti Kubernetes senza accesso amministrativo potrebbero dover ottenere l'autorizzazione dall'amministratore per creare un CR TASR nel namespace della propria applicazione.

Elimina un PV con gli snapshot associati

Quando si elimina un volume persistente con snapshot associati, il volume Trident corrispondente viene aggiornato allo "stato di eliminazione". Rimuovere gli snapshot del volume per eliminare il volume Trident .

Distribuisci un controller di snapshot del volume

Se la distribuzione Kubernetes non include il controller snapshot e i CRD, è possibile distribuirli come segue.

Passi

1. Creare CRD di snapshot del 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 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` al tuo namespace.

Link correlati

- "Istantanee del volume"
- "Classe VolumeSnapshot"

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 volumi). Questo snapshot del gruppo di volumi rappresenta copie di più volumi eseguite nello stesso momento.

VolumeGroupSnapshot è una funzionalità beta di Kubernetes con API beta. Kubernetes 1.32 è la versione minima richiesta per VolumeGroupSnapshot.

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 lavorare con gli snapshot è necessario disporre di un controller snapshot esterno e di definizioni di risorse personalizzate (CRD). Questa è responsabilità dell'orchestratore di Kubernetes (ad esempio: Kubeadm, GKE, OpenShift).

  Se la distribuzione Kubernetes non include il controller snapshot esterno e i CRD, fare riferimento aDistribuisci un controller di snapshot del volume .

  > Non creare un controller snapshot se si creano snapshot di gruppi di volumi su richiesta in un ambiente GKE. GKE utilizza un controller snapshot nascosto e integrato.

- Nel controller snapshot YAML, impostare `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.

Passi

- Creare una VolumeGroupSnapshotClass prima di creare una VolumeGroupSnapshot. Per maggiori informazioni, fare riferimento a"ClasseSnapshotVolumeGroup" .

```
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.

  L'esempio seguente crea il PVC utilizzando `pvc1-group-snap` come fonte di dati ed etichetta `consistentGroupSnapshot: groupA` . 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 gli snapshot individuali creati come parte dello snapshot del gruppo di volumi. Non è possibile ripristinare lo snapshot del gruppo di volumi come unità.

Utilizzare il ripristino dello snapshot del volume ONTAP CLI per ripristinare un volume a uno stato 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 loco da uno snapshot

Trident fornisce un rapido ripristino del volume in loco da uno snapshot utilizzando `TridentActionSnapshotRestore` (TASR) CR. Questa CR funziona come un'azione Kubernetes imperativa e non persiste dopo il completamento dell'operazione.

Per maggiori informazioni, vedere "Ripristino del volume in loco 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 complesso, 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.

Distribuisci un controller di snapshot del volume

Se la distribuzione Kubernetes non include il controller snapshot e i CRD, è possibile distribuirli come segue.

Passi

1. Creare CRD di snapshot del 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 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` al tuo namespace.

Link correlati

- "ClasseSnapshotVolumeGroup"
- "Istantanee del volume"

Gestire e monitorare Trident

Trident potenziato

Trident potenziato

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

Considerazioni prima dell'aggiornamento

Quando si esegue l'aggiornamento all'ultima versione di Trident, tenere presente quanto segue:

- Dovrebbe essere installata una sola istanza Trident in tutti gli spazi dei nomi di un determinato cluster Kubernetes.
- Trident 23.07 e versioni successive richiedono snapshot del volume v1 e non supportano più snapshot alpha o beta.
- Se hai creato il Cloud Volumes Service per Google Cloud in"Tipo di servizio CVS" , è necessario aggiornare la configurazione del backend per utilizzare `standardsw` O `zoneredundantstandardsw` livello di servizio durante l'aggiornamento da Trident 23.01. Mancato aggiornamento del `serviceLevel` nel backend potrebbe causare il fallimento dei volumi. Fare riferimento a "Esempi di tipi di servizio CVS" per i dettagli.
- Quando si esegue l'aggiornamento, è importante fornire `parameter.fsType` In `StorageClasses` utilizzato da Trident. Puoi eliminare e ricreare `StorageClasses` senza interrompere i volumi preesistenti.
  - Questo è un **requisito** per l'applicazione "contesti di sicurezza" per volumi SAN.
  - La directory https://github.com/NetApp/trident/tree/master/trident-installer/sample-input [sample input^] contiene esempi, come `storage-class-basic.yaml.templ` e collegamento: `storage-class-bronze-default.yaml` .
  - Per maggiori informazioni, fare riferimento a"Problemi noti" .

Passaggio 1: seleziona una versione

Le versioni Trident seguono un sistema basato sulla data `YY.MM` convenzione di denominazione, dove "YY" rappresenta le ultime due cifre dell'anno e "MM" è il mese. Le versioni Dot seguono un `YY.MM.X` convenzione, dove "X" è il livello di patch. Selezionerai la versione a cui effettuare l'aggiornamento in base alla versione da cui stai effettuando l'aggiornamento.

- È possibile eseguire un aggiornamento diretto a qualsiasi versione di destinazione che si trovi entro una finestra temporale di quattro versioni dalla versione installata. Ad esempio, è possibile effettuare l'aggiornamento direttamente dalla versione 24.06 (o da qualsiasi versione 24.06 dot) alla versione 25.06.
- Se si esegue l'aggiornamento da una versione al di fuori della finestra delle quattro versioni, eseguire un aggiornamento in più fasi. Utilizzare le istruzioni di aggiornamento per"versione precedente" stai eseguendo l'aggiornamento alla versione più recente che rientra nella finestra temporale delle quattro versioni. Ad esempio, se stai utilizzando la versione 23.07 e vuoi eseguire l'aggiornamento alla versione 25.06:

a. Primo aggiornamento dal 23.07 al 24.06.

b. Quindi aggiorna dalla versione 24.06 alla versione 25.06.

> Quando si esegue l'aggiornamento tramite l'operatore Trident su OpenShift Container Platform, è necessario eseguire l'aggiornamento a Trident 21.01.1 o versione successiva. L'operatore Trident rilasciato con la versione 21.01.0 contiene un problema noto che è stato risolto nella versione 21.01.1. Per maggiori dettagli fare riferimento al "dettagli del problema su GitHub" .

Passaggio 2: determinare il metodo di installazione originale

Per determinare quale versione hai utilizzato per installare originariamente Trident:

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

Passaggio 3: seleziona un metodo di aggiornamento

In genere, dovresti eseguire l'aggiornamento utilizzando lo stesso metodo utilizzato per l'installazione iniziale, tuttavia puoi"spostarsi tra i metodi di installazione" . Ci sono due opzioni per potenziare Trident.

- "Aggiorna utilizzando l'operatore Trident"

> Ti suggeriamo di rivedere"Comprendere il flusso di lavoro di aggiornamento dell'operatore" prima di effettuare l'aggiornamento con l'operatore.

*

Aggiorna con l'operatore

Comprendere il flusso di lavoro di aggiornamento dell'operatore

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

Gestione dell'aggiornamento dell'operatore Trident

Uno dei tanti"vantaggi dell'utilizzo dell'operatore Trident" per installare e aggiornare Trident è la gestione automatica degli oggetti Trident e Kubernetes senza interrompere i volumi montati esistenti. In questo modo, Trident può supportare gli aggiornamenti senza tempi di inattività, oppure"*aggiornamenti continui*" . In particolare, l'operatore Trident comunica con il cluster Kubernetes per:

- Eliminare e ricreare la distribuzione Trident Controller e il nodo DaemonSet.
- Sostituisci i Trident Controller Pod e i Trident Node Pod con nuove versioni.
  - Se un nodo non viene aggiornato, ciò non impedisce l'aggiornamento dei nodi rimanenti.
  - Solo i nodi con un Trident Node Pod in esecuzione possono montare volumi.

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

Flusso di lavoro di aggiornamento dell'operatore

Quando si avvia un aggiornamento utilizzando l'operatore Trident :

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

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

È possibile aggiornare Trident utilizzando l'operatore Trident manualmente o tramite Helm. È possibile effettuare l'aggiornamento da un'installazione dell'operatore Trident a un'altra installazione dell'operatore Trident o … `tridentctl` installazione su una versione dell'operatore Trident . Revisione"Seleziona un metodo di aggiornamento" prima di aggiornare l'installazione di un operatore Trident .

Aggiornare un'installazione manuale

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

Per eseguire l'aggiornamento da Trident installato utilizzando l'operatore con ambito namespace (versioni dalla 20.07 alla 20.10), utilizzare le istruzioni di aggiornamento per"la tua versione installata" del Trident.

Informazioni su questo compito

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

- Per i cluster che eseguono Kubernetes 1.24, utilizzare"bundle_pre_1_25.yaml" .
- Per i cluster che eseguono Kubernetes 1.25 o versioni successive, utilizzare"bundle_post_1_25.yaml" .

Prima di iniziare

Assicurati di utilizzare un cluster Kubernetes in esecuzione"una versione di Kubernetes supportata" .

Passi

1. Verifica la tua versione Trident :

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

2. Aggiorna il `operator.yaml` , `tridentorchestrator_cr.yaml` , E `post_1_25_bundle.yaml` con il registro e i percorsi immagine per la versione a cui si sta effettuando l'aggiornamento (ad esempio 25.06) e il segreto corretto.
3. Eliminare l'operatore Trident utilizzato per installare l'istanza Trident corrente. Ad esempio, se si esegue l'aggiornamento dalla versione 25.02, eseguire il seguente comando:

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

4. Se hai personalizzato l'installazione iniziale utilizzando `TridentOrchestrator` attributi, è possibile modificare il `TridentOrchestrator` oggetto per modificare i parametri di installazione. Ciò potrebbe includere modifiche apportate per specificare registri di immagini Trident e CSI speculari per la modalità offline, abilitare registri di debug o specificare segreti di estrazione delle immagini.
5. Installa Trident utilizzando il file YAML del bundle corretto per il tuo ambiente, dove *<bundle.yaml>* è `bundle_pre_1_25.yaml` O `bundle_post_1_25.yaml` in base alla versione di Kubernetes. Ad esempio, se si installa Trident 25.06.0, eseguire il seguente comando:

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

6. Modifica la torcia del tridente per includere l'immagine 25.06.0.

Aggiornare un'installazione di Helm

È possibile aggiornare un'installazione Trident Helm.

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

Se hai già aggiornato il tuo cluster Kubernetes dalla versione 1.24 alla versione 1.25 senza aggiornare Trident Helm, l'aggiornamento di Helm non riesce. Per completare l'aggiornamento del timone, è necessario eseguire questi passaggi come prerequisiti:

1. Installa il plugin helm-mapkubeapis da https://github.com/helm/helm-mapkubeapis .
2. Eseguire una prova di funzionamento della versione Trident nello spazio dei nomi in cui è installato Trident . Qui sono elencate le risorse che verranno ripulite.

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

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

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

Passi

1. Se tu"installato Trident tramite Helm" , puoi usare `helm upgrade trident netapp-trident/trident-operator --version 100.2506.0` per aggiornare in un unico passaggio. Se non hai aggiunto il repository Helm o non puoi utilizzarlo per l'aggiornamento:
   a. Scarica l'ultima versione Trident da"la sezione *Assets* su GitHub" .
   b. Utilizzare il `helm upgrade` comando dove `trident-operator-25.06.0.tgz` riflette la versione a cui si desidera effettuare l'aggiornamento.

```
helm upgrade <name> trident-operator-25.06.0.tgz
```

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

2. Correre `helm list` per verificare che sia la versione del grafico che quella dell'app siano state aggiornate. Correre `tridentctl logs` per rivedere eventuali messaggi di debug.

Aggiorna da un `tridentctl` installazione all'operatore Trident

È possibile effettuare l'aggiornamento all'ultima versione dell'operatore Trident da un `tridentctl` installazione. I backend e i PVC esistenti saranno automaticamente disponibili.

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

Passi

1. Scarica l'ultima versione Trident .

```
# Download the release required [25.06.0]
mkdir 25.06.0
cd 25.06.0
wget
https://github.com/NetApp/trident/releases/download/v25.06.0/trident-
installer-25.06.0.tar.gz
tar -xf trident-installer-25.06.0.tar.gz
cd trident-installer
```

2. Crea il `tridentorchestrator` CRD dal manifesto.

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

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

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

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

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

4. Crea un `TridentOrchestrator` CR per l'installazione Trident.

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

kubectl create -f deploy/crds/tridentorchestrator_cr.yaml

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

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

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

Message:                Trident installed
Namespace:              trident
Status:                 Installed
Version:                v25.06.0
```

Aggiorna con tridentctl

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

Informazioni su questo compito

Disinstallare e reinstallare Trident equivale a eseguire un aggiornamento. Quando si disinstalla Trident, il Persistent Volume Claim (PVC) e il Persistent Volume (PV) utilizzati dalla distribuzione Trident non vengono eliminati. I PV già forniti rimarranno disponibili mentre Trident è offline e Trident fornirà volumi per tutti i PVC creati nel frattempo, dopo essere tornato online.

Prima di iniziare

Revisione"Seleziona un metodo di aggiornamento" prima di effettuare l'aggiornamento utilizzando `tridentctl`.

Passi

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

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

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

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

Gestisci Trident usando tridentctl

IL "Pacchetto di installazione Trident" include il `tridentctl` utilità della riga di comando per fornire un accesso semplice a Trident. Gli utenti Kubernetes con privilegi sufficienti possono utilizzarlo per installare Trident o gestire lo spazio dei nomi che contiene il pod Trident .

Comandi e flag globali

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

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

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

Comandi

`create`
Aggiungi una risorsa a Trident.

`delete`
Rimuovi una o più risorse da Trident.

`get`
Ottieni una o più risorse da Trident.

`help`
Aiuto per qualsiasi comando.

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

`import`
Importa una risorsa esistente in Trident.

`install`
Installa Trident.

`logs`
Stampa i registri da Trident.

`send`
Invia una risorsa da Trident.

`uninstall`
Disinstallare Trident.

`update`
Modificare una risorsa in Trident.

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

`upgrade`
Aggiorna una risorsa in Trident.

`version`
Stampa la versione di Trident.

Bandiere globali

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

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

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

In alternativa, è possibile esportare il `KUBECONFIG` variabile per puntare a un cluster Kubernetes specifico e problema `tridentctl` comandi a quel cluster.

`-n, --namespace string`
Spazio dei nomi della distribuzione Trident .

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

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

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

Opzioni e flag dei comandi

creare

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

```
tridentctl create [option]
```

Opzioni

`backend`: Aggiungi un backend a Trident.

eliminare

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

```
tridentctl delete [option]
```

Opzioni

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

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

Ottenere

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

```
tridentctl get [option]
```

Opzioni

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

Bandiere

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

immagini

Utilizzo `images` flag per stampare una tabella delle immagini del contenitore di cui Trident ha bisogno.

```
tridentctl images [flags]
```

Bandiere

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

volume di importazione

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

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

Alias

`volume`, `v`

Bandiere

`-f`, `--filename string` : Percorso al file PVC YAML o JSON.
`-h` , `--help` : Aiuto per il volume.
`--no-manage` : Crea solo PV/PVC. Non dare per scontato che la gestione del ciclo di vita sia incentrata sul volume.

installare

Utilizzare il `install` flag per installare Trident.

```
tridentctl install [flags]
```

Bandiere

`--autosupport-image string`: L'immagine del contenitore per Autosupport Telemetry (predefinita "netapp/trident autosupport:<current-version>").
`--autosupport-proxy string` : Indirizzo/porta di un proxy per l'invio di dati di telemetria di Autosupport.
`--enable-node-prep` : Tentativo di installare i pacchetti richiesti sui nodi.
`--generate-custom-yaml` : Genera file YAML senza installare nulla.
`-h` , `--help` : Aiuto per l'installazione.
`--http-request-timeout` : Sostituisci il timeout della richiesta HTTP per l'API REST del controller Trident (predefinito 1m30s).
`--image-registry string` : L'indirizzo/porta di un registro di immagini interno.
`--k8s-timeout duration` : Timeout per tutte le operazioni Kubernetes (predefinito 3m0s).
`--kubelet-dir string` : Posizione host dello stato interno di kubelet (predefinito "/var/lib/kubelet").
`--log-format string` : Formato di registrazione Trident (testo, json) (predefinito "testo").
`--node-prep` : consente a Trident di preparare i nodi del cluster Kubernetes per gestire i volumi utilizzando il protocollo di archiviazione dati specificato. **Attualmente, `iscsi` è l'unico valore supportato. A partire da OpenShift 4.19, la versione minima Trident supportata per questa funzionalità è 25.06.1.**
`--pv string` : Il nome del PV legacy utilizzato da Trident, assicura che non esista (predefinito "trident").
`--pvc string` : Il nome del PVC legacy utilizzato da Trident, assicura che questo non esista (predefinito "trident").
`--silence-autosupport` : Non inviare automaticamente i bundle di supporto automatico a NetApp (valore predefinito: true).
`--silent` : Disabilita la maggior parte degli output durante l'installazione.
`--trident-image string` : L'immagine Trident da installare.
`--k8s-api-qps` : Limite di query al secondo (QPS) per le richieste API di Kubernetes (predefinito 100; facoltativo).
`--use-custom-yaml` : Utilizzare tutti i file YAML esistenti nella directory di installazione.
`--use-ipv6` : Utilizza IPv6 per la comunicazione di Trident.

registri

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

```
tridentctl logs [flags]
```

Bandiere

`-a`, `--archive` : Crea un archivio di supporto con tutti i log, salvo diversa indicazione.
`-h` , `--help` : Aiuto per i registri.
`-l` , `--log string` : Registro Trident da visualizzare. Uno tra trident|auto|trident-operator|all (predefinito "auto").
`--node string` : Nome del nodo Kubernetes da cui raccogliere i log dei pod dei nodi.
`-p` , `--previous` : Ottieni i log per l'istanza del contenitore precedente, se esiste.
`--sidecars` : Prendi i registri per i contenitori del sidecar.

Inviare

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

```
tridentctl send [option]
```

Opzioni

`autosupport`: Invia un archivio Autosupport a NetApp.

disinstallare

Utilizzo `uninstall` flag per disinstallare Trident.

```
tridentctl uninstall [flags]
```

Bandiere

`-h, --help`: Aiuto per la disinstallazione.
`--silent` : Disabilita la maggior parte degli output durante la disinstallazione.

aggiornamento

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

```
tridentctl update [option]
```

Opzioni

`backend`: Aggiorna un backend in Trident.

aggiorna lo stato del backend

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

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

Punti da considerare

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

  #### Bandiere

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

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

`ReconcileNodeAccess` rimangono disponibili.

Puoi anche aggiornare lo stato del backend utilizzando `userState` campo nel file di configurazione del backend `TridentBackendConfig` O `backend.json` . Per maggiori informazioni, fare riferimento a"Opzioni per la gestione dei backend" E"Eseguire la gestione del backend con kubectl" .

Esempio:

JSON

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

1. Modifica il `backend.json` file da includere `userState` campo con il suo valore impostato su "sospeso".
2. Aggiorna il backend utilizzando `tridentctl update backend` comando e il percorso per l'aggiornamento `backend.json` file.

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

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

YAML

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

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

versione

Utilizzo `version` bandiere per stampare la versione di `tridentctl` e il servizio Trident in funzione.

```
tridentctl version [flags]
```

Bandiere

`--client`: Solo versione client (non è richiesto alcun server).
`-h, --help` : Aiuto per la versione.

Supporto plugin

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

Monitor Trident

Trident fornisce un set di endpoint di metriche Prometheus che puoi utilizzare per monitorare le prestazioni Trident .

Panoramica

Le metriche fornite da Trident consentono di fare quanto segue:

- Tieni d'occhio lo stato di salute e la configurazione di Trident. È possibile verificare il successo delle operazioni e se è possibile comunicare con i backend come previsto.
- Esaminare le informazioni sull'utilizzo del backend e comprendere quanti volumi sono forniti su un backend, la quantità di spazio consumata e così via.
- Mantenere una mappatura della quantità di volumi forniti sui backend disponibili.
- Prestazioni in pista. Puoi dare un'occhiata a quanto tempo impiega Trident per comunicare con i backend ed eseguire le operazioni.

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

Cosa ti servirà

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

Fase 1: definire un obiettivo Prometheus

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

Kubernetes e come creare un ServiceMonitor per ottenere le metriche Trident .

Passaggio 2: creare un Prometheus ServiceMonitor

Per utilizzare le metriche Trident , è necessario creare un Prometheus ServiceMonitor che monitora il `trident-csi` servizio e ascolta sul `metrics` porta. Un esempio di ServiceMonitor si presenta così:

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

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

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

Passaggio 3: interrogare le metriche Trident con PromQL

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

Ecco alcune query PromQL che puoi utilizzare:

Ottieni informazioni sulla salute Trident

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

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

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

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

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

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

Ottieni informazioni sull'utilizzo Trident

- **Dimensione media del volume**

```
trident_volume_allocated_bytes/trident_volume_count
```

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

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

Ottieni l'utilizzo del volume individuale

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

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

```
kubelet_volume_stats_used_bytes / kubelet_volume_stats_capacity_bytes *
100
```

Scopri di più sulla telemetria di Trident AutoSupport

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

- Per impedire a Trident di inviare metriche Prometheus e informazioni di base sul backend a NetApp, passare il `--silence-autosupport` bandiera durante l'installazione Trident .
- Trident può anche inviare i log dei container al supporto NetApp su richiesta tramite `tridentctl send autosupport` . Sarà necessario attivare Trident per caricare i suoi registri. Prima di inviare i log, dovresti accettare i termini e le condizioni di NetApphttps://www.netapp.com/company/legal/privacy-policy/["politica sulla riservatezza"^] .

- Se non diversamente specificato, Trident recupera i log delle ultime 24 ore.
- È possibile specificare l'intervallo di tempo di conservazione del registro con `--since` bandiera. Per esempio: `tridentctl send autosupport --since=1h` . Queste informazioni vengono raccolte e inviate tramite un `trident-autosupport` contenitore installato accanto a Trident. È possibile ottenere l'immagine del contenitore su "Trident AutoSupport" .
- Trident AutoSupport non raccoglie né trasmette informazioni di identificazione personale (PII) o informazioni personali. Viene fornito con un "Contratto di licenza con l'utente finale" ciò non è applicabile all'immagine del contenitore Trident stesso. Puoi scoprire di più sull'impegno di NetApp per la sicurezza e l'affidabilità dei dati "Qui" .

Un esempio di payload inviato da Trident si presenta così:

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

- I messaggi AutoSupport vengono inviati all'endpoint AutoSupport di NetApp. Se si utilizza un registro privato per archiviare le immagini dei contenitori, è possibile utilizzare `--image-registry` bandiera.
- È anche possibile configurare gli URL proxy generando i file YAML di installazione. Questo può essere fatto utilizzando `tridentctl install --generate-custom-yaml` per creare i file YAML e aggiungere il `--proxy-url` argomento per la `trident-autosupport` contenitore in `trident-deployment.yaml` .

Disabilita le metriche Trident

Per **disabilitare** la segnalazione delle metriche, dovresti generare YAML personalizzati (utilizzando `--generate-custom-yaml` flag) e modificarli per rimuovere il `--metrics` flag dall'essere invocato per il `trident-main` contenitore.

Disinstallare Trident

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

Informazioni su questo compito

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

Prima di iniziare

Se si desidera dismettere i cluster Kubernetes, è necessario eliminare tutte le applicazioni che utilizzano volumi creati da Trident prima della disinstallazione. Ciò garantisce che i PVC non vengano pubblicati sui nodi Kubernetes prima di essere eliminati.

Determinare il metodo di installazione originale

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

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

Disinstallare un'installazione dell'operatore Trident

È possibile disinstallare manualmente un'installazione dell'operatore Trident oppure tramite Helm.

Disinstallare l'installazione manuale

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

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

   ```
   kubectl patch torc <trident-orchestrator-name> --type=merge -p
   '{"spec":{"uninstall":true}}'
   ```

   Quando il `uninstall` la bandiera è impostata su `true` , l'operatore Trident disinstalla Trident, ma non rimuove TridentOrchestrator stesso. Se vuoi installare nuovamente Trident, dovresti pulire TridentOrchestrator e crearne uno nuovo.

2. **Eliminare `TridentOrchestrator`** : Rimuovendo il `TridentOrchestrator` CR utilizzato per distribuire Trident, si chiede all'operatore di disinstallare Trident. L'operatore elabora la rimozione di `TridentOrchestrator` e procede alla rimozione della distribuzione e del daemonset Trident ,

eliminando i pod Trident creati come parte dell'installazione.

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

Disinstallare l'installazione di Helm

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

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

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

Disinstallare un `tridentctl` installazione

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

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

Trident per Docker

Prerequisiti per la distribuzione

Prima di poter distribuire Trident, è necessario installare e configurare i prerequisiti del protocollo necessari sul proprio host.

Verificare i requisiti

- Verifica che la tua distribuzione soddisfi tutti i requisiti"requisiti" .
- Verifica di aver installato una versione supportata di Docker. Se la tua versione di Docker non è aggiornata, "installarlo o aggiornarlo" .

```
docker --version
```

- Verificare che i prerequisiti del protocollo siano installati e configurati sul proprio host.

Strumenti NFS

Installare gli strumenti NFS utilizzando i comandi per il sistema operativo in uso.

RHEL 8+

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

Ubuntu

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

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

strumenti iSCSI

Installare gli strumenti iSCSI utilizzando i comandi del sistema operativo in uso.

RHEL 8+

1. Installare i seguenti pacchetti di sistema:

```
sudo yum install -y lsscsi iscsi-initiator-utils sg3_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. Imposta la scansione su manuale:

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

4. Abilita multipathing:

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

Garantire `etc/multipath.conf` contiene `find_multipaths no` Sotto `defaults` .

5. Assicurare che `iscsid` E `multipathd` stanno correndo:

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

6. Abilita e avvia `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 bionic) o 2.0.874-7.1ubuntu6.1 o successiva (per focal):

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

3. Imposta la scansione su manuale:

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

4. Abilita 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
```

Garantire `etc/multipath.conf` contiene `find_multipaths no` Sotto `defaults` .

5. Assicurare 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
```

Strumenti NVMe

Installa gli strumenti NVMe utilizzando i comandi per il 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, potrebbe essere necessario aggiornare la versione del kernel del nodo a una 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
```

Strumenti FC

Installa gli strumenti FC utilizzando i comandi per il tuo sistema operativo.

- Quando si utilizzano nodi worker che eseguono RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) con FC PV, specificare `discard` mountOption in StorageClass per eseguire il recupero dello spazio in linea. Fare riferimento a "Documentazione Red Hat" .

RHEL 8+

1. Installare i seguenti pacchetti di sistema:

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

2. Abilita multipathing:

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

Garantire `etc/multipath.conf` contiene `find_multipaths no` Sotto `defaults` .

3. Assicurare che `multipathd` è 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. Abilita 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
```

Garantire `etc/multipath.conf` contiene `find_multipaths no` Sotto `defaults` .

3. Assicurare che `multipath-tools` è abilitato e in esecuzione:

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

Distribuisci Trident

Trident per Docker fornisce l'integrazione diretta con l'ecosistema Docker per le piattaforme di storage NetApp . Supporta il provisioning e la gestione delle risorse di storage dalla piattaforma di storage agli host Docker, con un framework per l'aggiunta di ulteriori piattaforme in futuro.

Più istanze di Trident possono essere eseguite contemporaneamente sullo stesso host. Ciò consente connessioni simultanee a più sistemi di archiviazione e tipi di archiviazione, con la possibilità di personalizzare l'archiviazione utilizzata per i volumi Docker.

Cosa ti servirà

Vedi il"prerequisiti per la distribuzione" . Dopo aver verificato che i prerequisiti siano soddisfatti, sei pronto per distribuire Trident.

Metodo del plugin gestito da Docker (versione 1.13/17.03 e successive)

Prima di iniziare

Se hai utilizzato Trident prima di Docker 1.13/17.03 nel metodo daemon tradizionale, assicurati di arrestare il processo Trident e riavviare il daemon Docker prima di utilizzare il metodo del plugin gestito.

1. Arresta tutte le istanze in esecuzione:

```
pkill /usr/local/bin/netappdvp
pkill /usr/local/bin/trident
```

2. Riavviare Docker.

```
systemctl restart docker
```

3. Assicurati di aver installato Docker Engine 17.03 (nuovo 1.13) o versione successiva.

```
docker --version
```

Se la tua versione non è aggiornata, "installa o aggiorna la tua installazione" .

Passi

1. Creare un file di configurazione e specificare le opzioni come segue:
   - `config`: Il nome file predefinito è `config.json` , tuttavia puoi usare qualsiasi nome tu scelga specificando il `config` opzione con il nome del file. Il file di configurazione deve trovarsi nella directory `/etc/netappdvp` directory sul sistema host.
   - `log-level`: Specificare il livello di registrazione(`debug` , `info` , `warn` , `error` , `fatal`). L'impostazione predefinita è `info` .

- `debug`: Specifica se la registrazione del debug è abilitata. Il valore predefinito è falso. Se è vero, sostituisce il livello di registro.

  i. Creare una posizione per il file di configurazione:

  ```
  sudo mkdir -p /etc/netappdvp
  ```

  ii. Creare il file di configurazione:

  ```
  cat << EOF > /etc/netappdvp/config.json
  ```

  ```
  {
    "version": 1,
    "storageDriverName": "ontap-nas",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "username": "vsadmin",
    "password": "password",
    "aggregate": "aggr1"
  }
  EOF
  ```

2. Avvia Trident utilizzando il sistema di plugin gestiti. Sostituire `<version>` con la versione del plugin (xxx.xx.x) che stai utilizzando.

   ```
   docker plugin install --grant-all-permissions --alias netapp
   netapp/trident-plugin:<version> config=myConfigFile.json
   ```

3. Inizia a utilizzare Trident per utilizzare lo spazio di archiviazione del sistema configurato.

   a. Crea un volume denominato "firstVolume":

   ```
   docker volume create -d netapp --name firstVolume
   ```

   b. Crea un volume predefinito all'avvio del contenitore:

   ```
   docker run --rm -it --volume-driver netapp --volume
   secondVolume:/my_vol alpine ash
   ```

   c. Rimuovere il volume "firstVolume":

```
docker volume rm firstVolume
```

Metodo tradizionale (versione 1.12 o precedente)

Prima di iniziare

1. Assicurati di avere Docker versione 1.10 o successiva.

   ```
   docker --version
   ```

   Se la tua versione non è aggiornata, aggiorna l'installazione.

   ```
   curl -fsSL https://get.docker.com/ | sh
   ```

   O, "segui le istruzioni per la tua distribuzione" .

2. Assicurarsi che NFS e/o iSCSI siano configurati per il sistema.

Passi

1. Installa e configura il plugin NetApp Docker Volume:
   a. Scarica e decomprimi l'applicazione:

      ```
      wget
      https://github.com/NetApp/trident/releases/download/v25.06.0/trident-
      installer-25.06.0.tar.gz
      tar zxf trident-installer-25.06.0.tar.gz
      ```

   b. Spostarsi in una posizione nel percorso del cestino:

      ```
      sudo mv trident-installer/extras/bin/trident /usr/local/bin/
      sudo chown root:root /usr/local/bin/trident
      sudo chmod 755 /usr/local/bin/trident
      ```

   c. Creare una posizione per il file di configurazione:

      ```
      sudo mkdir -p /etc/netappdvp
      ```

   d. Creare il file di configurazione:

      ```
      cat << EOF > /etc/netappdvp/ontap-nas.json
      ```

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1"
}
EOF
```

2. Dopo aver posizionato il binario e creato il file di configurazione, avviare il demone Trident utilizzando il file di configurazione desiderato.

```
sudo trident --config=/etc/netappdvp/ontap-nas.json
```

> Se non diversamente specificato, il nome predefinito per il driver del volume è "netapp".

Dopo aver avviato il demone, è possibile creare e gestire i volumi utilizzando l'interfaccia Docker CLI.

3. Crea un volume:

```
docker volume create -d netapp --name trident_1
```

4. Fornire un volume Docker all'avvio di un contenitore:

```
docker run --rm -it --volume-driver netapp --volume trident_2:/my_vol
alpine ash
```

5. Rimuovere un volume Docker:

```
docker volume rm trident_1
```

```
docker volume rm trident_2
```

Avvia Trident all'avvio del sistema

Un file di unità di esempio per i sistemi basati su systemd può essere trovato qui `contrib/trident.service.example` nel repository Git. Per utilizzare il file con RHEL, procedere come segue:

1. Copiare il file nella posizione corretta.

   Se sono in esecuzione più istanze, è opportuno utilizzare nomi univoci per i file di unità.

   ```
   cp contrib/trident.service.example
   /usr/lib/systemd/system/trident.service
   ```

2. Modificare il file, cambiare la descrizione (riga 2) in modo che corrisponda al nome del driver e il percorso del file di configurazione (riga 9) in modo che rifletta il proprio ambiente.
3. Ricarica systemd affinché acquisisca le modifiche:

   ```
   systemctl daemon-reload
   ```

4. Abilita il servizio.

   Questo nome varia a seconda di come hai chiamato il file nel `/usr/lib/systemd/system` elenco.

   ```
   systemctl enable trident
   ```

5. Avvia il servizio.

   ```
   systemctl start trident
   ```

6. Visualizza lo stato.

   ```
   systemctl status trident
   ```

Ogni volta che modifichi il file dell'unità, esegui il comando `systemctl daemon-reload` comando affinché sia a conoscenza dei cambiamenti.

Aggiorna o disinstalla Trident

È possibile aggiornare Trident per Docker in tutta sicurezza, senza alcun impatto sui volumi in uso. Durante il processo di aggiornamento ci sarà un breve periodo in cui `docker volume` i comandi diretti al plugin non avranno esito positivo e le applicazioni non saranno in grado di montare volumi finché il plugin non sarà nuovamente in esecuzione. Nella maggior parte dei casi si tratta di una questione di secondi.

Aggiornamento

Per aggiornare Trident per Docker, procedere come segue.

Passi

1. Elenca i volumi esistenti:

```
docker volume ls
DRIVER              VOLUME NAME
netapp:latest       my_volume
```

2. Disattivare il plugin:

```
docker plugin disable -f netapp:latest
docker plugin ls
ID                  NAME                DESCRIPTION
ENABLED
7067f39a5df5        netapp:latest       nDVP - NetApp Docker Volume
Plugin   false
```

3. Aggiorna il plugin:

```
docker plugin upgrade --skip-remote-check --grant-all-permissions
netapp:latest netapp/trident-plugin:21.07
```

> La versione 18.01 di Trident sostituisce la versione nDVP. Dovresti effettuare l'aggiornamento direttamente da `netapp/ndvp-plugin` immagine al `netapp/trident-plugin` immagine.

4. Abilita il plugin:

```
docker plugin enable netapp:latest
```

5. Verificare che il plugin sia abilitato:

```
docker plugin ls
ID                  NAME                DESCRIPTION
ENABLED
7067f39a5df5        netapp:latest       Trident - NetApp Docker Volume
Plugin   true
```

6. Verificare che i volumi siano visibili:

```
docker volume ls
DRIVER              VOLUME NAME
netapp:latest       my_volume
```

Se stai effettuando l'aggiornamento da una vecchia versione di Trident (precedente alla 20.10) a Trident 20.10 o successiva, potresti riscontrare un errore. Per maggiori informazioni, fare riferimento a"Problemi noti" . Se riscontri questo errore, dovresti prima disabilitare il plugin, quindi rimuoverlo e infine installare la versione Trident richiesta passando un parametro di configurazione aggiuntivo: `docker plugin install netapp/trident-plugin:20.10 --alias netapp --grant-all-permissions config=config.json`

Disinstallare

Per disinstallare Trident per Docker, procedere come segue.

Passi

1. Rimuovere tutti i volumi creati dal plugin.
2. Disattivare il plugin:

```
docker plugin disable netapp:latest
docker plugin ls
ID                  NAME                DESCRIPTION
ENABLED
7067f39a5df5        netapp:latest       nDVP - NetApp Docker Volume
Plugin   false
```

3. Rimuovere il plugin:

```
docker plugin rm netapp:latest
```

Lavorare con i volumi

È possibile creare, clonare e rimuovere facilmente volumi utilizzando lo standard `docker volume` comandi con il nome del driver Trident specificato quando necessario.

Crea un volume

- Creare un volume con un driver utilizzando il nome predefinito:

```
docker volume create -d netapp --name firstVolume
```

- Crea un volume con un'istanza Trident specifica:

```
docker volume create -d ntap_bronze --name bronzeVolume
```

Se non specifichi alcun"opzioni" , vengono utilizzati i valori predefiniti per il driver.

- Sostituisci la dimensione predefinita del volume. Per creare un volume da 20 GiB con un driver, vedi l'esempio seguente:

```
docker volume create -d netapp --name my_vol --opt size=20G
```

Le dimensioni del volume sono espresse come stringhe contenenti un valore intero con unità facoltative (esempio: 10 G, 20 GB, 3 TiB). Se non viene specificata alcuna unità, l'impostazione predefinita è G. Le unità di misura possono essere espresse come potenze di 2 (B, KiB, MiB, GiB, TiB) o come potenze di 10 (B, KB, MB, GB, TB). Le unità abbreviate utilizzano potenze di 2 (G = GiB, T = TiB, …).

Rimuovere un volume

- Rimuovi il volume come qualsiasi altro volume Docker:

```
docker volume rm firstVolume
```

Quando si utilizza il `solidfire-san` driver, l'esempio precedente elimina e ripulisce il volume.

Per aggiornare Trident per Docker, procedere come segue.

Clonare un volume

Quando si utilizza il `ontap-nas` , `ontap-san` , `solidfire-san` , E `gcp-cvs storage drivers` , Trident può clonare i volumi. Quando si utilizza il `ontap-nas-flexgroup` O `ontap-nas-economy` driver, la clonazione non è supportata. La creazione di un nuovo volume da un volume esistente comporterà la creazione di un nuovo snapshot.

- Ispezionare il volume per enumerare gli snapshot:

```
docker volume inspect <volume_name>
```

- Crea un nuovo volume da un volume esistente. Verrà creato un nuovo snapshot:

```
docker volume create -d <driver_name> --name <new_name> -o from
=<source_docker_volume>
```

- Crea un nuovo volume da uno snapshot esistente su un volume. Questo non creerà un nuovo snapshot:

```
docker volume create -d <driver_name> --name <new_name> -o from
=<source_docker_volume> -o fromSnapshot=<source_snap_name>
```

Esempio

```
docker volume inspect firstVolume

[
  {
    "Driver": "ontap-nas",
    "Labels": null,
    "Mountpoint": "/var/lib/docker-volumes/ontap-
nas/netappdvp_firstVolume",
    "Name": "firstVolume",
    "Options": {},
    "Scope": "global",
    "Status": {
      "Snapshots": [
        {
          "Created": "2017-02-10T19:05:00Z",
          "Name": "hourly.2017-02-10_1505"
        }
      ]
    }
  }
]

docker volume create -d ontap-nas --name clonedVolume -o from=firstVolume
clonedVolume

docker volume rm clonedVolume
docker volume create -d ontap-nas --name volFromSnap -o from=firstVolume
-o fromSnapshot=hourly.2017-02-10_1505
volFromSnap

docker volume rm volFromSnap
```

Accedi ai volumi creati esternamente

È possibile accedere ai dispositivi a blocchi creati esternamente (o ai loro cloni) tramite contenitori che utilizzano Trident **solo** se non hanno partizioni e se il loro file system è supportato da Trident (ad esempio: un `ext4` -formattato `/dev/sdc1` non sarà accessibile tramite Trident).

Opzioni di volume specifiche del driver

Ogni driver di archiviazione ha un diverso set di opzioni, che è possibile specificare al momento della creazione del volume per personalizzare il risultato. Di seguito sono riportate le opzioni applicabili al sistema di archiviazione configurato.

Utilizzare queste opzioni durante l'operazione di creazione del volume è semplice. Fornire l'opzione e il valore utilizzando il `-o` operatore durante l'operazione CLI. Questi sovrascrivono tutti i valori equivalenti presenti nel file di configurazione JSON.

Opzioni del volume ONTAP

Le opzioni di creazione del volume per NFS, iSCSI e FC includono quanto segue:

Opzione	Descrizione
`size`	La dimensione del volume è predefinita e pari a 1 GiB.
`spaceReserve`	Sottile o spesso determinano il volume, l'impostazione predefinita è sottile. I valori validi sono `none` (sottile provisioning) e `volume` (spessore fornito).
`snapshotPolicy`	In questo modo la policy snapshot verrà impostata sul valore desiderato. L'impostazione predefinita è `none` , il che significa che non verranno creati automaticamente snapshot per il volume. A meno che non venga modificato dall'amministratore dell'archiviazione, su tutti i sistemi ONTAP esiste una policy denominata "default" che crea e conserva sei snapshot orari, due giornalieri e due settimanali. I dati conservati in uno snapshot possono essere recuperati navigando verso `.snapshot` directory in qualsiasi directory del volume.
`snapshotReserve`	In questo modo la riserva di snapshot verrà impostata sulla percentuale desiderata. L'impostazione predefinita è nessun valore, il che significa che ONTAP selezionerà snapshotReserve (in genere 5%) se è stata selezionata una snapshotPolicy oppure 0% se la snapshotPolicy è nessuna. È possibile impostare il valore snapshotReserve predefinito nel file di configurazione per tutti i backend ONTAP e utilizzarlo come opzione di creazione del volume per tutti i backend ONTAP, ad eccezione di ontap-nas-economy.

Opzione	Descrizione
`splitOnClone`	Quando si clona un volume, ONTAP divide immediatamente il clone dal suo elemento padre. L'impostazione predefinita è `false` . In alcuni casi d'uso per la clonazione dei volumi, il modo migliore è separare il clone dal suo elemento padre immediatamente dopo la creazione, perché è improbabile che vi siano opportunità di efficienza di archiviazione. Ad esempio, la clonazione di un database vuoto può comportare un notevole risparmio di tempo, ma poco di spazio di archiviazione, quindi è meglio dividere immediatamente la clonazione.
`encryption`	Abilita NetApp Volume Encryption (NVE) sul nuovo volume; l'impostazione predefinita è `false` . Per utilizzare questa opzione, NVE deve essere concesso in licenza e abilitato sul cluster. Se NAE è abilitato sul backend, qualsiasi volume fornito in Trident sarà abilitato per NAE. Per maggiori informazioni, fare riferimento a:"Come funziona Trident con NVE e NAE" .
`tieringPolicy`	Imposta la politica di suddivisione in livelli da utilizzare per il volume. In questo modo si decide se i dati vengono spostati nel livello cloud quando diventano inattivi (freddi).

Le seguenti opzioni aggiuntive sono **solo** per NFS:

Opzione	Descrizione
`unixPermissions`	Controlla il set di autorizzazioni per il volume stesso. Per impostazione predefinita, le autorizzazioni saranno impostate su `---rwxr-xr-x , o in notazione numerica 0755, e `root` sarà il proprietario. Funzionerà sia il formato testo che quello numerico.
`snapshotDir`	Impostando questo su `true` farà il `.snapshot` directory visibile ai client che accedono al volume. Il valore predefinito è `false` , il che significa che la visibilità del `.snapshot` la directory è disabilitata per impostazione predefinita. Alcune immagini, ad esempio l'immagine ufficiale di MySQL, non funzionano come previsto quando `.snapshot` la directory è visibile.
`exportPolicy`	Imposta la politica di esportazione da utilizzare per il volume. L'impostazione predefinita è `default` .

Opzione	Descrizione
`securityStyle`	Imposta lo stile di sicurezza da utilizzare per l'accesso al volume. L'impostazione predefinita è `unix` . I valori validi sono `unix` E `mixed` .

Le seguenti opzioni aggiuntive sono **solo** per iSCSI:

Opzione	Descrizione
`fileSystemType`	Imposta il file system utilizzato per formattare i volumi iSCSI. L'impostazione predefinita è `ext4` . I valori validi sono `ext3` , `ext4` , E `xfs` .
`spaceAllocation`	Impostando questo su `false` disattiverà la funzione di allocazione dello spazio della LUN. Il valore predefinito è `true` , ovvero ONTAP avvisa l'host quando il volume ha esaurito lo spazio e la LUN nel volume non può accettare scritture. Questa opzione consente inoltre a ONTAP di recuperare spazio automaticamente quando l'host elimina i dati.

Esempi

Vedi gli esempi qui sotto:

- Crea un volume da 10 GiB:

```
docker volume create -d netapp --name demo -o size=10G -o
encryption=true
```

- Crea un volume da 100 GiB con snapshot:

```
docker volume create -d netapp --name demo -o size=100G -o
snapshotPolicy=default -o snapshotReserve=10
```

- Creare un volume con il bit setUID abilitato:

```
docker volume create -d netapp --name demo -o unixPermissions=4755
```

La dimensione minima del volume è 20 MiB.

Se la riserva snapshot non è specificata e la policy snapshot è `none` , Trident utilizza una riserva snapshot dello 0%.

- Crea un volume senza policy snapshot e senza riserva snapshot:

```
docker volume create -d netapp --name my_vol --opt snapshotPolicy=none
```

- Crea un volume senza policy di snapshot e con una riserva di snapshot personalizzata del 10%:

```
docker volume create -d netapp --name my_vol --opt snapshotPolicy=none
--opt snapshotReserve=10
```

- Crea un volume con un criterio snapshot e una riserva snapshot personalizzata del 10%:

```
docker volume create -d netapp --name my_vol --opt
snapshotPolicy=myPolicy --opt snapshotReserve=10
```

- Creare un volume con una policy di snapshot e accettare la riserva di snapshot predefinita dell'ONTAP (in genere il 5%):

```
docker volume create -d netapp --name my_vol --opt
snapshotPolicy=myPolicy
```

Opzioni del volume del software Element

Le opzioni del software Element espongono le policy relative alle dimensioni e alla qualità del servizio (QoS) associate al volume. Quando il volume viene creato, la politica QoS ad esso associata viene specificata utilizzando `-o type=service_level` nomenclatura.

Il primo passo per definire un livello di servizio QoS con il driver Element è creare almeno un tipo e specificare gli IOPS minimi, massimi e burst associati a un nome nel file di configurazione.

Altre opzioni di creazione del volume del software Element includono quanto segue:

Opzione	Descrizione
`size`	La dimensione del volume, predefinita è 1 GiB o voce di configurazione … "defaults": {"size": "5G"}.
`blocksize`	Utilizzare 512 o 4096, il valore predefinito è 512 o la voce di configurazione DefaultBlockSize.

Esempio

Vedere il seguente file di configurazione di esempio con le definizioni QoS:

```
{
  "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
      }
    }
  ]
}
```

Nella configurazione soprastante, abbiamo tre definizioni di policy: Bronzo, Argento e Oro. Questi nomi sono arbitrari.

- Crea un volume Gold da 10 GiB:

```
docker volume create -d solidfire --name sfGold -o type=Gold -o size=10G
```

- Crea un volume Bronze da 100 GiB:

```
docker volume create -d solidfire --name sfBronze -o type=Bronze -o
size=100G
```

Raccogli i registri

È possibile raccogliere i registri per ottenere aiuto nella risoluzione dei problemi. Il metodo utilizzato per raccogliere i log varia in base alla modalità di esecuzione del plugin Docker.

Raccogliere i registri per la risoluzione dei problemi

Passi

1. Se stai eseguendo Trident utilizzando il metodo di plugin gestito consigliato (ad esempio, utilizzando `docker plugin` comandi), visualizzali come segue:

```
docker plugin ls
```

```
ID                  NAME                DESCRIPTION
ENABLED
4fb97d2b956b        netapp:latest       nDVP - NetApp Docker Volume
Plugin   false
journalctl -u docker | grep 4fb97d2b956b
```

   Il livello di registrazione standard dovrebbe consentire di diagnosticare la maggior parte dei problemi. Se ritieni che ciò non sia sufficiente, puoi abilitare la registrazione del debug.

2. Per abilitare la registrazione del debug, installare il plugin con la registrazione del debug abilitata:

```
docker plugin install netapp/trident-plugin:<version> --alias <alias>
debug=true
```

   Oppure, abilita la registrazione del debug quando il plugin è già installato:

```
docker plugin disable <plugin>
```

```
docker plugin set <plugin> debug=true
```

```
docker plugin enable <plugin>
```

3. Se si esegue il binario stesso sull'host, i registri sono disponibili nell'host `/var/log/netappdvp` elenco. Per abilitare la registrazione del debug, specificare `-debug` quando esegui il plugin.

Suggerimenti generali per la risoluzione dei problemi

- Il problema più comune che si verifica con i nuovi utenti è una configurazione errata che impedisce l'inizializzazione del plugin. Quando ciò accade, è probabile che venga visualizzato un messaggio come questo quando si tenta di installare o abilitare il plugin:

  ```
  Error response from daemon: dial unix /run/docker/plugins/<id>/netapp.sock:
  connect: no such file or directory
  ```

  Ciò significa che il plugin non è riuscito ad avviarsi. Fortunatamente, il plugin è stato creato con una funzionalità di registrazione completa che dovrebbe aiutarti a diagnosticare la maggior parte dei problemi che potresti incontrare.

- Se si verificano problemi con il montaggio di un fotovoltaico su un contenitore, assicurarsi che `rpcbind` è installato e funzionante. Utilizzare il gestore pacchetti richiesto per il sistema operativo host e verificare se `rpcbind` è in esecuzione. È possibile controllare lo stato del servizio rpcbind eseguendo un `systemctl status rpcbind` o il suo equivalente.

Gestisci più istanze Trident

Sono necessarie più istanze di Trident quando si desidera disporre contemporaneamente di più configurazioni di archiviazione. La chiave per più istanze è dare loro nomi diversi usando il `--alias` opzione con il plugin containerizzato, oppure `--volume-driver` opzione durante l'istanziazione Trident sull'host.

Passaggi per il plugin gestito da Docker (versione 1.13/17.03 o successiva)

1. Avviare la prima istanza specificando un alias e un file di configurazione.

   ```
   docker plugin install --grant-all-permissions --alias silver
   netapp/trident-plugin:21.07 config=silver.json
   ```

2. Avviare la seconda istanza, specificando un alias e un file di configurazione diversi.

   ```
   docker plugin install --grant-all-permissions --alias gold
   netapp/trident-plugin:21.07 config=gold.json
   ```

3. Creare volumi specificando l'alias come nome del driver.

   Ad esempio, per il volume dell'oro:

   ```
   docker volume create -d gold --name ntapGold
   ```

   Ad esempio, per il volume d'argento:

```
docker volume create -d silver --name ntapSilver
```

Passaggi per la versione tradizionale (versione 1.12 o precedente)

1. Avvia il plugin con una configurazione NFS utilizzando un ID driver personalizzato:

```
sudo trident --volume-driver=netapp-nas --config=/path/to/config
-nfs.json
```

2. Avviare il plugin con una configurazione iSCSI utilizzando un ID driver personalizzato:

```
sudo trident --volume-driver=netapp-san --config=/path/to/config
-iscsi.json
```

3. Fornire volumi Docker per ogni istanza del driver:

   Ad esempio, per NFS:

```
docker volume create -d netapp-nas --name my_nfs_vol
```

   Ad esempio, per iSCSI:

```
docker volume create -d netapp-san --name my_iscsi_vol
```

Opzioni di configurazione dell'archiviazione

Visualizza le opzioni di configurazione disponibili per le tue configurazioni Trident .

Opzioni di configurazione globali

Queste opzioni di configurazione si applicano a tutte le configurazioni Trident , indipendentemente dalla piattaforma di archiviazione utilizzata.

Opzione	Descrizione	Esempio
`version`	Numero di versione del file di configurazione	`1`

Opzione	Descrizione	Esempio
`storageDriverName`	Nome del driver di archiviazione	`ontap-nas`, `ontap-san` , `ontap-nas-economy` , `ontap-nas-flexgroup` , `solidfire-san`
`storagePrefix`	Prefisso facoltativo per i nomi dei volumi. Predefinito: `netappdvp_` .	`staging_`
`limitVolumeSize`	Restrizione facoltativa sulle dimensioni del volume. Predefinito: "" (non applicato)	`10g`

Non usare `storagePrefix` (incluso quello predefinito) per i backend Element. Per impostazione predefinita, il `solidfire-san` il driver ignorerà questa impostazione e non utilizzerà un prefisso. NetApp consiglia di utilizzare un tenantID specifico per la mappatura del volume Docker oppure di utilizzare i dati degli attributi popolati con la versione di Docker, le informazioni sul driver e il nome non elaborato di Docker nei casi in cui sia stato utilizzato un nome modificato.

Sono disponibili opzioni predefinite per evitare di doverle specificare su ogni volume creato. IL `size` L'opzione è disponibile per tutti i tipi di controller. Per un esempio su come impostare la dimensione predefinita del volume, vedere la sezione Configurazione ONTAP .

Opzione	Descrizione	Esempio
`size`	Dimensione predefinita facoltativa per i nuovi volumi. Predefinito: `1G`	`10G`

Configurazione ONTAP

Oltre ai valori di configurazione globali sopra indicati, quando si utilizza ONTAP, sono disponibili le seguenti opzioni di primo livello.

Opzione	Descrizione	Esempio
`managementLIF`	Indirizzo IP del LIF di gestione ONTAP . È possibile specificare un nome di dominio completo (FQDN).	`10.0.0.1`

Opzione	Descrizione	Esempio
`dataLIF`	Indirizzo IP del protocollo LIF. • Driver ONTAP NAS*: NetApp consiglia di specificare `dataLIF` . Se non specificato, Trident recupera i dataLIF dall'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 dataLIF. • Driver ONTAP SAN*: Non specificare per iSCSI o FC. Usi Trident"Mappa LUN selettiva ONTAP" per scoprire i LIF iSCSI o FC necessari per stabilire una sessione multi-percorso. Viene generato un avviso se `dataLIF` è definito esplicitamente.	`10.0.0.2`
`svm`	Macchina virtuale di archiviazione da utilizzare (obbligatoria se il LIF di gestione è un LIF del cluster)	`svm_nfs`
`username`	Nome utente per connettersi al dispositivo di archiviazione	`vsadmin`
`password`	Password per connettersi al dispositivo di archiviazione	`secret`
`aggregate`	Aggregato per il provisioning (facoltativo; se impostato, deve essere assegnato all'SVM). Per il `ontap-nas-flexgroup` driver, questa opzione viene ignorata. Tutti gli aggregati assegnati all'SVM vengono utilizzati per effettuare il provisioning di un volume FlexGroup .	`aggr1`
`limitAggregateUsage`	Facoltativo, il provisioning fallisce se l'utilizzo è superiore a questa percentuale	`75%`

Opzione	Descrizione	Esempio
`nfsMountOptions`	Controllo dettagliato delle opzioni di montaggio NFS; il valore predefinito è "-o nfsvers=3". **Disponibile solo per il `ontap-nas` E `ontap-nas-economy` autisti**. "Vedi le informazioni sulla configurazione dell'host NFS qui" .	`-o nfsvers=4`
`igroupName`	Trident crea e gestisce per nodo `igroups` COME `netappdvp` . Questo valore non può essere modificato o omesso. **Disponibile solo per il `ontap-san` autista**.	`netappdvp`
`limitVolumeSize`	Dimensione massima del volume richiedibile.	`300g`
`qtreesPerFlexvol`	Il numero massimo di qtree per FlexVol deve essere compreso nell'intervallo [50, 300], il valore predefinito è 200. **Per il `ontap-nas-economy` driver, questa opzione consente di personalizzare il numero massimo di qtree per FlexVol**.	`300`
`sanType`	**Supportato per `ontap-san` solo conducente.** Utilizzare per selezionare `iscsi` per iSCSI, `nvme` per NVMe/TCP o `fcp` per SCSI su Fibre Channel (FC).	`iscsi`se vuoto
`limitVolumePoolSize`	**Supportato per `ontap-san-economy` E `ontap-san-economy` solo conducenti.** Limita le dimensioni FlexVol nei driver ONTAP ontap-nas-economy e ontap-SAN-economy.	`300g`

Sono disponibili opzioni predefinite per evitare di doverle specificare su ogni volume creato:

Opzione	Descrizione	Esempio
`spaceReserve`	Modalità di prenotazione dello spazio; `none` (sottile provisioning) o `volume` (spesso)	`none`

Opzione	Descrizione	Esempio
`snapshotPolicy`	Criterio di snapshot da utilizzare, l'impostazione predefinita è `none`	`none`
`snapshotReserve`	Percentuale di riserva snapshot, il valore predefinito è "" per accettare il valore predefinito ONTAP	`10`
`splitOnClone`	Dividere un clone dal suo genitore al momento della creazione, per impostazione predefinita `false`	`false`
`encryption`	Abilita NetApp Volume Encryption (NVE) sul nuovo volume; il valore predefinito è `false` . Per utilizzare questa opzione, NVE deve essere concesso in licenza e abilitato sul cluster. Se NAE è abilitato sul backend, qualsiasi volume fornito in Trident sarà abilitato per NAE. Per maggiori informazioni, fare riferimento a:"Come funziona Trident con NVE e NAE" .	VERO
`unixPermissions`	Opzione NAS per volumi NFS forniti, impostazione predefinita `777`	`777`
`snapshotDir`	Opzione NAS per l'accesso al `.snapshot` elenco.	"true" per NFSv4 "false" per NFSv3
`exportPolicy`	Opzione NAS da utilizzare per la politica di esportazione NFS, predefinita `default`	`default`
`securityStyle`	Opzione NAS per l'accesso al volume NFS fornito. Supporti NFS `mixed` E `unix` stili di sicurezza. L'impostazione predefinita è `unix` .	`unix`
`fileSystemType`	Opzione SAN per selezionare il tipo di file system, il valore predefinito è `ext4`	`xfs`
`tieringPolicy`	Criterio di tiering da utilizzare, l'impostazione predefinita è `none` .	`none`

Opzioni di ridimensionamento

IL `ontap-nas` E `ontap-san` i driver creano un ONTAP FlexVol per ogni volume Docker. ONTAP supporta fino a 1000 FlexVol per nodo del cluster con un massimo di 12.000 volumi FlexVol per cluster. Se i requisiti del volume Docker rientrano in tale limitazione, `ontap-nas` driver è la soluzione NAS preferita grazie alle funzionalità aggiuntive offerte da FlexVols, come gli snapshot Docker-volume-granular e la clonazione.

Se hai bisogno di più volumi Docker di quelli che possono essere gestiti dai limiti FlexVol , scegli `ontap-nas-`

`economy` o il `ontap-san-economy` autista.

IL `ontap-nas-economy` il driver crea volumi Docker come Qtree ONTAP all'interno di un pool di volumi FlexVol gestiti automaticamente. I Qtree offrono una scalabilità molto maggiore, fino a 100.000 per nodo del cluster e 2.400.000 per cluster, a scapito di alcune funzionalità. IL `ontap-nas-economy` il driver non supporta snapshot Docker-volume-granular o clonazione.

> IL `ontap-nas-economy` Il driver non è attualmente supportato in Docker Swarm, perché Docker Swarm non orchestra la creazione di volumi su più nodi.

IL `ontap-san-economy` il driver crea volumi Docker come LUN ONTAP all'interno di un pool condiviso di volumi FlexVol gestiti automaticamente. In questo modo, ogni FlexVol non è limitato a una sola LUN e offre una migliore scalabilità per i carichi di lavoro SAN. A seconda dell'array di archiviazione, ONTAP supporta fino a 16384 LUN per cluster. Poiché i volumi sono LUN sottostanti, questo driver supporta snapshot e clonazione Docker-volume-granular.

Scegli il `ontap-nas-flexgroup` driver per aumentare il parallelismo su un singolo volume che può crescere fino a raggiungere l'ordine dei petabyte con miliardi di file. Alcuni casi d'uso ideali per FlexGroups includono AI/ML/DL, big data e analisi, build di software, streaming, repository di file e così via. Trident utilizza tutti gli aggregati assegnati a una SVM durante il provisioning di un volume FlexGroup . Il supporto FlexGroup in Trident deve tenere conto anche delle seguenti considerazioni:

- Richiede ONTAP versione 9.2 o successiva.
- Al momento in cui scriviamo, FlexGroups supporta solo NFS v3.
- Si consiglia di abilitare gli identificatori NFSv3 a 64 bit per SVM.
- La dimensione minima consigliata per il membro/volume FlexGroup è 100 GiB.
- La clonazione non è supportata per i volumi FlexGroup .

Per informazioni su FlexGroups e carichi di lavoro appropriati per FlexGroups, fare riferimento a "Guida alle best practice e all'implementazione del volume NetApp FlexGroup" .

Per ottenere funzionalità avanzate e su larga scala nello stesso ambiente, è possibile eseguire più istanze del Docker Volume Plugin, una delle quali utilizzando `ontap-nas` e un altro usando `ontap-nas-economy` .

Ruolo ONTAP personalizzato per Trident

È possibile creare un ruolo cluster ONTAP con privilegi minimi in modo da non dover utilizzare il ruolo di amministratore ONTAP per eseguire 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.

Fare riferimento a"Generatore di ruoli personalizzati Trident" per ulteriori informazioni sulla creazione di ruoli personalizzati Trident .

Utilizzo di ONTAP CLI

1. Crea un nuovo ruolo utilizzando il seguente comando:

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

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

3. Assegna 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

Eseguire i seguenti passaggi in ONTAP System Manager:

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 SVM, selezionare **Archiviazione > VM di archiviazione > `required SVM` > Impostazioni > Utenti e ruoli**.

   b. Selezionare l'icona della freccia (→) accanto a **Utenti e ruoli**.
   c. Selezionare **+Aggiungi** in **Ruoli**.
   d. Definisci le regole per il ruolo e clicca su **Salva**.
2. **Assegnare il ruolo all'utente Trident *: + Eseguire i seguenti passaggi nella pagina *Utenti e ruoli**:
   a. Selezionare Aggiungi icona **+** in **Utenti**.
   b. Selezionare il nome utente richiesto e selezionare un ruolo nel menu a discesa per **Ruolo**.
   c. Fare clic su **Salva**.

Per maggiori informazioni consultare le seguenti pagine:

- "Ruoli personalizzati per l'amministrazione di ONTAP"O"Definisci ruoli personalizzati"
- "Lavorare con ruoli e utenti"

Esempio di file di configurazione ONTAP

Esempio NFS per il driver <code>ontap-nas</code>

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "defaults": {
    "size": "10G",
    "spaceReserve": "none",
    "exportPolicy": "default"
  }
}
```

Esempio NFS per il driver <code>ontap-nas-flexgroup</code>

```
{
  "version": 1,
  "storageDriverName": "ontap-nas-flexgroup",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "defaults": {
    "size": "100G",
    "spaceReserve": "none",
    "exportPolicy": "default"
  }
}
```

Esempio NFS per il driver <code>ontap-nas-economy</code>

```
{
  "version": 1,
  "storageDriverName": "ontap-nas-economy",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1"
}
```

Esempio iSCSI per il driver <code>ontap-san</code>

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.3",
  "svm": "svm_iscsi",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "igroupName": "netappdvp"
}
```

Esempio NFS per il driver <code>ontap-san-economy</code>

```
{
  "version": 1,
  "storageDriverName": "ontap-san-economy",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.3",
  "svm": "svm_iscsi_eco",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "igroupName": "netappdvp"
}
```

Esempio NVMe/TCP per il driver <code>ontap-san</code>

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

Esempio SCSI su FC per il driver <code>ontap-san</code>

```
{
  "version": 1,
  "backendName": "ontap-san-backend",
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "sanType": "fcp",
  "svm": "trident_svm",
  "username": "vsadmin",
  "password": "password",
  "useREST": true
}
```

Configurazione del software Element

Oltre ai valori di configurazione globali, quando si utilizza il software Element (NetApp HCI/ SolidFire), sono disponibili queste opzioni.

Opzione	Descrizione	Esempio
`Endpoint`	https://<login>:<password>@<mvip>/json-rpc/<versione-elemento>	https://admin:admin@192.168.160.3/json-rpc/8.0
`SVIP`	Indirizzo IP e porta iSCSI	10.0.0.7:3260
`TenantName`	Tenant SolidFireF da utilizzare (creato se non trovato)	`docker`

Opzione	Descrizione	Esempio
`InitiatorIFace`	Specificare l'interfaccia quando si limita il traffico iSCSI all'interfaccia non predefinita	`default`
`Types`	Specifiche QoS	Vedi esempio qui sotto
`LegacyNamePrefix`	Prefisso per le installazioni Trident aggiornate. Se hai utilizzato una versione di Trident precedente alla 1.3.2 e esegui un aggiornamento con volumi esistenti, dovrai impostare questo valore per accedere ai vecchi volumi mappati tramite il metodo volume-name.	`netappdvp-`

IL `solidfire-san` il driver non supporta Docker Swarm.

File di configurazione del software Element di esempio

```
{
  "version": 1,
  "storageDriverName": "solidfire-san",
  "Endpoint": "https://admin:admin@192.168.160.3/json-rpc/8.0",
  "SVIP": "10.0.0.7:3260",
  "TenantName": "docker",
  "InitiatorIFace": "default",
  "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
      }
    }
  ]
}
```

Problemi noti e limitazioni

Trova informazioni sui problemi noti e sulle limitazioni quando si utilizza Trident con Docker.

L'aggiornamento del plugin Trident Docker Volume alla versione 20.10 e successive da versioni precedenti provoca un errore di aggiornamento con l'errore "Nessun file o directory presente".

Soluzione alternativa

1. Disattivare il plugin.

```
docker plugin disable -f netapp:latest
```

2. Rimuovere il plugin.

```
docker plugin rm -f netapp:latest
```

3. Reinstallare il plugin fornendo l'extra `config` parametro.

```
docker plugin install netapp/trident-plugin:20.10 --alias netapp --grant
-all-permissions config=config.json
```

I nomi dei volumi devono essere lunghi almeno 2 caratteri.

Questa è una limitazione del client Docker. Il client interpreterà il nome di un singolo carattere come un percorso Windows. "Vedi bug 25773" .

Docker Swarm presenta determinati comportamenti che impediscono a Trident di supportarlo con ogni combinazione di storage e driver.

- Attualmente Docker Swarm utilizza il nome del volume anziché l'ID del volume come identificatore univoco del volume.
- Le richieste di volume vengono inviate simultaneamente a ciascun nodo in un cluster Swarm.
- I plugin del volume (incluso Trident) devono essere eseguiti in modo indipendente su ciascun nodo in un cluster Swarm. A causa del modo in cui funziona ONTAP e del modo in cui `ontap-nas` E `ontap-san` i driver funzionano, sono gli unici che riescono a operare entro questi limiti.

Gli altri driver sono soggetti a problemi come condizioni di gara che possono comportare la creazione di un gran numero di volumi per una singola richiesta senza un chiaro "vincitore"; ad esempio, Element ha una funzionalità che consente ai volumi di avere lo stesso nome ma ID diversi.

NetApp ha fornito feedback al team Docker, ma non ha alcuna indicazione di possibili ricorsi futuri.

Se si sta predisponendo un FlexGroup , ONTAP non predispone un secondo FlexGroup se il secondo FlexGroup ha uno o più aggregati in comune con il FlexGroup in fase di provisioning.

Buone pratiche e raccomandazioni

Distribuzione

Quando distribuisci Trident, utilizza i consigli elencati qui.

Distribuisci in uno spazio dei nomi dedicato

"Spazi dei nomi"forniscono separazione amministrativa tra diverse applicazioni e rappresentano un ostacolo alla condivisione delle risorse. Ad esempio, un PVC di uno spazio dei nomi non può essere utilizzato da un altro. Trident fornisce risorse PV a tutti gli spazi dei nomi nel cluster Kubernetes e di conseguenza sfrutta un account di servizio con privilegi elevati.

Inoltre, l'accesso al pod Trident potrebbe consentire a un utente di accedere alle credenziali del sistema di archiviazione e ad altre informazioni sensibili. È importante assicurarsi che gli utenti dell'applicazione e le applicazioni di gestione non abbiano la possibilità di accedere alle definizioni degli oggetti Trident o ai pod stessi.

Utilizzare quote e limiti di intervallo per controllare il consumo di spazio di archiviazione

Kubernetes ha due funzionalità che, se combinate, forniscono un potente meccanismo per limitare il consumo di risorse da parte delle applicazioni. IL "meccanismo di quota di archiviazione" consente all'amministratore di implementare limiti di consumo di capacità e numero di oggetti globali e specifici per classe di archiviazione, in base allo spazio dei nomi. Inoltre, utilizzando un "limite di portata" garantisce che le richieste PVC rientrino in un valore minimo e massimo prima che la richiesta venga inoltrata al fornitore.

Questi valori sono definiti per ogni namespace, il che significa che per ogni namespace devono essere definiti valori in linea con i relativi requisiti di risorse. Vedi qui per informazioni su "come sfruttare le quote" .

Configurazione di archiviazione

Ogni piattaforma di storage nel portfolio NetApp ha funzionalità uniche che avvantaggiano le applicazioni, containerizzate o meno.

Panoramica della piattaforma

Trident funziona con ONTAP ed Element. Non esiste una piattaforma più adatta di un'altra a tutte le applicazioni e a tutti gli scenari; tuttavia, quando si sceglie una piattaforma, è necessario tenere conto delle esigenze dell'applicazione e del team che amministra il dispositivo.

Dovresti seguire le best practice di base per il sistema operativo host con il protocollo che stai sfruttando. Facoltativamente, potresti prendere in considerazione l'integrazione delle best practice dell'applicazione, quando disponibili, con le impostazioni backend, classe di archiviazione e PVC per ottimizzare l'archiviazione per applicazioni specifiche.

ONTAP e Cloud Volumes ONTAP

Scopri le best practice per la configurazione ONTAP e Cloud Volumes ONTAP per Trident.

Le seguenti raccomandazioni sono linee guida per la configurazione ONTAP per carichi di lavoro

containerizzati, che consumano volumi forniti dinamicamente da Trident. Ciascuna soluzione dovrebbe essere presa in considerazione e valutata per verificarne l'adeguatezza al proprio ambiente.

Utilizzare SVM dedicati a Trident

Le macchine virtuali di archiviazione (SVM) garantiscono isolamento e separazione amministrativa tra i tenant su un sistema ONTAP . Dedicare una SVM alle applicazioni consente la delega dei privilegi e consente di applicare le best practice per limitare il consumo di risorse.

Sono disponibili diverse opzioni per la gestione dell'SVM:

- Fornire l'interfaccia di gestione del cluster nella configurazione backend, insieme alle credenziali appropriate, e specificare il nome SVM.
- Creare un'interfaccia di gestione dedicata per l'SVM utilizzando ONTAP System Manager o la CLI.
- Condividere il ruolo di gestione con un'interfaccia dati NFS.

In ogni caso, l'interfaccia dovrebbe essere nel DNS e il nome DNS dovrebbe essere utilizzato durante la configurazione Trident. Ciò semplifica alcuni scenari DR, ad esempio SVM-DR senza l'utilizzo della conservazione dell'identità di rete.

Non esiste alcuna preferenza tra un LIF di gestione dedicato o condiviso per l'SVM; tuttavia, è necessario assicurarsi che le policy di sicurezza della rete siano allineate all'approccio scelto. In ogni caso, la gestione LIF dovrebbe essere accessibile tramite DNS per facilitare la massima flessibilità. "SVM-DR" essere utilizzato insieme a Trident.

Limita il conteggio massimo del volume

I sistemi di archiviazione ONTAP hanno un numero massimo di volumi, che varia in base alla versione del software e alla piattaforma hardware. Fare riferimento a "Hardware Universe NetApp" per la tua piattaforma specifica e la versione ONTAP per determinare i limiti esatti. Quando il conteggio del volume è esaurito, le operazioni di provisioning falliscono non solo per Trident, ma per tutte le richieste di archiviazione.

Trident's `ontap-nas` E `ontap-san` i driver forniscono un FlexVolume per ogni Kubernetes Persistent Volume (PV) creato. IL `ontap-nas-economy` il driver crea circa un FlexVolume ogni 200 PV (configurabile tra 50 e 300). IL `ontap-san-economy` il driver crea circa un FlexVolume ogni 100 PV (configurabile tra 50 e 200). Per impedire a Trident di consumare tutti i volumi disponibili sul sistema di archiviazione, è necessario impostare un limite sulla SVM. Puoi farlo dalla riga di comando:

```
vserver modify -vserver <svm_name> -max-volumes <num_of_volumes>
```

Il valore per `max-volumes` varia in base a diversi criteri specifici del tuo ambiente:

- Il numero di volumi esistenti nel cluster ONTAP
- Il numero di volumi che si prevede di fornire al di fuori di Trident per altre applicazioni
- Numero di volumi persistenti che si prevede saranno consumati dalle applicazioni Kubernetes

IL `max-volumes` Il valore è il totale dei volumi forniti su tutti i nodi del cluster ONTAP e non su un singolo nodo ONTAP . Di conseguenza, potrebbero verificarsi alcune condizioni in cui un nodo del cluster ONTAP potrebbe avere molti più o meno volumi Trident forniti rispetto a un altro nodo.

Ad esempio, un cluster ONTAP a due nodi ha la capacità di ospitare un massimo di 2000 volumi FlexVol .

Impostare il conteggio massimo del volume su 1250 sembra molto ragionevole. Tuttavia, se solo "aggregati" da un nodo vengono assegnati all'SVM oppure non è possibile eseguire il provisioning degli aggregati assegnati da un nodo (ad esempio, a causa della capacità), quindi l'altro nodo diventa la destinazione per tutti i volumi Trident provisionati. Ciò significa che il limite del volume potrebbe essere raggiunto per quel nodo prima del `max-volumes` viene raggiunto il valore, con conseguente impatto sia su Trident che su altre operazioni di volume che utilizzano quel nodo. **È possibile evitare questa situazione assicurandosi che gli aggregati di ciascun nodo del cluster vengano assegnati in numero uguale all'SVM utilizzato da Trident .**

Clonare un volume

NetApp Trident supporta la clonazione dei volumi quando si utilizza `ontap-nas` , `ontap-san` , `solidfire-san` , E `gcp-cvs` driver di archiviazione. Quando si utilizza il `ontap-nas-flexgroup` O `ontap-nas-economy` driver, la clonazione non è supportata. La creazione di un nuovo volume da un volume esistente comporterà la creazione di un nuovo snapshot.

> Evitare di clonare un PVC associato a una StorageClass diversa. Eseguire operazioni di clonazione all'interno della stessa StorageClass per garantire la compatibilità ed evitare comportamenti imprevisti.

Limita la dimensione massima dei volumi creati da Trident

Per configurare la dimensione massima dei volumi che possono essere creati da Trident, utilizzare `limitVolumeSize` parametro nel tuo `backend.json` definizione.

Oltre a controllare le dimensioni del volume nell'array di archiviazione, dovresti anche sfruttare le funzionalità di Kubernetes.

Limita la dimensione massima dei FlexVol creati da Trident

Per configurare la dimensione massima per FlexVols utilizzati come pool per i driver ontap-san-economy e ontap-nas-economy, utilizzare `limitVolumePoolSize` parametro nel tuo `backend.json` definizione.

Configurare Trident per utilizzare CHAP bidirezionale

È possibile specificare i nomi utente e le password dell'iniziatore CHAP e del target nella definizione del backend e fare in modo che Trident abiliti CHAP sull'SVM. Utilizzando il `useCHAP` parametro nella configurazione del backend, Trident autentica le connessioni iSCSI per i backend ONTAP con CHAP.

Creare e utilizzare una policy QoS SVM

L'utilizzo di una policy QoS ONTAP applicata all'SVM limita il numero di IOPS utilizzabili dai volumi Trident forniti. Questo aiuta a "prevenire un bullo" o un contenitore fuori controllo che influisca sui carichi di lavoro esterni alla Trident SVM.

È possibile creare una policy QoS per l'SVM in pochi passaggi. Per informazioni più precise, consultare la documentazione relativa alla propria versione di ONTAP . L'esempio seguente crea una policy QoS che limita il totale di IOPS disponibili per l'SVM a 5000.

```
# create the policy group for the SVM
qos policy-group create -policy-group <policy_name> -vserver <svm_name>
-max-throughput 5000iops

# assign the policy group to the SVM, note this will not work
# if volumes or files in the SVM have existing QoS policies
vserver modify -vserver <svm_name> -qos-policy-group <policy_name>
```

Inoltre, se la tua versione di ONTAP lo supporta, puoi prendere in considerazione l'utilizzo di un QoS minimo per garantire una certa quantità di throughput ai carichi di lavoro containerizzati. La QoS adattiva non è compatibile con una policy a livello SVM.

Il numero di IOPS dedicati ai carichi di lavoro containerizzati dipende da molti aspetti. Tra le altre cose, queste includono:

- Altri carichi di lavoro che utilizzano l'array di archiviazione. Se sono presenti altri carichi di lavoro, non correlati alla distribuzione di Kubernetes, che utilizzano le risorse di storage, è necessario prestare attenzione per garantire che tali carichi di lavoro non subiscano accidentalmente un impatto negativo.
- Carichi di lavoro previsti in esecuzione nei container. Se i carichi di lavoro con elevati requisiti IOPS vengono eseguiti nei container, una politica QoS bassa si traduce in un'esperienza negativa.

È importante ricordare che una policy QoS assegnata a livello SVM fa sì che tutti i volumi forniti alla SVM condividano lo stesso pool IOPS. Se una o un numero limitato di applicazioni containerizzate ha un elevato requisito di IOPS, potrebbe diventare un ostacolo per gli altri carichi di lavoro containerizzati. In tal caso, potresti prendere in considerazione l'utilizzo di un'automazione esterna per assegnare policy QoS per volume.

Dovresti assegnare il gruppo di policy QoS all'SVM **solo** se la versione ONTAP è precedente alla 9.8.

Creare gruppi di policy QoS per Trident

La qualità del servizio (QoS) garantisce che le prestazioni dei carichi di lavoro critici non vengano compromesse da carichi di lavoro concorrenti. I gruppi di policy QoS ONTAP forniscono opzioni QoS per i volumi e consentono agli utenti di definire il limite di throughput per uno o più carichi di lavoro. Per ulteriori informazioni su QoS, fare riferimento a "Garantire la produttività con QoS" . È possibile specificare gruppi di policy QoS nel backend o in un pool di archiviazione, che verranno applicati a ciascun volume creato in quel pool o backend.

ONTAP dispone di due tipi di gruppi di policy QoS: tradizionali e adattivi. I gruppi di policy tradizionali forniscono un throughput massimo fisso (o minimo, nelle versioni successive) in IOPS. La QoS adattiva adatta automaticamente la velocità di elaborazione alle dimensioni del carico di lavoro, mantenendo il rapporto tra IOPS e TB|GB al variare delle dimensioni del carico di lavoro. Ciò rappresenta un vantaggio significativo quando si gestiscono centinaia o migliaia di carichi di lavoro in una distribuzione di grandi dimensioni.

Quando si creano gruppi di policy QoS, tenere presente quanto segue:

- Dovresti impostare il `qosPolicy` inserisci la chiave `defaults` blocco della configurazione del backend. Vedere il seguente esempio di configurazione del backend:

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 0.0.0.0
dataLIF: 0.0.0.0
svm: svm0
username: user
password: pass
defaults:
  qosPolicy: standard-pg
storage:
  - labels:
      performance: extreme
    defaults:
      adaptiveQosPolicy: extremely-adaptive-pg
  - labels:
      performance: premium
    defaults:
      qosPolicy: premium-pg
```

- È necessario applicare i gruppi di policy per volume, in modo che ogni volume ottenga l'intera produttività specificata dal gruppo di policy. I gruppi di policy condivisi non sono supportati.

Per ulteriori informazioni sui gruppi di policy QoS, fare riferimento a "Riferimento al comando ONTAP" .

Limita l'accesso alle risorse di archiviazione ai membri del cluster Kubernetes

Limitare l'accesso ai volumi NFS, ai LUN iSCSI e ai LUN FC creati da Trident è un componente fondamentale della strategia di sicurezza per la distribuzione di Kubernetes. In questo modo si impedisce agli host che non fanno parte del cluster Kubernetes di accedere ai volumi e di modificare potenzialmente i dati in modo imprevisto.

È importante comprendere che gli spazi dei nomi rappresentano il confine logico per le risorse in Kubernetes. Si presuppone che le risorse nello stesso namespace possano essere condivise, ma, cosa importante, non esiste alcuna funzionalità cross-namespace. Ciò significa che, anche se i PV sono oggetti globali, quando sono associati a un PVC sono accessibili solo ai pod che si trovano nello stesso namespace. **È fondamentale garantire che gli spazi dei nomi vengano utilizzati per fornire la separazione quando appropriato.**

La preoccupazione principale per la maggior parte delle organizzazioni in merito alla sicurezza dei dati in un contesto Kubernetes è che un processo in un contenitore possa accedere a uno storage montato sull'host, ma che non è destinato al contenitore. "Spazi dei nomi" sono progettati per impedire questo tipo di compromissione. Esiste però un'eccezione: i contenitori privilegiati.

Un contenitore privilegiato è un contenitore che viene eseguito con autorizzazioni a livello di host sostanzialmente superiori al normale. Questi non vengono negati per impostazione predefinita, quindi assicurati di disabilitare la funzionalità utilizzando "politiche di sicurezza del pod" .

Per i volumi in cui è richiesto l'accesso sia da Kubernetes che da host esterni, lo storage dovrebbe essere gestito in modo tradizionale, con il PV introdotto dall'amministratore e non gestito da Trident. Ciò garantisce che il volume di archiviazione venga distrutto solo quando sia Kubernetes sia gli host esterni si sono

disconnessi e non utilizzano più il volume. Inoltre, è possibile applicare una policy di esportazione personalizzata, che consente l'accesso dai nodi del cluster Kubernetes e dai server di destinazione esterni al cluster Kubernetes.

Per le distribuzioni che dispongono di nodi infrastrutturali dedicati (ad esempio, OpenShift) o altri nodi che non sono in grado di pianificare le applicazioni utente, è opportuno utilizzare criteri di esportazione separati per limitare ulteriormente l'accesso alle risorse di archiviazione. Ciò include la creazione di una policy di esportazione per i servizi distribuiti su tali nodi infrastrutturali (ad esempio, i servizi OpenShift Metrics e Logging) e per le applicazioni standard distribuite su nodi non infrastrutturali.

Utilizzare una politica di esportazione dedicata

È necessario assicurarsi che esista una policy di esportazione per ciascun backend che consenta l'accesso solo ai nodi presenti nel cluster Kubernetes. Trident può creare e gestire automaticamente le policy di esportazione. In questo modo, Trident limita l'accesso ai volumi che fornisce ai nodi nel cluster Kubernetes e semplifica l'aggiunta/eliminazione dei nodi.

In alternativa, puoi anche creare manualmente una policy di esportazione e popolarla con una o più regole di esportazione che elaborano ogni richiesta di accesso al nodo:

- Utilizzare il `vserver export-policy create` Comando CLI ONTAP per creare la policy di esportazione.
- Aggiungere regole alla policy di esportazione utilizzando `vserver export-policy rule create` Comando CLI ONTAP .

L'esecuzione di questi comandi consente di limitare i nodi Kubernetes che hanno accesso ai dati.

Disabilitare `showmount` per l'applicazione SVM

IL `showmount` La funzionalità consente a un client NFS di interrogare l'SVM per ottenere un elenco delle esportazioni NFS disponibili. Un pod distribuito nel cluster Kubernetes può emettere `showmount -e` comando contro e ricevere un elenco delle cavalcature disponibili, comprese quelle a cui non ha accesso. Sebbene questo, di per sé, non costituisca una compromissione della sicurezza, fornisce informazioni non necessarie che potrebbero potenzialmente aiutare un utente non autorizzato a connettersi a un'esportazione NFS.

Dovresti disabilitare `showmount` utilizzando il comando CLI ONTAP a livello SVM:

```
vserver nfs modify -vserver <svm_name> -showmount disabled
```

Le migliori pratiche SolidFire

Scopri le best practice per configurare l'archiviazione SolidFire per Trident.

Crea un account Solidfire

Ogni account SolidFire rappresenta un proprietario di volume univoco e riceve il proprio set di credenziali Challenge-Handshake Authentication Protocol (CHAP). È possibile accedere ai volumi assegnati a un account utilizzando il nome dell'account e le relative credenziali CHAP oppure tramite un gruppo di accesso al volume. A un account possono essere assegnati fino a duemila volumi, ma un volume può appartenere a un solo account.

Creare una policy QoS

Utilizzare i criteri di qualità del servizio (QoS) SolidFire se si desidera creare e salvare un'impostazione di qualità del servizio standardizzata che può essere applicata a molti volumi.

È possibile impostare i parametri QoS in base al volume. Le prestazioni per ciascun volume possono essere garantite impostando tre parametri configurabili che definiscono la QoS: Min IOPS, Max IOPS e Burst IOPS.

Ecco i possibili valori IOPS minimi, massimi e burst per la dimensione del blocco da 4 Kb.

Parametro IOPS	Definizione	Valore minimo	Valore predefinito	Valore massimo (4Kb)
IOPS minimi	Il livello di prestazioni garantito per un volume.	50	50	15000
IOPS massimi	Le prestazioni non supereranno questo limite.	50	15000	200.000
IOPS a raffica	IOPS massimo consentito in uno scenario di burst breve.	50	15000	200.000

Sebbene i valori Max IOPS e Burst IOPS possano essere impostati fino a 200.000, le prestazioni massime reali di un volume sono limitate dall'utilizzo del cluster e dalle prestazioni per nodo.

La dimensione del blocco e la larghezza di banda hanno un'influenza diretta sul numero di IOPS. Con l'aumentare delle dimensioni dei blocchi, il sistema aumenta la larghezza di banda fino al livello necessario per elaborare blocchi di dimensioni maggiori. Con l'aumentare della larghezza di banda, diminuisce il numero di IOPS che il sistema è in grado di raggiungere. Fare riferimento a "Qualità del servizio SolidFire" per maggiori informazioni su QoS e prestazioni.

Autenticazione SolidFire

Element supporta due metodi di autenticazione: CHAP e Volume Access Groups (VAG). CHAP utilizza il protocollo CHAP per autenticare l'host al backend. Volume Access Groups controlla l'accesso ai volumi di cui si occupa. NetApp consiglia di utilizzare CHAP per l'autenticazione poiché è più semplice e non presenta limiti di scalabilità.

Trident con il provisioner CSI avanzato supporta l'uso dell'autenticazione CHAP. I VAG dovrebbero essere utilizzati solo nella tradizionale modalità di funzionamento non-CSI.

L'autenticazione CHAP (verifica che l'iniziatore sia l'utente del volume previsto) è supportata solo con il controllo degli accessi basato sull'account. Se si utilizza CHAP per l'autenticazione, sono disponibili due opzioni: CHAP unidirezionale e CHAP bidirezionale. Il CHAP unidirezionale autentica l'accesso al volume utilizzando il nome dell'account SolidFire e il segreto dell'iniziatore. L'opzione CHAP bidirezionale fornisce il modo più sicuro per autenticare il volume, perché il volume autentica l'host tramite il nome dell'account e il segreto dell'iniziatore, e quindi l'host autentica il volume tramite il nome dell'account e il segreto di

destinazione.

Tuttavia, se CHAP non può essere abilitato e sono necessari i VAG, creare il gruppo di accesso e aggiungere gli iniziatori host e i volumi al gruppo di accesso. Ogni IQN aggiunto a un gruppo di accesso può accedere a ciascun volume del gruppo con o senza autenticazione CHAP. Se l'iniziatore iSCSI è configurato per utilizzare l'autenticazione CHAP, viene utilizzato il controllo degli accessi basato sull'account. Se l'iniziatore iSCSI non è configurato per utilizzare l'autenticazione CHAP, viene utilizzato il controllo di accesso Volume Access Group.

Dove trovare maggiori informazioni?

Di seguito è riportata la documentazione di alcune delle migliori pratiche. Cerca il "Libreria NetApp" per le versioni più recenti.

- ONTAP*
- "Guida alle migliori pratiche e all'implementazione di NFS"
- "Amministrazione SAN"(per iSCSI)
- "Configurazione iSCSI Express per RHEL"

Software elementare

- "Configurazione SolidFire per Linux"
- NetApp HCI*
- "Prerequisiti per la distribuzione di NetApp HCI"
- "Accedi al motore di distribuzione NetApp"

Informazioni sulle migliori pratiche applicative

- "Le migliori pratiche per MySQL su ONTAP"
- "Best practice per MySQL su SolidFire"
- "NetApp SolidFire e Cassandra"
- "Le migliori pratiche di Oracle su SolidFire"
- "Le migliori pratiche di PostgreSQL su SolidFire"

Non tutte le applicazioni hanno linee guida specifiche, è importante lavorare con il tuo team NetApp e utilizzare le "Libreria NetApp" per trovare la documentazione più aggiornata.

Integra Trident

Per integrare Trident, è necessario integrare i seguenti elementi di progettazione e architettura: selezione e distribuzione del driver, progettazione della classe di archiviazione, progettazione del pool virtuale, impatto del Persistent Volume Claim (PVC) sul provisioning dell'archiviazione, operazioni sui volumi e distribuzione dei servizi OpenShift tramite Trident.

Selezione e distribuzione del driver

Seleziona e distribuisci un driver backend per il tuo sistema di archiviazione.

Driver backend ONTAP

I driver backend ONTAP si differenziano in base al protocollo utilizzato e al modo in cui i volumi vengono forniti sul sistema di archiviazione. Pertanto, è opportuno valutare attentamente la scelta del driver da implementare.

A un livello superiore, se l'applicazione ha componenti che necessitano di storage condiviso (più pod che accedono allo stesso PVC), i driver basati su NAS sarebbero la scelta predefinita, mentre i driver iSCSI basati su blocchi soddisfano le esigenze di storage non condiviso. Scegliere il protocollo in base ai requisiti dell'applicazione e al livello di comfort dei team di storage e infrastruttura. In generale, per la maggior parte delle applicazioni c'è poca differenza tra loro, quindi spesso la decisione si basa sulla necessità o meno di uno storage condiviso (in cui più di un pod necessita di accesso simultaneo).

I driver backend ONTAP disponibili sono:

- `ontap-nas`: Ogni PV fornito è un FlexVolume ONTAP completo.
- `ontap-nas-economy`: Ogni PV fornito è un qtree, con un numero configurabile di qtree per FlexVolume (il valore predefinito è 200).
- `ontap-nas-flexgroup`: Ogni PV è predisposto come un ONTAP FlexGroup completo e tutti gli aggregati assegnati a una SVM vengono utilizzati.
- `ontap-san`: Ogni PV fornito è una LUN all'interno del proprio FlexVolume.
- `ontap-san-economy`: Ogni PV fornito è una LUN, con un numero configurabile di LUN per FlexVolume (il valore predefinito è 100).

La scelta tra i tre driver NAS ha alcune implicazioni sulle funzionalità rese disponibili all'applicazione.

Si noti che nelle tabelle seguenti non tutte le funzionalità sono esposte tramite Trident. Alcune devono essere applicate dall'amministratore dell'archiviazione dopo il provisioning, se si desidera tale funzionalità. Le note a piè di pagina in apice distinguono la funzionalità per caratteristica e driver.

Driver NAS ONTAP	Istantanee	Cloni	Politiche di esportazione dinamiche	Multi-attacco	Qualità del servizio	Ridimensionare	Replicazione
`ontap-nas`	SÌ	SÌ	Sìnota a piè di pagina:5[]	SÌ	Sìnota a piè di pagina:1[]	SÌ	Sìnota a piè di pagina:1[]
`ontap-nas-economy`	NO [3]	NO [3]	Sìnota a piè di pagina:5[]	SÌ	NO [3]	SÌ	NO [3]
`ontap-nas-flexgroup`	Sìnota a piè di pagina:1[]	NO	Sìnota a piè di pagina:5[]	SÌ	Sìnota a piè di pagina:1[]	SÌ	Sìnota a piè di pagina:1[]

Trident offre 2 driver SAN per ONTAP, le cui capacità sono illustrate di seguito.

Driver ONTAP SAN	Istantanee	Cloni	Multi-attacco	CHAP bidirezionale	Qualità del servizio	Ridimensionare	Replicazione
`ontap-san`	SÌ	SÌ	Sìnota a piè di pagina:4[]	SÌ	Sìnota a piè di pagina:1[]	SÌ	Sìnota a piè di pagina:1[]
`ontap-san-economy`	SÌ	SÌ	Sìnota a piè di pagina:4[]	SÌ	NO [3]	SÌ	NO [3]

> Nota a piè di pagina per le tabelle sopra: Sì [1]: Non gestito da Trident Sì [2]: Gestito da Trident, ma non granulare PV NO [3]: Non gestito da Trident e non granulare PV Sì [4]: Supportato per volumi raw-block Sì [5]: Supportato da Trident

Le funzionalità che non sono granulari PV vengono applicate all'intero FlexVolume e tutti i PV (ovvero qtree o LUN nei FlexVol condivisi) condivideranno una pianificazione comune.

Come possiamo vedere nelle tabelle sopra, gran parte della funzionalità tra `ontap-nas` E `ontap-nas-economy` è lo stesso. Tuttavia, poiché il `ontap-nas-economy` il driver limita la possibilità di controllare la pianificazione con granularità per PV, il che può influire in particolare sulla pianificazione del disaster recovery e del backup. Per i team di sviluppo che desiderano sfruttare la funzionalità di clonazione PVC sullo storage ONTAP , ciò è possibile solo utilizzando `ontap-nas` , `ontap-san` O `ontap-san-economy` conducenti.

IL `solidfire-san` Il driver è anche in grado di clonare i PVC.

Driver backend Cloud Volumes ONTAP

Cloud Volumes ONTAP fornisce il controllo dei dati insieme a funzionalità di archiviazione di livello aziendale per vari casi d'uso, tra cui condivisioni di file e archiviazione a livello di blocco che servono protocolli NAS e SAN (NFS, SMB/CIFS e iSCSI). I driver compatibili per Cloud Volume ONTAP sono `ontap-nas` , `ontap-nas-economy` , `ontap-san` E `ontap-san-economy` . Sono applicabili a Cloud Volume ONTAP per Azure, Cloud Volume ONTAP per GCP.

Driver backend Amazon FSx per ONTAP

Amazon FSx for NetApp ONTAP ti consente di sfruttare le funzionalità, le prestazioni e le capacità amministrative 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 numerose funzionalità del file system ONTAP e API di amministrazione. I driver compatibili per Cloud Volume ONTAP sono `ontap-nas` , `ontap-nas-economy` , `ontap-nas-flexgroup` , `ontap-san` E `ontap-san-economy` .

Driver backend NetApp HCI/ SolidFire

IL `solidfire-san` driver utilizzato con le piattaforme NetApp HCI/ SolidFire , aiuta l'amministratore a configurare un backend Element per Trident in base ai limiti QoS. Se desideri progettare il tuo backend per impostare i limiti QoS specifici sui volumi forniti da Trident, usa `type` parametro nel file backend. L'amministratore può anche limitare la dimensione del volume che può essere creato sullo storage utilizzando `limitVolumeSize` parametro. Attualmente, le funzionalità di archiviazione Element come il ridimensionamento del volume e la replica del volume non sono supportate tramite `solidfire-san` autista. Queste operazioni devono essere eseguite manualmente tramite l'interfaccia utente web di Element Software.

Driver SolidFire	Istantanee	Cloni	Multi-attacco	CAPITOLO	Qualità del servizio	Ridimensionare	Replicazione
`solidfire-san`	SÌ	SÌ	Sìnota a piè di pagina:2[]	SÌ	SÌ	SÌ	Sìnota a piè di pagina:1[]

> Nota a piè di pagina: Sì [1]: Non gestito da Trident Sì [2]: Supportato per volumi raw-block

Driver backend Azure NetApp Files

Trident utilizza il `azure-netapp-files` autista per gestire il"Azure NetApp Files" servizio.

Ulteriori informazioni su questo driver e su come configurarlo possono essere trovate in"Configurazione del backend Trident per Azure NetApp Files" .

Driver Azure NetApp Files	Istantanee	Cloni	Multi-attacco	Qualità del servizio	Espandere	Replicazione
`azure-netapp-files`	SÌ	SÌ	SÌ	SÌ	SÌ	Sìnota a piè di pagina:1[]

> Nota a piè di pagina: Sì [1]: Non gestito da Trident

Driver backend Cloud Volumes Service su Google Cloud

Trident utilizza il `gcp-cvs` driver per il collegamento al Cloud Volumes Service su Google Cloud.

IL `gcp-cvs` Il driver utilizza pool virtuali per astrarre il backend e consentire a Trident di determinare il posizionamento del volume. L'amministratore definisce i pool virtuali nel `backend.json` file. Le classi di archiviazione utilizzano selettori per identificare i pool virtuali in base all'etichetta.

- Se nel backend sono definiti pool virtuali, Trident tenterà di creare un volume nei pool di archiviazione di Google Cloud a cui tali pool virtuali sono limitati.
- Se nel backend non sono definiti pool virtuali, Trident selezionerà un pool di archiviazione Google Cloud tra i pool di archiviazione disponibili nella regione.

Per configurare il backend di Google Cloud su Trident, è necessario specificare `projectNumber` , `apiRegion` , E `apiKey` nel file backend. Puoi trovare il numero del progetto nella console di Google Cloud. La chiave API viene ricavata dal file della chiave privata dell'account di servizio creato durante la configurazione dell'accesso API per Cloud Volumes Service su Google Cloud.

Per i dettagli sul Cloud Volumes Service sui tipi di servizio e sui livelli di servizio di Google Cloud, fare riferimento a"Scopri di più sul supporto Trident per CVS per GCP" .

Cloud Volumes Service per il driver Google Cloud	Istantanee	Cloni	Multi-attacco	Qualità del servizio	Espandere	Replicazione
`gcp-cvs`	SÌ	SÌ	SÌ	SÌ	SÌ	Disponibile solo per il tipo di servizio CVS-Performance.

Note di replicazione

- La replica non è gestita da Trident.
- Il clone verrà creato nello stesso pool di archiviazione del volume di origine.

Progettazione della classe di archiviazione

Per creare un oggetto Classe di archiviazione Kubernetes, è necessario configurare e applicare singole classi di archiviazione. In questa sezione viene illustrato come progettare una classe di archiviazione per la propria applicazione.

Utilizzo specifico del backend

È possibile utilizzare il filtraggio all'interno di un oggetto di classe di archiviazione specifico per determinare quale pool di archiviazione o set di pool utilizzare con quella specifica classe di archiviazione. Nella classe di archiviazione è possibile impostare tre set di filtri: `storagePools` , `additionalStoragePools` , e/o `excludeStoragePools` .

IL `storagePools` Il parametro consente di limitare l'archiviazione al set di pool che corrispondono a uno qualsiasi degli attributi specificati. IL `additionalStoragePools` Il parametro viene utilizzato per estendere il set di pool che Trident utilizza per il provisioning insieme al set di pool selezionati dagli attributi e `storagePools` parametri. È possibile utilizzare uno dei due parametri da solo o entrambi insieme per assicurarsi che venga selezionato il set appropriato di pool di archiviazione.

IL `excludeStoragePools` Il parametro viene utilizzato per escludere specificamente l'insieme elencato di pool che corrispondono agli attributi.

Emulare le policy QoS

Se si desidera progettare classi di archiviazione per emulare le policy di qualità del servizio, creare una classe di archiviazione con `media` attributo come `hdd` O `ssd` . Sulla base del `media` attributo menzionato nella classe di archiviazione, Trident selezionerà il backend appropriato che serve `hdd` O `ssd` aggregati per abbinare l'attributo multimediale e quindi indirizzare il provisioning dei volumi all'aggregato specifico. Pertanto possiamo creare una classe di archiviazione PREMIUM che avrebbe `media` attributo impostato come `ssd` che potrebbe essere classificata come politica QoS PREMIUM. Possiamo creare un'altra classe di archiviazione STANDARD che avrebbe l'attributo multimediale impostato su `hdd' e potrebbe essere classificata come policy QoS STANDARD. Potremmo anche utilizzare l'attributo ``IOPS'' nella classe di archiviazione per reindirizzare il provisioning a un'appliance Element che può essere definita come una politica QoS.

Utilizzare il backend in base a funzionalità specifiche

Le classi di archiviazione possono essere progettate per indirizzare il provisioning dei volumi su un backend specifico in cui sono abilitate funzionalità quali provisioning sottile e spesso, snapshot, cloni e crittografia. Per specificare quale storage utilizzare, creare classi di storage che specifichino il backend appropriato con la funzionalità richiesta abilitata.

Pool virtuali

I pool virtuali sono disponibili per tutti i backend Trident . È possibile definire pool virtuali per qualsiasi backend, utilizzando qualsiasi driver fornito da Trident .

I pool virtuali consentono a un amministratore di creare un livello di astrazione sui backend a cui è possibile fare riferimento tramite classi di archiviazione, per una maggiore flessibilità e un posizionamento efficiente dei volumi sui backend. È possibile definire backend diversi con la stessa classe di servizio. Inoltre, è possibile creare più pool di archiviazione sullo stesso backend, ma con caratteristiche diverse. Quando una classe di archiviazione viene configurata con un selettore con etichette specifiche, Trident sceglie un backend che corrisponda a tutte le etichette del selettore per posizionare il volume. Se le etichette del selettore della classe di archiviazione corrispondono a più pool di archiviazione, Trident ne sceglierà uno da cui eseguire il provisioning del volume.

Progettazione di piscine virtuali

Durante la creazione di un backend, è generalmente possibile specificare un set di parametri. Era impossibile per l'amministratore creare un altro backend con le stesse credenziali di storage e con un set di parametri diverso. Con l'introduzione dei pool virtuali, questo problema è stato risolto. Un pool virtuale è un'astrazione di livello introdotta tra il backend e la classe di storage di Kubernetes, in modo che l'amministratore possa definire parametri insieme a etichette a cui è possibile fare riferimento tramite le classi di storage di Kubernetes come selettore, in modo indipendente dal backend. I pool virtuali possono essere definiti per tutti i backend NetApp supportati con Trident. L'elenco include SolidFire/ NetApp HCI, ONTAP, Cloud Volumes Service su GCP e Azure NetApp Files.

Quando si definiscono pool virtuali, si consiglia di non tentare di riorganizzare l'ordine dei pool virtuali esistenti in una definizione backend. Si consiglia inoltre di non modificare gli attributi di un pool virtuale esistente e di definirne invece uno nuovo.

Emulazione di diversi livelli di servizio/QoS

È possibile progettare pool virtuali per emulare classi di servizi. Utilizzando l'implementazione del pool virtuale per Cloud Volume Service per Azure NetApp Files, esaminiamo come possiamo configurare diverse classi di servizi. Configurare il backend di Azure NetApp Files con più etichette, che rappresentano diversi livelli di prestazioni. Impostato `servicelevel` aspetto al livello di prestazione appropriato e aggiungere altri aspetti richiesti sotto ciascuna etichetta. Ora crea diverse classi di archiviazione Kubernetes che verranno mappate su diversi pool virtuali. Utilizzando il `parameters.selector` campo, ogni StorageClass richiama quali pool virtuali possono essere utilizzati per ospitare un volume.

Assegnazione di un insieme specifico di aspetti

È possibile progettare più pool virtuali con un set specifico di aspetti da un singolo backend di archiviazione. Per fare ciò, configura il backend con più etichette e imposta gli aspetti richiesti sotto ciascuna etichetta. Ora crea diverse classi di archiviazione Kubernetes utilizzando `parameters.selector` campo che verrebbe mappato su diversi pool virtuali. I volumi che vengono forniti sul backend avranno gli aspetti definiti nel pool virtuale scelto.

Caratteristiche del PVC che influenzano la fornitura di stoccaggio

Alcuni parametri che vanno oltre la classe di archiviazione richiesta potrebbero influire sul processo decisionale di provisioning Trident durante la creazione di un PVC.

Modalità di accesso

Quando si richiede l'archiviazione tramite PVC, uno dei campi obbligatori è la modalità di accesso. La modalità desiderata può influenzare il backend selezionato per ospitare la richiesta di archiviazione.

Trident tenterà di abbinare il protocollo di archiviazione utilizzato al metodo di accesso specificato in base alla seguente matrice. Ciò è indipendente dalla piattaforma di archiviazione sottostante.

	Leggi e scrivi una volta	ReadOnlyMany	LeggiScriviMolti
iSCSI	SÌ	SÌ	Sì (blocco grezzo)
NFS	SÌ	SÌ	SÌ

Una richiesta per un PVC ReadWriteMany inviata a una distribuzione Trident senza un backend NFS configurato non comporterà il provisioning di alcun volume. Per questo motivo, il richiedente deve utilizzare la modalità di accesso più adatta alla propria applicazione.

Operazioni di volume

Modificare i volumi persistenti

I volumi persistenti sono, con due eccezioni, oggetti immutabili in Kubernetes. Una volta creata, la politica di recupero e le dimensioni possono essere modificate. Tuttavia, ciò non impedisce che alcuni aspetti del volume vengano modificati al di fuori di Kubernetes. Ciò potrebbe essere utile per personalizzare il volume per applicazioni specifiche, per garantire che la capacità non venga consumata accidentalmente o semplicemente per spostare il volume su un controller di archiviazione diverso per qualsiasi motivo.

> Al momento, i provisioner in-tree di Kubernetes non supportano le operazioni di ridimensionamento del volume per PV NFS, iSCSI o FC. Trident supporta l'espansione di volumi NFS, iSCSI e FC.

I dettagli di connessione del PV non possono essere modificati dopo la creazione.

Crea snapshot di volumi su richiesta

Trident supporta la creazione di snapshot di volumi su richiesta e la creazione di PVC da snapshot utilizzando il framework CSI. Gli snapshot rappresentano un metodo pratico per conservare copie puntuali dei dati e hanno un ciclo di vita indipendente dal PV di origine in Kubernetes. Queste istantanee possono essere utilizzate per clonare i PVC.

Crea volumi da snapshot

Trident supporta anche la creazione di PersistentVolume da snapshot di volume. Per fare ciò, basta creare un PersistentVolumeClaim e menzionare il `datasource` come snapshot richiesto da cui creare il volume. Trident gestirà questo PVC creando un volume con i dati presenti nello snapshot. Grazie a questa funzionalità è possibile duplicare i dati tra regioni diverse, creare ambienti di test, sostituire completamente un volume di produzione danneggiato o corrotto oppure recuperare file e directory specifici e trasferirli su un altro volume collegato.

Spostare i volumi nel cluster

Gli amministratori di storage hanno la possibilità di spostare volumi tra aggregati e controller nel cluster ONTAP senza interrompere l'attività del consumatore di storage. Questa operazione non ha alcun effetto Trident o sul cluster Kubernetes, a condizione che l'aggregato di destinazione sia uno a cui ha accesso l'SVM utilizzato da Trident . È importante sottolineare che se l'aggregato è stato appena aggiunto all'SVM, il backend dovrà essere aggiornato aggiungendolo nuovamente a Trident. Ciò indurrà Trident a rielaborare l'inventario dell'SVM in modo che il nuovo aggregato venga riconosciuto.

Tuttavia, lo spostamento di volumi tra backend non è supportato automaticamente da Trident. Ciò include tra SVM nello stesso cluster, tra cluster o su una piattaforma di archiviazione diversa (anche se tale sistema di archiviazione è connesso a Trident).

Se un volume viene copiato in un'altra posizione, è possibile utilizzare la funzionalità di importazione del volume per importare i volumi correnti in Trident.

Espandi i volumi

Trident supporta il ridimensionamento di PV NFS, iSCSI e FC. Ciò consente agli utenti di ridimensionare i propri volumi direttamente tramite il livello Kubernetes. L'espansione del volume è possibile per tutte le principali piattaforme di storage NetApp , inclusi i backend ONTAP, SolidFire/ NetApp HCI e Cloud Volumes Service . Per consentire una possibile espansione successiva, impostare `allowVolumeExpansion` A `true` nella StorageClass associata al volume. Ogni volta che è necessario ridimensionare il volume persistente, modificare il `spec.resources.requests.storage` annotazione nella Persistent Volume Claim alla dimensione del volume richiesta. Trident si occuperà automaticamente di ridimensionare il volume sul cluster di archiviazione.

Importa un volume esistente in Kubernetes

L'importazione di volumi consente di importare un volume di archiviazione esistente in un ambiente Kubernetes. Questo è attualmente supportato da `ontap-nas` , `ontap-nas-flexgroup` , `solidfire-san` , `azure-netapp-files` , E `gcp-cvs` conducenti. Questa funzionalità è utile quando si trasferisce un'applicazione esistente in Kubernetes o durante scenari di disaster recovery.

Quando si utilizza ONTAP e `solidfire-san` driver, utilizzare il comando `tridentctl import volume <backend-name> <volume-name> -f /path/pvc.yaml` per importare un volume esistente in Kubernetes da gestire tramite Trident. Il file PVC YAML o JSON utilizzato nel comando di importazione del volume punta a una classe di archiviazione che identifica Trident come provisioner. Quando si utilizza un backend NetApp HCI/ SolidFire , assicurarsi che i nomi dei volumi siano univoci. Se i nomi dei volumi sono duplicati, clonare il volume con un nome univoco in modo che la funzionalità di importazione dei volumi possa distinguerli.

Se il `azure-netapp-files` O `gcp-cvs` viene utilizzato il driver, utilizzare il comando `tridentctl import volume <backend-name> <volume path> -f /path/pvc.yaml` per importare il volume in Kubernetes affinché venga gestito da Trident. Ciò garantisce un riferimento volumetrico univoco.

Quando viene eseguito il comando sopra, Trident troverà il volume sul backend e ne leggerà le dimensioni. Aggiungerà automaticamente (e sovrascriverà se necessario) la dimensione del volume del PVC configurato. Trident crea quindi il nuovo PV e Kubernetes lega il PVC al PV.

Se un contenitore è stato distribuito in modo tale da richiedere lo specifico PVC importato, rimarrà in uno stato di attesa finché la coppia PVC/PV non verrà associata tramite il processo di importazione del volume. Una volta legata la coppia PVC/PV, il contenitore dovrebbe sollevarsi, a meno che non ci siano altri problemi.

Servizio di registro

L'implementazione e la gestione dell'archiviazione per il registro sono state documentate su"netapp.io" nel"blog" .

Servizio di registrazione

Come altri servizi OpenShift, il servizio di registrazione viene distribuito tramite Ansible con parametri di configurazione forniti dal file di inventario, noto anche come host, fornito al playbook. Verranno trattati due metodi di installazione: l'implementazione della registrazione durante l'installazione iniziale di OpenShift e l'implementazione della registrazione dopo l'installazione di OpenShift.

A partire dalla versione 3.9 di Red Hat OpenShift, la documentazione ufficiale sconsiglia l'utilizzo di NFS per il servizio di registrazione a causa di preoccupazioni relative al danneggiamento dei dati. Ciò si basa sui test effettuati da Red Hat sui propri prodotti. Il server ONTAP NFS non presenta questi problemi e può facilmente supportare una distribuzione di registrazione. In definitiva, la scelta del protocollo per il servizio di registrazione spetta a te, sappi solo che entrambi funzioneranno alla grande quando si utilizzano le piattaforme NetApp e non c'è motivo di evitare NFS se questa è la tua preferenza.

Se si sceglie di utilizzare NFS con il servizio di registrazione, sarà necessario impostare la variabile Ansible `openshift_enable_unsupported_configurations` A `true` per evitare che il programma di installazione fallisca.

Iniziare

Facoltativamente, il servizio di registrazione può essere distribuito sia per le applicazioni che per le operazioni principali del cluster OpenShift stesso. Se si sceglie di distribuire la registrazione delle operazioni, specificando la variabile `openshift_logging_use_ops` COME `true` , verranno create due istanze del servizio. Le variabili che controllano l'istanza di registrazione per le operazioni contengono "ops", mentre l'istanza per le applicazioni no.

La configurazione delle variabili Ansible in base al metodo di distribuzione è importante per garantire che i servizi sottostanti utilizzino lo spazio di archiviazione corretto. Diamo un'occhiata alle opzioni per ciascuno dei metodi di distribuzione.

Le tabelle seguenti contengono solo le variabili rilevanti per la configurazione dell'archiviazione in relazione al servizio di registrazione. Puoi trovare altre opzioni in"Documentazione sulla registrazione di Red Hat OpenShift" che dovrebbe essere rivisto, configurato e utilizzato in base alla tua distribuzione.

Le variabili nella tabella sottostante faranno sì che il playbook Ansible crei un PV e un PVC per il servizio di registrazione utilizzando i dettagli forniti. Questo metodo è notevolmente meno flessibile rispetto all'utilizzo del playbook di installazione dei componenti dopo l'installazione di OpenShift; tuttavia, se si hanno volumi esistenti disponibili, è un'opzione.

Variabile	Dettagli
`openshift_logging_storage_kind`	Impostato su `nfs` per far sì che il programma di installazione crei un PV NFS per il servizio di registrazione.

Variabile	Dettagli
`openshift_logging_storage_host`	Il nome host o l'indirizzo IP dell'host NFS. Dovrebbe essere impostato sul dataLIF della tua macchina virtuale.
`openshift_logging_storage_nfs_directory`	Percorso di montaggio per l'esportazione NFS. Ad esempio, se il volume è giuntato come `/openshift_logging` , dovresti usare quel percorso per questa variabile.
`openshift_logging_storage_volume_name`	Il nome, ad esempio `pv_ose_logs` , del PV da creare.
`openshift_logging_storage_volume_size`	La dimensione dell'esportazione NFS, ad esempio `100Gi` .

Se il cluster OpenShift è già in esecuzione e quindi Trident è stato distribuito e configurato, il programma di installazione può utilizzare il provisioning dinamico per creare i volumi. Sarà necessario configurare le seguenti variabili.

Variabile	Dettagli
`openshift_logging_es_pvc_dynamic`	Impostare su true per utilizzare volumi con provisioning dinamico.
`openshift_logging_es_pvc_storage_class_name`	Nome della classe di archiviazione che verrà utilizzata nel PVC.
`openshift_logging_es_pvc_size`	La dimensione del volume richiesto nel PVC.
`openshift_logging_es_pvc_prefix`	Prefisso per i PVC utilizzati dal servizio di registrazione.
`openshift_logging_es_ops_pvc_dynamic`	Impostato su `true` per utilizzare volumi forniti dinamicamente per l'istanza di registrazione delle operazioni.
`openshift_logging_es_ops_pvc_storage_class_name`	Nome della classe di archiviazione per l'istanza di registrazione delle operazioni.
`openshift_logging_es_ops_pvc_size`	La dimensione della richiesta di volume per l'istanza ops.
`openshift_logging_es_ops_pvc_prefix`	Un prefisso per i PVC delle istanze ops.

Distribuisci lo stack di registrazione

Se si distribuisce la registrazione come parte del processo di installazione iniziale di OpenShift, è sufficiente seguire la procedura di distribuzione standard. Ansible configurerà e distribuirà i servizi e gli oggetti OpenShift necessari in modo che il servizio sia disponibile non appena Ansible sarà completato.

Tuttavia, se si esegue la distribuzione dopo l'installazione iniziale, il playbook dei componenti dovrà essere utilizzato da Ansible. Questo processo potrebbe cambiare leggermente con diverse versioni di OpenShift, quindi assicurati di leggere e seguire"Documentazione di Red Hat OpenShift Container Platform 3.11" per la tua versione.

Servizio di metriche

Il servizio di metriche fornisce all'amministratore informazioni preziose sullo stato, l'utilizzo delle risorse e la disponibilità del cluster OpenShift. È inoltre necessario per la funzionalità di scalabilità automatica dei pod e molte organizzazioni utilizzano i dati del servizio di metriche per le loro applicazioni di addebito e/o visualizzazione.

Come per il servizio di registrazione e OpenShift nel suo complesso, Ansible viene utilizzato per distribuire il servizio di metriche. Inoltre, come il servizio di registrazione, il servizio di metriche può essere distribuito durante la configurazione iniziale del cluster o dopo la sua operatività tramite il metodo di installazione dei componenti. Le tabelle seguenti contengono le variabili importanti quando si configura l'archiviazione persistente per il servizio di metriche.

> Le tabelle seguenti contengono solo le variabili rilevanti per la configurazione dell'archiviazione in relazione al servizio di metriche. Nella documentazione sono presenti molte altre opzioni che dovrebbero essere esaminate, configurate e utilizzate in base alla propria distribuzione.

Variabile	Dettagli
`openshift_metrics_storage_kind`	Impostato su `nfs` per far sì che il programma di installazione crei un PV NFS per il servizio di registrazione.
`openshift_metrics_storage_host`	Il nome host o l'indirizzo IP dell'host NFS. Dovrebbe essere impostato sul dataLIF per il tuo SVM.
`openshift_metrics_storage_nfs_directory`	Percorso di montaggio per l'esportazione NFS. Ad esempio, se il volume è giuntato come `/openshift_metrics` , dovresti usare quel percorso per questa variabile.
`openshift_metrics_storage_volume_name`	Il nome, ad esempio `pv_ose_metrics` , del PV da creare.
`openshift_metrics_storage_volume_size`	La dimensione dell'esportazione NFS, ad esempio `100Gi` .

Se il cluster OpenShift è già in esecuzione e quindi Trident è stato distribuito e configurato, il programma di installazione può utilizzare il provisioning dinamico per creare i volumi. Sarà necessario configurare le seguenti variabili.

Variabile	Dettagli
`openshift_metrics_cassandra_pvc_prefix`	Prefisso da utilizzare per le metriche PVC.
`openshift_metrics_cassandra_pvc_size`	La dimensione dei volumi da richiedere.
`openshift_metrics_cassandra_storage_type`	Il tipo di archiviazione da utilizzare per le metriche deve essere impostato su dinamico affinché Ansible possa creare PVC con la classe di archiviazione appropriata.
`openshift_metrics_cassanda_pvc_storage_class_name`	Nome della classe di archiviazione da utilizzare.

Distribuisci il servizio di metriche

Dopo aver definito le variabili Ansible appropriate nel file hosts/inventory, distribuisci il servizio utilizzando Ansible. Se si esegue la distribuzione al momento dell'installazione di OpenShift, il PV verrà creato e utilizzato automaticamente. Se si esegue la distribuzione utilizzando i playbook dei componenti, dopo l'installazione di OpenShift, Ansible crea tutti i PVC necessari e, dopo che Trident ha predisposto lo storage per essi, distribuisce il servizio.

Le variabili sopra indicate e il processo di distribuzione potrebbero cambiare con ogni versione di OpenShift. Assicurati di rivedere e seguire"Guida alla distribuzione di OpenShift di Red Hat" per la tua versione in modo che sia configurata per il tuo ambiente.

Protezione dei dati e ripristino di emergenza

Scopri le opzioni di protezione e ripristino per Trident e i volumi creati utilizzando Trident. Dovresti avere una strategia di protezione e ripristino dei dati per ogni applicazione con un requisito di persistenza.

Replicazione e recupero Trident

È possibile creare un backup per ripristinare Trident in caso di disastro.

Replicazione Trident

Trident utilizza i CRD di Kubernetes per archiviare e gestire il proprio stato e il cluster etcd di Kubernetes per archiviare i propri metadati.

Passi

1. Eseguire il backup del cluster Kubernetes etcd utilizzando"Kubernetes: backup di un cluster etcd" .
2. Posizionare gli artefatti di backup su un FlexVol volume

> NetApp consiglia di proteggere l'SVM in cui risiede FlexVol con una relazione SnapMirror con un altro SVM.

Recupero Trident

Utilizzando i CRD di Kubernetes e lo snapshot etcd del cluster Kubernetes, è possibile ripristinare Trident.

Passi

1. Dall'SVM di destinazione, montare il volume contenente i file di dati e i certificati etcd di Kubernetes sull'host che verrà configurato come nodo master.
2. Copia tutti i certificati richiesti relativi al cluster Kubernetes in `/etc/kubernetes/pki` e i file membri etcd sotto `/var/lib/etcd` .
3. Ripristina il cluster Kubernetes dal backup etcd utilizzando"Kubernetes: ripristino di un cluster etcd" .
4. Correre `kubectl get crd` per verificare che tutte le risorse personalizzate Trident siano state attivate e recuperare gli oggetti Trident per verificare che tutti i dati siano disponibili.

Replica e ripristino SVM

Trident non può configurare relazioni di replica, tuttavia, l'amministratore di storage può utilizzare "ONTAP SnapMirror" per replicare una SVM.

In caso di disastro, è possibile attivare la SVM di destinazione SnapMirror per iniziare a fornire dati. È possibile tornare al sistema primario quando i sistemi vengono ripristinati.

Informazioni su questo compito

Quando si utilizza la funzionalità di replica SVM SnapMirror, tenere presente quanto segue:

- Dovresti creare un backend distinto per ogni SVM con SVM-DR abilitato.
- Configurare le classi di archiviazione per selezionare i backend replicati solo quando necessario, per evitare che i volumi che non necessitano di replica vengano forniti sui backend che supportano SVM-DR.
- Gli amministratori delle applicazioni devono comprendere i costi aggiuntivi e la complessità associati alla replica e valutare attentamente il proprio piano di ripristino prima di iniziare questo processo.

Replicazione SVM

Puoi usare"ONTAP: replica SVM SnapMirror" per creare la relazione di replicazione SVM.

SnapMirror consente di impostare opzioni per controllare cosa replicare. Dovrai sapere quali opzioni hai selezionato durante l'esecuzioneRipristino SVM tramite Trident .

- "-identità-preserva vero"replica l'intera configurazione SVM.
- "-discard-configs rete"esclude i LIF e le relative impostazioni di rete.
- "-identità-preserva falso"replica solo i volumi e la configurazione di sicurezza.

Ripristino SVM tramite Trident

Trident non rileva automaticamente i guasti SVM. In caso di disastro, l'amministratore può avviare manualmente il failover Trident sul nuovo SVM.

Passi

1. Annullare i trasferimenti SnapMirror pianificati e in corso, interrompere la relazione di replica, arrestare l'SVM di origine e quindi attivare l'SVM di destinazione SnapMirror .
2. Se hai specificato `-identity-preserve false` O `-discard-config network` quando si configura la replica SVM, aggiornare `managementLIF` E `dataLIF` nel file di definizione del backend Trident .
3. Confermare `storagePrefix` è presente nel file di definizione del backend Trident . Questo parametro non può essere modificato. Omettendo `storagePrefix` causerà il fallimento dell'aggiornamento del backend.
4. Aggiornare tutti i backend richiesti per riflettere il nuovo nome SVM di destinazione utilizzando:

```
./tridentctl update backend <backend-name> -f <backend-json-file> -n
<namespace>
```

5. Se hai specificato `-identity-preserve false` O `discard-config network` , è necessario rimbalzare tutti i pod dell'applicazione.

Se hai specificato `-identity-preserve true` , tutti i volumi forniti da Trident iniziano a fornire dati quando viene attivata la SVM di destinazione.

Replicazione e ripristino del volume

Trident non può configurare le relazioni di replica SnapMirror , tuttavia, l'amministratore dell'archiviazione può utilizzare"Replica e ripristino ONTAP SnapMirror" per replicare i volumi creati da Trident.

È quindi possibile importare i volumi recuperati in Trident utilizzando"importazione volume tridentctl" .

L'importazione non è supportata su `ontap-nas-economy` , `ontap-san-economy` , O `ontap-flexgroup-economy` conducenti.

Protezione dei dati snapshot

È possibile proteggere e ripristinare i dati utilizzando:

- Un controller di snapshot esterno e CRD per creare snapshot di volumi Kubernetes di volumi persistenti (PV).

  "Istantanee del volume"

- Snapshot ONTAP per ripristinare l'intero contenuto di un volume o per recuperare singoli file o LUN.

  "Istantanee ONTAP"

Sicurezza

Sicurezza

Per garantire la sicurezza dell'installazione Trident , attenersi alle raccomandazioni elencate qui.

Esegui Trident nel suo namespace

È importante impedire alle applicazioni, agli amministratori delle applicazioni, agli utenti e alle applicazioni di gestione di accedere alle definizioni degli oggetti Trident o ai pod per garantire un'archiviazione affidabile e bloccare potenziali attività dannose.

Per separare le altre applicazioni e gli utenti da Trident, installare sempre Trident nel proprio namespace Kubernetes(`trident`). Inserire Trident nel proprio namespace garantisce che solo il personale amministrativo di Kubernetes abbia accesso al pod Trident e agli artefatti (come i segreti backend e CHAP, se applicabile) archiviati negli oggetti CRD con namespace. Dovresti assicurarti di consentire solo agli amministratori l'accesso allo spazio dei nomi Trident e quindi l'accesso a `tridentctl` applicazione.

Utilizzare l'autenticazione CHAP con i backend ONTAP SAN

Trident supporta l'autenticazione basata su CHAP per carichi di lavoro ONTAP SAN (utilizzando `ontap-san` E `ontap-san-economy` conducenti). NetApp consiglia di utilizzare CHAP bidirezionale con Trident per l'autenticazione tra un host e il backend di storage.

Per i backend ONTAP che utilizzano i driver di archiviazione SAN, Trident può impostare CHAP bidirezionale e gestire i nomi utente e i segreti CHAP tramite `tridentctl` . Fare riferimento a"Prepararsi a configurare il backend con i driver ONTAP SAN" per capire come Trident configura CHAP sui backend ONTAP .

Utilizzare l'autenticazione CHAP con i backend NetApp HCI e SolidFire

NetApp consiglia di implementare CHAP bidirezionale per garantire l'autenticazione tra un host e i backend NetApp HCI e SolidFire . Trident utilizza un oggetto segreto che include due password CHAP per tenant. Quando Trident è installato, gestisce i segreti CHAP e li memorizza in un `tridentvolume` Oggetto CR per il rispettivo PV. Quando si crea un PV, Trident utilizza i segreti CHAP per avviare una sessione iSCSI e comunicare con il sistema NetApp HCI e SolidFire tramite CHAP.

I volumi creati da Trident non sono associati ad alcun Volume Access Group.

Utilizzare Trident con NVE e NAE

NetApp ONTAP fornisce la crittografia dei dati a riposo per proteggere i dati sensibili nel caso in cui un disco venga rubato, restituito o riutilizzato. Per i dettagli, fare riferimento a"Panoramica sulla configurazione della crittografia del volume NetApp" .

- Se NAE è abilitato sul backend, qualsiasi volume fornito in Trident sarà abilitato per NAE.
  - È possibile impostare il flag di crittografia NVE su `""` per creare volumi abilitati per NAE.
- Se NAE non è abilitato sul backend, qualsiasi volume fornito in Trident sarà abilitato per NVE a meno che il flag di crittografia NVE non sia impostato su `false` (valore predefinito) nella configurazione del backend.

I volumi creati in Trident su un backend abilitato NAE devono essere crittografati tramite NVE o NAE.

- È possibile impostare il flag di crittografia NVE su `true` nella configurazione del backend Trident per ignorare la crittografia NAE e utilizzare una chiave di crittografia specifica per ogni volume.
- Impostazione del flag di crittografia NVE su `false` su un backend abilitato NAE crea un volume abilitato NAE. Non è possibile disabilitare la crittografia NAE impostando il flag di crittografia NVE su `false` .

- È possibile creare manualmente un volume NVE in Trident impostando esplicitamente il flag di crittografia NVE su `true` .

Per maggiori informazioni sulle opzioni di configurazione del backend, fare riferimento a:

- "Opzioni di configurazione SAN ONTAP"
- "Opzioni di configurazione NAS ONTAP"

Configurazione della chiave unificata Linux (LUKS)

È possibile abilitare Linux Unified Key Setup (LUKS) per crittografare i volumi ONTAP SAN e ONTAP SAN ECONOMY su Trident. Trident supporta la rotazione delle passphrase e l'espansione del volume per i volumi crittografati con LUKS.

In Trident, i volumi crittografati LUKS utilizzano la cifratura e la modalità aes-xts-plain64, come raccomandato

da"NIST" .

> La crittografia LUKS non è supportata per i sistemi ASA r2. Per informazioni sui sistemi ASA r2, vedere"Scopri di più sui sistemi di archiviazione ASA r2" .

Prima di iniziare

- Sui nodi worker deve essere installato cryptsetup 2.1 o versione successiva (ma inferiore a 3.0). Per maggiori informazioni, visita"Gitlab: cryptsetup" .
- Per motivi di prestazioni, NetApp consiglia che i nodi worker supportino Advanced Encryption Standard New Instructions (AES-NI). Per verificare il supporto AES-NI, eseguire il seguente comando:

```
grep "aes" /proc/cpuinfo
```

Se non viene restituito nulla, il processore non supporta AES-NI. Per maggiori informazioni su AES-NI, visitare:"Intel: Istruzioni per lo standard di crittografia avanzata (AES-NI)" .

Abilita la crittografia LUKS

È possibile abilitare la crittografia lato host per volume utilizzando Linux Unified Key Setup (LUKS) per i volumi ONTAP SAN e ONTAP SAN ECONOMY.

Passi

1. Definire gli attributi di crittografia LUKS nella configurazione del backend. Per ulteriori informazioni sulle opzioni di configurazione backend per ONTAP SAN, fare riferimento a"Opzioni di configurazione SAN ONTAP" .

```
{
  "storage": [
    {
      "labels": {
        "luks": "true"
      },
      "zone": "us_east_1a",
      "defaults": {
        "luksEncryption": "true"
      }
    },
    {
      "labels": {
        "luks": "false"
      },
      "zone": "us_east_1a",
      "defaults": {
        "luksEncryption": "false"
      }
    }
  ]
}
```

2. Utilizzo `parameters.selector` per definire i pool di archiviazione utilizzando la crittografia LUKS. Per esempio:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: csi.trident.netapp.io
parameters:
  selector: "luks=true"
  csi.storage.k8s.io/node-stage-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

3. Crea un segreto che contenga la passphrase LUKS. Per esempio:

```
kubectl -n trident create -f luks-pvc1.yaml
apiVersion: v1
kind: Secret
metadata:
  name: luks-pvc1
stringData:
  luks-passphrase-name: A
  luks-passphrase: secretA
```

Limitazioni

I volumi crittografati con LUKS non possono sfruttare la deduplicazione e la compressione ONTAP .

Configurazione backend per l'importazione di volumi LUKS

Per importare un volume LUKS, è necessario impostare `luksEncryption` A(`true` sul backend. IL `luksEncryption` l'opzione indica a Trident se il volume è conforme a LUKS(`true`) o non conforme a LUKS(`false`) come mostrato nell'esempio seguente.

```
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: trident_svm
username: admin
password: password
defaults:
  luksEncryption: 'true'
  spaceAllocation: 'false'
  snapshotPolicy: default
  snapshotReserve: '10'
```

Configurazione PVC per l'importazione di volumi LUKS

Per importare dinamicamente i volumi LUKS, impostare l'annotazione `trident.netapp.io/luksEncryption` A `true` e includere una classe di archiviazione abilitata LUKS nel PVC come mostrato in questo esempio.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: luks-pvc
  namespace: trident
  annotations:
    trident.netapp.io/luksEncryption: "true"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: luks-sc
```

Ruota una passphrase LUKS

È possibile ruotare la passphrase LUKS e confermare la rotazione.

> Non dimenticare una passphrase finché non hai verificato che non sia più referenziata da alcun volume, snapshot o segreto. Se si perde una passphrase di riferimento, potrebbe non essere possibile montare il volume e i dati rimarranno crittografati e inaccessibili.

Informazioni su questo compito

La rotazione della passphrase LUKS si verifica quando viene creato un pod che monta il volume dopo aver specificato una nuova passphrase LUKS. Quando viene creato un nuovo pod, Trident confronta la passphrase LUKS sul volume con la passphrase attiva nel segreto.

- Se la passphrase sul volume non corrisponde alla passphrase attiva nel segreto, si verifica la rotazione.
- Se la passphrase sul volume corrisponde alla passphrase attiva nel segreto, il `previous-luks-passphrase` il parametro viene ignorato.

Passi

1. Aggiungi il `node-publish-secret-name` E `node-publish-secret-namespace` Parametri StorageClass. Per esempio:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-san
provisioner: csi.trident.netapp.io
parameters:
  trident.netapp.io/backendType: "ontap-san"
  csi.storage.k8s.io/node-stage-secret-name: luks
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
  csi.storage.k8s.io/node-publish-secret-name: luks
  csi.storage.k8s.io/node-publish-secret-namespace: ${pvc.namespace}
```

2. Identificare le passphrase esistenti sul volume o sullo snapshot.

Volume

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames:["A"]
```

Istantanea

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames:["A"]
```

3. Aggiornare il segreto LUKS per il volume per specificare le passphrase nuove e precedenti. Garantire `previous-luke-passphrase-name` E `previous-luks-passphrase` corrisponde alla passphrase precedente.

```
apiVersion: v1
kind: Secret
metadata:
  name: luks-pvc1
stringData:
  luks-passphrase-name: B
  luks-passphrase: secretB
  previous-luks-passphrase-name: A
  previous-luks-passphrase: secretA
```

4. Crea un nuovo pod montando il volume. Ciò è necessario per avviare la rotazione.
5. Verificare che la passphrase sia stata ruotata.

Volume

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames:["B"]
```

Istantanea

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames:["B"]
```

Risultati

La passphrase è stata ruotata quando sul volume e sullo snapshot è stata restituita solo la nuova passphrase.

> Se vengono restituite due passphrase, ad esempio `luksPassphraseNames: ["B", "A"]`, la rotazione è incompleta. È possibile attivare un nuovo pod per tentare di completare la rotazione.

Abilita l'espansione del volume

È possibile abilitare l'espansione del volume su un volume crittografato con LUKS.

Passi

1. Abilita il `CSINodeExpandSecret` feature gate (beta 1.25+). Fare riferimento a "Kubernetes 1.25: utilizzare i segreti per l'espansione dei volumi CSI basata sui nodi" per i dettagli.
2. Aggiungi il `node-expand-secret-name` E `node-expand-secret-namespace` Parametri StorageClass. Per esempio:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: csi.trident.netapp.io
parameters:
  selector: "luks=true"
  csi.storage.k8s.io/node-stage-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
  csi.storage.k8s.io/node-expand-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-expand-secret-namespace: ${pvc.namespace}
allowVolumeExpansion: true
```

Risultati

Quando si avvia l'espansione dello storage online, kubelet passa le credenziali appropriate al driver.

Crittografia Kerberos in volo

Utilizzando la crittografia in-flight Kerberos, puoi migliorare la sicurezza dell'accesso ai dati abilitando la crittografia per il traffico tra il cluster gestito e il backend di archiviazione.

Trident supporta la crittografia Kerberos per ONTAP come backend di archiviazione:

- * ONTAP on-premise* - Trident supporta la crittografia Kerberos su connessioni NFSv3 e NFSv4 da Red Hat OpenShift e cluster Kubernetes upstream a volumi ONTAP on-premise.

È possibile creare, eliminare, ridimensionare, creare snapshot, clonare, clonare in sola lettura e importare volumi che utilizzano la crittografia NFS.

Configurare la crittografia Kerberos in-flight con volumi ONTAP on-premise

È possibile abilitare la crittografia Kerberos sul traffico di archiviazione tra il cluster gestito e un backend di archiviazione ONTAP locale.

> La crittografia Kerberos per il traffico NFS con backend di archiviazione ONTAP on-premise è supportata solo utilizzando `ontap-nas` driver di archiviazione.

Prima di iniziare

- Assicurati di avere accesso a `tridentctl` utilità.
- Assicurati di avere accesso come amministratore al backend di archiviazione ONTAP .
- Assicurati di conoscere il nome del volume o dei volumi che condividerai dal backend di archiviazione ONTAP .
- Assicurarsi di aver preparato la VM di archiviazione ONTAP per supportare la crittografia Kerberos per i volumi NFS. Fare riferimento a "Abilita Kerberos su un dataLIF" per istruzioni.
- Assicurarsi che tutti i volumi NFSv4 utilizzati con la crittografia Kerberos siano configurati correttamente. Fare riferimento alla sezione Configurazione del dominio NetApp NFSv4 (pagina 13) del manuale "Guida ai miglioramenti e alle best practice NetApp NFSv4" .

Aggiungere o modificare le policy di esportazione ONTAP

È necessario aggiungere regole alle policy di esportazione ONTAP esistenti o creare nuove policy di esportazione che supportino la crittografia Kerberos per il volume radice della VM di archiviazione ONTAP e per tutti i volumi ONTAP condivisi con il cluster Kubernetes upstream. Le regole dei criteri di esportazione aggiunti o i nuovi criteri di esportazione creati devono supportare i seguenti protocolli di accesso e autorizzazioni di accesso:

Protocolli di accesso

Configurare la policy di esportazione con i protocolli di accesso NFS, NFSv3 e NFSv4.

Dettagli di accesso

È possibile configurare una delle tre diverse versioni della crittografia Kerberos, a seconda delle esigenze del volume:

- **Kerberos 5** - (autenticazione e crittografia)

- **Kerberos 5i** - (autenticazione e crittografia con protezione dell'identità)
- **Kerberos 5p** - (autenticazione e crittografia con protezione dell'identità e della privacy)

Configurare la regola di policy di esportazione ONTAP con le autorizzazioni di accesso appropriate. Ad esempio, se i cluster monteranno i volumi NFS con una combinazione di crittografia Kerberos 5i e Kerberos 5p, utilizzare le seguenti impostazioni di accesso:

Tipo	Accesso di sola lettura	Accesso in lettura/scrittura	Accesso superutente
UNIX	Abilitato	Abilitato	Abilitato
Kerberos 5i	Abilitato	Abilitato	Abilitato
Kerberos 5p	Abilitato	Abilitato	Abilitato

Per informazioni su come creare policy di esportazione ONTAP e regole di policy di esportazione, fare riferimento alla seguente documentazione:

- "Creare una politica di esportazione"
- "Aggiungere una regola a un criterio di esportazione"

Creare un backend di archiviazione

È possibile creare una configurazione backend di archiviazione Trident che includa la funzionalità di crittografia Kerberos.

Informazioni su questo compito

Quando si crea un file di configurazione del backend di archiviazione che configura la crittografia Kerberos, è possibile specificare una delle tre diverse versioni della crittografia Kerberos utilizzando `spec.nfsMountOptions` parametro:

- `spec.nfsMountOptions: sec=krb5`(autenticazione e crittografia)
- `spec.nfsMountOptions: sec=krb5i`(autenticazione e crittografia con protezione dell'identità)
- `spec.nfsMountOptions: sec=krb5p`(autenticazione e crittografia con protezione dell'identità e della privacy)

Specificare un solo livello Kerberos. Se si specifica più di un livello di crittografia Kerberos nell'elenco dei parametri, verrà utilizzata solo la prima opzione.

Passi

1. Nel cluster gestito, creare un file di configurazione del backend di archiviazione utilizzando il seguente esempio. Sostituisci i valori tra parentesi <> con le informazioni del tuo ambiente:

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-ontap-nas-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-ontap-nas
spec:
  version: 1
  storageDriverName: "ontap-nas"
  managementLIF: <STORAGE_VM_MGMT_LIF_IP_ADDRESS>
  dataLIF: <PROTOCOL_LIF_FQDN_OR_IP_ADDRESS>
  svm: <STORAGE_VM_NAME>
  username: <STORAGE_VM_USERNAME_CREDENTIAL>
  password: <STORAGE_VM_PASSWORD_CREDENTIAL>
  nasType: nfs
  nfsMountOptions: ["sec=krb5i"] #can be krb5, krb5i, or krb5p
  qtreesPerFlexvol:
  credentials:
    name: backend-ontap-nas-secret
```

2. Utilizzare il file di configurazione creato nel passaggio precedente per creare il backend:

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

Se la creazione del backend fallisce, c'è qualcosa che non va nella configurazione del backend. È possibile visualizzare i registri 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.

Creare una classe di archiviazione

È possibile creare una classe di archiviazione per eseguire il provisioning dei volumi con crittografia Kerberos.

Informazioni su questo compito

Quando si crea un oggetto di classe di archiviazione, è possibile specificare una delle tre diverse versioni della crittografia Kerberos utilizzando `mountOptions` parametro:

- `mountOptions: sec=krb5`(autenticazione e crittografia)
- `mountOptions: sec=krb5i`(autenticazione e crittografia con protezione dell'identità)
- `mountOptions: sec=krb5p`(autenticazione e crittografia con protezione dell'identità e della privacy)

Specificare un solo livello Kerberos. Se si specifica più di un livello di crittografia Kerberos nell'elenco dei parametri, verrà utilizzata solo la prima opzione. Se il livello di crittografia specificato nella configurazione del backend di archiviazione è diverso dal livello specificato nell'oggetto della classe di archiviazione, l'oggetto della classe di archiviazione ha la precedenza.

Passi

1. Creare un oggetto StorageClass Kubernetes, utilizzando il seguente esempio:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-nas-sc
provisioner: csi.trident.netapp.io
mountOptions:
  - sec=krb5i #can be krb5, krb5i, or krb5p
parameters:
  backendType: ontap-nas
  storagePools: ontapnas_pool
  trident.netapp.io/nasType: nfs
allowVolumeExpansion: true
```

2. Creare la classe di archiviazione:

```
kubectl create -f sample-input/storage-class-ontap-nas-sc.yaml
```

3. Assicurarsi che la classe di archiviazione sia stata creata:

```
kubectl get sc ontap-nas-sc
```

   Dovresti vedere un output simile al seguente:

```
NAME          PROVISIONER             AGE
ontap-nas-sc    csi.trident.netapp.io   15h
```

Volumi di fornitura

Dopo aver creato un backend di archiviazione e una classe di archiviazione, è possibile effettuare il provisioning di un volume. Per le istruzioni, fare riferimento a "Fornire un volume" .

Configurare la crittografia Kerberos in-flight con volumi Azure NetApp Files

È possibile abilitare la crittografia Kerberos sul traffico di archiviazione tra il cluster gestito e un singolo backend di archiviazione Azure NetApp Files o un pool virtuale di backend di archiviazione di Azure NetApp Files .

Prima di iniziare

- Assicurati di aver abilitato Trident sul cluster Red Hat OpenShift gestito.
- Assicurati di avere accesso a `tridentctl` utilità.
- Assicurarsi di aver preparato il backend di archiviazione Azure NetApp Files per la crittografia Kerberos prendendo nota dei requisiti e seguendo le istruzioni in "Documentazione Azure NetApp Files" .
- Assicurarsi che tutti i volumi NFSv4 utilizzati con la crittografia Kerberos siano configurati correttamente. Fare riferimento alla sezione Configurazione del dominio NetApp NFSv4 (pagina 13) del manuale "Guida ai miglioramenti e alle best practice NetApp NFSv4" .

Creare un backend di archiviazione

È possibile creare una configurazione back-end di archiviazione di Azure NetApp Files che includa la funzionalità di crittografia Kerberos.

Informazioni su questo compito

Quando si crea un file di configurazione del backend di archiviazione che configura la crittografia Kerberos, è possibile definirlo in modo che venga applicato a uno dei due livelli possibili:

- Il **livello di backend di archiviazione** che utilizza `spec.kerberos` campo
- Il **livello della piscina virtuale** utilizzando il `spec.storage.kerberos` campo

Quando si definisce la configurazione a livello di pool virtuale, il pool viene selezionato utilizzando l'etichetta nella classe di archiviazione.

A entrambi i livelli è possibile specificare una delle tre diverse versioni della crittografia Kerberos:

- `kerberos: sec=krb5`(autenticazione e crittografia)
- `kerberos: sec=krb5i`(autenticazione e crittografia con protezione dell'identità)
- `kerberos: sec=krb5p`(autenticazione e crittografia con protezione dell'identità e della privacy)

Passi

1. Nel cluster gestito, creare un file di configurazione del backend di archiviazione utilizzando uno degli esempi seguenti, a seconda di dove è necessario definire il backend di archiviazione (livello di backend di archiviazione o livello di pool virtuale). Sostituisci i valori tra parentesi <> con le informazioni del tuo ambiente:

Esempio di livello di backend di archiviazione

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-secret
```

Esempio di livello di piscina virtuale

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  storage:
    - labels:
        type: encryption
      kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-secret
```

2. Utilizzare il file di configurazione creato nel passaggio precedente per creare il backend:

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

Se la creazione del backend fallisce, c'è qualcosa che non va nella configurazione del backend. È possibile visualizzare i registri 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.

Creare una classe di archiviazione

È possibile creare una classe di archiviazione per eseguire il provisioning dei volumi con crittografia Kerberos.

Passi

1. Creare un oggetto StorageClass Kubernetes, utilizzando il seguente esempio:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-nfs
provisioner: csi.trident.netapp.io
parameters:
  backendType: azure-netapp-files
  trident.netapp.io/nasType: nfs
  selector: type=encryption
```

2. Creare la classe di archiviazione:

```
kubectl create -f sample-input/storage-class-sc-nfs.yaml
```

3. Assicurarsi che la classe di archiviazione sia stata creata:

```
kubectl get sc -sc-nfs
```

Dovresti vedere un output simile al seguente:

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

Volumi di fornitura

Dopo aver creato un backend di archiviazione e una classe di archiviazione, è possibile effettuare il provisioning di un volume. Per le istruzioni, fare riferimento a "Fornire un volume" .

Proteggi le applicazioni con Trident Protect

Scopri di più su Trident Protect

NetApp Trident Protect offre funzionalità avanzate di gestione dei dati delle applicazioni che migliorano la funzionalità e la disponibilità delle applicazioni Kubernetes con stato supportate dai sistemi di storage NetApp ONTAP e dal provisioner di storage NetApp Trident CSI. Trident Protect semplifica la gestione, la protezione e lo spostamento dei carichi di lavoro containerizzati tra cloud pubblici e ambienti on-premise. Offre inoltre funzionalità di automazione tramite API e CLI.

È possibile proteggere le applicazioni con Trident Protect creando risorse personalizzate (CR) o utilizzando la CLI Trident Protect.

Cosa succederà ora?

Puoi informarti sui requisiti Trident Protect prima di installarlo:

- "Requisiti di protezione Trident"

Installa Trident Protect

Requisiti di protezione Trident

Per iniziare, verifica la prontezza del tuo ambiente operativo, dei cluster applicativi, delle applicazioni e delle licenze. Assicurati che il tuo ambiente soddisfi questi requisiti per distribuire e utilizzare Trident Protect.

Trident protegge la compatibilità del cluster Kubernetes

Trident Protect è compatibile con un'ampia gamma di offerte Kubernetes completamente gestite e autogestite, tra cui:

- Servizio Amazon Elastic Kubernetes (EKS)
- Motore Google Kubernetes (GKE)
- Servizio Microsoft Azure Kubernetes (AKS)
- Red Hat OpenShift
- SUSE Rancher
- Portafoglio VMware Tanzu
- Kubernetes a monte

- I backup Trident Protect sono supportati solo sui nodi di elaborazione Linux. I nodi di elaborazione Windows non sono supportati per le operazioni di backup.
- Assicurarsi che il cluster su cui si installa Trident Protect sia configurato con un controller snapshot in esecuzione e i relativi CRD. Per installare un controller snapshot, fare riferimento a "queste istruzioni" .

Compatibilità del backend di archiviazione Trident Protect

Trident Protect supporta i seguenti backend di archiviazione:

- Amazon FSx for NetApp ONTAP
- Cloud Volumes ONTAP
- Array di archiviazione ONTAP
- Google Cloud NetApp Volumes
- Azure NetApp Files

Assicurati che il tuo backend di archiviazione soddisfi i seguenti requisiti:

- Assicurarsi che lo storage NetApp connesso al cluster utilizzi Trident 24.02 o una versione successiva (si consiglia Trident 24.10).
- Assicurati di disporre di un backend di archiviazione NetApp ONTAP .
- Assicurati di aver configurato un bucket di archiviazione degli oggetti per l'archiviazione dei backup.
- Crea tutti gli spazi dei nomi delle applicazioni che intendi utilizzare per le applicazioni o per le operazioni di gestione dei dati delle applicazioni. Trident Protect non crea questi namespace per te; se specifichi uno namespace inesistente in una risorsa personalizzata, l'operazione non riuscirà.

Requisiti per i volumi nas-economy

Trident Protect supporta le operazioni di backup e ripristino sui volumi nas-economy. Snapshot, cloni e replica SnapMirror su volumi nas-economy non sono attualmente supportati. È necessario abilitare una directory snapshot per ogni volume nas-economy che si intende utilizzare con Trident Protect.

Alcune applicazioni non sono compatibili con i volumi che utilizzano una directory snapshot. Per queste applicazioni, è necessario nascondere la directory degli snapshot eseguendo il seguente comando sul sistema di archiviazione ONTAP :

```
nfs modify -vserver <svm> -v3-hide-snapshot enabled
```

È possibile abilitare la directory snapshot eseguendo il seguente comando per ogni volume nas-economy, sostituendo `<volume-UUID>` con l'UUID del volume che vuoi modificare:

```
tridentctl update volume <volume-UUID> --snapshot-dir=true --pool-level
=true -n trident
```

È possibile abilitare le directory snapshot per impostazione predefinita per i nuovi volumi impostando l'opzione di configurazione del backend Trident `snapshotDir` A `true` . I volumi esistenti non sono interessati.

Protezione dei dati con le VM KubeVirt

Trident Protect 24.10 e 24.10.1 e versioni successive hanno un comportamento diverso quando si proteggono le applicazioni in esecuzione su VM KubeVirt. Per entrambe le versioni è possibile abilitare o disabilitare il blocco e lo sblocco del file system durante le operazioni di protezione dei dati.

Durante le operazioni di ripristino, qualsiasi `VirtualMachineSnapshots` creati per una macchina virtuale (VM) non vengono ripristinati.

Trident protect 24.10

Trident Protect 24.10 non garantisce automaticamente uno stato coerente per i file system delle VM KubeVirt durante le operazioni di protezione dei dati. Se si desidera proteggere i dati della VM KubeVirt utilizzando Trident Protect 24.10, è necessario abilitare manualmente la funzionalità di congelamento/scongelamento per i file system prima dell'operazione di protezione dei dati. Ciò garantisce che i file system siano in uno stato coerente.

È possibile configurare Trident Protect 24.10 per gestire il blocco e lo sblocco del file system della VM durante le operazioni di protezione dei dati tramite"configurazione della virtualizzazione" e quindi utilizzando il seguente comando:

```
kubectl set env deployment/trident-protect-controller-manager
NEPTUNE_VM_FREEZE=true -n trident-protect
```

Trident Protect 24.10.1 e versioni successive

A partire da Trident Protect 24.10.1, Trident Protect blocca e sblocca automaticamente i file system KubeVirt durante le operazioni di protezione dei dati. Facoltativamente, è possibile disattivare questo comportamento automatico utilizzando il seguente comando:

```
kubectl set env deployment/trident-protect-controller-manager
NEPTUNE_VM_FREEZE=false -n trident-protect
```

Requisiti per la replica SnapMirror

La replica NetApp SnapMirror è disponibile per l'uso con Trident Protect per le seguenti soluzioni ONTAP :

- Cluster NetApp FAS, AFF e ASA on-premise
- ONTAP Select NetApp ONTAP
- NetApp Cloud Volumes ONTAP
- Amazon FSx for NetApp ONTAP

Requisiti del cluster ONTAP per la replica SnapMirror

Se si prevede di utilizzare la replica SnapMirror, assicurarsi che il cluster ONTAP soddisfi i seguenti requisiti:

- * NetApp Trident*: NetApp Trident deve essere presente sia sul cluster Kubernetes di origine che su quello di destinazione che utilizzano ONTAP come backend. Trident Protect supporta la replica 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)

- **Licenze**: le licenze asincrone ONTAP SnapMirror che utilizzano il bundle Data Protection devono essere abilitate sia sul cluster ONTAP di origine che su quello di destinazione. Fare riferimento a "Panoramica delle licenze SnapMirror in ONTAP" per maggiori informazioni.

  A partire da ONTAP 9.10.1, tutte le licenze vengono fornite come file di licenza NetApp (NLF), ovvero un singolo file che abilita più funzionalità. Fare riferimento a"Licenze incluse con ONTAP One" per maggiori informazioni.

  > È supportata solo la protezione asincrona SnapMirror .

Considerazioni sul peering per la replica SnapMirror

Se intendi utilizzare il peering backend di archiviazione, assicurati che il tuo ambiente soddisfi i seguenti requisiti:

- **Cluster e SVM**: i backend di archiviazione ONTAP devono essere peered. Fare riferimento a "Panoramica del peering di cluster e SVM" per maggiori informazioni.

  > Assicurarsi che i nomi SVM utilizzati nella relazione di replica tra due cluster ONTAP siano univoci.

- * NetApp Trident e SVM*: le SVM remote peered devono essere disponibili per NetApp Trident sul cluster di destinazione.
- **Backend gestiti**: è necessario aggiungere e gestire i backend di archiviazione ONTAP in Trident Protect per creare una relazione di replica.

Configurazione Trident / ONTAP per la replica SnapMirror

Trident Protect richiede la configurazione di almeno un backend di archiviazione che supporti la replica sia per i cluster di origine che per quelli di destinazione. Se i cluster di origine e di destinazione sono gli stessi, l'applicazione di destinazione dovrebbe utilizzare un backend di archiviazione diverso da quello dell'applicazione di origine per ottenere la migliore resilienza.

Requisiti del cluster Kubernetes per la replica SnapMirror

Assicurati che i tuoi cluster Kubernetes soddisfino i seguenti requisiti:

- **Accessibilità ad AppVault**: sia i cluster di origine che quelli di destinazione devono avere accesso alla rete per leggere e scrivere su AppVault per la replica degli oggetti applicativi.
- **Connettività di rete**: configura le regole del firewall, le autorizzazioni dei bucket e le liste consentite di IP per abilitare la comunicazione tra entrambi i cluster e AppVault attraverso le WAN.

> Molti ambienti aziendali implementano rigide policy firewall sulle connessioni WAN. Verificare questi requisiti di rete con il team dell'infrastruttura prima di configurare la replica.

Installa e configura Trident Protect

Se il tuo ambiente soddisfa i requisiti per Trident Protect, puoi seguire questi passaggi per installare Trident Protect sul tuo cluster. Puoi ottenere Trident Protect da NetApp oppure installarlo dal tuo registro privato. L'installazione da un registro privato è utile se il cluster non riesce ad accedere a Internet.

Installa Trident Protect

Installa Trident Protect da NetApp

Passi

1. Aggiungere il repository Trident Helm:

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

2. Utilizzare Helm per installare Trident Protect. Sostituire `<name-of-cluster>` con un nome cluster, che verrà assegnato al cluster e utilizzato per identificare i backup e gli snapshot del cluster:

```
helm install trident-protect netapp-trident-protect/trident-protect
--set clusterName=<name-of-cluster> --version 100.2506.0 --create
-namespace --namespace trident-protect
```

Installa Trident Protect da un registro privato

È possibile installare Trident Protect da un registro di immagini privato se il cluster Kubernetes non riesce ad accedere a Internet. In questi esempi, sostituisci i valori tra parentesi con le informazioni provenienti dal tuo ambiente:

Passi

1. Estrai le seguenti immagini sul tuo computer locale, aggiorna i tag e poi inseriscile nel tuo registro privato:

```
netapp/controller:25.06.0
netapp/restic:25.06.0
netapp/kopia:25.06.0
netapp/trident-autosupport:25.06.0
netapp/exechook:25.06.0
netapp/resourcebackup:25.06.0
netapp/resourcerestore:25.06.0
netapp/resourcedelete:25.06.0
bitnami/kubectl:1.30.2
kubebuilder/kube-rbac-proxy:v0.16.0
```

   Per esempio:

```
docker pull netapp/controller:25.06.0
```

```
docker tag netapp/controller:25.06.0 <private-registry-
url>/controller:25.06.0
```

```
docker push <private-registry-url>/controller:25.06.0
```

2. Creare lo spazio dei nomi del sistema di protezione Trident :

```
kubectl create ns trident-protect
```

3. Accedi al registro:

```
helm registry login <private-registry-url> -u <account-id> -p <api-token>
```

4. Crea un segreto pull da utilizzare per l'autenticazione del registro privato:

```
kubectl create secret docker-registry regcred --docker-username=<registry-username> --docker-password=<api-token> -n trident-protect --docker-server=<private-registry-url>
```

5. Aggiungere il repository Trident Helm:

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

6. Crea un file denominato `protectValues.yaml` . Assicurarsi che contenga le seguenti impostazioni di protezione Trident :

```
---
image:
  registry: <private-registry-url>
imagePullSecrets:
  - name: regcred
controller:
  image:
    registry: <private-registry-url>
rbacProxy:
  image:
    registry: <private-registry-url>
crCleanup:
  imagePullSecrets:
    - name: regcred
webhooksCleanup:
  imagePullSecrets:
    - name: regcred
```

7. Utilizzare Helm per installare Trident Protect. Sostituire `<name_of_cluster>` con un nome cluster, che verrà assegnato al cluster e utilizzato per identificare i backup e gli snapshot del cluster:

```
helm install trident-protect netapp-trident-protect/trident-protect
--set clusterName=<name_of_cluster> --version 100.2506.0 --create
-namespace --namespace trident-protect -f protectValues.yaml
```

Installa il plugin Trident Protect CLI

È possibile utilizzare il plugin della riga di comando Trident Protect, che è un'estensione di Trident `tridentctl` utilità, per creare e interagire con le risorse personalizzate (CR) di protezione Trident .

Installa il plugin Trident Protect CLI

Prima di utilizzare l'utilità della riga di comando, è necessario installarla sul computer utilizzato per accedere al cluster. Seguire questi passaggi, a seconda che il computer utilizzi una CPU x64 o ARM .

Scarica il plugin per le CPU Linux AMD64

Passi

1. Scarica il plugin Trident Protect CLI:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-linux-amd64
```

Scarica il plugin per le CPU Linux ARM64

Passi

1. Scarica il plugin Trident Protect CLI:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-linux-arm64
```

Scarica il plugin per le CPU AMD64 di Mac

Passi

1. Scarica il plugin Trident Protect CLI:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-macos-amd64
```

Scarica il plugin per le CPU Mac ARM64

Passi

1. Scarica il plugin Trident Protect CLI:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/25.06.0/tridentctl-protect-macos-arm64
```

1. Abilita i permessi di esecuzione per il binario del plugin:

```
chmod +x tridentctl-protect
```

2. Copia il file binario del plugin in una posizione definita nella variabile PATH. Per esempio, `/usr/bin` O `/usr/local/bin` (potrebbero essere necessari privilegi elevati):

```
cp ./tridentctl-protect /usr/local/bin/
```

3. Facoltativamente, puoi copiare il file binario del plugin in una posizione nella tua directory home. In questo caso, si consiglia di assicurarsi che la posizione faccia parte della variabile PATH:

```
cp ./tridentctl-protect ~/bin/
```

Copiando il plugin in una posizione nella variabile PATH è possibile utilizzare il plugin digitando `tridentctl-protect` O `tridentctl protect` da qualsiasi luogo.

Visualizza la guida del plugin Trident CLI

È possibile utilizzare le funzionalità di aiuto integrate nel plugin per ottenere assistenza dettagliata sulle funzionalità del plugin:

Passi

1. Utilizzare la funzione di aiuto per visualizzare le istruzioni per l'uso:

```
tridentctl-protect help
```

Abilita il completamento automatico dei comandi

Dopo aver installato il plugin Trident Protect CLI, è possibile abilitare il completamento automatico per determinati comandi.

Abilita il completamento automatico per la shell Bash

Passi

1. Scarica lo script di completamento:

```
curl -L -O https://github.com/NetApp/tridentctl-
protect/releases/download/25.06.0/tridentctl-completion.bash
```

2. Crea una nuova directory nella tua directory home per contenere lo script:

```
mkdir -p ~/.bash/completions
```

3. Sposta lo script scaricato in `~/.bash/completions` elenco:

```
mv tridentctl-completion.bash ~/.bash/completions/
```

4. Aggiungere la seguente riga al `~/.bashrc` file nella tua directory home:

```
source ~/.bash/completions/tridentctl-completion.bash
```

Abilita il completamento automatico per la shell Z

Passi

1. Scarica lo script di completamento:

```
curl -L -O https://github.com/NetApp/tridentctl-
protect/releases/download/25.06.0/tridentctl-completion.zsh
```

2. Crea una nuova directory nella tua directory home per contenere lo script:

```
mkdir -p ~/.zsh/completions
```

3. Sposta lo script scaricato in `~/.zsh/completions` elenco:

```
mv tridentctl-completion.zsh ~/.zsh/completions/
```

4. Aggiungere la seguente riga al `~/.zprofile` file nella tua directory home:

```
source ~/.zsh/completions/tridentctl-completion.zsh
```

Risultato

Al successivo accesso alla shell, è possibile utilizzare il completamento automatico dei comandi con il plugin tridentctl-protect.

Personalizza l'installazione Trident Protect

È possibile personalizzare la configurazione predefinita di Trident Protect per soddisfare i requisiti specifici del proprio ambiente.

Specificare i limiti delle risorse del contenitore di protezione Trident

Dopo aver installato Trident Protect, è possibile utilizzare un file di configurazione per specificare i limiti delle risorse per i contenitori Trident Protect. Impostando i limiti delle risorse è possibile controllare la quantità di risorse del cluster consumata dalle operazioni di protezione Trident .

Passi

1. Crea un file denominato `resourceLimits.yaml` .
2. Compilare il file con le opzioni di limite delle risorse per i contenitori Trident Protect in base alle esigenze del proprio ambiente.

   Il seguente file di configurazione di esempio mostra le impostazioni disponibili e contiene i valori predefiniti per ciascun limite di risorse:

```
---
jobResources:
  defaults:
    limits:
      cpu: 8000m
      memory: 10000Mi
      ephemeralStorage: ""
    requests:
      cpu: 100m
      memory: 100Mi
      ephemeralStorage: ""
  resticVolumeBackup:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  resticVolumeRestore:
    limits:
      cpu: ""
      memory: ""
```

```
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  kopiaVolumeBackup:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  kopiaVolumeRestore:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
```

3. Applicare i valori da `resourceLimits.yaml` file:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f resourceLimits.yaml --reuse-values
```

Personalizza i vincoli del contesto di sicurezza

È possibile utilizzare un file di configurazione per modificare i vincoli di contesto di sicurezza (SCC) di OpenShift per i contenitori Trident Protect dopo aver installato Trident Protect. Questi vincoli definiscono le restrizioni di sicurezza per i pod in un cluster Red Hat OpenShift.

Passi

1. Crea un file denominato `sccconfig.yaml` .
2. Aggiungere l'opzione SCC al file e modificare i parametri in base alle esigenze del proprio ambiente.

   L'esempio seguente mostra i valori predefiniti dei parametri per l'opzione SCC:

```
scc:
  create: true
  name: trident-protect-job
  priority: 1
```

Questa tabella descrive i parametri per l'opzione SCC:

Parametro	Descrizione	Predefinito
creare	Determina se è possibile creare una risorsa SCC. Una risorsa SCC verrà creata solo se `scc.create` è impostato su `true` e il processo di installazione di Helm identifica un ambiente OpenShift. Se non si opera su OpenShift, o se `scc.create` è impostato su `false` , non verrà creata alcuna risorsa SCC.	VERO
nome	Specifica il nome dell'SCC.	trident-protect-job
priorità	Definisce la priorità dell'SCC. Gli SCC con valori di priorità più elevati vengono valutati prima di quelli con valori più bassi.	1

3. Applicare i valori da `sccconfig.yaml` file:

```
helm upgrade trident-protect netapp-trident-protect/trident-protect -f
sccconfig.yaml --reuse-values
```

Ciò sostituirà i valori predefiniti con quelli specificati nel `sccconfig.yaml` file.

Configurare le impostazioni aggiuntive del grafico del timone di protezione Trident

È possibile personalizzare le impostazioni AutoSupport e il filtraggio degli spazi dei nomi in base alle proprie esigenze specifiche. La tabella seguente descrive i parametri di configurazione disponibili:

Parametro	Tipo	Descrizione
autoSupport.proxy	corda	Configura un URL proxy per le connessioni NetApp AutoSupport . Utilizzare questa opzione per instradare i caricamenti dei pacchetti di supporto tramite un server proxy. Esempio: `http://my.proxy.url` .

Parametro	Tipo	Descrizione
autoSupport.insicuro	booleano	Salta la verifica TLS per le connessioni proxy AutoSupport quando impostato su `true` . Utilizzare solo per connessioni proxy non sicure. (predefinito: `false`)
autoSupport.abilitato	booleano	Abilita o disabilita i caricamenti giornalieri del bundle Trident Protect AutoSupport . Quando impostato su `false` , i caricamenti giornalieri programmati sono disabilitati, ma puoi comunque generare manualmente i pacchetti di supporto. (predefinito: `true`)
restoreSkipNamespaceAnnotations	corda	Elenco separato da virgole di annotazioni dello spazio dei nomi da escludere dalle operazioni di backup e ripristino. Consente di filtrare gli spazi dei nomi in base alle annotazioni.
ripristina Salta le etichette dello spazio dei nomi	corda	Elenco separato da virgole delle etichette degli spazi dei nomi da escludere dalle operazioni di backup e ripristino. Consente di filtrare gli spazi dei nomi in base alle etichette.

È possibile configurare queste opzioni utilizzando un file di configurazione YAML o i flag della riga di comando:

Utilizzare il file YAML

Passi

1. Crea un file di configurazione e assegnagli un nome `values.yaml` .
2. Nel file creato, aggiungi le opzioni di configurazione che desideri personalizzare.

```
autoSupport:
  enabled: false
  proxy: http://my.proxy.url
  insecure: true
restoreSkipNamespaceAnnotations: "annotation1,annotation2"
restoreSkipNamespaceLabels: "label1,label2"
```

3. Dopo aver popolato il `values.yaml` file con i valori corretti, applicare il file di configurazione:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f values.yaml --reuse-values
```

Usa il flag CLI

Passi

1. Utilizzare il seguente comando con il `--set` flag per specificare parametri individuali:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect \
  --set autoSupport.enabled=false \
  --set autoSupport.proxy=http://my.proxy.url \
  --set restoreSkipNamespaceAnnotations="annotation1,annotation2" \
  --set restoreSkipNamespaceLabels="label1,label2" \
  --reuse-values
```

Limita i pod di protezione Trident a nodi specifici

È possibile utilizzare il vincolo di selezione dei nodi Kubernetes nodeSelector per controllare quali nodi sono idonei a eseguire i pod Trident Protect, in base alle etichette dei nodi. Per impostazione predefinita, Trident Protect è limitato ai nodi che eseguono Linux. È possibile personalizzare ulteriormente questi vincoli in base alle proprie esigenze.

Passi

1. Crea un file denominato `nodeSelectorConfig.yaml` .
2. Aggiungere l'opzione nodeSelector al file e modificare il file per aggiungere o modificare le etichette dei nodi in modo da limitare le restrizioni in base alle esigenze del proprio ambiente. Ad esempio, il file seguente contiene la restrizione predefinita del sistema operativo, ma ha anche come target una regione

specifica e un nome di app:

```
nodeSelector:
  kubernetes.io/os: linux
  region: us-west
  app.kubernetes.io/name: mysql
```

3. Applicare i valori da `nodeSelectorConfig.yaml` file:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f nodeSelectorConfig.yaml --reuse-values
```

Questo sostituisce le restrizioni predefinite con quelle specificate nel `nodeSelectorConfig.yaml` file.

Gestisci Trident proteggi

Gestisci l'autorizzazione di protezione e il controllo degli accessi Trident

Trident Protect utilizza il modello Kubernetes di controllo degli accessi basato sui ruoli (RBAC). Per impostazione predefinita, Trident Protect fornisce un singolo namespace di sistema e il relativo account di servizio predefinito. Se la tua organizzazione ha molti utenti o esigenze di sicurezza specifiche, puoi utilizzare le funzionalità RBAC di Trident Protect per ottenere un controllo più granulare sull'accesso alle risorse e agli spazi dei nomi.

L'amministratore del cluster ha sempre accesso alle risorse predefinite `trident-protect` namespace e può anche accedere alle risorse in tutti gli altri namespace. Per controllare l'accesso alle risorse e alle applicazioni, è necessario creare namespace aggiuntivi e aggiungere risorse e applicazioni a tali namespace.

Si noti che nessun utente può creare CR di gestione dei dati delle applicazioni nell'impostazione predefinita `trident-protect` spazio dei nomi. È necessario creare CR di gestione dei dati dell'applicazione in uno spazio dei nomi dell'applicazione (come buona pratica, creare CR di gestione dei dati dell'applicazione nello stesso spazio dei nomi dell'applicazione associata).

> Solo gli amministratori dovrebbero avere accesso agli oggetti di risorse personalizzati protetti Trident privilegiati, tra cui:
>
> - **AppVault**: richiede i dati delle credenziali del bucket
> - **AutoSupportBundle**: raccoglie metriche, registri e altri dati sensibili Trident Protect
> - **AutoSupportBundleSchedule**: Gestisce le pianificazioni della raccolta dei log
>
> Come buona pratica, utilizzare RBAC per limitare l'accesso agli oggetti privilegiati ai soli amministratori.

Per ulteriori informazioni su come RBAC regola l'accesso alle risorse e agli spazi dei nomi, fare riferimento a "Documentazione di Kubernetes RBAC" .

Per informazioni sugli account di servizio, fare riferimento a "Documentazione dell'account di servizio Kubernetes" .

Esempio: Gestire l'accesso per due gruppi di utenti

Ad esempio, un'organizzazione ha un amministratore del cluster, un gruppo di utenti di ingegneria e un gruppo di utenti di marketing. L'amministratore del cluster dovrebbe completare le seguenti attività per creare un ambiente in cui il gruppo di ingegneria e il gruppo di marketing abbiano accesso solo alle risorse assegnate ai rispettivi namespace.

Passaggio 1: creare uno spazio dei nomi per contenere le risorse per ciascun gruppo

La creazione di uno spazio dei nomi consente di separare logicamente le risorse e di controllare meglio chi ha accesso a tali risorse.

Passi

1. Creare uno spazio dei nomi per il gruppo di ingegneria:

```
kubectl create ns engineering-ns
```

2. Crea uno spazio dei nomi per il gruppo di marketing:

```
kubectl create ns marketing-ns
```

Passaggio 2: creare nuovi account di servizio per interagire con le risorse in ogni spazio dei nomi

Ogni nuovo namespace creato è dotato di un account di servizio predefinito, ma è consigliabile creare un account di servizio per ogni gruppo di utenti, in modo da poter suddividere ulteriormente i privilegi tra i gruppi in futuro, se necessario.

Passi

1. Creare un account di servizio per il gruppo di ingegneria:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: eng-user
  namespace: engineering-ns
```

2. Crea un account di servizio per il gruppo marketing:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: mkt-user
  namespace: marketing-ns
```

Passaggio 3: creare un segreto per ogni nuovo account di servizio

Un segreto dell'account di servizio viene utilizzato per l'autenticazione con l'account di servizio e può essere facilmente eliminato e ricreato se compromesso.

Passi

1. Crea un segreto per l'account del servizio di ingegneria:

```
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: eng-user
  name: eng-user-secret
  namespace: engineering-ns
type: kubernetes.io/service-account-token
```

2. Crea un segreto per l'account del servizio di marketing:

```
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: mkt-user
  name: mkt-user-secret
  namespace: marketing-ns
type: kubernetes.io/service-account-token
```

Passaggio 4: creare un oggetto RoleBinding per associare l'oggetto ClusterRole a ciascun nuovo account di servizio

Quando si installa Trident Protect, viene creato un oggetto ClusterRole predefinito. È possibile associare questo ClusterRole all'account di servizio creando e applicando un oggetto RoleBinding.

Passi

1. Associare ClusterRole all'account del servizio di ingegneria:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: engineering-ns-tenant-rolebinding
  namespace: engineering-ns
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: trident-protect-tenant-cluster-role
subjects:
- kind: ServiceAccount
  name: eng-user
  namespace: engineering-ns
```

2. Associa ClusterRole all'account del servizio di marketing:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: marketing-ns-tenant-rolebinding
  namespace: marketing-ns
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: trident-protect-tenant-cluster-role
subjects:
- kind: ServiceAccount
  name: mkt-user
  namespace: marketing-ns
```

Passaggio 5: testare le autorizzazioni

Verificare che i permessi siano corretti.

Passi

1. Verificare che gli utenti di ingegneria possano accedere alle risorse di ingegneria:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get applications.protect.trident.netapp.io -n engineering-ns
```

2. Verificare che gli utenti di ingegneria non possano accedere alle risorse di marketing:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get applications.protect.trident.netapp.io -n marketing-ns
```

Passaggio 6: Concedi l'accesso agli oggetti AppVault

Per eseguire attività di gestione dei dati quali backup e snapshot, l'amministratore del cluster deve concedere l'accesso agli oggetti AppVault ai singoli utenti.

Passi

1. Crea e applica un file YAML con combinazione di AppVault e segreto che garantisce a un utente l'accesso a un AppVault. Ad esempio, il seguente CR concede l'accesso a un AppVault all'utente `eng-user` :

```
apiVersion: v1
data:
  accessKeyID: <ID_value>
  secretAccessKey: <key_value>
kind: Secret
metadata:
  name: appvault-for-eng-user-only-secret
  namespace: trident-protect
type: Opaque
---
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: appvault-for-eng-user-only
  namespace: trident-protect # Trident protect system namespace
spec:
  providerConfig:
    azure:
      accountName: ""
      bucketName: ""
      endpoint: ""
    gcp:
      bucketName: ""
      projectID: ""
    s3:
      bucketName: testbucket
      endpoint: 192.168.0.1:30000
      secure: "false"
      skipCertValidation: "true"
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: appvault-for-eng-user-only-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: appvault-for-eng-user-only-secret
  providerType: GenericS3
```

2. Creare e applicare un CR di ruolo per consentire agli amministratori del cluster di concedere l'accesso a risorse specifiche in uno spazio dei nomi. Per esempio:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: eng-user-appvault-reader
  namespace: trident-protect
rules:
- apiGroups:
  - protect.trident.netapp.io
  resourceNames:
  - appvault-for-enguser-only
  resources:
  - appvaults
  verbs:
  - get
```

3. Creare e applicare un CR RoleBinding per associare le autorizzazioni all'utente eng-user. Per esempio:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: eng-user-read-appvault-binding
  namespace: trident-protect
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: eng-user-appvault-reader
subjects:
- kind: ServiceAccount
  name: eng-user
  namespace: engineering-ns
```

4. Verificare che le autorizzazioni siano corrette.

   a. Tentativo di recuperare le informazioni sugli oggetti AppVault per tutti gli spazi dei nomi:

   ```
   kubectl get appvaults -n trident-protect
   --as=system:serviceaccount:engineering-ns:eng-user
   ```

   Dovresti vedere un output simile al seguente:

```
Error from server (Forbidden): appvaults.protect.trident.netapp.io is
forbidden: User "system:serviceaccount:engineering-ns:eng-user"
cannot list resource "appvaults" in API group
"protect.trident.netapp.io" in the namespace "trident-protect"
```

b. Prova a verificare se l'utente riesce a ottenere le informazioni di AppVault a cui ora ha l'autorizzazione ad accedere:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get appvaults.protect.trident.netapp.io/appvault-for-eng-user-only -n
trident-protect
```

Dovresti vedere un output simile al seguente:

```
yes
```

Risultato

Gli utenti a cui hai concesso le autorizzazioni AppVault devono essere in grado di utilizzare oggetti AppVault autorizzati per le operazioni di gestione dei dati dell'applicazione e non devono essere in grado di accedere a risorse esterne agli spazi dei nomi assegnati o di creare nuove risorse a cui non hanno accesso.

Monitora le risorse di protezione Trident

È possibile utilizzare gli strumenti open source kube-state-metrics, Prometheus e Alertmanager per monitorare lo stato di integrità delle risorse protette da Trident Protect.

Il servizio kube-state-metrics genera metriche dalla comunicazione API di Kubernetes. Utilizzandolo con Trident Protect, puoi ottenere informazioni utili sullo stato delle risorse nel tuo ambiente.

Prometheus è un toolkit in grado di acquisire i dati generati da kube-state-metrics e presentarli come informazioni facilmente leggibili su questi oggetti. Insieme, kube-state-metrics e Prometheus ti consentono di monitorare lo stato e l'integrità delle risorse che gestisci con Trident Protect.

Alertmanager è un servizio che acquisisce gli avvisi inviati da strumenti come Prometheus e li indirizza verso destinazioni configurate dall'utente.

> Le configurazioni e le istruzioni incluse in questi passaggi sono solo esempi; è necessario personalizzarle in base al proprio ambiente. Per istruzioni e supporto specifici, fare riferimento alla seguente documentazione ufficiale:
>
> - "documentazione di kube-state-metrics"
> - "Documentazione di Prometheus"
> - "Documentazione di Alertmanager"

Passaggio 1: installare gli strumenti di monitoraggio

Per abilitare il monitoraggio delle risorse in Trident Protect, è necessario installare e configurare kube-state-metrics, Prometheus e Alertmanager.

Installa kube-state-metrics

Puoi installare kube-state-metrics tramite Helm.

Passi

1. Aggiungere il grafico Helm kube-state-metrics. Per esempio:

```
helm repo add prometheus-community https://prometheus-
community.github.io/helm-charts
helm repo update
```

2. Applicare il CRD di Prometheus ServiceMonitor al cluster:

```
kubectl apply -f https://raw.githubusercontent.com/prometheus-
operator/prometheus-operator/main/example/prometheus-operator-
crd/monitoring.coreos.com_servicemonitors.yaml
```

3. Creare un file di configurazione per il grafico Helm (ad esempio, `metrics-config.yaml`). È possibile personalizzare la seguente configurazione di esempio in base al proprio ambiente:

metrics-config.yaml: configurazione del grafico Helm kube-state-metrics

```
---
extraArgs:
  # Collect only custom metrics
  - --custom-resource-state-only=true

customResourceState:
  enabled: true
  config:
    kind: CustomResourceStateMetrics
    spec:
      resources:
      - groupVersionKind:
          group: protect.trident.netapp.io
          kind: "Backup"
          version: "v1"
        labelsFromPath:
          backup_uid: [metadata, uid]
          backup_name: [metadata, name]
          creation_time: [metadata, creationTimestamp]
        metrics:
        - name: backup_info
          help: "Exposes details about the Backup state"
          each:
            type: Info
            info:
              labelsFromPath:
                appVaultReference: ["spec", "appVaultRef"]
                appReference: ["spec", "applicationRef"]
rbac:
  extraRules:
  - apiGroups: ["protect.trident.netapp.io"]
    resources: ["backups"]
    verbs: ["list", "watch"]

# Collect metrics from all namespaces
namespaces: ""

# Ensure that the metrics are collected by Prometheus
prometheus:
  monitor:
    enabled: true
```

4. Installa kube-state-metrics distribuendo il grafico Helm. Per esempio:

```
helm install custom-resource -f metrics-config.yaml prometheus-
community/kube-state-metrics --version 5.21.0
```

5. Configurare kube-state-metrics per generare metriche per le risorse personalizzate utilizzate da Trident Protect seguendo le istruzioni in "documentazione delle risorse personalizzate kube-state-metrics" .

Installa Prometheus

Puoi installare Prometheus seguendo le istruzioni nel "Documentazione di Prometheus" .

Installa Alertmanager

Puoi installare Alertmanager seguendo le istruzioni nel "Documentazione di Alertmanager" .

Passaggio 2: configurare gli strumenti di monitoraggio affinché funzionino insieme

Dopo aver installato gli strumenti di monitoraggio, è necessario configurarli affinché funzionino insieme.

Passi

1. Integra kube-state-metrics con Prometheus. Modifica il file di configurazione di Prometheus(`prometheus.yaml`) e aggiungere le informazioni sul servizio kube-state-metrics. Per esempio:

   prometheus.yaml: integrazione del servizio kube-state-metrics con Prometheus

```
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: prometheus-config
  namespace: trident-protect
data:
  prometheus.yaml: |
    global:
      scrape_interval: 15s
    scrape_configs:
      - job_name: 'kube-state-metrics'
        static_configs:
          - targets: ['kube-state-metrics.trident-protect.svc:8080']
```

2. Configurare Prometheus per indirizzare gli avvisi ad Alertmanager. Modifica il file di configurazione di Prometheus(`prometheus.yaml`) e aggiungere la seguente sezione:

prometheus.yaml: Invia avvisi ad Alertmanager

```
alerting:
  alertmanagers:
    - static_configs:
        - targets:
            - alertmanager.trident-protect.svc:9093
```

Risultato

Prometheus ora può raccogliere metriche da kube-state-metrics e può inviare avvisi ad Alertmanager. Ora sei pronto per configurare quali condizioni attivano un avviso e dove devono essere inviati gli avvisi.

Passaggio 3: configurare gli avvisi e le destinazioni degli avvisi

Dopo aver configurato gli strumenti affinché funzionino insieme, è necessario configurare il tipo di informazioni che attivano gli avvisi e dove devono essere inviati.

Esempio di avviso: errore di backup

L'esempio seguente definisce un avviso critico che viene attivato quando lo stato della risorsa personalizzata di backup è impostato su `Error` per 5 secondi o più. Puoi personalizzare questo esempio per adattarlo al tuo ambiente e includere questo frammento YAML nel tuo `prometheus.yaml` file di configurazione:

rules.yaml: Definisci un avviso Prometheus per i backup non riusciti

```
rules.yaml: |
  groups:
    - name: fail-backup
        rules:
          - alert: BackupFailed
            expr: kube_customresource_backup_info{status="Error"}
            for: 5s
            labels:
              severity: critical
            annotations:
              summary: "Backup failed"
              description: "A backup has failed."
```

Configura Alertmanager per inviare avvisi ad altri canali

È possibile configurare Alertmanager per inviare notifiche ad altri canali, come e-mail, PagerDuty, Microsoft Teams o altri servizi di notifica specificando la rispettiva configurazione in `alertmanager.yaml` file.

L'esempio seguente configura Alertmanager per inviare notifiche a un canale Slack. Per personalizzare questo esempio in base al tuo ambiente, sostituisci il valore di `api_url` chiave con l'URL del webhook Slack utilizzato nel tuo ambiente:

alertmanager.yaml: invia avvisi a un canale Slack

```
data:
  alertmanager.yaml: |
    global:
      resolve_timeout: 5m
    route:
      receiver: 'slack-notifications'
    receivers:
      - name: 'slack-notifications'
        slack_configs:
          - api_url: '<your-slack-webhook-url>'
            channel: '#failed-backups-channel'
            send_resolved: false
```

Genera un pacchetto di supporto Trident Protect

Trident Protect consente agli amministratori di generare bundle che includono informazioni utili al supporto NetApp , tra cui registri, metriche e informazioni sulla topologia dei cluster e delle app in gestione. Se sei connesso a Internet, puoi caricare i bundle di supporto sul sito di supporto NetApp (NSS) utilizzando un file di risorse personalizzato (CR).

Crea un pacchetto di supporto utilizzando una CR

Passi

1. Crea il file di risorse personalizzate (CR) e assegnagli un nome (ad esempio, `trident-protect-support-bundle.yaml`).
2. Configurare i seguenti attributi:
   - **metadata.name**: (*Obbligatorio*) Il nome di questa risorsa personalizzata; scegli un nome univoco e sensato per il tuo ambiente.
   - **spec.triggerType**: (*Obbligatorio*) Determina se il pacchetto di supporto viene generato immediatamente o pianificato. La generazione programmata del bundle avviene alle 00:00 UTC. Valori possibili:
     - Programmato
     - Manuale
   - **spec.uploadEnabled**: (*Facoltativo*) Controlla se il bundle di supporto deve essere caricato sul sito di supporto NetApp dopo essere stato generato. Se non specificato, il valore predefinito è `false` . Valori possibili:
     - VERO
     - falso (predefinito)
   - **spec.dataWindowStart**: (*Facoltativo*) Una stringa di data nel formato RFC 3339 che specifica la data e l'ora in cui deve iniziare la finestra dei dati inclusi nel bundle di supporto. Se non specificato, il valore predefinito è 24 ore fa. La prima data possibile è 7 giorni fa.

     Esempio YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: AutoSupportBundle
metadata:
  name: trident-protect-support-bundle
spec:
  triggerType: Manual
  uploadEnabled: true
  dataWindowStart: 2024-05-05T12:30:00Z
```

3. Dopo aver popolato il `trident-protect-support-bundle.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-support-bundle.yaml -n trident-protect
```

Crea un pacchetto di supporto utilizzando la CLI

Passi

1. Crea il pacchetto di supporto, sostituendo i valori tra parentesi con le informazioni provenienti dal tuo

ambiente. IL `trigger-type` determina se il bundle viene creato immediatamente o se il tempo di creazione è dettato dalla pianificazione e può essere `Manual` O `Scheduled` . L'impostazione predefinita è `Manual` .

Per esempio:

```
tridentctl-protect create autosupportbundle <my-bundle-name>
--trigger-type <trigger-type> -n trident-protect
```

Monitorare e recuperare il pacchetto di supporto

Dopo aver creato un pacchetto di supporto utilizzando uno dei due metodi, puoi monitorarne l'avanzamento della generazione e recuperarlo nel tuo sistema locale.

Passi

1. Aspetta il `status.generationState` raggiungere `Completed` stato. È possibile monitorare l'avanzamento della generazione con il seguente comando:

   ```
   kubectl get autosupportbundle trident-protect-support-bundle -n trident-
   protect
   ```

2. Recupera il pacchetto di supporto sul tuo sistema locale. Ottieni il comando di copia dal bundle AutoSupport completato:

   ```
   kubectl describe autosupportbundle trident-protect-support-bundle -n
   trident-protect
   ```

   Trova il `kubectl cp` comando dall'output ed eseguilo, sostituendo l'argomento di destinazione con la directory locale preferita.

Aggiorna la protezione Trident

È possibile aggiornare Trident Protect all'ultima versione per sfruttare le nuove funzionalità o le correzioni di bug.

> Quando si esegue l'aggiornamento dalla versione 24.10, gli snapshot in esecuzione durante l'aggiornamento potrebbero non funzionare. Questo errore non impedisce la creazione di snapshot futuri, manuali o pianificati. Se uno snapshot non riesce durante l'aggiornamento, puoi crearne manualmente uno nuovo per garantire la protezione dell'applicazione.
>
> Per evitare potenziali errori, è possibile disattivare tutte le pianificazioni degli snapshot prima dell'aggiornamento e riattivarle in seguito. Tuttavia, ciò comporta la perdita di tutti gli snapshot pianificati durante il periodo di aggiornamento.

Per aggiornare Trident Protect, procedere come segue.

Passi

1. Aggiorna il repository Trident Helm:

```
helm repo update
```

2. Aggiorna i CRD di protezione Trident :

> Questo passaggio è necessario se si esegue l'aggiornamento da una versione precedente alla 25.06, poiché i CRD sono ora inclusi nella tabella Trident Protect Helm.

a. Eseguire questo comando per spostare la gestione dei CRD da `trident-protect-crds` A `trident-protect` :

```
kubectl get crd | grep protect.trident.netapp.io | awk '{print $1}' |
xargs -I {} kubectl patch crd {} --type merge -p '{"metadata":
{"annotations":{"meta.helm.sh/release-name": "trident-protect"}}}'
```

b. Eseguire questo comando per eliminare il segreto Helm per `trident-protect-crds` grafico:

> Non disinstallare il `trident-protect-crds` grafico utilizzando Helm, poiché ciò potrebbe rimuovere i CRD e tutti i dati correlati.

```
kubectl delete secret -n trident-protect -l name=trident-protect-
crds,owner=helm
```

3. Aggiorna la protezione Trident :

```
helm upgrade trident-protect netapp-trident-protect/trident-protect
--version 100.2506.0 --namespace trident-protect
```

Gestire e proteggere le applicazioni

Utilizzare Trident per proteggere gli oggetti AppVault per gestire i bucket

La risorsa personalizzata del bucket (CR) per Trident Protect è nota come AppVault. Gli oggetti AppVault sono la rappresentazione dichiarativa del flusso di lavoro Kubernetes di un bucket di archiviazione. Un CR di AppVault contiene le configurazioni necessarie affinché un bucket venga utilizzato nelle operazioni di protezione, come backup, snapshot, operazioni di ripristino e replica SnapMirror . Solo gli amministratori possono creare AppVault.

Quando si eseguono operazioni di protezione dei dati su un'applicazione, è necessario creare una CR di

AppVault manualmente o dalla riga di comando. La CR di AppVault è specifica per il proprio ambiente e gli esempi in questa pagina possono essere utilizzati come guida per la creazione di CR di AppVault.

> Assicurarsi che la CR di AppVault si trovi sul cluster in cui è installato Trident Protect. Se la CR di AppVault non esiste o non è possibile accedervi, la riga di comando mostrerà un errore.

Configurare l'autenticazione e le password di AppVault

Prima di creare una CR di AppVault, assicurati che AppVault e il data mover scelto possano autenticarsi con il provider e con tutte le risorse correlate.

Password del repository del data mover

Quando si creano oggetti AppVault utilizzando le CR o il plugin Trident Protect CLI, è possibile specificare un segreto Kubernetes con password personalizzate per la crittografia Restic e Kopia. Se non si specifica un segreto, Trident Protect utilizza una password predefinita.

- Quando si creano manualmente CR di AppVault, utilizzare il campo **spec.dataMoverPasswordSecretRef** per specificare il segreto.
- Quando si creano oggetti AppVault utilizzando la CLI Trident Protect, utilizzare `--data-mover -password-secret-ref` argomento per specificare il segreto.

Crea una password segreta per il repository del data mover

Utilizzare i seguenti esempi per creare la password segreta. Quando si creano oggetti AppVault, è possibile indicare a Trident Protect di utilizzare questo segreto per l'autenticazione con il repository del data mover.

> - A seconda del data mover utilizzato, è sufficiente includere la password corrispondente per quel data mover. Ad esempio, se utilizzi Restic e non prevedi di utilizzare Kopia in futuro, puoi includere solo la password Restic quando crei il segreto.
> - Conservare la password in un luogo sicuro. Sarà necessaria per ripristinare i dati sullo stesso cluster o su uno diverso. Se il cluster o il `trident-protect` Se lo spazio dei nomi viene eliminato, non sarà possibile ripristinare i backup o gli snapshot senza la password.

Utilizzare un CR

```
---
apiVersion: v1
data:
  KOPIA_PASSWORD: <base64-encoded-password>
  RESTIC_PASSWORD: <base64-encoded-password>
kind: Secret
metadata:
  name: my-optional-data-mover-secret
  namespace: trident-protect
type: Opaque
```

Utilizzare la CLI

```
kubectl create secret generic my-optional-data-mover-secret \
--from-literal=KOPIA_PASSWORD=<plain-text-password> \
--from-literal=RESTIC_PASSWORD=<plain-text-password> \
-n trident-protect
```

Autorizzazioni IAM di archiviazione compatibili con S3

Quando si accede a un archivio compatibile con S3 come Amazon S3, Generic S3, "StorageGrid S3" , O "ONTAP S3" Utilizzando Trident Protect, è necessario assicurarsi che le credenziali utente fornite dispongano delle autorizzazioni necessarie per accedere al bucket. Di seguito è riportato un esempio di policy che concede le autorizzazioni minime richieste per l'accesso con Trident Protect. È possibile applicare questo criterio all'utente che gestisce i criteri dei bucket compatibili con S3.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:ListBucket",
        "s3:DeleteObject"
      ],
      "Resource": "*"
    }
  ]
}
```

Per ulteriori informazioni sulle policy di Amazon S3, fare riferimento agli esempi nel "Documentazione di Amazon S3" .

Identità pod EKS per l'autenticazione Amazon S3 (AWS)

Trident Protect supporta EKS Pod Identity per le operazioni di spostamento dati Kopia. Questa funzionalità consente l'accesso sicuro ai bucket S3 senza dover archiviare le credenziali AWS nei segreti di Kubernetes.

*Requisiti per l'identità del pod EKS con protezione Trident *

Prima di utilizzare EKS Pod Identity con Trident Protect, accertarsi di quanto segue:

- Il tuo cluster EKS ha l'identità Pod abilitata.
- Hai creato un ruolo IAM con le autorizzazioni necessarie per il bucket S3. Per saperne di più, fare riferimento a "Autorizzazioni IAM di archiviazione compatibili con S3".
- Il ruolo IAM è associato ai seguenti account di servizio Trident Protect:
  - `<trident-protect>-controller-manager`
  - `<trident-protect>-resource-backup`
  - `<trident-protect>-resource-restore`
  - `<trident-protect>-resource-delete`

Per istruzioni dettagliate sull'abilitazione di Pod Identity e sull'associazione dei ruoli IAM agli account di servizio, fare riferimento a "Documentazione sull'identità del pod AWS EKS" .

Configurazione AppVault Quando si utilizza EKS Pod Identity, configurare il CR AppVault con `useIAM: true` flag invece di credenziali esplicite:

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: eks-protect-vault
  namespace: trident-protect
spec:
  providerType: AWS
  providerConfig:
    s3:
      bucketName: trident-protect-aws
      endpoint: s3.example.com
      useIAM: true
```

Esempi di generazione di chiavi AppVault per i provider cloud

Quando si definisce un CR AppVault, è necessario includere le credenziali per accedere alle risorse ospitate dal provider, a meno che non si utilizzi l'autenticazione IAM. Il modo in cui vengono generate le chiavi per le credenziali varia a seconda del provider. Di seguito sono riportati esempi di generazione di chiavi da riga di comando per diversi provider. È possibile utilizzare gli esempi seguenti per creare chiavi per le credenziali di ciascun provider cloud.

Google Cloud

```
kubectl create secret generic <secret-name> \
--from-file=credentials=<mycreds-file.json> \
-n trident-protect
```

Amazon S3 (AWS)

```
kubectl create secret generic <secret-name> \
--from-literal=accessKeyID=<objectstorage-accesskey> \
--from-literal=secretAccessKey=<amazon-s3-trident-protect-src-bucket
-secret> \
-n trident-protect
```

Microsoft Azure

```
kubectl create secret generic <secret-name> \
--from-literal=accountKey=<secret-name> \
-n trident-protect
```

S3 generico

```
kubectl create secret generic <secret-name> \
--from-literal=accessKeyID=<objectstorage-accesskey> \
--from-literal=secretAccessKey=<generic-s3-trident-protect-src-bucket
-secret> \
-n trident-protect
```

ONTAP S3

```
kubectl create secret generic <secret-name> \
--from-literal=accessKeyID=<objectstorage-accesskey> \
--from-literal=secretAccessKey=<ontap-s3-trident-protect-src-bucket
-secret> \
-n trident-protect
```

StorageGrid S3

```
kubectl create secret generic <secret-name> \
--from-literal=accessKeyID=<objectstorage-accesskey> \
--from-literal=secretAccessKey=<storagegrid-s3-trident-protect-src
-bucket-secret> \
-n trident-protect
```

Esempi di creazione di AppVault

Di seguito sono riportati esempi di definizioni di AppVault per ciascun provider.

Esempi di CR di AppVault

È possibile utilizzare i seguenti esempi di CR per creare oggetti AppVault per ciascun provider cloud.

- Facoltativamente, puoi specificare un segreto Kubernetes che contenga password personalizzate per la crittografia dei repository Restic e Kopia. Fare riferimento aPassword del repository del data mover per maggiori informazioni.
- Per gli oggetti Amazon S3 (AWS) AppVault, è possibile specificare facoltativamente un sessionToken, utile se si utilizza l'accesso Single Sign-On (SSO) per l'autenticazione. Questo token viene creato quando si generano le chiavi per il provider inEsempi di generazione di chiavi AppVault per i provider cloud .
- Per gli oggetti S3 AppVault, è possibile specificare facoltativamente un URL proxy di uscita per il traffico S3 in uscita utilizzando `spec.providerConfig.S3.proxyURL` chiave.

Google Cloud

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: gcp-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: GCP
  providerConfig:
    gcp:
      bucketName: trident-protect-src-bucket
      projectID: project-id
  providerCredentials:
    credentials:
      valueFromSecret:
        key: credentials
        name: gcp-trident-protect-src-bucket-secret
```

Amazon S3 (AWS)

```
---
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: amazon-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: AWS
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
    sessionToken:
      valueFromSecret:
        key: sessionToken
        name: s3-secret
```

Per gli ambienti EKS che utilizzano Pod Identity con Kopia Data Mover, è possibile rimuovere `providerCredentials` sezione e aggiungi `useIAM: true` sotto il `s3` configurazione invece.

Microsoft Azure

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: azure-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: Azure
  providerConfig:
    azure:
      accountName: account-name
      bucketName: trident-protect-src-bucket
  providerCredentials:
    accountKey:
      valueFromSecret:
        key: accountKey
        name: azure-trident-protect-src-bucket-secret
```

S3 generico

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: generic-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: GenericS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
```

ONTAP S3

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: ontap-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: OntapS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
```

StorageGrid S3

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: storagegrid-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: StorageGridS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
```

Esempi di creazione di AppVault utilizzando la CLI Trident Protect

È possibile utilizzare i seguenti esempi di comandi CLI per creare CR AppVault per ciascun provider.

- Facoltativamente, puoi specificare un segreto Kubernetes che contenga password personalizzate per la crittografia dei repository Restic e Kopia. Fare riferimento aPassword del repository del data mover per maggiori informazioni.
- Per gli oggetti S3 AppVault, è possibile specificare facoltativamente un URL proxy di uscita per il traffico S3 in uscita utilizzando `--proxy-url <ip_address:port>` discussione.

Google Cloud

```
tridentctl-protect create vault GCP <vault-name> \
--bucket <mybucket> \
--project <my-gcp-project> \
--secret <secret-name>/credentials \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

Amazon S3 (AWS)

```
tridentctl-protect create vault AWS <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

Microsoft Azure

```
tridentctl-protect create vault Azure <vault-name> \
--account <account-name> \
--bucket <bucket-name> \
--secret <secret-name> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

S3 generico

```
tridentctl-protect create vault GenericS3 <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

ONTAP S3

```
tridentctl-protect create vault OntapS3 <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

StorageGrid S3

```
tridentctl-protect create vault StorageGridS3 <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

Visualizza le informazioni di AppVault

È possibile utilizzare il plug-in Trident Protect CLI per visualizzare informazioni sugli oggetti AppVault creati nel cluster.

Passi

1. Visualizza il contenuto di un oggetto AppVault:

```
tridentctl-protect get appvaultcontent gcp-vault \
--show-resources all \
-n trident-protect
```

Esempio di output:

```
+-------------+-------+----------+-----------------------------
+---------------------------+
|   CLUSTER   |  APP  |   TYPE   |            NAME             |
TIMESTAMP         |
+-------------+-------+----------+-----------------------------
+---------------------------+
|             | mysql | snapshot | mysnap                      | 2024-
08-09 21:02:11 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815180300 | 2024-
08-15 18:03:06 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815190300 | 2024-
08-15 19:03:06 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815200300 | 2024-
08-15 20:03:06 (UTC) |
| production1 | mysql | backup   | hourly-e7db6-20240815180300 | 2024-
08-15 18:04:25 (UTC) |
| production1 | mysql | backup   | hourly-e7db6-20240815190300 | 2024-
08-15 19:03:30 (UTC) |
| production1 | mysql | backup   | hourly-e7db6-20240815200300 | 2024-
08-15 20:04:21 (UTC) |
| production1 | mysql | backup   | mybackup5                   | 2024-
08-09 22:25:13 (UTC) |
|             | mysql | backup   | mybackup                    | 2024-
08-09 21:02:52 (UTC) |
+-------------+-------+----------+-----------------------------
+---------------------------+
```

2. Facoltativamente, per visualizzare l'AppVaultPath per ogni risorsa, utilizzare il flag `--show-paths` .

   Il nome del cluster nella prima colonna della tabella è disponibile solo se è stato specificato un nome del cluster nell'installazione di Trident Protect Helm. Per esempio: `--set clusterName=production1` .

Rimuovere un AppVault

È possibile rimuovere un oggetto AppVault in qualsiasi momento.

> Non rimuovere il `finalizers` digitare nella CR di AppVault prima di eliminare l'oggetto AppVault. In tal caso, potrebbero esserci dati residui nel bucket AppVault e risorse orfane nel cluster.

Prima di iniziare

Assicurati di aver eliminato tutti gli snapshot e i CR di backup utilizzati dall'AppVault che desideri eliminare.

Rimuovere un AppVault utilizzando la CLI di Kubernetes

1. Rimuovere l'oggetto AppVault, sostituendolo `appvault-name` con il nome dell'oggetto AppVault da rimuovere:

```
kubectl delete appvault <appvault-name> \
-n trident-protect
```

Rimuovere un AppVault utilizzando la CLI Trident Protect

1. Rimuovere l'oggetto AppVault, sostituendolo `appvault-name` con il nome dell'oggetto AppVault da rimuovere:

```
tridentctl-protect delete appvault <appvault-name> \
-n trident-protect
```

Definisci un'applicazione per la gestione con Trident Protect

È possibile definire un'applicazione che si desidera gestire con Trident Protect creando una CR dell'applicazione e una CR AppVault associata.

Crea una CR AppVault

È necessario creare un AppVault CR che verrà utilizzato durante l'esecuzione di operazioni di protezione dei dati sull'applicazione e il AppVault CR deve risiedere nel cluster in cui è installato Trident Protect. Il CR di AppVault è specifico per il tuo ambiente; per esempi di CR di AppVault, fai riferimento a"Risorse personalizzate di AppVault."

Definire un'applicazione

È necessario definire ciascuna applicazione che si desidera gestire con Trident Protect. È possibile definire un'applicazione per la gestione creando manualmente un CR dell'applicazione oppure utilizzando la CLI Trident Protect.

Aggiungere un'applicazione utilizzando un CR

Passi

1. Creare il file CR dell'applicazione di destinazione:
   a. Crea il file di risorse personalizzate (CR) e assegnagli un nome (ad esempio, `maria-app.yaml`).
   b. Configurare i seguenti attributi:
      - **metadata.name**: (*Obbligatorio*) Nome della risorsa personalizzata dell'applicazione. Prendi nota del nome scelto perché altri file CR necessari per le operazioni di protezione fanno riferimento a questo valore.
      - **spec.includedNamespaces**: (*Obbligatorio*) Utilizza il selettore di namespace ed etichette per specificare i namespace e le risorse utilizzati dall'applicazione. Lo spazio dei nomi dell'applicazione deve far parte di questo elenco. Il selettore di etichette è facoltativo e può essere utilizzato per filtrare le risorse all'interno di ogni namespace specificato.
      - **spec.includedClusterScopedResources**: (*Facoltativo*) Utilizzare questo attributo per specificare le risorse con ambito cluster da includere nella definizione dell'applicazione. Questo attributo consente di selezionare queste risorse in base al gruppo, alla versione, al tipo e alle etichette.
        - **groupVersionKind**: (*Obbligatorio*) Specifica il gruppo API, la versione e il tipo della risorsa con ambito cluster.
        - **labelSelector**: (*Facoltativo*) Filtra le risorse con ambito cluster in base alle loro etichette.
      - **metadata.annotations.protect.trident.netapp.io/skip-vm-freeze**: (*Facoltativo*) Questa annotazione è applicabile solo alle applicazioni definite da macchine virtuali, come negli ambienti KubeVirt, in cui i blocchi del file system si verificano prima degli snapshot. Specificare se questa applicazione può scrivere sul file system durante uno snapshot. Se impostato su true, l'applicazione ignora l'impostazione globale e può scrivere sul file system durante uno snapshot. Se impostato su false, l'applicazione ignora l'impostazione globale e il file system viene bloccato durante uno snapshot. Se specificato ma l'applicazione non ha macchine virtuali nella definizione dell'applicazione, l'annotazione viene ignorata. Se non specificato, l'applicazione segue la"impostazione di protezione antigelo Trident globale" .

Se è necessario applicare questa annotazione dopo che un'applicazione è già stata creata, è possibile utilizzare il seguente comando:

```
kubectl annotate application -n <application CR namespace> <application CR
name> protect.trident.netapp.io/skip-vm-freeze="true"
```

+
Esempio YAML:

+

```
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  annotations:
    protect.trident.netapp.io/skip-vm-freeze: "false"
  name: my-app-name
  namespace: my-app-namespace
spec:
  includedNamespaces:
    - namespace: namespace-1
      labelSelector:
        matchLabels:
          app: example-app
    - namespace: namespace-2
      labelSelector:
        matchLabels:
          app: another-example-app
  includedClusterScopedResources:
    - groupVersionKind:
        group: rbac.authorization.k8s.io
        kind: ClusterRole
        version: v1
      labelSelector:
        matchLabels:
          mylabel: test
```

1. (*Facoltativo*) Aggiungi un filtro che includa o escluda le risorse contrassegnate con etichette particolari:

   - **resourceFilter.resourceSelectionCriteria**: (Obbligatorio per il filtraggio) Utilizzare `Include` O `Exclude` per includere o escludere una risorsa definita in resourceMatchers. Aggiungere i seguenti parametri resourceMatchers per definire le risorse da includere o escludere:

     - **resourceFilter.resourceMatchers**: un array di oggetti resourceMatcher. Se si definiscono più elementi in questo array, essi corrispondono come un'operazione OR e i campi all'interno di ciascun elemento (gruppo, tipo, versione) corrispondono come un'operazione AND.
       - **resourceMatchers[].group**: (*Facoltativo*) Gruppo della risorsa da filtrare.
       - **resourceMatchers[].kind**: (*Facoltativo*) Tipo di risorsa da filtrare.
       - **resourceMatchers[].version**: (*Facoltativo*) Versione della risorsa da filtrare.
       - **resourceMatchers[].names**: (*Facoltativo*) Nomi nel campo metadata.name di Kubernetes della risorsa da filtrare.

- **resourceMatchers[].namespaces**: (*Facoltativo*) Spazi dei nomi nel campo metadata.name di Kubernetes della risorsa da filtrare.
- **resourceMatchers[].labelSelectors**: (*Facoltativo*) Stringa del selettore di etichetta nel campo metadata.name di Kubernetes della risorsa come definito in "Documentazione di Kubernetes" . Per esempio: `"trident.netapp.io/os=linux"` .

> ⓘ Quando entrambi `resourceFilter` E `labelSelector` vengono utilizzati, `resourceFilter` corre prima, e poi `labelSelector` viene applicato alle risorse risultanti.

Per esempio:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

2. Dopo aver creato la CR dell'applicazione adatta al tuo ambiente, applica la CR. Per esempio:

```
kubectl apply -f maria-app.yaml
```

Passi

1. Crea e applica la definizione dell'applicazione utilizzando uno degli esempi seguenti, sostituendo i valori tra parentesi con le informazioni provenienti dal tuo ambiente. È possibile includere namespace e risorse nella definizione dell'applicazione utilizzando elenchi separati da virgole con gli argomenti mostrati negli esempi.

   Facoltativamente, quando si crea un'app è possibile utilizzare un'annotazione per specificare se l'applicazione può scrivere sul file system durante uno snapshot. Ciò è applicabile solo alle applicazioni definite da macchine virtuali, come negli ambienti KubeVirt, in cui si verificano blocchi del file system prima degli snapshot. Se si imposta l'annotazione su `true` , l'applicazione ignora l'impostazione globale e può scrivere sul file system durante uno snapshot. Se lo imposti su `false` , l'applicazione ignora l'impostazione globale e il file system viene bloccato durante uno snapshot. Se si utilizza l'annotazione ma l'applicazione non ha macchine virtuali nella definizione dell'applicazione,

l'annotazione viene ignorata. Se non si utilizza l'annotazione, l'applicazione segue la"impostazione di protezione antigelo Trident globale" .

Per specificare l'annotazione quando si utilizza la CLI per creare un'applicazione, è possibile utilizzare `--annotation` bandiera.

- Creare l'applicazione e utilizzare l'impostazione globale per il comportamento di blocco del file system:

```
tridentctl-protect create application <my_new_app_cr_name>
--namespaces <namespaces_to_include> --csr
<cluster_scoped_resources_to_include> --namespace <my-app-
namespace>
```

- Creare l'applicazione e configurare le impostazioni dell'applicazione locale per il comportamento di blocco del file system:

```
tridentctl-protect create application <my_new_app_cr_name>
--namespaces <namespaces_to_include> --csr
<cluster_scoped_resources_to_include> --namespace <my-app-
namespace> --annotation protect.trident.netapp.io/skip-vm-freeze
=<"true"|"false">
```

Puoi usare `--resource-filter-include` E `--resource-filter-exclude` flag per includere o escludere risorse in base a `resourceSelectionCriteria` come gruppo, tipo, versione, etichette, nomi e namespace, come mostrato nel seguente esempio:

```
tridentctl-protect create application <my_new_app_cr_name>
--namespaces <namespaces_to_include> --csr
<cluster_scoped_resources_to_include> --namespace <my-app-namespace>
--resource-filter-include
'[{"Group":"apps","Kind":"Deployment","Version":"v1","Names":["my-
deployment"],"Namespaces":["my-
namespace"],"LabelSelectors":["app=my-app"]}]'
```

Proteggi le applicazioni utilizzando Trident Protect

È possibile proteggere tutte le app gestite da Trident Protect eseguendo snapshot e backup tramite una policy di protezione automatizzata o su base ad hoc.

È possibile configurare Trident Protect in modo che blocchi e sblocchi i file system durante le operazioni di protezione dei dati. "Scopri di più sulla configurazione del congelamento del file system con Trident Protect".

Crea uno snapshot on-demand

È possibile creare uno snapshot on-demand in qualsiasi momento.

> Le risorse con ambito cluster vengono incluse in un backup, in uno snapshot o in un clone se sono esplicitamente referenziate nella definizione dell'applicazione o se contengono riferimenti a uno qualsiasi degli spazi dei nomi dell'applicazione.

Crea uno snapshot utilizzando un CR

Passi

1. Crea il file di risorse personalizzate (CR) e assegnagli un nome `trident-protect-snapshot-cr.yaml` .
2. Nel file creato, configura i seguenti attributi:
   - **metadata.name**: (*Obbligatorio*) Il nome di questa risorsa personalizzata; scegli un nome univoco e sensato per il tuo ambiente.
   - **spec.applicationRef**: Nome Kubernetes dell'applicazione di cui eseguire lo snapshot.
   - **spec.appVaultRef**: (*Obbligatorio*) Nome dell'AppVault in cui devono essere archiviati i contenuti dello snapshot (metadati).
   - **spec.reclaimPolicy**: (*Facoltativo*) Definisce cosa accade all'AppArchive di uno snapshot quando il CR dello snapshot viene eliminato. Ciò significa che anche quando impostato su `Retain` , lo snapshot verrà eliminato. Opzioni valide:
     - `Retain`(predefinito)
     - `Delete`

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: Snapshot
     metadata:
       namespace: my-app-namespace
       name: my-cr-name
     spec:
       applicationRef: my-application
       appVaultRef: appvault-name
       reclaimPolicy: Delete
     ```
3. Dopo aver popolato il `trident-protect-snapshot-cr.yaml` file con i valori corretti, applicare CR:

   ```
   kubectl apply -f trident-protect-snapshot-cr.yaml
   ```

Creare uno snapshot utilizzando la CLI

Passi

1. Crea lo snapshot sostituendo i valori tra parentesi con le informazioni provenienti dal tuo ambiente. Per esempio:

   ```
   tridentctl-protect create snapshot <my_snapshot_name> --appvault
   <my_appvault_name> --app <name_of_app_to_snapshot> -n
   <application_namespace>
   ```

Crea un backup su richiesta

Puoi eseguire il backup di un'app in qualsiasi momento.

> Le risorse con ambito cluster vengono incluse in un backup, in uno snapshot o in un clone se sono esplicitamente referenziate nella definizione dell'applicazione o se contengono riferimenti a uno qualsiasi degli spazi dei nomi dell'applicazione.

Prima di iniziare

Assicurarsi che la scadenza del token di sessione AWS sia sufficiente per qualsiasi operazione di backup S3 di lunga durata. Se il token scade durante l'operazione di backup, l'operazione potrebbe non riuscire.

- Fare riferimento al "Documentazione AWS API" per ulteriori informazioni sulla verifica della scadenza del token della sessione corrente.
- Fare riferimento al "Documentazione AWS IAM" per ulteriori informazioni sulle credenziali con le risorse AWS.

Crea un backup utilizzando un CR

Passi

1. Crea il file di risorse personalizzate (CR) e assegnagli un nome `trident-protect-backup-cr.yaml` .
2. Nel file creato, configura i seguenti attributi:
   - **metadata.name**: (*Obbligatorio*) Il nome di questa risorsa personalizzata; scegli un nome univoco e sensato per il tuo ambiente.
   - **spec.applicationRef**: (*Obbligatorio*) Nome Kubernetes dell'applicazione di cui eseguire il backup.
   - **spec.appVaultRef**: (*Obbligatorio*) Nome dell'AppVault in cui devono essere archiviati i contenuti del backup.
   - **spec.dataMover**: (*Facoltativo*) Una stringa che indica quale strumento di backup utilizzare per l'operazione di backup. Valori possibili (con distinzione tra maiuscole e minuscole):
     - `Restic`
     - `Kopia`(predefinito)
   - **spec.reclaimPolicy**: (*Facoltativo*) Definisce cosa accade a un backup quando viene rilasciato dalla sua richiesta. Valori possibili:
     - `Delete`
     - `Retain`(predefinito)
   - **spec.snapshotRef**: (*Facoltativo*): Nome dello snapshot da utilizzare come origine del backup. Se non specificato, verrà creato uno snapshot temporaneo e ne verrà eseguito il backup.
   - **metadata.annotations.protect.trident.netapp.io/full-backup** : (*Facoltativo*) Questa annotazione viene utilizzata per specificare se un backup deve essere non incrementale. Per impostazione predefinita, tutti i backup sono incrementali. Tuttavia, se questa annotazione è impostata su `true` , il backup diventa non incrementale. Se non specificato, il backup segue l'impostazione di backup incrementale predefinita. È consigliabile eseguire periodicamente un backup completo e poi eseguire backup incrementali tra un backup completo e l'altro, per ridurre al minimo i rischi associati ai ripristini.

     Esempio YAML:

```yaml
---
apiVersion: protect.trident.netapp.io/v1
kind: Backup
metadata:
  namespace: my-app-namespace
  name: my-cr-name
  annotations:
    protect.trident.netapp.io/full-backup: "true"
spec:
  applicationRef: my-application
  appVaultRef: appvault-name
  dataMover: Kopia
```

3. Dopo aver popolato il `trident-protect-backup-cr.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-backup-cr.yaml
```

Creare un backup utilizzando la CLI

Passi

1. Crea il backup sostituendo i valori tra parentesi con le informazioni del tuo ambiente. Per esempio:

```
tridentctl-protect create backup <my_backup_name> --appvault <my-
vault-name> --app <name_of_app_to_back_up> --data-mover
<Kopia_or_Restic> -n <application_namespace>
```

Facoltativamente puoi usare il `--full-backup` flag per specificare se un backup deve essere non incrementale. Per impostazione predefinita, tutti i backup sono incrementali. Quando si utilizza questo flag, il backup diventa non incrementale. È consigliabile eseguire periodicamente un backup completo e poi eseguire backup incrementali tra un backup completo e l'altro, per ridurre al minimo i rischi associati ai ripristini.

Creare un programma di protezione dei dati

Una policy di protezione protegge un'app creando snapshot, backup o entrambi secondo una pianificazione definita. È possibile scegliere di creare snapshot e backup orari, giornalieri, settimanali e mensili e specificare il numero di copie da conservare. È possibile pianificare un backup completo non incrementale utilizzando l'annotazione full-backup-rule. Per impostazione predefinita, tutti i backup sono incrementali. L'esecuzione periodica di un backup completo, insieme a backup incrementali intermedi, aiuta a ridurre il rischio associato ai ripristini.

- È possibile creare pianificazioni solo per gli snapshot impostando `backupRetention` a zero e `snapshotRetention` a un valore maggiore di zero. Collocamento `snapshotRetention` a zero significa che tutti i backup pianificati creeranno comunque degli snapshot, ma questi saranno temporanei e verranno eliminati immediatamente dopo il completamento del backup.
- Le risorse con ambito cluster vengono incluse in un backup, in uno snapshot o in un clone se sono esplicitamente referenziate nella definizione dell'applicazione o se contengono riferimenti a uno qualsiasi degli spazi dei nomi dell'applicazione.

Creare una pianificazione utilizzando un CR

Passi

1. Crea il file di risorse personalizzate (CR) e assegnagli un nome `trident-protect-schedule-cr.yaml` .
2. Nel file creato, configura i seguenti attributi:
   - **metadata.name**: (*Obbligatorio*) Il nome di questa risorsa personalizzata; scegli un nome univoco e sensato per il tuo ambiente.
   - **spec.dataMover**: (*Facoltativo*) Una stringa che indica quale strumento di backup utilizzare per l'operazione di backup. Valori possibili (con distinzione tra maiuscole e minuscole):
     - `Restic`
     - `Kopia`(predefinito)
   - **spec.applicationRef**: Nome Kubernetes dell'applicazione di cui eseguire il backup.
   - **spec.appVaultRef**: (*Obbligatorio*) Nome dell'AppVault in cui devono essere archiviati i contenuti del backup.
   - **spec.backupRetention**: Numero di backup da conservare. Zero indica che non devono essere creati backup (solo snapshot).
   - **spec.snapshotRetention**: Numero di snapshot da conservare. Zero indica che non devono essere creati snapshot.
   - **spec.granularity**: la frequenza con cui la pianificazione deve essere eseguita. Valori possibili, insieme ai campi associati obbligatori:
     - `Hourly`(richiede che tu specifichi `spec.minute`)
     - `Daily`(richiede che tu specifichi `spec.minute` E `spec.hour`)
     - `Weekly`(richiede che tu specifichi `spec.minute, spec.hour` , E `spec.dayOfWeek`)
     - `Monthly`(richiede che tu specifichi `spec.minute, spec.hour` , E `spec.dayOfMonth`)
     - `Custom`
   - **spec.dayOfMonth**: (*Facoltativo*) Il giorno del mese (1 - 31) in cui la pianificazione deve essere eseguita. Questo campo è obbligatorio se la granularità è impostata su `Monthly` . Il valore deve essere fornito come stringa.
   - **spec.dayOfWeek**: (*Facoltativo*) Il giorno della settimana (0 - 7) in cui deve essere eseguita la pianificazione. I valori 0 o 7 indicano domenica. Questo campo è obbligatorio se la granularità è impostata su `Weekly` . Il valore deve essere fornito come stringa.
   - **spec.hour**: (*Facoltativo*) L'ora del giorno (0 - 23) in cui la pianificazione deve essere eseguita. Questo campo è obbligatorio se la granularità è impostata su `Daily` , `Weekly` , O `Monthly` . Il valore deve essere fornito come stringa.
   - **spec.minute**: (*Facoltativo*) Il minuto dell'ora (0 - 59) in cui la pianificazione deve essere eseguita. Questo campo è obbligatorio se la granularità è impostata su `Hourly` , `Daily` , `Weekly` , O `Monthly` . Il valore deve essere fornito come stringa.
   - **metadata.annotations.protect.trident.netapp.io/full-backup-rule**: (*Facoltativo*) Questa annotazione viene utilizzata per specificare la regola per la pianificazione del backup completo. Puoi impostarlo su `always` per un backup completo costante o personalizzarlo in base alle tue esigenze. Ad esempio, se si sceglie la granularità giornaliera, è possibile specificare i giorni della

settimana in cui deve essere eseguito il backup completo.

Esempio di YAML per la pianificazione di backup e snapshot:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  namespace: my-app-namespace
  name: my-cr-name
  annotations:
    protect.trident.netapp.io/full-backup-rule: "Monday,Thursday"
spec:
  dataMover: Kopia
  applicationRef: my-application
  appVaultRef: appvault-name
  backupRetention: "15"
  snapshotRetention: "15"
  granularity: Daily
  hour: "0"
  minute: "0"
```

Esempio di YAML per la pianificazione solo snapshot:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  namespace: my-app-namespace
  name: my-snapshot-schedule
spec:
  applicationRef: my-application
  appVaultRef: appvault-name
  backupRetention: "0"
  snapshotRetention: "15"
  granularity: Daily
  hour: "2"
  minute: "0"
```

3. Dopo aver popolato il `trident-protect-schedule-cr.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-schedule-cr.yaml
```

Creare una pianificazione utilizzando la CLI

Passi

1. Crea la pianificazione della protezione, sostituendo i valori tra parentesi con le informazioni provenienti dal tuo ambiente. Per esempio:

   Puoi usare `tridentctl-protect create schedule --help` per visualizzare informazioni di aiuto dettagliate per questo comando.

   ```
   tridentctl-protect create schedule <my_schedule_name> --appvault
   <my_appvault_name> --app <name_of_app_to_snapshot> --backup
   -retention <how_many_backups_to_retain> --data-mover
   <Kopia_or_Restic> --day-of-month <day_of_month_to_run_schedule>
   --day-of-week <day_of_month_to_run_schedule> --granularity
   <frequency_to_run> --hour <hour_of_day_to_run> --minute
   <minute_of_hour_to_run> --recurrence-rule <recurrence> --snapshot
   -retention <how_many_snapshots_to_retain> -n <application_namespace>
   --full-backup-rule <string>
   ```

   Puoi impostare il `--full-backup-rule` bandiera a `always` per un backup completo costante o personalizzarlo in base alle tue esigenze. Ad esempio, se si sceglie la granularità giornaliera, è possibile specificare i giorni della settimana in cui deve essere eseguito il backup completo. Ad esempio, utilizzare `--full-backup-rule "Monday,Thursday"` per programmare un backup completo il lunedì e il giovedì.

   Per pianificazioni solo snapshot, impostare `--backup-retention 0` e specificare un valore maggiore di 0 per `--snapshot-retention` .

Elimina uno snapshot

Elimina gli snapshot pianificati o su richiesta di cui non hai più bisogno.

Passi

1. Rimuovere lo snapshot CR associato allo snapshot:

   ```
   kubectl delete snapshot <snapshot_name> -n my-app-namespace
   ```

Elimina un backup

Elimina i backup pianificati o su richiesta di cui non hai più bisogno.

Assicurarsi che la politica di recupero sia impostata su `Delete` per rimuovere tutti i dati di backup dall'archiviazione degli oggetti. L'impostazione predefinita della policy è `Retain` per evitare la perdita accidentale di dati. Se la politica non viene modificata in `Delete` , i dati di backup rimarranno nell'archivio oggetti e richiederanno l'eliminazione manuale.

Passi

1. Rimuovere il CR di backup associato al backup:

```
kubectl delete backup <backup_name> -n my-app-namespace
```

Controllare lo stato di un'operazione di backup

È possibile utilizzare la riga di comando per verificare lo stato di un'operazione di backup in corso, completata o non riuscita.

Passi

1. Utilizzare il seguente comando per recuperare lo stato dell'operazione di backup, sostituendo i valori tra parentesi con le informazioni provenienti dal proprio ambiente:

```
kubectl get backup -n <namespace_name> <my_backup_cr_name> -o jsonpath
='{.status}'
```

Abilita backup e ripristino per le operazioni azure-netapp-files (ANF)

Se hai installato Trident Protect, puoi abilitare la funzionalità di backup e ripristino a basso consumo di spazio per i backend di archiviazione che utilizzano la classe di archiviazione azure-netapp-files e sono stati creati prima di Trident 24.06. Questa funzionalità funziona con volumi NFSv4 e non consuma spazio aggiuntivo dal pool di capacità.

Prima di iniziare

Assicurarsi che:

- Hai installato Trident Protect.
- Hai definito un'applicazione in Trident Protect. Questa applicazione avrà funzionalità di protezione limitate finché non avrai completato questa procedura.
- Hai `azure-netapp-files` selezionata come classe di archiviazione predefinita per il backend di archiviazione.

Espandi per i passaggi di configurazione

1. Eseguire le seguenti operazioni in Trident se il volume ANF è stato creato prima dell'aggiornamento a Trident 24.10:

   a. Abilitare la directory snapshot per ogni PV basato su azure-netapp-files e associato all'applicazione:

   ```
   tridentctl update volume <pv name> --snapshot-dir=true -n trident
   ```

   b. Verificare che la directory snapshot sia stata abilitata per ogni PV associato:

   ```
   tridentctl get volume <pv name> -n trident -o yaml | grep
   snapshotDir
   ```

   Risposta:

   ```
   snapshotDirectory: "true"
   ```

   +
   Quando la directory degli snapshot non è abilitata, Trident Protect sceglie la funzionalità di backup normale, che consuma temporaneamente spazio nel pool di capacità durante il processo di backup. In questo caso, assicurarsi che nel pool di capacità sia disponibile spazio sufficiente per creare un volume temporaneo delle dimensioni del volume sottoposto a backup.

Risultato

L'applicazione è pronta per il backup e il ripristino tramite Trident Protect. Ogni PVC può essere utilizzato anche da altre applicazioni per backup e ripristini.

Ripristinare le applicazioni

Ripristina le applicazioni utilizzando Trident Protect

Puoi utilizzare Trident Protect per ripristinare la tua applicazione da uno snapshot o da un backup. Il ripristino da uno snapshot esistente sarà più rapido se si ripristina l'applicazione nello stesso cluster.

- Quando si ripristina un'applicazione, tutti gli hook di esecuzione configurati per l'applicazione vengono ripristinati con l'applicazione. Se è presente un hook di esecuzione post-ripristino, questo viene eseguito automaticamente come parte dell'operazione di ripristino.
- Per i volumi qtree è supportato il ripristino da un backup a uno spazio dei nomi diverso o allo spazio dei nomi originale. Tuttavia, il ripristino da uno snapshot a uno spazio dei nomi diverso o allo spazio dei nomi originale non è supportato per i volumi qtree.
- È possibile utilizzare le impostazioni avanzate per personalizzare le operazioni di ripristino. Per saperne di più, fare riferimento a "Utilizzare le impostazioni di ripristino avanzate Trident Protect".

Ripristina da un backup a uno spazio dei nomi diverso

Quando si ripristina un backup in uno spazio dei nomi diverso utilizzando un CR BackupRestore, Trident Protect ripristina l'applicazione in un nuovo spazio dei nomi e crea un CR dell'applicazione per l'applicazione ripristinata. Per proteggere l'applicazione ripristinata, creare backup o snapshot su richiesta oppure stabilire una pianificazione di protezione.

Il ripristino di un backup in uno spazio dei nomi diverso con risorse esistenti non modificherà le risorse che condividono i nomi con quelle presenti nel backup. Per ripristinare tutte le risorse nel backup, eliminare e ricreare lo spazio dei nomi di destinazione oppure ripristinare il backup in un nuovo spazio dei nomi.

Prima di iniziare

Assicurarsi che la scadenza del token di sessione AWS sia sufficiente per qualsiasi operazione di ripristino S3 di lunga durata. Se il token scade durante l'operazione di ripristino, l'operazione potrebbe non riuscire.

- Fare riferimento al "Documentazione AWS API" per ulteriori informazioni sulla verifica della scadenza del token della sessione corrente.
- Fare riferimento al "Documentazione AWS IAM" per ulteriori informazioni sulle credenziali con le risorse AWS.

Quando si ripristinano i backup utilizzando Kopia come strumento di spostamento dati, è possibile specificare facoltativamente annotazioni nel CR o tramite la CLI per controllare il comportamento dell'archiviazione temporanea utilizzata da Kopia. Fare riferimento al "Documentazione Kopia" per maggiori informazioni sulle opzioni che puoi configurare. Utilizzare il `tridentctl-protect create --help` comando per ulteriori informazioni sulla specifica delle annotazioni con la CLI Trident Protect.

Utilizzare un CR

Passi

1. Crea il file di risorse personalizzate (CR) e assegnagli un nome `trident-protect-backup-restore-cr.yaml` .
2. Nel file creato, configura i seguenti attributi:
   - **metadata.name**: (*Obbligatorio*) Il nome di questa risorsa personalizzata; scegli un nome univoco e sensato per il tuo ambiente.
   - **spec.appArchivePath**: il percorso all'interno di AppVault in cui sono archiviati i contenuti del backup. Per trovare questo percorso puoi usare il seguente comando:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

   - **spec.appVaultRef**: (*Obbligatorio*) Nome dell'AppVault in cui sono archiviati i contenuti del backup.
   - **spec.namespaceMapping**: la mappatura dello spazio dei nomi di origine dell'operazione di ripristino allo spazio dei nomi di destinazione. Sostituire `my-source-namespace` E `my-destination-namespace` con informazioni provenienti dal tuo ambiente.

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: BackupRestore
     metadata:
       name: my-cr-name
       namespace: my-destination-namespace
     spec:
       appArchivePath: my-backup-path
       appVaultRef: appvault-name
       namespaceMapping: [{"source": "my-source-namespace",
     "destination": "my-destination-namespace"}]
     ```

3. (*Facoltativo*) Se è necessario selezionare solo determinate risorse dell'applicazione da ripristinare, aggiungere un filtro che includa o escluda le risorse contrassegnate con etichette particolari:

   > Trident Protect seleziona automaticamente alcune risorse in base alla loro relazione con le risorse selezionate. Ad esempio, se selezioni una risorsa di richiesta di volume persistente e a questa è associato un pod, Trident Protect ripristinerà anche il pod associato.

   - **resourceFilter.resourceSelectionCriteria**: (Obbligatorio per il filtraggio) Utilizzare `Include` O `Exclude` per includere o escludere una risorsa definita in resourceMatchers. Aggiungere i seguenti parametri resourceMatchers per definire le risorse da includere o escludere:
     - **resourceFilter.resourceMatchers**: un array di oggetti resourceMatcher. Se si definiscono più elementi in questo array, essi corrispondono come un'operazione OR e i campi all'interno di

ciascun elemento (gruppo, tipo, versione) corrispondono come un'operazione AND.

- **resourceMatchers[].group**: (*Facoltativo*) Gruppo della risorsa da filtrare.
- **resourceMatchers[].kind**: (*Facoltativo*) Tipo di risorsa da filtrare.
- **resourceMatchers[].version**: (*Facoltativo*) Versione della risorsa da filtrare.
- **resourceMatchers[].names**: (*Facoltativo*) Nomi nel campo metadata.name di Kubernetes della risorsa da filtrare.
- **resourceMatchers[].namespaces**: (*Facoltativo*) Spazi dei nomi nel campo metadata.name di Kubernetes della risorsa da filtrare.
- **resourceMatchers[].labelSelectors**: (*Facoltativo*) Stringa del selettore di etichetta nel campo metadata.name di Kubernetes della risorsa come definito in "Documentazione di Kubernetes" . Per esempio: `"trident.netapp.io/os=linux"` .

  Per esempio:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Dopo aver popolato il `trident-protect-backup-restore-cr.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-backup-restore-cr.yaml
```

Utilizzare la CLI

Passi

1. Ripristina il backup in uno spazio dei nomi diverso, sostituendo i valori tra parentesi con le informazioni del tuo ambiente. IL `namespace-mapping` l'argomento utilizza namespace separati da due punti per mappare i namespace di origine ai namespace di destinazione corretti nel formato `source1:dest1,source2:dest2` . Per esempio:

```
tridentctl-protect create backuprestore <my_restore_name> \
--backup <backup_namespace>/<backup_to_restore> \
--namespace-mapping <source_to_destination_namespace_mapping> \
-n <application_namespace>
```

Ripristina da un backup allo spazio dei nomi originale

È possibile ripristinare un backup nello spazio dei nomi originale in qualsiasi momento.

Prima di iniziare

Assicurarsi che la scadenza del token di sessione AWS sia sufficiente per qualsiasi operazione di ripristino S3 di lunga durata. Se il token scade durante l'operazione di ripristino, l'operazione potrebbe non riuscire.

- Fare riferimento al "Documentazione AWS API" per ulteriori informazioni sulla verifica della scadenza del token della sessione corrente.
- Fare riferimento al "Documentazione AWS IAM" per ulteriori informazioni sulle credenziali con le risorse AWS.

> Quando si ripristinano i backup utilizzando Kopia come strumento di spostamento dati, è possibile specificare facoltativamente annotazioni nel CR o tramite la CLI per controllare il comportamento dell'archiviazione temporanea utilizzata da Kopia. Fare riferimento al "Documentazione Kopia" per maggiori informazioni sulle opzioni che puoi configurare. Utilizzare il `tridentctl-protect create --help` comando per ulteriori informazioni sulla specifica delle annotazioni con la CLI Trident Protect.

Utilizzare un CR

Passi

1. Crea il file di risorse personalizzate (CR) e assegnagli un nome `trident-protect-backup-ipr-cr.yaml` .
2. Nel file creato, configura i seguenti attributi:
   - **metadata.name**: (*Obbligatorio*) Il nome di questa risorsa personalizzata; scegli un nome univoco e sensato per il tuo ambiente.
   - **spec.appArchivePath**: il percorso all'interno di AppVault in cui sono archiviati i contenuti del backup. Per trovare questo percorso puoi usare il seguente comando:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o jsonpath='{.status.appArchivePath}'
     ```

   - **spec.appVaultRef**: (*Obbligatorio*) Nome dell'AppVault in cui sono archiviati i contenuti del backup.

     Per esempio:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: BackupInplaceRestore
metadata:
  name: my-cr-name
  namespace: my-app-namespace
spec:
  appArchivePath: my-backup-path
  appVaultRef: appvault-name
```

3. (*Facoltativo*) Se è necessario selezionare solo determinate risorse dell'applicazione da ripristinare, aggiungere un filtro che includa o escluda le risorse contrassegnate con etichette particolari:

   Trident Protect seleziona automaticamente alcune risorse in base alla loro relazione con le risorse selezionate. Ad esempio, se selezioni una risorsa di richiesta di volume persistente e a questa è associato un pod, Trident Protect ripristinerà anche il pod associato.

   - **resourceFilter.resourceSelectionCriteria**: (Obbligatorio per il filtraggio) Utilizzare `Include` O `Exclude` per includere o escludere una risorsa definita in resourceMatchers. Aggiungere i seguenti parametri resourceMatchers per definire le risorse da includere o escludere:
     - **resourceFilter.resourceMatchers**: un array di oggetti resourceMatcher. Se si definiscono più elementi in questo array, essi corrispondono come un'operazione OR e i campi all'interno di ciascun elemento (gruppo, tipo, versione) corrispondono come un'operazione AND.
       - **resourceMatchers[].group**: (*Facoltativo*) Gruppo della risorsa da filtrare.
       - **resourceMatchers[].kind**: (*Facoltativo*) Tipo di risorsa da filtrare.

- **resourceMatchers[].version**: (*Facoltativo*) Versione della risorsa da filtrare.
- **resourceMatchers[].names**: (*Facoltativo*) Nomi nel campo metadata.name di Kubernetes della risorsa da filtrare.
- **resourceMatchers[].namespaces**: (*Facoltativo*) Spazi dei nomi nel campo metadata.name di Kubernetes della risorsa da filtrare.
- **resourceMatchers[].labelSelectors**: (*Facoltativo*) Stringa del selettore di etichetta nel campo metadata.name di Kubernetes della risorsa come definito in "Documentazione di Kubernetes" . Per esempio: `"trident.netapp.io/os=linux"` .

  Per esempio:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Dopo aver popolato il `trident-protect-backup-ipr-cr.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-backup-ipr-cr.yaml
```

Utilizzare la CLI

Passi

1. Ripristina il backup nello spazio dei nomi originale, sostituendo i valori tra parentesi con le informazioni del tuo ambiente. IL `backup` l'argomento utilizza uno spazio dei nomi e un nome di backup nel formato `<namespace>/<name>` . Per esempio:

```
tridentctl-protect create backupinplacerestore <my_restore_name> \
--backup <namespace/backup_to_restore> \
-n <application_namespace>
```

Ripristina da un backup a un cluster diverso

È possibile ripristinare un backup su un cluster diverso se si verifica un problema con il cluster originale.

> Quando si ripristinano i backup utilizzando Kopia come strumento di spostamento dati, è possibile specificare facoltativamente annotazioni nel CR o tramite la CLI per controllare il comportamento dell'archiviazione temporanea utilizzata da Kopia. Fare riferimento al "Documentazione Kopia" per maggiori informazioni sulle opzioni che puoi configurare. Utilizzare il `tridentctl-protect create --help` comando per ulteriori informazioni sulla specifica delle annotazioni con la CLI Trident Protect.

Prima di iniziare

Assicurarsi che siano soddisfatti i seguenti prerequisiti:

- Nel cluster di destinazione è installato Trident Protect.
- Il cluster di destinazione ha accesso al percorso del bucket dello stesso AppVault del cluster di origine, in cui è archiviato il backup.
- Assicurarsi che l'ambiente locale possa connettersi al bucket di archiviazione degli oggetti definito in AppVault CR durante l'esecuzione di `tridentctl-protect get appvaultcontent` comando. Se le restrizioni di rete impediscono l'accesso, eseguire invece la CLI Trident Protect dall'interno di un pod sul cluster di destinazione.
- Assicurarsi che la scadenza del token di sessione AWS sia sufficiente per qualsiasi operazione di ripristino di lunga durata. Se il token scade durante l'operazione di ripristino, l'operazione potrebbe non riuscire.
  - Fare riferimento al "Documentazione AWS API" per ulteriori informazioni sulla verifica della scadenza del token della sessione corrente.
  - Fare riferimento al "Documentazione AWS" per ulteriori informazioni sulle credenziali con le risorse AWS.

Passi

1. Verificare la disponibilità di AppVault CR sul cluster di destinazione utilizzando il plug-in Trident Protect CLI:

```
tridentctl-protect get appvault --context <destination_cluster_name>
```

> Assicurarsi che lo spazio dei nomi destinato al ripristino dell'applicazione esista nel cluster di destinazione.

2. Visualizza il contenuto del backup dell'AppVault disponibile dal cluster di destinazione:

```
tridentctl-protect get appvaultcontent <appvault_name> \
--show-resources backup \
--show-paths \
--context <destination_cluster_name>
```

L'esecuzione di questo comando visualizza i backup disponibili in AppVault, inclusi i cluster di origine, i nomi delle applicazioni corrispondenti, i timestamp e i percorsi di archivio.

Esempio di output:

```
+-------------+-----------+--------+-----------------
+--------------------------+-------------+
|   CLUSTER   |    APP    |  TYPE  |      NAME       |        TIMESTAMP
|    PATH     |
+-------------+-----------+--------+-----------------
+--------------------------+-------------+
| production1 | wordpress | backup | wordpress-bkup-1| 2024-10-30
08:37:40 (UTC)| backuppath1 |
| production1 | wordpress | backup | wordpress-bkup-2| 2024-10-30
08:37:40 (UTC)| backuppath2 |
+-------------+-----------+--------+-----------------
+--------------------------+-------------+
```

3. Ripristinare l’applicazione nel cluster di destinazione utilizzando il nome AppVault e il percorso di archivio:

Utilizzare un CR

1. Crea il file di risorse personalizzate (CR) e assegnagli un nome `trident-protect-backup-restore-cr.yaml` .
2. Nel file creato, configura i seguenti attributi:
   - **metadata.name**: (*Obbligatorio*) Il nome di questa risorsa personalizzata; scegli un nome univoco e sensato per il tuo ambiente.
   - **spec.appVaultRef**: (*Obbligatorio*) Nome dell'AppVault in cui sono archiviati i contenuti del backup.
   - **spec.appArchivePath**: il percorso all'interno di AppVault in cui sono archiviati i contenuti del backup. Per trovare questo percorso puoi usare il seguente comando:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

     Se BackupRestore CR non è disponibile, è possibile utilizzare il comando menzionato nel passaggio 2 per visualizzare il contenuto del backup.

   - **spec.namespaceMapping**: la mappatura dello spazio dei nomi di origine dell'operazione di ripristino allo spazio dei nomi di destinazione. Sostituire `my-source-namespace` E `my-destination-namespace` con informazioni provenienti dal tuo ambiente.

     Per esempio:

```
apiVersion: protect.trident.netapp.io/v1
kind: BackupRestore
metadata:
  name: my-cr-name
  namespace: my-destination-namespace
spec:
  appVaultRef: appvault-name
  appArchivePath: my-backup-path
  namespaceMapping: [{"source": "my-source-namespace", "
destination": "my-destination-namespace"}]
```

3. Dopo aver popolato il `trident-protect-backup-restore-cr.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-backup-restore-cr.yaml
```

Utilizzare la CLI

1. Utilizzare il seguente comando per ripristinare l'applicazione, sostituendo i valori tra parentesi con le informazioni provenienti dal proprio ambiente. L'argomento namespace-mapping utilizza namespace

separati da due punti per mappare i namespace di origine ai namespace di destinazione corretti nel formato source1:dest1,source2:dest2. Per esempio:

```
tridentctl-protect create backuprestore <restore_name> \
--namespace-mapping <source_to_destination_namespace_mapping> \
--appvault <appvault_name> \
--path <backup_path> \
--context <destination_cluster_name> \
-n <application_namespace>
```

Ripristina da uno snapshot a uno spazio dei nomi diverso

È possibile ripristinare i dati da uno snapshot utilizzando un file di risorse personalizzato (CR) in uno spazio dei nomi diverso o nello spazio dei nomi di origine originale. Quando si ripristina uno snapshot in un namespace diverso utilizzando un CR SnapshotRestore, Trident Protect ripristina l'applicazione in un nuovo namespace e crea un CR per l'applicazione ripristinata. Per proteggere l'applicazione ripristinata, creare backup o snapshot su richiesta oppure stabilire una pianificazione di protezione.

> SnapshotRestore supporta il `spec.storageClassMapping` attributo, ma solo quando le classi di archiviazione di origine e di destinazione utilizzano lo stesso backend di archiviazione. Se si tenta di ripristinare un `StorageClass` che utilizza un backend di archiviazione diverso, l'operazione di ripristino non riuscirà.

Prima di iniziare

Assicurarsi che la scadenza del token di sessione AWS sia sufficiente per qualsiasi operazione di ripristino S3 di lunga durata. Se il token scade durante l'operazione di ripristino, l'operazione potrebbe non riuscire.

- Fare riferimento al "Documentazione AWS API" per ulteriori informazioni sulla verifica della scadenza del token della sessione corrente.
- Fare riferimento al "Documentazione AWS IAM" per ulteriori informazioni sulle credenziali con le risorse AWS.

Utilizzare un CR

Passi

1. Crea il file di risorse personalizzate (CR) e assegnagli un nome `trident-protect-snapshot-restore-cr.yaml` .
2. Nel file creato, configura i seguenti attributi:
   - **metadata.name**: (*Obbligatorio*) Il nome di questa risorsa personalizzata; scegli un nome univoco e sensato per il tuo ambiente.
   - **spec.appVaultRef**: (*Obbligatorio*) Nome dell'AppVault in cui sono archiviati i contenuti dello snapshot.
   - **spec.appArchivePath**: il percorso all'interno di AppVault in cui sono archiviati i contenuti dello snapshot. Per trovare questo percorso puoi usare il seguente comando:

     ```
     kubectl get snapshots <SNAPHOT_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

   - **spec.namespaceMapping**: la mappatura dello spazio dei nomi di origine dell'operazione di ripristino allo spazio dei nomi di destinazione. Sostituire `my-source-namespace` E `my-destination-namespace` con informazioni provenienti dal tuo ambiente.

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: SnapshotRestore
     metadata:
       name: my-cr-name
       namespace: my-app-namespace
     spec:
       appVaultRef: appvault-name
       appArchivePath: my-snapshot-path
       namespaceMapping: [{"source": "my-source-namespace",
     "destination": "my-destination-namespace"}]
     ```

3. (*Facoltativo*) Se è necessario selezionare solo determinate risorse dell'applicazione da ripristinare, aggiungere un filtro che includa o escluda le risorse contrassegnate con etichette particolari:

   > Trident Protect seleziona automaticamente alcune risorse in base alla loro relazione con le risorse selezionate. Ad esempio, se selezioni una risorsa di richiesta di volume persistente e a questa è associato un pod, Trident Protect ripristinerà anche il pod associato.

   - **resourceFilter.resourceSelectionCriteria**: (Obbligatorio per il filtraggio) Utilizzare `Include` O `Exclude` per includere o escludere una risorsa definita in resourceMatchers. Aggiungere i seguenti parametri resourceMatchers per definire le risorse da includere o escludere:
     - **resourceFilter.resourceMatchers**: un array di oggetti resourceMatcher. Se si definiscono più elementi in questo array, essi corrispondono come un'operazione OR e i campi all'interno di

ciascun elemento (gruppo, tipo, versione) corrispondono come un'operazione AND.

- **resourceMatchers[].group**: (*Facoltativo*) Gruppo della risorsa da filtrare.
- **resourceMatchers[].kind**: (*Facoltativo*) Tipo di risorsa da filtrare.
- **resourceMatchers[].version**: (*Facoltativo*) Versione della risorsa da filtrare.
- **resourceMatchers[].names**: (*Facoltativo*) Nomi nel campo metadata.name di Kubernetes della risorsa da filtrare.
- **resourceMatchers[].namespaces**: (*Facoltativo*) Spazi dei nomi nel campo metadata.name di Kubernetes della risorsa da filtrare.
- **resourceMatchers[].labelSelectors**: (*Facoltativo*) Stringa del selettore di etichetta nel campo metadata.name di Kubernetes della risorsa come definito in "Documentazione di Kubernetes" . Per esempio: `"trident.netapp.io/os=linux"` .

  Per esempio:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Dopo aver popolato il `trident-protect-snapshot-restore-cr.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-snapshot-restore-cr.yaml
```

Utilizzare la CLI

Passi

1. Ripristina lo snapshot in uno spazio dei nomi diverso, sostituendo i valori tra parentesi con le informazioni del tuo ambiente.

   - IL `snapshot` l'argomento utilizza uno spazio dei nomi e un nome di snapshot nel formato `<namespace>/<name>` .
   - IL `namespace-mapping` l'argomento utilizza namespace separati da due punti per mappare i

namespace di origine ai namespace di destinazione corretti nel formato `source1:dest1,source2:dest2`.

Per esempio:

```
tridentctl-protect create snapshotrestore <my_restore_name> \
--snapshot <namespace/snapshot_to_restore> \
--namespace-mapping <source_to_destination_namespace_mapping> \
-n <application_namespace>
```

Ripristina da uno snapshot allo spazio dei nomi originale

È possibile ripristinare uno snapshot nello spazio dei nomi originale in qualsiasi momento.

Prima di iniziare

Assicurarsi che la scadenza del token di sessione AWS sia sufficiente per qualsiasi operazione di ripristino S3 di lunga durata. Se il token scade durante l'operazione di ripristino, l'operazione potrebbe non riuscire.

- Fare riferimento al "Documentazione AWS API" per ulteriori informazioni sulla verifica della scadenza del token della sessione corrente.
- Fare riferimento al "Documentazione AWS IAM" per ulteriori informazioni sulle credenziali con le risorse AWS.

Utilizzare un CR

Passi

1. Crea il file di risorse personalizzate (CR) e assegnagli un nome `trident-protect-snapshot-ipr-cr.yaml` .
2. Nel file creato, configura i seguenti attributi:
   - **metadata.name**: (*Obbligatorio*) Il nome di questa risorsa personalizzata; scegli un nome univoco e sensato per il tuo ambiente.
   - **spec.appVaultRef**: (*Obbligatorio*) Nome dell'AppVault in cui sono archiviati i contenuti dello snapshot.
   - **spec.appArchivePath**: il percorso all'interno di AppVault in cui sono archiviati i contenuti dello snapshot. Per trovare questo percorso puoi usare il seguente comando:

     ```
     kubectl get snapshots <SNAPSHOT_NAME> -n my-app-namespace -o jsonpath='{.status.appArchivePath}'
     ```

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: SnapshotInplaceRestore
     metadata:
       name: my-cr-name
       namespace: my-app-namespace
     spec:
       appVaultRef: appvault-name
         appArchivePath: my-snapshot-path
     ```

3. (*Facoltativo*) Se è necessario selezionare solo determinate risorse dell'applicazione da ripristinare, aggiungere un filtro che includa o escluda le risorse contrassegnate con etichette particolari:

   > Trident Protect seleziona automaticamente alcune risorse in base alla loro relazione con le risorse selezionate. Ad esempio, se selezioni una risorsa di richiesta di volume persistente e a questa è associato un pod, Trident Protect ripristinerà anche il pod associato.

   - **resourceFilter.resourceSelectionCriteria**: (Obbligatorio per il filtraggio) Utilizzare `Include` O `Exclude` per includere o escludere una risorsa definita in resourceMatchers. Aggiungere i seguenti parametri resourceMatchers per definire le risorse da includere o escludere:
     - **resourceFilter.resourceMatchers**: un array di oggetti resourceMatcher. Se si definiscono più elementi in questo array, essi corrispondono come un'operazione OR e i campi all'interno di ciascun elemento (gruppo, tipo, versione) corrispondono come un'operazione AND.
       - **resourceMatchers[].group**: (*Facoltativo*) Gruppo della risorsa da filtrare.
       - **resourceMatchers[].kind**: (*Facoltativo*) Tipo di risorsa da filtrare.
       - **resourceMatchers[].version**: (*Facoltativo*) Versione della risorsa da filtrare.

- **resourceMatchers[].names**: (*Facoltativo*) Nomi nel campo metadata.name di Kubernetes della risorsa da filtrare.
- **resourceMatchers[].namespaces**: (*Facoltativo*) Spazi dei nomi nel campo metadata.name di Kubernetes della risorsa da filtrare.
- **resourceMatchers[].labelSelectors**: (*Facoltativo*) Stringa del selettore di etichetta nel campo metadata.name di Kubernetes della risorsa come definito in "Documentazione di Kubernetes" . Per esempio: `"trident.netapp.io/os=linux"` .

  Per esempio:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Dopo aver popolato il `trident-protect-snapshot-ipr-cr.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-snapshot-ipr-cr.yaml
```

Utilizzare la CLI

Passi

1. Ripristina lo snapshot nello spazio dei nomi originale, sostituendo i valori tra parentesi con le informazioni del tuo ambiente. Per esempio:

```
tridentctl-protect create snapshotinplacerestore <my_restore_name> \
--snapshot <snapshot_to_restore> \
-n <application_namespace>
```

Controllare lo stato di un'operazione di ripristino

È possibile utilizzare la riga di comando per verificare lo stato di un'operazione di ripristino in corso, completata o non riuscita.

Passi

1. Utilizzare il seguente comando per recuperare lo stato dell'operazione di ripristino, sostituendo i valori tra parentesi con le informazioni provenienti dal proprio ambiente:

```
kubectl get backuprestore -n <namespace_name> <my_restore_cr_name> -o
jsonpath='{.status}'
```

Utilizzare le impostazioni di ripristino avanzate Trident Protect

È possibile personalizzare le operazioni di ripristino utilizzando impostazioni avanzate quali annotazioni, impostazioni dello spazio dei nomi e opzioni di archiviazione per soddisfare esigenze specifiche.

Annotazioni e etichette dello spazio dei nomi durante le operazioni di ripristino e failover

Durante le operazioni di ripristino e failover, le etichette e le annotazioni nello spazio dei nomi di destinazione vengono create in modo da corrispondere alle etichette e alle annotazioni nello spazio dei nomi di origine. Vengono aggiunte etichette o annotazioni provenienti dallo spazio dei nomi di origine che non esistono nello spazio dei nomi di destinazione e tutte le etichette o annotazioni già esistenti vengono sovrascritte in modo che corrispondano al valore dello spazio dei nomi di origine. Le etichette o le annotazioni che esistono solo nello spazio dei nomi di destinazione rimangono invariate.

> Se si utilizza Red Hat OpenShift, è importante tenere presente il ruolo fondamentale delle annotazioni dello spazio dei nomi negli ambienti OpenShift. Le annotazioni dello spazio dei nomi garantiscono che i pod ripristinati aderiscano alle autorizzazioni appropriate e alle configurazioni di sicurezza definite dai vincoli del contesto di sicurezza (SCC) di OpenShift e possano accedere ai volumi senza problemi di autorizzazione. Per maggiori informazioni, fare riferimento al "Documentazione sui vincoli del contesto di sicurezza di OpenShift" .

È possibile impedire che annotazioni specifiche nello spazio dei nomi di destinazione vengano sovrascritte impostando la variabile di ambiente Kubernetes `RESTORE_SKIP_NAMESPACE_ANNOTATIONS` prima di eseguire l'operazione di ripristino o failover. Per esempio:

```
helm upgrade trident-protect --set
restoreSkipNamespaceAnnotations=<annotation_key_to_skip_1>,<annotation_key
_to_skip_2> --reuse-values
```

> Quando si esegue un'operazione di ripristino o failover, tutte le annotazioni e le etichette dello spazio dei nomi specificate in `restoreSkipNamespaceAnnotations` E `restoreSkipNamespaceLabels` sono esclusi dall'operazione di ripristino o failover. Assicurarsi che queste impostazioni siano configurate durante l'installazione iniziale di Helm. Per saperne di più, fare riferimento a "Configurare le opzioni di filtraggio AutoSupport e namespace".

Se hai installato l'applicazione sorgente utilizzando Helm con `--create-namespace` bandiera, un trattamento speciale è riservato al `name` etichetta chiave. Durante il processo di ripristino o failover, Trident Protect copia questa etichetta nello spazio dei nomi di destinazione, ma aggiorna il valore al valore dello spazio dei nomi di destinazione se il valore dell'origine corrisponde allo spazio dei nomi di origine. Se questo valore non corrisponde allo spazio dei nomi di origine, viene copiato nello spazio dei nomi di destinazione senza modifiche.

Esempio

L'esempio seguente presenta uno spazio dei nomi di origine e di destinazione, ciascuno con annotazioni ed etichette diverse. È possibile visualizzare lo stato dello spazio dei nomi di destinazione prima e dopo l'operazione e il modo in cui le annotazioni e le etichette vengono combinate o sovrascritte nello spazio dei nomi di destinazione.

Prima dell'operazione di ripristino o failover

La tabella seguente illustra lo stato degli spazi dei nomi di origine e di destinazione di esempio prima dell'operazione di ripristino o failover:

Spazio dei nomi	Annotazioni	Etichette
Namespace ns-1 (fonte)	• annotation.one/key: "updatedvalue" • annotation.two/key: "true"	• ambiente=produzione • conformità=hipaa • nome=ns-1
Namespace ns-2 (destinazione)	• annotation.one/key: "true" • annotazione.tre/chiave: "falso"	• ruolo=database

Dopo l'operazione di ripristino

La tabella seguente illustra lo stato dello spazio dei nomi di destinazione di esempio dopo l'operazione di ripristino o failover. Alcune chiavi sono state aggiunte, alcune sono state sovrascritte e `name` l'etichetta è stata aggiornata per corrispondere allo spazio dei nomi di destinazione:

Spazio dei nomi	Annotazioni	Etichette
Namespace ns-2 (destinazione)	• annotation.one/key: "updatedvalue" • annotation.two/key: "true" • annotazione.tre/chiave: "falso"	• nome=ns-2 • conformità=hipaa • ambiente=produzione • ruolo=database

Campi supportati

Questa sezione descrive i campi aggiuntivi disponibili per le operazioni di ripristino.

Mappatura delle classi di archiviazione

IL `spec.storageClassMapping` L'attributo definisce una mappatura da una classe di archiviazione presente nell'applicazione di origine a una nuova classe di archiviazione nel cluster di destinazione. È possibile

utilizzarlo durante la migrazione di applicazioni tra cluster con classi di archiviazione diverse o quando si modifica il backend di archiviazione per le operazioni BackupRestore.

Esempio:

```
storageClassMapping:
  - destination: "destinationStorageClass1"
    source: "sourceStorageClass1"
  - destination: "destinationStorageClass2"
    source: "sourceStorageClass2"
```

Annotazioni supportate

Questa sezione elenca le annotazioni supportate per la configurazione di vari comportamenti nel sistema. Se un'annotazione non viene impostata esplicitamente dall'utente, il sistema utilizzerà il valore predefinito.

Annotazione	Tipo	Descrizione	Valore predefinito
protect.trident.netapp.io/data-mover-timeout-sec	corda	Tempo massimo (in secondi) consentito per l'interruzione dell'operazione di spostamento dei dati.	"300"
protect.trident.netapp.io/kopia-content-cache-size-limit-mb	corda	Limite massimo di dimensione (in megabyte) per la cache dei contenuti di Kopia.	"1000"

Replica le applicazioni utilizzando NetApp SnapMirror e Trident Protect

Utilizzando Trident Protect, è possibile sfruttare le funzionalità di replica asincrona della tecnologia NetApp SnapMirror per replicare i dati e le modifiche delle applicazioni da un backend di storage all'altro, sullo stesso cluster o tra cluster diversi.

Annotazioni e etichette dello spazio dei nomi durante le operazioni di ripristino e failover

Durante le operazioni di ripristino e failover, le etichette e le annotazioni nello spazio dei nomi di destinazione vengono create in modo da corrispondere alle etichette e alle annotazioni nello spazio dei nomi di origine. Vengono aggiunte etichette o annotazioni provenienti dallo spazio dei nomi di origine che non esistono nello spazio dei nomi di destinazione e tutte le etichette o annotazioni già esistenti vengono sovrascritte in modo che corrispondano al valore dello spazio dei nomi di origine. Le etichette o le annotazioni che esistono solo nello spazio dei nomi di destinazione rimangono invariate.

Se si utilizza Red Hat OpenShift, è importante tenere presente il ruolo fondamentale delle annotazioni dello spazio dei nomi negli ambienti OpenShift. Le annotazioni dello spazio dei nomi garantiscono che i pod ripristinati aderiscano alle autorizzazioni appropriate e alle configurazioni di sicurezza definite dai vincoli del contesto di sicurezza (SCC) di OpenShift e possano accedere ai volumi senza problemi di autorizzazione. Per maggiori informazioni, fare riferimento al "Documentazione sui vincoli del contesto di sicurezza di OpenShift" .

È possibile impedire che annotazioni specifiche nello spazio dei nomi di destinazione vengano sovrascritte impostando la variabile di ambiente Kubernetes `RESTORE_SKIP_NAMESPACE_ANNOTATIONS` prima di eseguire l'operazione di ripristino o failover. Per esempio:

```
helm upgrade trident-protect --set
restoreSkipNamespaceAnnotations=<annotation_key_to_skip_1>,<annotation_key
_to_skip_2> --reuse-values
```

> Quando si esegue un'operazione di ripristino o failover, tutte le annotazioni e le etichette dello spazio dei nomi specificate in `restoreSkipNamespaceAnnotations` E `restoreSkipNamespaceLabels` sono esclusi dall'operazione di ripristino o failover. Assicurarsi che queste impostazioni siano configurate durante l'installazione iniziale di Helm. Per saperne di più, fare riferimento a "Configurare le opzioni di filtraggio AutoSupport e namespace".

Se hai installato l'applicazione sorgente utilizzando Helm con `--create-namespace` bandiera, un trattamento speciale è riservato al `name` etichetta chiave. Durante il processo di ripristino o failover, Trident Protect copia questa etichetta nello spazio dei nomi di destinazione, ma aggiorna il valore al valore dello spazio dei nomi di destinazione se il valore dell'origine corrisponde allo spazio dei nomi di origine. Se questo valore non corrisponde allo spazio dei nomi di origine, viene copiato nello spazio dei nomi di destinazione senza modifiche.

Esempio

L'esempio seguente presenta uno spazio dei nomi di origine e di destinazione, ciascuno con annotazioni ed etichette diverse. È possibile visualizzare lo stato dello spazio dei nomi di destinazione prima e dopo l'operazione e il modo in cui le annotazioni e le etichette vengono combinate o sovrascritte nello spazio dei nomi di destinazione.

Prima dell'operazione di ripristino o failover

La tabella seguente illustra lo stato degli spazi dei nomi di origine e di destinazione di esempio prima dell'operazione di ripristino o failover:

Spazio dei nomi	Annotazioni	Etichette
Namespace ns-1 (fonte)	• annotation.one/key: "updatedvalue" • annotation.two/key: "true"	• ambiente=produzione • conformità=hipaa • nome=ns-1
Namespace ns-2 (destinazione)	• annotation.one/key: "true" • annotazione.tre/chiave: "falso"	• ruolo=database

Dopo l'operazione di ripristino

La tabella seguente illustra lo stato dello spazio dei nomi di destinazione di esempio dopo l'operazione di ripristino o failover. Alcune chiavi sono state aggiunte, alcune sono state sovrascritte e `name` l'etichetta è stata aggiornata per corrispondere allo spazio dei nomi di destinazione:

Spazio dei nomi	Annotazioni	Etichette
Namespace ns-2 (destinazione)	• annotation.one/key: "updatedvalue" • annotation.two/key: "true" • annotazione.tre/chiave: "falso"	• nome=ns-2 • conformità=hipaa • ambiente=produzione • ruolo=database

È possibile configurare Trident Protect in modo che blocchi e sblocchi i file system durante le operazioni di protezione dei dati. "Scopri di più sulla configurazione del congelamento del file system con Trident Protect".

Hook di esecuzione durante le operazioni di failover e reverse

Quando si utilizza la relazione AppMirror per proteggere l'applicazione, è necessario essere a conoscenza di comportamenti specifici relativi agli hook di esecuzione durante le operazioni di failover e reverse.

- Durante il failover, gli hook di esecuzione vengono copiati automaticamente dal cluster di origine al cluster di destinazione. Non è necessario ricrearli manualmente. Dopo il failover, gli hook di esecuzione sono presenti nell'applicazione e verranno eseguiti durante tutte le azioni rilevanti.
- Durante la sincronizzazione inversa o la risincronizzazione inversa, tutti gli hook di esecuzione esistenti sull'applicazione vengono rimossi. Quando l'applicazione di origine diventa l'applicazione di destinazione, questi hook di esecuzione non sono validi e vengono eliminati per impedirne l'esecuzione.

Per saperne di più sugli hook di esecuzione, fare riferimento a"Gestisci i ganci di esecuzione Trident Protect" .

Impostare una relazione di replicazione

L'impostazione di una relazione di replicazione comporta quanto segue:

- Scegliere la frequenza con cui si desidera che Trident Protect esegua uno snapshot dell'app (che include le risorse Kubernetes dell'app e gli snapshot del volume per ciascuno dei volumi dell'app)
- Scelta della pianificazione della replica (include risorse Kubernetes e dati di volume persistenti)
- Impostazione dell'ora in cui verrà scattata l'istantanea

Passi

1. Nel cluster di origine, creare un AppVault per l'applicazione di origine. A seconda del provider di archiviazione, modifica un esempio in"Risorse personalizzate di AppVault" per adattarsi al tuo ambiente:

Crea un AppVault utilizzando una CR

a. Crea il file di risorse personalizzate (CR) e assegnagli un nome (ad esempio, `trident-protect-appvault-primary-source.yaml`).

b. Configurare i seguenti attributi:

- **metadata.name**: (*Obbligatorio*) Nome della risorsa personalizzata di AppVault. Prendi nota del nome scelto, perché altri file CR necessari per una relazione di replica fanno riferimento a questo valore.
- **spec.providerConfig**: (*Obbligatorio*) Memorizza la configurazione necessaria per accedere ad AppVault utilizzando il provider specificato. Scegli un bucketName e tutti gli altri dettagli necessari per il tuo provider. Prendi nota dei valori scelti, perché altri file CR necessari per una relazione di replicazione fanno riferimento a questi valori. Fare riferimento a"Risorse personalizzate di AppVault" per esempi di CR AppVault con altri provider.
- **spec.providerCredentials**: (*Obbligatorio*) Memorizza i riferimenti a qualsiasi credenziale richiesta per accedere ad AppVault utilizzando il provider specificato.
  - **spec.providerCredentials.valueFromSecret**: (*Obbligatorio*) Indica che il valore delle credenziali deve provenire da un segreto.
    - **key**: (*Obbligatorio*) La chiave valida del segreto da cui effettuare la selezione.
    - **name**: (*Obbligatorio*) Nome del segreto contenente il valore per questo campo. Devono trovarsi nello stesso namespace.
  - **spec.providerCredentials.secretAccessKey**: (*Obbligatorio*) La chiave di accesso utilizzata per accedere al provider. Il **nome** deve corrispondere a **spec.providerCredentials.valueFromSecret.name**.
- **spec.providerType**: (*Obbligatorio*) Determina chi fornisce il backup; ad esempio, NetApp ONTAP S3, S3 generico, Google Cloud o Microsoft Azure. Valori possibili:
  - aws
  - azzurro
  - gcp
  - generico-s3
  - ontap-s3
  - storagegrid-s3

c. Dopo aver popolato il `trident-protect-appvault-primary-source.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-appvault-primary-source.yaml -n
trident-protect
```

Creare un AppVault utilizzando la CLI

a. Crea AppVault, sostituendo i valori tra parentesi con le informazioni del tuo ambiente:

```
tridentctl-protect create vault Azure <vault-name> --account
<account-name> --bucket <bucket-name> --secret <secret-name>
```

2. Nel cluster di origine, creare l'applicazione di origine CR:

Creare l'applicazione sorgente utilizzando un CR

a. Crea il file di risorse personalizzate (CR) e assegnagli un nome (ad esempio, `trident-protect-app-source.yaml`).

b. Configurare i seguenti attributi:

- **metadata.name**: (*Obbligatorio*) Nome della risorsa personalizzata dell'applicazione. Prendi nota del nome scelto, perché altri file CR necessari per una relazione di replica fanno riferimento a questo valore.
- **spec.includedNamespaces**: (*Obbligatorio*) Un array di namespace ed etichette associate. Utilizzare i nomi degli spazi dei nomi e, facoltativamente, restringere l'ambito degli spazi dei nomi con etichette per specificare le risorse presenti negli spazi dei nomi elencati qui. Lo spazio dei nomi dell'applicazione deve far parte di questo array.

  Esempio YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  name: my-app-name
  namespace: my-app-namespace
spec:
  includedNamespaces:
    - namespace: my-app-namespace
      labelSelector: {}
```

c. Dopo aver popolato il `trident-protect-app-source.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-app-source.yaml -n my-app-namespace
```

Creare l'applicazione sorgente utilizzando la CLI

a. Creare l'applicazione sorgente. Per esempio:

```
tridentctl-protect create app <my-app-name> --namespaces <namespaces-to-be-included> -n <my-app-namespace>
```

3. Facoltativamente, sul cluster di origine, eseguire uno snapshot dell'applicazione di origine. Questo snapshot viene utilizzato come base per l'applicazione sul cluster di destinazione. Se si salta questo passaggio, sarà necessario attendere l'esecuzione del prossimo snapshot pianificato per avere uno

snapshot recente.

Oltre alla pianificazione fornita di seguito, si consiglia di creare una pianificazione di snapshot giornalieri separata con un periodo di conservazione di 7 giorni per mantenere uno snapshot comune tra i cluster ONTAP peered. Ciò garantisce che gli snapshot siano disponibili fino a 7 giorni, ma il periodo di conservazione può essere personalizzato in base alle esigenze dell'utente.

In caso di failover, il sistema può utilizzare questi snapshot per un massimo di 7 giorni per le operazioni inverse. Questo approccio rende il processo inverso più rapido ed efficiente, perché verranno trasferite solo le modifiche apportate dall'ultimo snapshot, non tutti i dati.

Se una pianificazione esistente per l'applicazione soddisfa già i requisiti di conservazione desiderati, non sono necessarie pianificazioni aggiuntive.

Scatta un'istantanea utilizzando un CR

a. Creare una pianificazione di replica per l'applicazione di origine:

i. Crea il file di risorse personalizzate (CR) e assegnagli un nome (ad esempio, `trident-protect-schedule.yaml`).

ii. Configurare i seguenti attributi:

- **metadata.name**: (*Obbligatorio*) Nome della risorsa personalizzata della pianificazione.
- **spec.AppVaultRef**: (*Obbligatorio*) Questo valore deve corrispondere al campo metadata.name dell'AppVault per l'applicazione di origine.
- **spec.ApplicationRef**: (*Obbligatorio*) Questo valore deve corrispondere al campo metadata.name del CR dell'applicazione di origine.
- **spec.backupRetention**: (*Obbligatorio*) Questo campo è obbligatorio e il valore deve essere impostato su 0.
- **spec.enabled**: Deve essere impostato su true.
- **spec.granularity**: Deve essere impostato su `Custom` .
- **spec.recurrenceRule**: definisce una data di inizio in ora UTC e un intervallo di ricorrenza.
- **spec.snapshotRetention**: Deve essere impostato su 2.

Esempio YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  name: appmirror-schedule-0e1f88ab-f013-4bce-8ae9-6afed9df59a1
  namespace: my-app-namespace
spec:
  appVaultRef: generic-s3-trident-protect-src-bucket-04b6b4ec-
46a3-420a-b351-45795e1b5e34
  applicationRef: my-app-name
  backupRetention: "0"
  enabled: true
  granularity: custom
  recurrenceRule: |-
    DTSTART:20220101T000200Z
    RRULE:FREQ=MINUTELY;INTERVAL=5
  snapshotRetention: "2"
```

i. Dopo aver popolato il `trident-protect-schedule.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-schedule.yaml -n my-app-
namespace
```

Scatta uno snapshot utilizzando la CLI

a. Crea lo snapshot sostituendo i valori tra parentesi con le informazioni provenienti dal tuo ambiente. Per esempio:

```
tridentctl-protect create snapshot <my_snapshot_name> --appvault
<my_appvault_name> --app <name_of_app_to_snapshot> -n
<application_namespace>
```

4. Nel cluster di destinazione, crea un'applicazione di origine AppVault CR identica all'AppVault CR applicata nel cluster di origine e assegnale un nome (ad esempio, `trident-protect-appvault-primary-destination.yaml`).
5. Applicare il CR:

```
kubectl apply -f trident-protect-appvault-primary-destination.yaml -n
my-app-namespace
```

6. Creare un CR AppVault di destinazione per l'applicazione di destinazione sul cluster di destinazione. A seconda del provider di archiviazione, modificare un esempio in"Risorse personalizzate di AppVault" per adattarsi al tuo ambiente:
   a. Crea il file di risorse personalizzate (CR) e assegnagli un nome (ad esempio, `trident-protect-appvault-secondary-destination.yaml`).
   b. Configurare i seguenti attributi:
      - **metadata.name**: (*Obbligatorio*) Nome della risorsa personalizzata di AppVault. Prendi nota del nome scelto, perché altri file CR necessari per una relazione di replica fanno riferimento a questo valore.
      - **spec.providerConfig**: (*Obbligatorio*) Memorizza la configurazione necessaria per accedere ad AppVault utilizzando il provider specificato. Scegli un `bucketName` e qualsiasi altro dettaglio necessario per il tuo fornitore. Prendi nota dei valori scelti, perché altri file CR necessari per una relazione di replicazione fanno riferimento a questi valori. Fare riferimento a"Risorse personalizzate di AppVault" per esempi di CR AppVault con altri provider.
      - **spec.providerCredentials**: (*Obbligatorio*) Memorizza i riferimenti a qualsiasi credenziale richiesta per accedere ad AppVault utilizzando il provider specificato.
        - **spec.providerCredentials.valueFromSecret**: (*Obbligatorio*) Indica che il valore delle credenziali deve provenire da un segreto.
          - **key**: (*Obbligatorio*) La chiave valida del segreto da cui effettuare la selezione.
          - **name**: (*Obbligatorio*) Nome del segreto contenente il valore per questo campo. Devono trovarsi nello stesso namespace.
        - **spec.providerCredentials.secretAccessKey**: (*Obbligatorio*) La chiave di accesso utilizzata

per accedere al provider. Il **nome** deve corrispondere a **spec.providerCredentials.valueFromSecret.name**.

- **spec.providerType**: (*Obbligatorio*) Determina chi fornisce il backup; ad esempio, NetApp ONTAP S3, S3 generico, Google Cloud o Microsoft Azure. Valori possibili:
  - aws
  - azzurro
  - gcp
  - generico-s3
  - ontap-s3
  - storagegrid-s3

c. Dopo aver popolato il `trident-protect-appvault-secondary-destination.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-appvault-secondary-destination.yaml
-n my-app-namespace
```

7. Nel cluster di destinazione, creare un file CR AppMirrorRelationship:

Creare un AppMirrorRelationship utilizzando un CR

a. Crea il file di risorse personalizzate (CR) e assegnagli un nome (ad esempio, `trident-protect-relationship.yaml`).

b. Configurare i seguenti attributi:

- **metadata.name:** (obbligatorio) Nome della risorsa personalizzata AppMirrorRelationship.
- **spec.destinationAppVaultRef**: (*Obbligatorio*) Questo valore deve corrispondere al nome dell'AppVault per l'applicazione di destinazione sul cluster di destinazione.
- **spec.namespaceMapping**: (*Obbligatorio*) Gli spazi dei nomi di destinazione e di origine devono corrispondere allo spazio dei nomi dell'applicazione definito nel rispettivo CR dell'applicazione.
- **spec.sourceAppVaultRef**: (*Obbligatorio*) Questo valore deve corrispondere al nome dell'AppVault per l'applicazione di origine.
- **spec.sourceApplicationName**: (*Obbligatorio*) Questo valore deve corrispondere al nome dell'applicazione sorgente definita nel CR dell'applicazione sorgente.
- **spec.storageClassName**: (*Obbligatorio*) Scegli il nome di una classe di archiviazione valida sul cluster. La classe di archiviazione deve essere collegata a una VM di archiviazione ONTAP collegata in peering con l'ambiente di origine.
- **spec.recurrenceRule**: definisce una data di inizio in ora UTC e un intervallo di ricorrenza.

Esempio YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: AppMirrorRelationship
metadata:
  name: amr-16061e80-1b05-4e80-9d26-d326dc1953d8
  namespace: my-app-namespace
spec:
  desiredState: Established
  destinationAppVaultRef: generic-s3-trident-protect-dst-bucket-
8fe0b902-f369-4317-93d1-ad7f2edc02b5
  namespaceMapping:
    - destination: my-app-namespace
      source: my-app-namespace
  recurrenceRule: |-
    DTSTART:20220101T000200Z
    RRULE:FREQ=MINUTELY;INTERVAL=5
  sourceAppVaultRef: generic-s3-trident-protect-src-bucket-
b643cc50-0429-4ad5-971f-ac4a83621922
  sourceApplicationName: my-app-name
  sourceApplicationUID: 7498d32c-328e-4ddd-9029-122540866aeb
  storageClassName: sc-vsim-2
```

c. Dopo aver popolato il `trident-protect-relationship.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-
namespace
```

Creare un AppMirrorRelationship utilizzando la CLI

a. Crea e applica l'oggetto AppMirrorRelationship, sostituendo i valori tra parentesi con le informazioni provenienti dal tuo ambiente. Per esempio:

```
tridentctl-protect create appmirrorrelationship
<name_of_appmirorrelationship> --destination-app-vault
<my_vault_name> --recurrence-rule <rule> --source-app
<my_source_app> --source-app-vault <my_source_app_vault> -n
<application_namespace>
```

8. (*Facoltativo*) Nel cluster di destinazione, controllare lo stato e la relazione di replica:

```
kubectl get amr -n my-app-namespace <relationship name> -o=jsonpath
='{.status}' | jq
```

Failover sul cluster di destinazione

Utilizzando Trident Protect, è possibile eseguire il failover delle applicazioni replicate su un cluster di destinazione. Questa procedura interrompe la relazione di replica e porta l'app online sul cluster di destinazione. Trident Protect non arresta l'app sul cluster di origine se era operativa.

Passi

1. Nel cluster di destinazione, modificare il file CR AppMirrorRelationship (ad esempio, `trident-protect-relationship.yaml`) e modificare il valore di **spec.desiredState** in `Promoted` .
2. Salvare il file CR.
3. Applicare il CR:

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-namespace
```

4. (*Facoltativo*) Creare tutte le pianificazioni di protezione necessarie sull'applicazione sottoposta a failover.
5. (*Facoltativo*) Controllare lo stato e la situazione della relazione di replicazione:

```
kubectl get amr -n my-app-namespace <relationship name> -o=jsonpath
='{.status}' | jq
```

Risincronizzare una relazione di replicazione non riuscita

L'operazione di risincronizzazione ristabilisce la relazione di replica. Dopo aver eseguito un'operazione di risincronizzazione, l'applicazione di origine diventa l'applicazione in esecuzione e tutte le modifiche apportate all'applicazione in esecuzione sul cluster di destinazione vengono ignorate.

Il processo arresta l'app sul cluster di destinazione prima di ristabilire la replica.

> Tutti i dati scritti nell'applicazione di destinazione durante il failover andranno persi.

Passi

1. Facoltativo: sul cluster di origine, creare uno snapshot dell'applicazione di origine. Ciò garantisce che vengano acquisite le ultime modifiche dal cluster di origine.
2. Nel cluster di destinazione, modificare il file CR AppMirrorRelationship (ad esempio, `trident-protect-relationship.yaml`) e modificare il valore di spec.desiredState in `Established` .
3. Salvare il file CR.
4. Applicare il CR:

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-namespace
```

5. Se sono state create delle pianificazioni di protezione sul cluster di destinazione per proteggere l'applicazione sottoposta a failover, rimuoverle. Tutte le pianificazioni rimanenti causano errori negli snapshot del volume.

Risincronizzazione inversa di una relazione di replicazione non riuscita

Quando si esegue la risincronizzazione inversa di una relazione di replicazione non riuscita, l'applicazione di destinazione diventa l'applicazione di origine e l'origine diventa la destinazione. Le modifiche apportate all'applicazione di destinazione durante il failover vengono mantenute.

Passi

1. Nel cluster di destinazione originale, eliminare il CR AppMirrorRelationship. Ciò fa sì che la destinazione diventi la sorgente. Se sul nuovo cluster di destinazione sono presenti pianificazioni di protezione rimanenti, rimuoverle.
2. Imposta una relazione di replica applicando i file CR utilizzati originariamente per impostare la relazione ai cluster opposti.
3. Assicurarsi che la nuova destinazione (cluster di origine originale) sia configurata con entrambi i CR di AppVault.
4. Impostare una relazione di replicazione sul cluster opposto, configurando i valori per la direzione inversa.

Invertire la direzione di replicazione dell'applicazione

Quando si inverte la direzione della replica, Trident Protect sposta l'applicazione sul backend di archiviazione di destinazione, continuando a replicare sul backend di archiviazione di origine. Trident Protect arresta l'applicazione di origine e replica i dati nella destinazione prima di eseguire il failover sull'app di destinazione.

In questa situazione si scambiano la sorgente e la destinazione.

Passi

1. Nel cluster di origine, creare uno snapshot di arresto:

Creare uno snapshot di arresto utilizzando un CR

a. Disattivare le pianificazioni dei criteri di protezione per l'applicazione di origine.

b. Creare un file CR ShutdownSnapshot:

   i. Crea il file di risorse personalizzate (CR) e assegnagli un nome (ad esempio, `trident-protect-shutdownsnapshot.yaml`).

   ii. Configurare i seguenti attributi:

   - **metadata.name**: (*Obbligatorio*) Nome della risorsa personalizzata.
   - **spec.AppVaultRef**: (*Obbligatorio*) Questo valore deve corrispondere al campo metadata.name dell'AppVault per l'applicazione di origine.
   - **spec.ApplicationRef**: (*Obbligatorio*) Questo valore deve corrispondere al campo metadata.name del file CR dell'applicazione di origine.

     Esempio YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: ShutdownSnapshot
metadata:
  name: replication-shutdown-snapshot-afc4c564-e700-4b72-86c3-
c08a5dbe844e
  namespace: my-app-namespace
spec:
  appVaultRef: generic-s3-trident-protect-src-bucket-04b6b4ec-
46a3-420a-b351-45795e1b5e34
  applicationRef: my-app-name
```

c. Dopo aver popolato il `trident-protect-shutdownsnapshot.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-shutdownsnapshot.yaml -n my-app-
namespace
```

Creare uno snapshot di arresto utilizzando la CLI

a. Crea lo snapshot di arresto, sostituendo i valori tra parentesi con le informazioni provenienti dal tuo ambiente. Per esempio:

```
tridentctl-protect create shutdownsnapshot <my_shutdown_snapshot>
--appvault <my_vault> --app <app_to_snapshot> -n
<application_namespace>
```

2. Nel cluster di origine, una volta completato lo snapshot di arresto, ottenere lo stato dello snapshot di arresto:

```
kubectl get shutdownsnapshot -n my-app-namespace
<shutdown_snapshot_name> -o yaml
```

3. Nel cluster di origine, trova il valore di **shutdownsnapshot.status.appArchivePath** utilizzando il seguente comando e registra l'ultima parte del percorso del file (chiamato anche basename; sarà tutto ciò che segue l'ultima barra):

```
k get shutdownsnapshot -n my-app-namespace <shutdown_snapshot_name> -o
jsonpath='{.status.appArchivePath}'
```

4. Eseguire un failover dal nuovo cluster di destinazione al nuovo cluster di origine, con la seguente modifica:

> Nel passaggio 2 della procedura di failover, includere `spec.promotedSnapshot` nel file CR AppMirrorRelationship e impostarne il valore sul nome base registrato nel passaggio 3 precedente.

5. Eseguire i passaggi di risincronizzazione inversa inRisincronizzazione inversa di una relazione di replicazione non riuscita .
6. Abilitare le pianificazioni di protezione sul nuovo cluster di origine.

Risultato

A causa della replicazione inversa si verificano le seguenti azioni:

- Viene creato uno snapshot delle risorse Kubernetes dell'app sorgente originale.
- I pod dell'app sorgente originale vengono arrestati correttamente eliminando le risorse Kubernetes dell'app (lasciando in posizione PVC e PV).
- Dopo aver arrestato i pod, vengono acquisiti e replicati gli snapshot dei volumi dell'app.
- Le relazioni SnapMirror vengono interrotte, rendendo i volumi di destinazione pronti per la lettura/scrittura.
- Le risorse Kubernetes dell'app vengono ripristinate dallo snapshot precedente all'arresto, utilizzando i dati del volume replicati dopo l'arresto dell'app sorgente originale.
- La replicazione viene ristabilita nella direzione inversa.

Eseguire il failback delle applicazioni al cluster di origine originale

Utilizzando Trident Protect, è possibile ottenere il "fail back" dopo un'operazione di failover utilizzando la seguente sequenza di operazioni. In questo flusso di lavoro per ripristinare la direzione di replicazione originale, Trident Protect replica (risincronizza) tutte le modifiche apportate all'applicazione di origine prima di invertire la direzione di replicazione.

Questo processo inizia da una relazione che ha completato un failover verso una destinazione e prevede i seguenti passaggi:

- Inizia con uno stato di failover.

- Risincronizza inversamente la relazione di replicazione.

> Non eseguire una normale operazione di risincronizzazione, poiché ciò eliminerà i dati scritti nel cluster di destinazione durante la procedura di failover.

- Invertire la direzione della replicazione.

Passi

1. Eseguire ilRisincronizzazione inversa di una relazione di replicazione non riuscita passi.
2. Eseguire ilInvertire la direzione di replicazione dell'applicazione passi.

Elimina una relazione di replicazione

È possibile eliminare una relazione di replicazione in qualsiasi momento. Quando si elimina la relazione di replica dell'applicazione, si ottengono due applicazioni separate senza alcuna relazione tra loro.

Passi

1. Nel cluster di destinazione corrente, eliminare il CR AppMirrorRelationship:

```
kubectl delete -f trident-protect-relationship.yaml -n my-app-namespace
```

Migrare le applicazioni utilizzando Trident Protect

È possibile migrare le applicazioni tra cluster o in classi di archiviazione diverse ripristinando i dati di backup.

> Quando si esegue la migrazione di un'applicazione, tutti gli hook di esecuzione configurati per l'applicazione vengono migrati insieme all'applicazione. Se è presente un hook di esecuzione post-ripristino, questo viene eseguito automaticamente come parte dell'operazione di ripristino.

Operazioni di backup e ripristino

Per eseguire operazioni di backup e ripristino per i seguenti scenari, è possibile automatizzare attività di backup e ripristino specifiche.

Clona nello stesso cluster

Per clonare un'applicazione nello stesso cluster, creare uno snapshot o un backup e ripristinare i dati nello stesso cluster.

Passi

1. Eseguire una delle seguenti operazioni:
   a. "Crea uno snapshot".
   b. "Crea un backup".
2. Sullo stesso cluster, esegui una delle seguenti operazioni, a seconda che tu abbia creato uno snapshot o un backup:
   a. "Ripristina i tuoi dati dallo snapshot".

b. "Ripristina i tuoi dati dal backup".

Clona in cluster diversi

Per clonare un'applicazione su un cluster diverso (eseguire un clone tra cluster), creare un backup sul cluster di origine, quindi ripristinare il backup su un cluster diverso. Assicurarsi che Trident Protect sia installato sul cluster di destinazione.

È possibile replicare un'applicazione tra cluster diversi utilizzando"Replica SnapMirror" .

Passi

1. "Crea un backup".
2. Assicurarsi che il CR di AppVault per il bucket di archiviazione degli oggetti che contiene il backup sia stato configurato sul cluster di destinazione.
3. Sul cluster di destinazione,"ripristina i tuoi dati dal backup" .

Migrare le applicazioni da una classe di archiviazione a un'altra classe di archiviazione

È possibile migrare le applicazioni da una classe di archiviazione a una diversa ripristinando un backup nella classe di archiviazione di destinazione.

Ad esempio (escludendo i segreti dal ripristino CR):

```
apiVersion: protect.trident.netapp.io/v1
kind: SnapshotRestore
metadata:
  name: "${snapshotRestoreCRName}"
spec:
  appArchivePath: "${snapshotArchivePath}"
  appVaultRef: "${appVaultCRName}"
  namespaceMapping:
    - destination: "${destinationNamespace}"
      source: "${sourceNamespace}"
  storageClassMapping:
    - destination: "${destinationStorageClass}"
      source: "${sourceStorageClass}"
  resourceFilter:
    resourceMatchers:
      kind: Secret
      version: v1
    resourceSelectionCriteria: exclude
```

Ripristinare lo snapshot utilizzando un CR

Passi

1. Crea il file di risorse personalizzate (CR) e assegnagli un nome `trident-protect-snapshot-restore-cr.yaml` .
2. Nel file creato, configura i seguenti attributi:
   - **metadata.name**: (*Obbligatorio*) Il nome di questa risorsa personalizzata; scegli un nome univoco e sensato per il tuo ambiente.
   - **spec.appArchivePath**: il percorso all'interno di AppVault in cui sono archiviati i contenuti dello snapshot. Per trovare questo percorso puoi usare il seguente comando:

     ```
     kubectl get snapshots <my-snapshot-name> -n trident-protect -o jsonpath='{.status.appArchivePath}'
     ```

   - **spec.appVaultRef**: (*Obbligatorio*) Nome dell'AppVault in cui sono archiviati i contenuti dello snapshot.
   - **spec.namespaceMapping**: la mappatura dello spazio dei nomi di origine dell'operazione di ripristino allo spazio dei nomi di destinazione. Sostituire `my-source-namespace` E `my-destination-namespace` con informazioni provenienti dal tuo ambiente.

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: SnapshotRestore
     metadata:
       name: my-cr-name
       namespace: trident-protect
     spec:
       appArchivePath: my-snapshot-path
       appVaultRef: appvault-name
       namespaceMapping: [{"source": "my-source-namespace",
     "destination": "my-destination-namespace"}]
     ```

3. Facoltativamente, se è necessario selezionare solo determinate risorse dell'applicazione da ripristinare, aggiungere un filtro che includa o escluda le risorse contrassegnate con etichette particolari:
   - **resourceFilter.resourceSelectionCriteria**: (obbligatorio per il filtraggio) Usa `include or exclude` per includere o escludere una risorsa definita in resourceMatchers. Aggiungere i seguenti parametri resourceMatchers per definire le risorse da includere o escludere:
     - **resourceFilter.resourceMatchers**: un array di oggetti resourceMatcher. Se si definiscono più elementi in questo array, essi corrispondono come un'operazione OR e i campi all'interno di ciascun elemento (gruppo, tipo, versione) corrispondono come un'operazione AND.
       - **resourceMatchers[].group**: (*Facoltativo*) Gruppo della risorsa da filtrare.
       - **resourceMatchers[].kind**: (*Facoltativo*) Tipo di risorsa da filtrare.

- **resourceMatchers[].version**: (*Facoltativo*) Versione della risorsa da filtrare.
- **resourceMatchers[].names**: (*Facoltativo*) Nomi nel campo metadata.name di Kubernetes della risorsa da filtrare.
- **resourceMatchers[].namespaces**: (*Facoltativo*) Spazi dei nomi nel campo metadata.name di Kubernetes della risorsa da filtrare.
- **resourceMatchers[].labelSelectors**: (*Facoltativo*) Stringa del selettore di etichetta nel campo metadata.name di Kubernetes della risorsa come definito in "Documentazione di Kubernetes" . Per esempio: `"trident.netapp.io/os=linux"` .

  Per esempio:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Dopo aver popolato il `trident-protect-snapshot-restore-cr.yaml` file con i valori corretti, applicare CR:

```
kubectl apply -f trident-protect-snapshot-restore-cr.yaml
```

Ripristinare lo snapshot utilizzando la CLI

Passi

1. Ripristina lo snapshot in uno spazio dei nomi diverso, sostituendo i valori tra parentesi con le informazioni del tuo ambiente.

   - IL `snapshot` l'argomento utilizza uno spazio dei nomi e un nome di snapshot nel formato `<namespace>/<name>` .
   - IL `namespace-mapping` l'argomento utilizza namespace separati da due punti per mappare i namespace di origine ai namespace di destinazione corretti nel formato `source1:dest1,source2:dest2` .

     Per esempio:

```
tridentctl-protect create snapshotrestore <my_restore_name>
--snapshot <namespace/snapshot_to_restore> --namespace-mapping
<source_to_destination_namespace_mapping>
```

Gestisci i ganci di esecuzione Trident Protect

Un hook di esecuzione è un'azione personalizzata che è possibile configurare per essere eseguita insieme a un'operazione di protezione dei dati di un'app gestita. Ad esempio, se si dispone di un'app di database, è possibile utilizzare un hook di esecuzione per mettere in pausa tutte le transazioni del database prima di uno snapshot e riprendere le transazioni al termine dello snapshot. Ciò garantisce snapshot coerenti con l'applicazione.

Tipi di ganci di esecuzione

Trident Protect supporta i seguenti tipi di hook di esecuzione, in base al momento in cui possono essere eseguiti:

- Pre-istantanea
- Post-istantanea
- Pre-backup
- Post-backup
- Post-ripristino
- Post-failover

Ordine di esecuzione

Quando viene eseguita un'operazione di protezione dei dati, gli eventi di hook di esecuzione si verificano nel seguente ordine:

1. Tutti gli hook di esecuzione pre-operazione personalizzati applicabili vengono eseguiti sui contenitori appropriati. È possibile creare ed eseguire tutti i hook pre-operazione personalizzati di cui si ha bisogno, ma l'ordine di esecuzione di questi hook prima dell'operazione non è né garantito né configurabile.
2. Se applicabile, si verificano blocchi del file system. "Scopri di più sulla configurazione del congelamento del file system con Trident Protect".
3. L'operazione di protezione dei dati è eseguita.
4. I file system congelati vengono sbloccati, se applicabile.
5. Tutti gli hook di esecuzione post-operazione personalizzati applicabili vengono eseguiti sui contenitori appropriati. È possibile creare ed eseguire tutti i hook post-operazione personalizzati di cui si ha bisogno, ma l'ordine di esecuzione di questi hook dopo l'operazione non è né garantito né configurabile.

Se si creano più hook di esecuzione dello stesso tipo (ad esempio, pre-snapshot), l'ordine di esecuzione di tali hook non è garantito. Tuttavia, l'ordine di esecuzione dei ganci di diverso tipo è garantito. Ad esempio, ecco l'ordine di esecuzione di una configurazione che presenta tutti i diversi tipi di hook:

1. Eseguiti i pre-snapshot hook
2. Eseguiti i ganci post-snapshot
3. Hook pre-backup eseguiti
4. Hook post-backup eseguiti

L'esempio dell'ordine precedente si applica solo quando si esegue un backup che non utilizza uno snapshot esistente.

Dovresti sempre testare gli script di esecuzione prima di abilitarli in un ambiente di produzione. È possibile utilizzare il comando 'kubectl exec' per testare comodamente gli script. Dopo aver abilitato gli hook di esecuzione in un ambiente di produzione, testare gli snapshot e i backup risultanti per assicurarsi che siano coerenti. È possibile farlo clonando l'app in uno spazio dei nomi temporaneo, ripristinando lo snapshot o il backup e quindi testando l'app.

Se un hook di esecuzione pre-snapshot aggiunge, modifica o rimuove risorse Kubernetes, tali modifiche vengono incluse nello snapshot o nel backup e in qualsiasi successiva operazione di ripristino.

Note importanti sui ganci di esecuzione personalizzati

Quando pianifichi gli hook di esecuzione per le tue app, tieni presente quanto segue.

- Un hook di esecuzione deve utilizzare uno script per eseguire azioni. Molti hook di esecuzione possono fare riferimento allo stesso script.
- Trident Protect richiede che gli script utilizzati dagli hook di esecuzione siano scritti nel formato degli script shell eseguibili.
- La dimensione dello script è limitata a 96 KB.
- Trident Protect utilizza le impostazioni dell'hook di esecuzione e tutti i criteri corrispondenti per determinare quali hook sono applicabili a un'operazione di snapshot, backup o ripristino.

Poiché gli hook di esecuzione spesso riducono o disabilitano completamente la funzionalità dell'applicazione su cui vengono eseguiti, dovresti sempre cercare di ridurre al minimo il tempo impiegato per l'esecuzione degli hook di esecuzione personalizzati. Se si avvia un'operazione di backup o snapshot con hook di esecuzione associati ma poi la si annulla, gli hook possono comunque essere eseguiti se l'operazione di backup o snapshot è già iniziata. Ciò significa che la logica utilizzata in un hook di esecuzione post-backup non può presumere che il backup sia stato completato.

Filtri di hook di esecuzione

Quando aggiungi o modifichi un hook di esecuzione per un'applicazione, puoi aggiungere filtri all'hook di esecuzione per gestire i contenitori a cui l'hook corrisponderà. I filtri sono utili per le applicazioni che utilizzano la stessa immagine contenitore su tutti i contenitori, ma potrebbero utilizzare ciascuna immagine per uno scopo diverso (ad esempio Elasticsearch). I filtri consentono di creare scenari in cui gli hook di esecuzione vengono eseguiti su alcuni contenitori identici, ma non necessariamente su tutti. Se si creano più filtri per un singolo hook di esecuzione, questi vengono combinati con un operatore logico AND. È possibile avere fino a 10 filtri attivi per ogni hook di esecuzione.

Ogni filtro aggiunto a un hook di esecuzione utilizza un'espressione regolare per abbinare i contenitori nel

cluster. Quando un hook corrisponde a un contenitore, eseguirà lo script associato su quel contenitore. Le espressioni regolari per i filtri utilizzano la sintassi Regular Expression 2 (RE2), che non supporta la creazione di un filtro che escluda i contenitori dall'elenco delle corrispondenze. Per informazioni sulla sintassi supportata Trident Protect per le espressioni regolari nei filtri di hook di esecuzione, vedere "Supporto della sintassi Regular Expression 2 (RE2)" .

> Se si aggiunge un filtro namespace a un hook di esecuzione eseguito dopo un'operazione di ripristino o clonazione e l'origine e la destinazione del ripristino o della clonazione si trovano in namespace diversi, il filtro namespace viene applicato solo al namespace di destinazione.

Esempi di hook di esecuzione

Visita il "Progetto GitHub NetApp Verda" per scaricare veri e propri hook di esecuzione per app popolari come Apache Cassandra ed Elasticsearch. Puoi anche vedere esempi e trarre spunti per strutturare i tuoi hook di esecuzione personalizzati.

Creare un hook di esecuzione

È possibile creare un hook di esecuzione personalizzato per un'app utilizzando Trident Protect. Per creare hook di esecuzione è necessario disporre delle autorizzazioni di Proprietario, Amministratore o Membro.

Utilizzare un CR

Passi

1. Crea il file di risorse personalizzate (CR) e assegnagli un nome `trident-protect-hook.yaml` .
2. Configurare i seguenti attributi in modo che corrispondano all'ambiente di protezione Trident e alla configurazione del cluster:
   - **metadata.name**: (*Obbligatorio*) Il nome di questa risorsa personalizzata; scegli un nome univoco e sensato per il tuo ambiente.
   - **spec.applicationRef**: (*Obbligatorio*) Nome Kubernetes dell'applicazione per cui eseguire l'hook di esecuzione.
   - **spec.stage**: (*Obbligatorio*) Una stringa che indica in quale fase dell'azione deve essere eseguito l'hook di esecuzione. Valori possibili:
     - Pre
     - Inviare
   - **spec.action**: (*Obbligatorio*) Una stringa che indica quale azione intraprenderà l'hook di esecuzione, supponendo che tutti i filtri dell'hook di esecuzione specificati corrispondano. Valori possibili:
     - Istantanea
     - Backup
     - Ripristinare
     - Failover
   - **spec.enabled**: (*Facoltativo*) Indica se questo hook di esecuzione è abilitato o disabilitato. Se non specificato, il valore predefinito è true.
   - **spec.hookSource**: (*Obbligatorio*) Una stringa contenente lo script hook codificato in base64.
   - **spec.timeout**: (*Facoltativo*) Un numero che definisce per quanti minuti è consentita l'esecuzione dell'hook. Il valore minimo è 1 minuto e il valore predefinito è 25 minuti se non specificato.
   - **spec.arguments**: (*Facoltativo*) Un elenco YAML di argomenti che è possibile specificare per l'hook di esecuzione.
   - **spec.matchingCriteria**: (*Facoltativo*) Un elenco facoltativo di coppie chiave-valore di criteri, ciascuna delle quali costituisce un filtro di hook di esecuzione. È possibile aggiungere fino a 10 filtri per ogni hook di esecuzione.
   - **spec.matchingCriteria.type**: (*Facoltativo*) Una stringa che identifica il tipo di filtro dell'hook di esecuzione. Valori possibili:
     - Immagine contenitore
     - NomeContenitore
     - NomePod
     - Etichetta Pod
     - Nome dello spazio dei nomi
   - **spec.matchingCriteria.value**: (*Facoltativo*) Una stringa o espressione regolare che identifica il valore del filtro dell'hook di esecuzione.

     Esempio YAML:

```
apiVersion: protect.trident.netapp.io/v1
kind: ExecHook
metadata:
  name: example-hook-cr
  namespace: my-app-namespace
  annotations:
    astra.netapp.io/astra-control-hook-source-id:
/account/test/hookSource/id
spec:
  applicationRef: my-app-name
  stage: Pre
  action: Snapshot
  enabled: true
  hookSource: IyEvYmluL2Jhc2gKZWNobyAiZXhhbXBsZSBzY3JpcHQiCg==
  timeout: 10
  arguments:
    - FirstExampleArg
    - SecondExampleArg
  matchingCriteria:
    - type: containerName
      value: mysql
    - type: containerImage
      value: bitnami/mysql
    - type: podName
      value: mysql
    - type: namespaceName
      value: mysql-a
    - type: podLabel
      value: app.kubernetes.io/component=primary
    - type: podLabel
      value: helm.sh/chart=mysql-10.1.0
    - type: podLabel
      value: deployment-type=production
```

3. Dopo aver popolato il file CR con i valori corretti, applicare il CR:

```
kubectl apply -f trident-protect-hook.yaml
```

Utilizzare la CLI

Passi

1. Crea l'hook di esecuzione, sostituendo i valori tra parentesi con le informazioni provenienti dal tuo ambiente. Per esempio:

```
tridentctl-protect create exechook <my_exec_hook_name> --action
<action_type> --app <app_to_use_hook> --stage <pre_or_post_stage>
--source-file <script-file> -n <application_namespace>
```

Eseguire manualmente un hook di esecuzione

È possibile eseguire manualmente un hook di esecuzione per scopi di test o se è necessario rieseguirlo manualmente dopo un errore. Per eseguire manualmente gli hook di esecuzione è necessario disporre delle autorizzazioni di Proprietario, Amministratore o Membro.

L'esecuzione manuale di un hook di esecuzione consiste in due passaggi fondamentali:

1. Crea un backup delle risorse, che raccoglie le risorse e ne crea un backup, determinando dove verrà eseguito l'hook
2. Eseguire l'hook di esecuzione sul backup

Passaggio 1: creare un backup delle risorse

Utilizzare un CR

Passi

1. Crea il file di risorse personalizzate (CR) e assegnagli un nome `trident-protect-resource-backup.yaml` .
2. Configurare i seguenti attributi in modo che corrispondano all'ambiente di protezione Trident e alla configurazione del cluster:
   - **metadata.name**: (*Obbligatorio*) Il nome di questa risorsa personalizzata; scegli un nome univoco e sensato per il tuo ambiente.
   - **spec.applicationRef**: (*Obbligatorio*) Nome Kubernetes dell'applicazione per cui creare il backup delle risorse.
   - **spec.appVaultRef**: (*Obbligatorio*) Nome dell'AppVault in cui sono archiviati i contenuti del backup.
   - **spec.appArchivePath**: il percorso all'interno di AppVault in cui sono archiviati i contenuti del backup. Per trovare questo percorso puoi usare il seguente comando:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o jsonpath='{.status.appArchivePath}'
     ```

     Esempio YAML:

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: ResourceBackup
     metadata:
       name: example-resource-backup
     spec:
       applicationRef: my-app-name
       appVaultRef: my-appvault-name
       appArchivePath: example-resource-backup
     ```

3. Dopo aver popolato il file CR con i valori corretti, applicare il CR:

   ```
   kubectl apply -f trident-protect-resource-backup.yaml
   ```

Utilizzare la CLI

Passi

1. Crea il backup sostituendo i valori tra parentesi con le informazioni del tuo ambiente. Per esempio:

```
tridentctl protect create resourcebackup <my_backup_name> --app
<my_app_name> --appvault <my_appvault_name> -n
<my_app_namespace> --app-archive-path <app_archive_path>
```

2. Visualizza lo stato del backup. È possibile utilizzare questo comando di esempio ripetutamente fino al completamento dell'operazione:

```
tridentctl protect get resourcebackup -n <my_app_namespace>
<my_backup_name>
```

3. Verificare che il backup sia andato a buon fine:

```
kubectl describe resourcebackup <my_backup_name>
```

Passaggio 2: eseguire l’hook di esecuzione

Utilizzare un CR

Passi

1. Crea il file di risorse personalizzate (CR) e assegnagli un nome `trident-protect-hook-run.yaml` .
2. Configurare i seguenti attributi in modo che corrispondano all'ambiente di protezione Trident e alla configurazione del cluster:
   - **metadata.name**: (*Obbligatorio*) Il nome di questa risorsa personalizzata; scegli un nome univoco e sensato per il tuo ambiente.
   - **spec.applicationRef**: (*Obbligatorio*) Assicurati che questo valore corrisponda al nome dell'applicazione dal CR ResourceBackup creato nel passaggio 1.
   - **spec.appVaultRef**: (*Obbligatorio*) Assicurati che questo valore corrisponda all'appVaultRef del CR ResourceBackup creato nel passaggio 1.
   - **spec.appArchivePath**: assicurati che questo valore corrisponda all'appArchivePath del CR ResourceBackup creato nel passaggio 1.

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

   - **spec.action**: (*Obbligatorio*) Una stringa che indica quale azione intraprenderà l'hook di esecuzione, supponendo che tutti i filtri dell'hook di esecuzione specificati corrispondano. Valori possibili:
     - Istantanea
     - Backup
     - Ripristinare
     - Failover
   - **spec.stage**: (*Obbligatorio*) Una stringa che indica in quale fase dell'azione deve essere eseguito l'hook di esecuzione. Questa serie di hook non prevede hook in nessun'altra fase. Valori possibili:
     - Pre
     - Inviare

     Esempio YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: ExecHooksRun
metadata:
  name: example-hook-run
spec:
  applicationRef: my-app-name
  appVaultRef: my-appvault-name
  appArchivePath: example-resource-backup
  stage: Post
  action: Failover
```

3. Dopo aver popolato il file CR con i valori corretti, applicare il CR:

```
kubectl apply -f trident-protect-hook-run.yaml
```

Utilizzare la CLI

Passi

1. Creare la richiesta di esecuzione manuale dell'hook:

```
tridentctl protect create exechooksrun <my_exec_hook_run_name>
-n <my_app_namespace> --action snapshot --stage <pre_or_post>
--app <my_app_name> --appvault <my_appvault_name> --path
<my_backup_name>
```

2. Controllare lo stato dell'esecuzione del gancio. È possibile eseguire questo comando più volte fino al completamento dell'operazione:

```
tridentctl protect get exechooksrun -n <my_app_namespace>
<my_exec_hook_run_name>
```

3. Descrivi l'oggetto exechooksrun per vedere i dettagli finali e lo stato:

```
kubectl -n <my_app_namespace> describe exechooksrun
<my_exec_hook_run_name>
```

Disinstallare Trident Protect

Potrebbe essere necessario rimuovere i componenti di protezione Trident se si esegue l'aggiornamento da una versione di prova a una versione completa del prodotto.

Per rimuovere Trident Protect, procedere come segue.

Passi

1. Rimuovere i file CR di protezione Trident :

   Questo passaggio non è necessario per la versione 25.06 e successive.

   ```
   helm uninstall -n trident-protect trident-protect-crds
   ```

2. Rimuovere la protezione Trident :

   ```
   helm uninstall -n trident-protect trident-protect
   ```

3. Rimuovere lo spazio dei nomi Trident protect:

   ```
   kubectl delete ns trident-protect
   ```

Trident e Trident proteggono i blog

Puoi trovare alcuni ottimi blog NetApp Trident e Trident Protect qui:

Blog Trident

- 9 maggio 2025:"Configurazione automatica del backend Trident per FSx per ONTAP con il componente aggiuntivo Amazon EKS"
- 19 agosto 2025:"Miglioramento della coerenza dei dati: snapshot del gruppo di volumi nella virtualizzazione OpenShift con Trident"
- 15 aprile 2025:"NetApp Trident con Google Cloud NetApp Volumes per il protocollo SMB"
- 14 aprile 2025:"Utilizzo del protocollo Fibre Channel con Trident 25.02 per l'archiviazione persistente su Kubernetes"
- 14 aprile 2025:"Sfruttare la potenza dei sistemi NetApp ASA r2 per l'archiviazione a blocchi di Kubernetes"
- 31 marzo 2025:"Semplificazione dell'installazione Trident su Red Hat OpenShift con il nuovo operatore certificato"
- 27 marzo 2025:"Provisioning Trident per PMI con Google Cloud NetApp Volumes"
- 5 marzo 2025:"Sblocca l'integrazione perfetta dello storage iSCSI: una guida a FSxN su cluster ROSA per AWS"
- 27 febbraio 2025:"Distribuzione dell'identità cloud con Trident, GKE e Google Cloud NetApp Volumes"
- 12 dicembre 2024:"Introduzione del supporto Fibre Channel in Trident"
- 26 novembre 2024:"Trident 25.01: Miglioramento dell'esperienza di archiviazione Kubernetes con nuove funzionalità e miglioramenti"
- 11 novembre 2024:"NetApp Trident con Google Cloud NetApp Volumes"
- 29 ottobre 2024:"Amazon FSx for NetApp ONTAP con Red Hat OpenShift Service su AWS (ROSA) utilizzando Trident"
- 29 ottobre 2024:"Migrazione live di VM con OpenShift Virtualization su ROSA e Amazon FSx for NetApp ONTAP"
- 8 luglio 2024:"Utilizzo di NVMe/TCP per consumare lo storage ONTAP per le tue moderne app containerizzate su Amazon EKS"
- 1 luglio 2024:"Archiviazione Kubernetes senza interruzioni con Google Cloud NetApp Volumes Flex e Astra Trident"
- 11 giugno 2024:"ONTAP come archivio backend per il registro immagini integrato in OpenShift"

Trident protegge i blog

- 16 maggio 2025:"Automazione del failover del registro per il ripristino di emergenza con i ganci post-ripristino Trident Protect"
- 16 maggio 2025:"Ripristino di emergenza della virtualizzazione OpenShift con NetApp Trident Protect"
- 13 maggio 2025:"Migrazione della classe di archiviazione con Trident Protect Backup Restore"
- 9 maggio 2025:"Ridimensiona le applicazioni Kubernetes con i ganci di protezione post-ripristino Trident"
- 3 aprile 2025:"Trident Protect Power Up: replica di Kubernetes per protezione e ripristino di emergenza"

- 13 marzo 2025:"Operazioni di backup e ripristino coerenti con gli arresti anomali per le VM di virtualizzazione OpenShift"
- 11 marzo 2025:"Estensione dei modelli GitOps alla protezione dei dati delle applicazioni con NetApp Trident"
- 03 marzo 2025:"Trident 25.02: Migliorare l'esperienza Red Hat OpenShift con nuove entusiasmanti funzionalità"
- 15 gennaio 2025:"Presentazione del controllo degli accessi basato sui ruoli Trident Protect"
- 11 novembre 2024: "Presentazione di tridentctl protect: la potente CLI per Trident Protect"
- 11 novembre 2024:"Gestione dei dati basata su Kubernetes: la nuova era con Trident Protect"

Conoscenza e supporto

Domande frequenti

Trova le risposte alle domande più frequenti sull'installazione, la configurazione, l'aggiornamento e la risoluzione dei problemi Trident.

Domande generali

Con quale frequenza viene rilasciato Trident ?

A partire dalla versione 24.02, Trident verrà rilasciato ogni quattro mesi: febbraio, giugno e ottobre.

Trident supporta tutte le funzionalità rilasciate in una particolare versione di Kubernetes?

Solitamente Trident non supporta le funzionalità alpha in Kubernetes. Trident potrebbe supportare le funzionalità beta nelle due versioni Trident che seguiranno la versione beta di Kubernetes.

Il funzionamento Trident dipende in qualche modo da altri prodotti NetApp ?

Trident non ha alcuna dipendenza da altri prodotti software NetApp e funziona come applicazione autonoma. Tuttavia, dovresti disporre di un dispositivo di archiviazione back-end NetApp .

Come posso ottenere i dettagli completi della configurazione Trident ?

Utilizzare il `tridentctl get` comando per ottenere maggiori informazioni sulla configurazione del tuo Trident .

Posso ottenere metriche su come Trident fornisce lo storage?

Sì. Endpoint Prometheus che possono essere utilizzati per raccogliere informazioni sul funzionamento Trident , come il numero di backend gestiti, il numero di volumi forniti, i byte consumati e così via. Puoi anche usare"Cloud Insights" per il monitoraggio e l'analisi.

L'esperienza utente cambia quando si utilizza Trident come CSI Provisioner?

No. Non ci sono cambiamenti per quanto riguarda l'esperienza utente e le funzionalità. Il nome del fornitore utilizzato è `csi.trident.netapp.io` . Questo metodo di installazione Trident è consigliato se si desidera utilizzare tutte le nuove funzionalità fornite dalle versioni attuali e future.

Installa e usa Trident su un cluster Kubernetes

Trident supporta l'installazione offline da un registro privato?

Sì, Trident può essere installato offline. Fare riferimento a"Scopri di più sull'installazione Trident" .

Posso installare Trident da remoto?

Sì. Trident 18.10 e versioni successive supportano la capacità di installazione remota da qualsiasi macchina che abbia `kubectl` accesso al cluster. Dopo `kubectl` l'accesso è verificato (ad esempio, avviare un `kubectl get nodes` comando dalla macchina remota per verificare), seguire le istruzioni di installazione.

Posso configurare l'alta disponibilità con Trident?

Trident è installato come distribuzione Kubernetes (ReplicaSet) con un'istanza e quindi ha HA integrato. Non dovresti aumentare il numero di repliche nella distribuzione. Se il nodo su cui è installato Trident viene perso o il pod non è accessibile, Kubernetes ridistribuisce automaticamente il pod su un nodo funzionante nel cluster. Trident è un aereo di controllo esclusivo, quindi i pod attualmente montati non saranno interessati dal Trident riposizionamento.

Trident ha bisogno di accedere allo spazio dei nomi kube-system?

Trident legge dal server API di Kubernetes per determinare quando le applicazioni richiedono nuovi PVC, quindi ha bisogno di accedere a kube-system.

Quali sono i ruoli e i privilegi utilizzati da Trident?

Il programma di installazione Trident crea un Kubernetes ClusterRole, che ha accesso specifico alle risorse PersistentVolume, PersistentVolumeClaim, StorageClass e Secret del cluster Kubernetes. Fare riferimento a"Personalizza l'installazione di tridentctl" .

Posso generare localmente gli esatti file manifest utilizzati Trident per l'installazione?

Se necessario, è possibile generare e modificare localmente i file manifest esatti utilizzati Trident per l'installazione. Fare riferimento a"Personalizza l'installazione di tridentctl" .

Posso condividere lo stesso SVM backend ONTAP per due istanze Trident separate per due cluster Kubernetes separati?

Sebbene non sia consigliato, è possibile utilizzare lo stesso backend SVM per due istanze Trident . Specificare un nome di volume univoco per ogni istanza durante l'installazione e/o specificare un nome di volume univoco `StoragePrefix` parametro nel `setup/backend.json` file. Ciò serve a garantire che non venga utilizzato lo stesso FlexVol volume per entrambe le istanze.

È possibile installare Trident su ContainerLinux (in precedenza CoreOS)?

Trident è semplicemente un pod Kubernetes e può essere installato ovunque sia in esecuzione Kubernetes.

Posso utilizzare Trident con NetApp Cloud Volumes ONTAP?

Sì, Trident è supportato su AWS, Google Cloud e Azure.

Trident funziona con Cloud Volumes Services?

Sì, Trident supporta il servizio Azure NetApp Files in Azure e il Cloud Volumes Service in GCP.

Risoluzione dei problemi e supporto

NetApp supporta Trident?

Sebbene Trident sia open source e fornito gratuitamente, NetApp lo supporta pienamente, a condizione che il backend NetApp sia supportato.

Come posso inviare un caso di supporto?

Per inviare un caso di supporto, procedere in uno dei seguenti modi:

1. Contatta il tuo Account Manager di supporto e ricevi aiuto per aprire un ticket.
2. Invia un caso di supporto contattando "Supporto NetApp" .

Come posso generare un pacchetto di log di supporto?

Puoi creare un pacchetto di supporto eseguendo `tridentctl logs -a` . Oltre ai log acquisiti nel bundle, acquisisci il log kubelet per diagnosticare i problemi di montaggio sul lato Kubernetes. Le istruzioni per ottenere il log kubelet variano in base al modo in cui Kubernetes è installato.

Cosa devo fare se devo inoltrare una richiesta per una nuova funzionalità?

Crea un problema su "Trident Github" e menzionare **RFE** nell'oggetto e nella descrizione del problema.

Dove posso segnalare un difetto?

Crea un problema su "Trident Github" . Assicurati di includere tutte le informazioni e i registri necessari relativi al problema.

Cosa succede se ho una domanda veloce su Trident e ho bisogno di chiarimenti? Esiste una community o un forum?

Se hai domande, problemi o richieste, contattaci tramite il nostro Trident"Canale Discord" o GitHub.

La password del mio sistema di archiviazione è cambiata e Trident non funziona più. Come posso fare per ripristinarla?

Aggiorna la password del backend con `tridentctl update backend myBackend -f </path/to_new_backend.json> -n trident` . Sostituire `myBackend` nell'esempio con il nome del tuo backend e `` `/path/to_new_backend.json `` con il percorso corretto `backend.json` file.

Trident non riesce a trovare il mio nodo Kubernetes. Come posso risolvere questo problema?

Esistono due possibili scenari per cui Trident non riesce a trovare un nodo Kubernetes. Potrebbe essere dovuto a un problema di rete all'interno di Kubernetes o a un problema DNS. Il daemonset del nodo Trident in esecuzione su ciascun nodo Kubernetes deve essere in grado di comunicare con il controller Trident per registrare il nodo con Trident. Se si sono verificate modifiche alla rete dopo l'installazione Trident , questo problema si verifica solo con i nuovi nodi Kubernetes aggiunti al cluster.

Se il pod Trident viene distrutto, perderò i dati?

I dati non andranno persi se il pod Trident verrà distrutto. I metadati Trident sono memorizzati negli oggetti CRD. Tutti i PV forniti da Trident funzioneranno normalmente.

Trident potenziato

Posso effettuare l'aggiornamento direttamente da una versione precedente a una versione più recente (saltando alcune versioni)?

NetApp supporta l'aggiornamento Trident da una versione principale alla successiva versione principale immediata. È possibile effettuare l'aggiornamento dalla versione 18.xx alla 19.xx, dalla 19.xx alla 20.xx e così via. Dovresti testare l'aggiornamento in laboratorio prima di distribuirlo in produzione.

È possibile effettuare il downgrade Trident a una versione precedente?

Se hai bisogno di una correzione per bug osservati dopo un aggiornamento, problemi di dipendenza o un aggiornamento non riuscito o incompleto, dovresti"disinstallare Trident" e reinstallare la versione precedente seguendo le istruzioni specifiche per quella versione. Questo è l'unico metodo consigliato per eseguire il downgrade a una versione precedente.

Gestire backend e volumi

Devo definire sia Management che DataLIF in un file di definizione backend ONTAP ?

La gestione LIF è obbligatoria. DataLIF varia:

- ONTAP SAN: non specificare per iSCSI. Usi Trident"Mappa LUN selettiva ONTAP" per scoprire gli iSCI LIF necessari per stabilire una sessione multi-percorso. Viene generato un avviso se `dataLIF` è definito esplicitamente. Fare riferimento a "Opzioni ed esempi di configurazione SAN ONTAP" per i dettagli.
- ONTAP NAS: NetApp consiglia di specificare `dataLIF` . Se non specificato, Trident recupera i dataLIF dall'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 dataLIF. Fare riferimento a"Opzioni ed esempi di configurazione del NAS ONTAP" per i dettagli

Trident può configurare CHAP per i backend ONTAP ?

Sì. Trident supporta CHAP bidirezionale per i backend ONTAP . Ciò richiede l'impostazione `useCHAP=true` nella configurazione del backend.

Come posso gestire le politiche di esportazione con Trident?

Trident può creare e gestire dinamicamente le policy di esportazione a partire dalla versione 20.04. Ciò consente all'amministratore dell'archiviazione di fornire uno o più blocchi CIDR nella configurazione backend e di far sì che Trident aggiunga gli IP dei nodi che rientrano in questi intervalli a una policy di esportazione creata. In questo modo, Trident gestisce automaticamente l'aggiunta e l'eliminazione di regole per i nodi con IP all'interno dei CIDR specificati.

È possibile utilizzare gli indirizzi IPv6 per Management e DataLIF?

Trident supporta la definizione di indirizzi IPv6 per:

- `managementLIF`E `dataLIF` per i backend NAS ONTAP .
- `managementLIF`per i backend ONTAP SAN. Non puoi specificare `dataLIF` su un backend ONTAP SAN.

Il Trident deve essere installato utilizzando la bandiera `--use-ipv6` (per `tridentctl` installazione), `IPv6` (per l'operatore Trident), o `tridentTPv6` (per l'installazione di Helm) affinché funzioni su IPv6.

È possibile aggiornare il Management LIF sul backend?

Sì, è possibile aggiornare il backend Management LIF utilizzando `tridentctl update backend` comando.

È possibile aggiornare DataLIF sul backend?

È possibile aggiornare DataLIF su `ontap-nas` E `ontap-nas-economy` soltanto.

Posso creare più backend in Trident per Kubernetes?

Trident può supportare molti backend contemporaneamente, sia con lo stesso driver che con driver diversi.

In che modo Trident memorizza le credenziali backend?

Trident memorizza le credenziali del backend come segreti di Kubernetes.

In che modo Trident seleziona un backend specifico?

Se gli attributi backend non possono essere utilizzati per selezionare automaticamente i pool corretti per una classe, `storagePools` E `additionalStoragePools` I parametri vengono utilizzati per selezionare un set specifico di pool.

Come posso assicurarmi che Trident non effettui il provisioning da un backend specifico?

IL `excludeStoragePools` Il parametro viene utilizzato per filtrare l'insieme di pool che Trident utilizza per il provisioning e rimuoverà tutti i pool corrispondenti.

Se sono presenti più backend dello stesso tipo, in che modo Trident seleziona quale backend utilizzare?

Se sono presenti più backend configurati dello stesso tipo, Trident seleziona il backend appropriato in base ai parametri presenti in `StorageClass` E `PersistentVolumeClaim` . Ad esempio, se sono presenti più backend del driver ontap-nas, Trident tenta di abbinare i parametri in `StorageClass` E `PersistentVolumeClaim` combinati e abbinano un backend in grado di soddisfare i requisiti elencati in `StorageClass` E `PersistentVolumeClaim` . Se ci sono più backend che corrispondono alla richiesta, Trident ne seleziona uno a caso.

Trident supporta CHAP bidirezionale con Element/ SolidFire?

Sì.

In che modo Trident distribuisce Qtrees su un volume ONTAP ? Quanti Qtree possono essere distribuiti su un singolo volume?

IL `ontap-nas-economy` il driver crea fino a 200 Qtree nello stesso FlexVol volume (configurabile tra 50 e 300), 100.000 Qtree per nodo del cluster e 2,4 milioni per cluster. Quando si entra in un nuovo `PersistentVolumeClaim` gestito dal driver economy, il driver verifica se esiste già un FlexVol volume in grado di gestire il nuovo Qtree. Se non esiste un FlexVol volume in grado di servire Qtree, viene creato un nuovo FlexVol volume .

Come posso impostare le autorizzazioni Unix per i volumi forniti su ONTAP NAS?

È possibile impostare le autorizzazioni Unix sul volume fornito da Trident impostando un parametro nel file di definizione del backend.

Come posso configurare un set esplicito di opzioni di montaggio ONTAP NFS durante il provisioning di un volume?

Per impostazione predefinita, Trident non imposta le opzioni di montaggio su alcun valore con Kubernetes. Per specificare le opzioni di montaggio nella classe di archiviazione Kubernetes, seguire l'esempio fornito"Qui" .

Come faccio a impostare i volumi forniti su una policy di esportazione specifica?

Per consentire agli host appropriati di accedere a un volume, utilizzare `exportPolicy` parametro configurato nel file di definizione del backend.

Come posso impostare la crittografia del volume tramite Trident con ONTAP?

È possibile impostare la crittografia sul volume fornito da Trident utilizzando il parametro di crittografia nel file di definizione del backend. Per maggiori informazioni, fare riferimento a:"Come funziona Trident con NVE e NAE"

Qual è il modo migliore per implementare QoS per ONTAP tramite Trident?

Utilizzo `StorageClasses` per implementare QoS per ONTAP.

Come posso specificare il provisioning sottile o spesso tramite Trident?

I driver ONTAP supportano sia il provisioning sottile che quello spesso. Per impostazione predefinita, i driver ONTAP utilizzano il thin provisioning. Se si desidera un provisioning spesso, è necessario configurare il file di definizione del backend o il `StorageClass` . Se entrambi sono configurati, `StorageClass` ha la precedenza. Configurare quanto segue per ONTAP:

1. SU `StorageClass` , impostare il `provisioningType` attributo come spesso.
2. Nel file di definizione del backend, abilitare i volumi spessi impostando `backend spaceReserve parameter` come volume.

Come posso assicurarmi che i volumi utilizzati non vengano eliminati anche se elimino accidentalmente il PVC?

La protezione PVC è abilitata automaticamente su Kubernetes a partire dalla versione 1.10.

Posso coltivare PVC NFS creati da Trident?

Sì. È possibile espandere un PVC creato da Trident. Si noti che l'aumento automatico del volume è una funzionalità ONTAP non applicabile a Trident.

Posso importare un volume mentre è in modalità SnapMirror Data Protection (DP) o offline?

L'importazione del volume non riesce se il volume esterno è in modalità DP o è offline. Viene visualizzato il seguente messaggio di errore:

```
Error: could not import volume: volume import failed to get size of
volume: volume <name> was not found (400 Bad Request) command terminated
with exit code 1.
Make sure to remove the DP mode or put the volume online before importing
the volume.
```

Come viene tradotta la quota di risorse in un cluster NetApp ?

La quota delle risorse di archiviazione di Kubernetes dovrebbe funzionare finché lo storage NetApp ha capacità. Quando lo storage NetApp non riesce a rispettare le impostazioni delle quote di Kubernetes a causa

della mancanza di capacità, Trident tenta di effettuare il provisioning ma genera un errore.

Posso creare snapshot del volume utilizzando Trident?

Sì. Trident supporta la creazione di snapshot di volumi su richiesta e di volumi persistenti da snapshot. Per creare PV da snapshot, assicurarsi che `VolumeSnapshotDataSource` feature gate è stato abilitato.

Quali sono i driver che supportano gli snapshot del volume Trident ?

Da oggi, il supporto snapshot on-demand è disponibile per il nostro `ontap-nas` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy` , `solidfire-san` , `gcp-cvs` , E `azure-netapp-files` driver backend.

Come posso eseguire un backup snapshot di un volume fornito da Trident con ONTAP?

Questo è disponibile su `ontap-nas` , `ontap-san` , E `ontap-nas-flexgroup` conducenti. Puoi anche specificare un `snapshotPolicy` per il `ontap-san-economy` driver a livello FlexVol .

Questo è disponibile anche su `ontap-nas-economy` driver ma sulla granularità a livello di FlexVol volume e non su quella a livello di qtree. Per abilitare la possibilità di creare snapshot dei volumi forniti da Trident, impostare l'opzione del parametro backend `snapshotPolicy` alla policy di snapshot desiderata come definita nel backend ONTAP . Trident non conosce gli snapshot acquisiti dal controller di archiviazione.

Posso impostare una percentuale di riserva snapshot per un volume fornito tramite Trident?

Sì, è possibile riservare una percentuale specifica di spazio su disco per l'archiviazione delle copie snapshot tramite Trident impostando `snapshotReserve` attributo nel file di definizione del backend. Se hai configurato `snapshotPolicy` E `snapshotReserve` nel file di definizione del backend, la percentuale di riserva degli snapshot è impostata in base a `snapshotReserve` percentuale menzionata nel file backend. Se il `snapshotReserve` Se non viene menzionato il numero percentuale, ONTAP per impostazione predefinita considera la percentuale di riserva degli snapshot pari a 5. Se il `snapshotPolicy` Se l'opzione è impostata su nessuno, la percentuale di riserva degli snapshot è impostata su 0.

Posso accedere direttamente alla directory degli snapshot del volume e copiare i file?

Sì, puoi accedere alla directory snapshot sul volume fornito da Trident impostando `snapshotDir` parametro nel file di definizione del backend.

Posso configurare SnapMirror per i volumi tramite Trident?

Attualmente, SnapMirror deve essere impostato esternamente tramite ONTAP CLI o OnCommand System Manager.

Come posso ripristinare i volumi persistenti in uno snapshot ONTAP specifico?

Per ripristinare un volume in uno snapshot ONTAP , procedere come segue:

1. Disattiva il pod dell'applicazione che utilizza il volume persistente.
2. Ripristinare lo snapshot richiesto tramite ONTAP CLI o OnCommand System Manager.
3. Riavviare il pod dell'applicazione.

Trident può effettuare il provisioning dei volumi su SVM in cui è configurato un Load-Sharing Mirror?

È possibile creare mirror di condivisione del carico per i volumi root delle SVM che gestiscono i dati tramite NFS. ONTAP aggiorna automaticamente i mirror di condivisione del carico per i volumi creati da Trident. Ciò potrebbe comportare ritardi nell'aumento dei volumi. Quando vengono creati più volumi utilizzando Trident, il provisioning di un volume dipende dall'aggiornamento del mirror di condivisione del carico ONTAP .

Come posso separare l'utilizzo della classe di archiviazione per ciascun cliente/tenant?

Kubernetes non consente classi di archiviazione negli spazi dei nomi. Tuttavia, è possibile utilizzare Kubernetes per limitare l'utilizzo di una specifica classe di archiviazione per ogni namespace tramite le quote delle risorse di archiviazione, che sono per ogni namespace. Per negare a uno specifico namespace l'accesso a uno specifico archivio, impostare la quota delle risorse su 0 per quella classe di archiviazione.

Risoluzione dei problemi

Utilizzare i suggerimenti forniti qui per risolvere i problemi che potrebbero verificarsi durante l'installazione e l'utilizzo Trident.

Per assistenza con Trident, crea un pacchetto di supporto utilizzando `tridentctl logs -a -n trident` e inviarlo al supporto NetApp .

Risoluzione dei problemi generali

- Se il pod Trident non riesce a sollevarsi correttamente (ad esempio, quando il pod Trident è bloccato nel `ContainerCreating` fase con meno di due contenitori pronti), in esecuzione `kubectl -n trident describe deployment trident` E `kubectl -n trident describe pod trident-**` può fornire ulteriori approfondimenti. Ottenere i log kubelet (ad esempio, tramite `journalctl -xeu kubelet`) può anche essere utile.
- Se non ci sono abbastanza informazioni nei registri Trident , puoi provare ad abilitare la modalità di debug per Trident passando il `-d` flag al parametro di installazione in base all'opzione di installazione.

  Quindi confermare che il debug è impostato utilizzando `./tridentctl logs -n trident` e cercando `level=debug msg` nel registro.

  Installato con operatore

  ```
  kubectl patch torc trident -n <namespace> --type=merge -p
  '{"spec":{"debug":true}}'
  ```

  Questa operazione riavvierà tutti i pod Trident , operazione che potrebbe richiedere diversi secondi. È possibile verificarlo osservando la colonna 'AGE' nell'output di `kubectl get pod -n trident` .

  Per l'uso Trident 20.07 e 20.10 `tprov` al posto di `torc` .

  Installato con Helm

  ```
  helm upgrade <name> trident-operator-21.07.1-custom.tgz --set
  tridentDebug=true`
  ```

Installato con tridentctl

```
./tridentctl uninstall -n trident
./tridentctl install -d -n trident
```

- È anche possibile ottenere i log di debug per ciascun backend includendo `debugTraceFlags` nella definizione del backend. Ad esempio, includere `debugTraceFlags: {"api":true, "method":true,}` per ottenere chiamate API e attraversamenti di metodi nei log Trident . I backend esistenti possono avere `debugTraceFlags` configurato con un `tridentctl backend update` .
- Quando si utilizza Red Hat Enterprise Linux CoreOS (RHCOS), assicurarsi che `iscsid` è abilitato sui nodi worker e avviato per impostazione predefinita. Ciò può essere fatto utilizzando OpenShift MachineConfigs o modificando i modelli di accensione.
- Un problema comune che potresti riscontrare quando usi Trident con "Azure NetApp Files" si verifica quando i segreti del tenant e del client provengono da una registrazione dell'app con autorizzazioni insufficienti. Per un elenco completo dei requisiti Trident , fare riferimento a"Azure NetApp Files" configurazione.
- Se si verificano problemi con il montaggio di un fotovoltaico su un contenitore, assicurarsi che `rpcbind` è installato e funzionante. Utilizzare il gestore di pacchetti richiesto per il sistema operativo host e verificare se `rpcbind` è in esecuzione. Puoi controllare lo stato del `rpcbind` servizio eseguendo un `systemctl status rpcbind` o il suo equivalente.
- Se un backend Trident segnala che è nel `failed` stato nonostante abbia funzionato in precedenza, è probabile che sia causato dalla modifica delle credenziali SVM/admin associate al backend. Aggiornamento delle informazioni di backend utilizzando `tridentctl update backend` oppure facendo rimbalzare il pod Trident il problema verrà risolto.
- Se riscontri problemi di autorizzazione durante l'installazione Trident con Docker come runtime del contenitore, prova a installare Trident con `--in cluster=false` bandiera. Ciò non utilizzerà un pod di installazione ed eviterà problemi di autorizzazione riscontrati a causa di `trident-installer` utente.
- Utilizzare il `uninstall parameter <Uninstalling Trident>` per la pulizia dopo una corsa fallita. Per impostazione predefinita, lo script non rimuove i CRD creati da Trident, rendendo sicura la disinstallazione e la reinstallazione anche durante una distribuzione in esecuzione.
- Se vuoi effettuare il downgrade a una versione precedente di Trident, esegui prima il `tridentctl uninstall` comando per rimuovere Trident. Scarica il file desiderato "Versione Trident" e installare utilizzando il `tridentctl install` comando.
- Dopo un'installazione riuscita, se un PVC rimane bloccato nel `Pending` fase, in esecuzione `kubectl describe pvc` può fornire ulteriori informazioni sul motivo per cui Trident non è riuscita a fornire un PV per questo PVC.

Dispiegamento Trident non riuscito tramite l'operatore

Se si sta distribuendo Trident utilizzando l'operatore, lo stato di `TridentOrchestrator` cambiamenti da `Installing` A `Installed` . Se osservi il `Failed` stato e l'operatore non è in grado di ripristinarsi autonomamente, è necessario controllare i registri dell'operatore eseguendo il seguente comando:

```
tridentctl logs -l trident-operator
```

L'analisi dei log del container Trident-Operator può indicare dove risiede il problema. Ad esempio, uno di questi

problemi potrebbe essere l'impossibilità di estrarre le immagini dei container richieste dai registri upstream in un ambiente airgapped.

Per capire perché l'installazione di Trident non ha avuto successo, dovresti dare un'occhiata a `TridentOrchestrator` stato.

```
kubectl describe torc trident-2
Name:         trident-2
Namespace:
Labels:       <none>
Annotations:  <none>
API Version:  trident.netapp.io/v1
Kind:         TridentOrchestrator
...
Status:
  Current Installation Params:
    IPv6:
    Autosupport Hostname:
    Autosupport Image:
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:
    Image Pull Secrets:         <nil>
    Image Registry:
    k8sTimeout:
    Kubelet Dir:
    Log Format:
    Silence Autosupport:
    Trident Image:
  Message:                      Trident is bound to another CR 'trident'
  Namespace:                    trident-2
  Status:                       Error
  Version:
Events:
  Type     Reason  Age                From                          Message
  ----     ------  ----               ----                          -------
  Warning  Error   16s (x2 over 16s)  trident-operator.netapp.io  Trident
is bound to another CR 'trident'
```

Questo errore indica che esiste già un `TridentOrchestrator` che è stato utilizzato per installare Trident. Poiché ogni cluster Kubernetes può avere solo un'istanza di Trident, l'operatore garantisce che in qualsiasi momento esista solo un'istanza attiva `TridentOrchestrator` che può creare.

Inoltre, osservare lo stato dei baccelli Trident può spesso indicare se qualcosa non va.

```
kubectl get pods -n trident

NAME                                READY   STATUS             RESTARTS
AGE
trident-csi-4p5kq                   1/2     ImagePullBackOff   0
5m18s
trident-csi-6f45bfd8b6-vfrkw        4/5     ImagePullBackOff   0
5m19s
trident-csi-9q5xc                   1/2     ImagePullBackOff   0
5m18s
trident-csi-9v95z                   1/2     ImagePullBackOff   0
5m18s
trident-operator-766f7b8658-ldzsv   1/1     Running            0
8m17s
```

È possibile vedere chiaramente che i pod non sono in grado di inizializzarsi completamente perché una o più immagini del contenitore non sono state recuperate.

Per risolvere il problema, dovresti modificare il `TridentOrchestrator` CR. In alternativa, puoi eliminare `TridentOrchestrator` e crearne uno nuovo con la definizione modificata e accurata.

Dispiegamento Trident non riuscito utilizzando `tridentctl`

Per capire cosa è andato storto, puoi eseguire nuovamente il programma di installazione utilizzando`-d` argomento, che attiverà la modalità debug e ti aiuterà a capire qual è il problema:

```
./tridentctl install -n trident -d
```

Dopo aver risolto il problema, è possibile pulire l'installazione come segue, quindi eseguire il `tridentctl install` comando di nuovo:

```
./tridentctl uninstall -n trident
INFO Deleted Trident deployment.
INFO Deleted cluster role binding.
INFO Deleted cluster role.
INFO Deleted service account.
INFO Removed Trident user from security context constraint.
INFO Trident uninstallation succeeded.
```

Rimuovere completamente Trident e CRD

È possibile rimuovere completamente Trident e tutti i CRD creati e le risorse personalizzate associate.

Questa operazione non può essere annullata. Non farlo a meno che tu non voglia un'installazione completamente nuova di Trident. Per disinstallare Trident senza rimuovere i CRD, fare riferimento a"Disinstallare Trident" .

Operatore Trident

Per disinstallare Trident e rimuovere completamente i CRD utilizzando l'operatore Trident :

```
kubectl patch torc <trident-orchestrator-name> --type=merge -p
'{"spec":{"wipeout":["crds"],"uninstall":true}}'
```

Timone

Per disinstallare Trident e rimuovere completamente i CRD utilizzando Helm:

```
kubectl patch torc trident --type=merge -p
'{"spec":{"wipeout":["crds"],"uninstall":true}}'
```

<code>tridentctl</code>

Per rimuovere completamente i CRD dopo aver disinstallato Trident utilizzando `tridentctl`

```
tridentctl obliviate crd
```

Errore di destaging del nodo NVMe con namespace di blocchi raw RWX o Kubernetes 1.26

Se si esegue Kubernetes 1.26, l'unstaging del nodo potrebbe non riuscire quando si utilizza NVMe/TCP con namespace di blocchi raw RWX. Gli scenari seguenti forniscono una soluzione alternativa al problema. In alternativa, puoi aggiornare Kubernetes alla versione 1.27.

Eliminato lo spazio dei nomi e il pod

Si consideri uno scenario in cui si dispone di uno spazio dei nomi gestito Trident (volume persistente NVMe) collegato a un pod. Se si elimina lo spazio dei nomi direttamente dal backend ONTAP , il processo di unstaging si blocca dopo il tentativo di eliminare il pod. Questo scenario non ha alcun impatto sul cluster Kubernetes o su altre funzionalità.

Soluzione alternativa

Smontare il volume persistente (corrispondente a quello spazio dei nomi) dal rispettivo nodo ed eliminarlo.

DataLIF bloccati

```
 If you block (or bring down) all the dataLIFs of the NVMe Trident
backend, the unstaging process gets stuck when you attempt to delete the
pod. In this scenario, you cannot run any NVMe CLI commands on the
Kubernetes node.
.Soluzione alternativa
Avviare dataLIFS per ripristinare la piena funzionalità.
```

Mappatura dello spazio dei nomi eliminata

```
 If you remove the `hostNQN` of the worker node from the corresponding
subsystem, the unstaging process gets stuck when you attempt to delete the
pod. In this scenario, you cannot run any NVMe CLI commands on the
Kubernetes node.
.Soluzione alternativa
Aggiungi il `hostNQN` torniamo al sottosistema.
```

I client NFSv4.2 segnalano "argomento non valido" dopo l'aggiornamento ONTAP quando si prevede che "v4.2-xattrs" sia abilitato

Dopo l'aggiornamento ONTAP, i client NFSv4.2 potrebbero segnalare errori di tipo "argomento non valido" quando tentano di montare esportazioni NFSv4.2. Questo problema si verifica quando il `v4.2-xattrs` l'opzione non è abilitata sulla SVM. .Soluzione alternativa Abilitare il `v4.2-xattrs` opzione sull'SVM o eseguire l'aggiornamento a ONTAP 9.12.1 o versione successiva, dove questa opzione è abilitata per impostazione predefinita.

Supporto

NetApp fornisce supporto per Trident in vari modi. Sono disponibili ampie opzioni di auto-assistenza gratuite 24 ore su 24, 7 giorni su 7, come articoli della knowledge base (KB) e un canale Discord.

Supporto Trident

Trident offre tre livelli di supporto in base alla versione. Fare riferimento a"Supporto della versione software NetApp per le definizioni" .

Supporto completo
: Trident fornisce supporto completo per dodici mesi dalla data di rilascio.

Supporto limitato
: Trident fornisce supporto limitato per i mesi dal 13° al 24° dalla data di rilascio.

Autosufficienza
: La documentazione Trident sarà disponibile per i mesi dal 25° al 36° dalla data di uscita.

Versione	Supporto completo	Supporto limitato	Autosufficienza

"25,06"	Giugno 2026	Giugno 2027	Giugno 2028
"25,02"	Febbraio 2026	Febbraio 2027	Febbraio 2028
"24,10"	Ottobre 2025	Ottobre 2026	Ottobre 2027
"24,06"	Giugno 2025	Giugno 2026	Giugno 2027
"24,02"	Febbraio 2025	Febbraio 2026	Febbraio 2027
"23,10"	—	Ottobre 2025	Ottobre 2026
"23,07"	—	Luglio 2025	Luglio 2026
"23,04"	—	Aprile 2025	Aprile 2026
"23,01"	—	—	Gennaio 2026
"22,10"	—	—	Ottobre 2025

Autosufficienza

Per un elenco completo degli articoli sulla risoluzione dei problemi, fare riferimento a "Knowledgebase NetApp (richiesto accesso)" .

Supporto della comunità

Esiste una vivace comunità pubblica di utenti di container (inclusi gli sviluppatori Trident) sul nostro"Canale Discord" . Questo è il posto ideale per porre domande generali sul progetto e discutere di argomenti correlati con colleghi che la pensano come te.

Supporto tecnico NetApp

Per assistenza con Trident, crea un pacchetto di supporto utilizzando `tridentctl logs -a -n trident` e inviarlo a `NetApp Support <Getting Help>` .

Per maggiori informazioni

- "Risorse Trident"
- "Hub Kubernetes"

Riferimento

Porte Trident

Scopri di più sulle porte utilizzate Trident per le comunicazioni.

Porte Trident

Trident utilizza le seguenti porte per la comunicazione all'interno di Kubernetes:

Porta	Scopo
8443	Backchannel HTTPS
8001	Endpoint delle metriche di Prometheus
8000	Server REST Trident
17546	Porta di sonda di attività/prontezza utilizzata dai pod del daemonset Trident

La porta della sonda di attività/prontezza può essere modificata durante l'installazione utilizzando `--probe-port` bandiera. È importante assicurarsi che questa porta non sia utilizzata da un altro processo sui nodi worker.

API REST Trident

Mentre"comandi e opzioni tridentctl" rappresentano il modo più semplice per interagire con l'API REST Trident ; se preferisci, puoi utilizzare direttamente l'endpoint REST.

Quando utilizzare l'API REST

L'API REST è utile per installazioni avanzate che utilizzano Trident come binario autonomo in distribuzioni non Kubernetes.

Per una maggiore sicurezza, il Trident `REST API` è limitato per impostazione predefinita a localhost quando viene eseguito all'interno di un pod. Per modificare questo comportamento, è necessario impostare Trident `-address` argomento nella sua configurazione pod.

Utilizzo dell'API REST

Per esempi di come vengono chiamate queste API, passare il debug(`-d`) bandiera. Per maggiori informazioni, fare riferimento a"Gestisci Trident usando tridentctl" .

L'API funziona come segue:

OTTENERE

`GET <trident-address>/trident/v1/<object-type>`

Elenca tutti gli oggetti di quel tipo.

`GET <trident-address>/trident/v1/<object-type>/<object-name>`

Ottiene i dettagli dell'oggetto denominato.

INVIARE

`POST <trident-address>/trident/v1/<object-type>`

Crea un oggetto del tipo specificato.

- Richiede una configurazione JSON per l'oggetto da creare. Per la specifica di ciascun tipo di oggetto, fare riferimento a"Gestisci Trident usando tridentctl" .
- Se l'oggetto esiste già, il comportamento varia: i backend aggiornano l'oggetto esistente, mentre tutti gli altri tipi di oggetti non riusciranno nell'operazione.

ELIMINARE

`DELETE <trident-address>/trident/v1/<object-type>/<object-name>`

Elimina la risorsa indicata.

> I volumi associati ai backend o alle classi di archiviazione continueranno a esistere; dovranno essere eliminati separatamente. Per maggiori informazioni, fare riferimento a"Gestisci Trident usando tridentctl" .

Opzioni della riga di comando

Trident espone diverse opzioni della riga di comando per l'orchestratore Trident . Puoi utilizzare queste opzioni per modificare la tua distribuzione.

Registrazione

`-debug`

Abilita l'output di debug.

`-loglevel <level>`

Imposta il livello di registrazione (debug, info, warn, error, fatal). Per impostazione predefinita è info.

Kubernetes

`-k8s_pod`

Utilizzare questa opzione o `-k8s_api_server` per abilitare il supporto Kubernetes. Impostando questa opzione, Trident utilizzerà le credenziali dell'account di servizio Kubernetes del pod contenente per contattare il server API. Funziona solo quando Trident viene eseguito come pod in un cluster Kubernetes con account di servizio abilitati.

`-k8s_api_server <insecure-address:insecure-port>`

Utilizzare questa opzione o `-k8s_pod` per abilitare il supporto Kubernetes. Se specificato, Trident si connette al server API Kubernetes utilizzando l'indirizzo e la porta non sicuri forniti. Ciò consente di distribuire Trident al di fuori di un pod; tuttavia, supporta solo connessioni non sicure al server API. Per connettersi in modo sicuro, distribuire Trident in un pod con `-k8s_pod` opzione.

Docker

`-volume_driver <name>`

Nome del driver utilizzato durante la registrazione del plugin Docker. Predefinito su `netapp` .

`-driver_port <port-number>`

Ascolta su questa porta anziché su un socket di dominio UNIX.

`-config <file>`

Obbligatorio; è necessario specificare questo percorso per un file di configurazione backend.

RIPOSO

`-address <ip-or-host>`

Specifica l'indirizzo su cui il server REST di Trident deve essere in ascolto. Il valore predefinito è localhost. Quando si ascolta su localhost ed è in esecuzione all'interno di un pod Kubernetes, l'interfaccia REST non è direttamente accessibile dall'esterno del pod. Utilizzo `-address ""` per rendere l'interfaccia REST accessibile dall'indirizzo IP del pod.

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

`-port <port-number>`

Specifica la porta su cui il server REST di Trident deve essere in ascolto. Il valore predefinito è 8000.

`-rest`

Abilita l'interfaccia REST. Il valore predefinito è true.

Oggetti Kubernetes e Trident

È possibile interagire con Kubernetes e Trident utilizzando le API REST leggendo e scrivendo oggetti risorsa. Esistono diversi oggetti risorsa che determinano la relazione tra Kubernetes e Trident, Trident e storage e Kubernetes e storage. Alcuni di questi oggetti sono gestiti tramite Kubernetes, mentre altri sono gestiti tramite Trident.

Come interagiscono gli oggetti tra loro?

Forse il modo più semplice per comprendere gli oggetti, a cosa servono e come interagiscono è seguire una singola richiesta di archiviazione da parte di un utente Kubernetes:

1. Un utente crea un `PersistentVolumeClaim` richiedendo un nuovo `PersistentVolume` di una dimensione particolare da un Kubernetes `StorageClass` che era stato precedentemente configurato dall'amministratore.
2. Il Kubernetes `StorageClass` identifica Trident come suo fornitore e include parametri che indicano a Trident come fornire un volume per la classe richiesta.
3. Trident guarda se stesso `StorageClass` con lo stesso nome che identifica la corrispondenza `Backends` E `StoragePools` che può utilizzare per fornire volumi per la classe.
4. Trident fornisce spazio di archiviazione su un backend corrispondente e crea due oggetti: un

`PersistentVolume` in Kubernetes che indica a Kubernetes come trovare, montare e trattare il volume, e un volume in Trident che mantiene la relazione tra `PersistentVolume` e l'archiviazione vera e propria.

5. Kubernetes lega il `PersistentVolumeClaim` al nuovo `PersistentVolume` . Baccelli che includono il `PersistentVolumeClaim` montare quel PersistentVolume su qualsiasi host su cui viene eseguito.
6. Un utente crea un `VolumeSnapshot` di un PVC esistente, utilizzando un `VolumeSnapshotClass` che punta a Trident.
7. Trident identifica il volume associato al PVC e crea uno snapshot del volume sul suo backend. Crea anche un `VolumeSnapshotContent` che indica a Kubernetes come identificare lo snapshot.
8. Un utente può creare un `PersistentVolumeClaim` usando `VolumeSnapshot` come fonte.
9. Trident identifica lo snapshot richiesto ed esegue la stessa serie di passaggi coinvolti nella creazione di un `PersistentVolume` e un `Volume` .

Per ulteriori approfondimenti sugli oggetti Kubernetes, ti consigliamo vivamente di leggere "Volumi persistenti" sezione della documentazione di Kubernetes.

Kubernetes `PersistentVolumeClaim` oggetti

Un Kubernetes `PersistentVolumeClaim` L'oggetto è una richiesta di archiviazione effettuata da un utente del cluster Kubernetes.

Oltre alle specifiche standard, Trident consente agli utenti di specificare le seguenti annotazioni specifiche del volume se desiderano sovrascrivere le impostazioni predefinite impostate nella configurazione del backend:

Annotazione	Opzione volume	Driver supportati
trident.netapp.io/fileSystem	file System	ontap-san, solidfire-san,ontap-san-economy
trident.netapp.io/cloneFromPVC	cloneSourceVolume	ontap-nas, ontap-san, solidfire-san, azure-netapp-files, gcp-cvs, ontap-san-economy
trident.netapp.io/splitOnClone	splitOnClone	ontap-nas, ontap-san
trident.netapp.io/protocollo	protocollo	Qualunque
trident.netapp.io/exportPolicy	Politica di esportazione	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup
trident.netapp.io/snapshotPolicy	snapshotPolicy	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san
trident.netapp.io/snapshotReserve	snapshotReserve	ontap-nas, ontap-nas-flexgroup, ontap-san, gcp-cvs
trident.netapp.io/snapshotDirectory	directoryistantanea	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup
trident.netapp.io/unixPermissions	Permessi unix	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup
trident.netapp.io/blockSize	dimensione del blocco	solidfire-san

Se il PV creato ha il `Delete` politica di recupero, Trident elimina sia il PV che il volume di supporto quando il

PV viene rilasciato (ovvero quando l'utente elimina il PVC). Se l'operazione di eliminazione fallisce, Trident contrassegna il PV come tale e riprova periodicamente finché non riesce o finché il PV non viene eliminato manualmente. Se il PV utilizza il `Retain` policy, Trident la ignora e presume che l'amministratore la ripulirà da Kubernetes e dal backend, consentendo il backup o l'ispezione del volume prima della sua rimozione. Si noti che l'eliminazione del PV non comporta l'eliminazione del volume di supporto Trident . Dovresti rimuoverlo usando l'API REST(`tridentctl`).

Trident supporta la creazione di snapshot del volume utilizzando la specifica CSI: è possibile creare uno snapshot del volume e utilizzarlo come origine dati per clonare PVC esistenti. In questo modo, le copie point-in-time dei PV possono essere esposte a Kubernetes sotto forma di snapshot. Gli snapshot possono quindi essere utilizzati per creare nuovi PV. Dai un'occhiata a `On-Demand Volume Snapshots` per vedere come funzionerebbe.

Trident fornisce anche il `cloneFromPVC` E `splitOnClone` annotazioni per la creazione di cloni. È possibile utilizzare queste annotazioni per clonare un PVC senza dover ricorrere all'implementazione CSI.

Ecco un esempio: se un utente ha già un PVC chiamato `mysql` , l'utente può creare un nuovo PVC chiamato `mysqlclone` utilizzando l'annotazione, come ad esempio `trident.netapp.io/cloneFromPVC: mysql` . Con questo set di annotazioni, Trident clona il volume corrispondente al PVC mysql, anziché effettuare il provisioning di un volume da zero.

Considera i seguenti punti:

- NetApp consiglia di clonare un volume inattivo.
- Un PVC e il suo clone devono trovarsi nello stesso namespace Kubernetes e avere la stessa classe di archiviazione.
- Con il `ontap-nas` E `ontap-san` driver, potrebbe essere desiderabile impostare l'annotazione PVC `trident.netapp.io/splitOnClone` in collaborazione con `trident.netapp.io/cloneFromPVC` . Con `trident.netapp.io/splitOnClone` impostato su `true` Trident divide il volume clonato dal volume padre e, quindi, disaccoppia completamente il ciclo di vita del volume clonato dal suo padre, a scapito della perdita di efficienza di archiviazione. Non impostare `trident.netapp.io/splitOnClone` o impostandolo su `false` comporta una riduzione del consumo di spazio sul backend a scapito della creazione di dipendenze tra i volumi padre e clone, in modo che il volume padre non possa essere eliminato a meno che non venga eliminato prima il clone. Uno scenario in cui la suddivisione del clone ha senso è la clonazione di un volume di database vuoto in cui ci si aspetta che il volume e il suo clone divergano notevolmente e non traggano vantaggio dalle efficienze di archiviazione offerte da ONTAP.

IL `sample-input` La directory contiene esempi di definizioni PVC da utilizzare con Trident. Fare riferimento a per una descrizione completa dei parametri e delle impostazioni associati ai volumi Trident .

Kubernetes `PersistentVolume` oggetti

Un Kubernetes `PersistentVolume` L'oggetto rappresenta un elemento di storage reso disponibile al cluster Kubernetes. Ha un ciclo di vita indipendente dal pod che lo utilizza.

> Trident crea `PersistentVolume` oggetti e li registra automaticamente nel cluster Kubernetes in base ai volumi che fornisce. Non ci si aspetta che tu li gestisca da solo.

Quando si crea un PVC che fa riferimento a un Trident-based `StorageClass` Trident predispone un nuovo volume utilizzando la classe di archiviazione corrispondente e registra un nuovo PV per tale volume. Nella configurazione del volume fornito e del PV corrispondente, Trident segue le seguenti regole:

- Trident genera un nome PV per Kubernetes e un nome interno che utilizza per il provisioning dello storage. In entrambi i casi, si garantisce che i nomi siano univoci nel loro ambito.
- La dimensione del volume corrisponde il più possibile alla dimensione richiesta nel PVC, anche se potrebbe essere arrotondata alla quantità allocabile più vicina, a seconda della piattaforma.

Kubernetes `StorageClass` oggetti

Kubernetes `StorageClass` gli oggetti sono specificati per nome in `PersistentVolumeClaims` per fornire storage con un set di proprietà. La classe di archiviazione stessa identifica il provisioner da utilizzare e definisce il set di proprietà in termini comprensibili al provisioner.

È uno dei due oggetti di base che devono essere creati e gestiti dall'amministratore. L'altro è l'oggetto backend Trident .

Un Kubernetes `StorageClass` L'oggetto che utilizza Trident si presenta così:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: <Name>
provisioner: csi.trident.netapp.io
mountOptions: <Mount Options>
parameters: <Trident Parameters>
allowVolumeExpansion: true
volumeBindingMode: Immediate
```

Questi parametri sono specifici di Trident e indicano a Trident come effettuare il provisioning dei volumi per la classe.

I parametri della classe di archiviazione sono:

Attributo	Tipo	Necessario	Descrizione
attributi	mappa[stringa]stringa	NO	Vedi la sezione attributi qui sotto
pool di stoccaggio	map[string]StringList	NO	Mappa dei nomi backend agli elenchi dei pool di archiviazione all'interno
pool di archiviazione aggiuntivi	map[string]StringList	NO	Mappa dei nomi backend agli elenchi dei pool di archiviazione all'interno
Escludi pool di archiviazione	map[string]StringList	NO	Mappa dei nomi backend agli elenchi dei pool di archiviazione all'interno

Gli attributi di archiviazione e i loro possibili valori possono essere classificati in attributi di selezione del pool di archiviazione e attributi Kubernetes.

Attributi di selezione del pool di archiviazione

Questi parametri determinano quali pool di archiviazione gestiti da Trident devono essere utilizzati per fornire volumi di un determinato tipo.

Attributo	Tipo	Valori	Offerta	Richiesta	Supportato da
media[1]	corda	hdd, ibrido, ssd	Il pool contiene supporti di questo tipo; ibrido significa entrambi	Tipo di supporto specificato	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san
provisioningType	corda	sottile, spesso	Il pool supporta questo metodo di provisioning	Metodo di provisioning specificato	spesso: tutto ontap; sottile: tutto ontap e solidfire-san
tipo backend	corda	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san, gcp-cvs, azure-netapp-files, ontap-san-economy	Pool appartiene a questo tipo di backend	Backend specificato	Tutti i conducenti
istantanee	bool	vero, falso	Il pool supporta volumi con snapshot	Volume con snapshot abilitati	ontap-nas, ontap-san, solidfire-san, gcp-cvs
cloni	bool	vero, falso	Il pool supporta la clonazione dei volumi	Volume con cloni abilitati	ontap-nas, ontap-san, solidfire-san, gcp-cvs
crittografia	bool	vero, falso	Il pool supporta volumi crittografati	Volume con crittografia abilitata	ontap-nas, ontap-nas-economy, ontap-nas-flexgroups, ontap-san
IOPS	interno	intero positivo	Pool è in grado di garantire IOPS in questo intervallo	Volume garantito per questi IOPS	solidfire-san

[1]: Non supportato dai sistemi ONTAP Select

Nella maggior parte dei casi, i valori richiesti influenzano direttamente il provisioning; ad esempio, la richiesta di provisioning spesso produce un volume con provisioning spesso. Tuttavia, un pool di archiviazione Element utilizza i valori IOPS minimi e massimi offerti per impostare i valori QoS, anziché il valore richiesto. In questo

caso, il valore richiesto viene utilizzato solo per selezionare il pool di archiviazione.

Idealmente, puoi usare `attributes` da solo per modellare le qualità dello spazio di archiviazione necessario a soddisfare le esigenze di una particolare classe. Trident rileva e seleziona automaticamente i pool di archiviazione che corrispondono a *tutti* i `attributes` che specifichi.

Se ti accorgi di non essere in grado di utilizzare `attributes` per selezionare automaticamente le piscine giuste per una classe, puoi usare `storagePools` E `additionalStoragePools` parametri per perfezionare ulteriormente i pool o addirittura per selezionare un set specifico di pool.

Puoi usare il `storagePools` parametro per limitare ulteriormente l'insieme di pool che corrispondono a qualsiasi specificato `attributes` . In altre parole, Trident utilizza l'intersezione dei pool identificati dal `attributes` E `storagePools` parametri per il provisioning. È possibile utilizzare uno dei due parametri da solo oppure entrambi insieme.

Puoi usare il `additionalStoragePools` parametro per estendere il set di pool che Trident utilizza per il provisioning, indipendentemente dai pool selezionati da `attributes` E `storagePools` parametri.

Puoi usare il `excludeStoragePools` parametro per filtrare l'insieme di pool che Trident utilizza per il provisioning. Utilizzando questo parametro vengono rimossi tutti i pool corrispondenti.

Nel `storagePools` E `additionalStoragePools` parametri, ogni voce assume la forma `<backend>:<storagePoolList>` , Dove `<storagePoolList>` è un elenco separato da virgole di pool di archiviazione per il backend specificato. Ad esempio, un valore per `additionalStoragePools` potrebbe sembrare `ontapnas_192.168.1.100:aggr1,aggr2;solidfire_192.168.1.101:bronze` . Questi elenchi accettano valori regex sia per i valori backend che per quelli dell'elenco. Puoi usare `tridentctl get backend` per ottenere l'elenco dei backend e dei loro pool.

Attributi di Kubernetes

Questi attributi non hanno alcun impatto sulla selezione dei pool di archiviazione/backend da parte di Trident durante il provisioning dinamico. Questi attributi, invece, forniscono semplicemente parametri supportati dai volumi persistenti di Kubernetes. I nodi worker sono responsabili delle operazioni di creazione del file system e potrebbero richiedere utilità del file system, come xfsprogs.

Attributo	Tipo	Valori	Descrizione	Fattori rilevanti	Versione di Kubernetes
fsType	corda	ext4, ext3, xfs	Il tipo di file system per i volumi a blocchi	solidfire-san, ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economy	Tutto

Attributo	Tipo	Valori	Descrizione	Fattori rilevanti	Versione di Kubernetes
consentiEspansioneVolume	booleano	vero, falso	Abilita o disabilita il supporto per l'aumento delle dimensioni del PVC	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economy, solidfire-san, gcp-cvs, azure-netapp-files	1.11+
volumeBindingMode	corda	Immediato, WaitForFirstConsumer	Scegli quando si verifica il binding del volume e il provisioning dinamico	Tutto	1,19 - 1,26

- IL `fsType` Il parametro viene utilizzato per controllare il tipo di file system desiderato per le LUN SAN. Inoltre, Kubernetes sfrutta anche la presenza di `fsType` in una classe di archiviazione per indicare l'esistenza di un file system. La proprietà del volume può essere controllata utilizzando `fsGroup` contesto di sicurezza di un pod solo se `fsType` è impostato. Fare riferimento a"Kubernetes: configurare un contesto di sicurezza per un pod o un contenitore" per una panoramica sull'impostazione della proprietà del volume utilizzando `fsGroup` contesto. Kubernetes applicherà il `fsGroup` valore solo se:
  - `fsType`è impostato nella classe di archiviazione.
  - La modalità di accesso al PVC è RWO.

  Per i driver di archiviazione NFS, esiste già un file system come parte dell'esportazione NFS. Per poter utilizzare `fsGroup` la classe di archiviazione deve ancora specificare un `fsType` Puoi impostarlo su `nfs` o qualsiasi valore non nullo.
- Fare riferimento a"Espandi i volumi" per ulteriori dettagli sull'espansione del volume.
- Il pacchetto di installazione Trident fornisce diverse definizioni di classi di archiviazione di esempio da utilizzare con Trident in`sample-input/storage-class-*.yaml` . L'eliminazione di una classe di archiviazione Kubernetes comporta l'eliminazione anche della classe di archiviazione Trident corrispondente.

Kubernetes `VolumeSnapshotClass` oggetti

Kubernetes `VolumeSnapshotClass` gli oggetti sono analoghi a `StorageClasses` . Consentono di definire più classi di archiviazione e sono referenziati dagli snapshot del volume per associare lo snapshot alla classe di snapshot richiesta. Ogni snapshot del volume è associato a una singola classe di snapshot del volume.

UN `VolumeSnapshotClass` deve essere definito da un amministratore per poter creare snapshot. Viene creata una classe snapshot del volume con la seguente definizione:

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: csi-snapclass
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

IL `driver` specifica a Kubernetes che le richieste per gli snapshot del volume di `csi-snapclass` la classe è gestita da Trident. IL `deletionPolicy` specifica l'azione da intraprendere quando è necessario eliminare uno snapshot. Quando `deletionPolicy` è impostato su `Delete` , gli oggetti snapshot del volume e lo snapshot sottostante sul cluster di archiviazione vengono rimossi quando uno snapshot viene eliminato. In alternativa, impostandolo su `Retain` significa che `VolumeSnapshotContent` e l'istantanea fisica vengono mantenute.

Kubernetes `VolumeSnapshot` oggetti

Un Kubernetes `VolumeSnapshot` object è una richiesta per creare uno snapshot di un volume. Proprio come un PVC rappresenta una richiesta effettuata da un utente per un volume, uno snapshot del volume è una richiesta effettuata da un utente per creare uno snapshot di un PVC esistente.

Quando arriva una richiesta di snapshot del volume, Trident gestisce automaticamente la creazione dello snapshot per il volume sul backend ed espone lo snapshot creando un'immagine univoca `VolumeSnapshotContent` oggetto. È possibile creare snapshot da PVC esistenti e utilizzarli come DataSource durante la creazione di nuovi PVC.

> Il ciclo di vita di un VolumeSnapshot è indipendente dal PVC di origine: uno snapshot persiste anche dopo l'eliminazione del PVC di origine. Quando si elimina un PVC a cui sono associati snapshot, Trident contrassegna il volume di supporto per questo PVC in uno stato **Eliminazione**, ma non lo rimuove completamente. Il volume viene rimosso quando vengono eliminati tutti gli snapshot associati.

Kubernetes `VolumeSnapshotContent` oggetti

Un Kubernetes `VolumeSnapshotContent` L'oggetto rappresenta uno snapshot preso da un volume già sottoposto a provisioning. È analogo a un `PersistentVolume` e indica uno snapshot fornito sul cluster di archiviazione. Simile a `PersistentVolumeClaim` E `PersistentVolume` oggetti, quando viene creato uno snapshot, il `VolumeSnapshotContent` l'oggetto mantiene una mappatura uno a uno con `VolumeSnapshot` oggetto che aveva richiesto la creazione dello snapshot.

IL `VolumeSnapshotContent` l'oggetto contiene dettagli che identificano in modo univoco lo snapshot, come ad esempio `snapshotHandle` . Questo `snapshotHandle` è una combinazione unica del nome del PV e del nome del `VolumeSnapshotContent` oggetto.

Quando arriva una richiesta di snapshot, Trident crea lo snapshot nel backend. Dopo aver creato lo snapshot, Trident configura un `VolumeSnapshotContent` oggetto e quindi espone lo snapshot all'API Kubernetes.

> In genere, non è necessario gestire il `VolumeSnapshotContent` oggetto. Un'eccezione a questo è quando vuoi"importare uno snapshot del volume" creato al di fuori di Trident.

Kubernetes `VolumeGroupSnapshotClass` oggetti

Kubernetes `VolumeGroupSnapshotClass` gli oggetti sono analoghi a `VolumeSnapshotClass` . Consentono di definire più classi di archiviazione e sono referenziati dagli snapshot del gruppo di volumi per associare lo snapshot alla classe di snapshot richiesta. Ogni snapshot del gruppo di volumi è associato a una singola classe di snapshot del gruppo di volumi.

UN `VolumeGroupSnapshotClass` dovrebbe essere definito da un amministratore per creare un gruppo di snapshot. Viene creata una classe snapshot del gruppo di volumi con la seguente definizione:

```
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
```

IL `driver` specifica a Kubernetes che le richieste per gli snapshot del gruppo di volumi del `csi-group-snap-class` la classe è gestita da Trident. IL `deletionPolicy` specifica l'azione da intraprendere quando è necessario eliminare uno snapshot di gruppo. Quando `deletionPolicy` è impostato su `Delete` , gli oggetti snapshot del gruppo di volumi e lo snapshot sottostante sul cluster di archiviazione vengono rimossi quando uno snapshot viene eliminato. In alternativa, impostandolo su `Retain` significa che `VolumeGroupSnapshotContent` e l'istantanea fisica vengono mantenute.

Kubernetes `VolumeGroupSnapshot` oggetti

Un Kubernetes `VolumeGroupSnapshot` object è una richiesta per creare uno snapshot di più volumi. Proprio come un PVC rappresenta una richiesta effettuata da un utente per un volume, uno snapshot del gruppo di volumi è una richiesta effettuata da un utente per creare uno snapshot di un PVC esistente.

Quando arriva una richiesta di snapshot del gruppo di volumi, Trident gestisce automaticamente la creazione dello snapshot del gruppo per i volumi sul backend ed espone lo snapshot creando un'immagine univoca `VolumeGroupSnapshotContent` oggetto. È possibile creare snapshot da PVC esistenti e utilizzarli come DataSource durante la creazione di nuovi PVC.

> Il ciclo di vita di un VolumeGroupSnapshot è indipendente dal PVC di origine: uno snapshot persiste anche dopo l'eliminazione del PVC di origine. Quando si elimina un PVC a cui sono associati snapshot, Trident contrassegna il volume di supporto per questo PVC in uno stato **Eliminazione**, ma non lo rimuove completamente. Lo snapshot del gruppo di volumi viene rimosso quando vengono eliminati tutti gli snapshot associati.

Kubernetes `VolumeGroupSnapshotContent` oggetti

Un Kubernetes `VolumeGroupSnapshotContent` L'oggetto rappresenta uno snapshot di gruppo preso da un volume già sottoposto a provisioning. È analogo a un `PersistentVolume` e indica uno snapshot fornito sul cluster di archiviazione. Simile a `PersistentVolumeClaim` E `PersistentVolume` oggetti, quando viene creato uno snapshot, il `VolumeSnapshotContent` l'oggetto mantiene una mappatura uno a uno con

`VolumeSnapshot` oggetto che aveva richiesto la creazione dello snapshot.

IL `VolumeGroupSnapshotContent` l'oggetto contiene dettagli che identificano il gruppo di snapshot, come ad esempio `volumeGroupSnapshotHandle` e singoli volumiSnapshotHandles esistenti sul sistema di archiviazione.

Quando arriva una richiesta di snapshot, Trident crea lo snapshot del gruppo di volumi sul backend. Dopo aver creato lo snapshot del gruppo di volumi, Trident configura un `VolumeGroupSnapshotContent` oggetto e quindi espone lo snapshot all'API Kubernetes.

Kubernetes `CustomResourceDefinition` oggetti

Le risorse personalizzate di Kubernetes sono endpoint nell'API di Kubernetes definiti dall'amministratore e utilizzati per raggruppare oggetti simili. Kubernetes supporta la creazione di risorse personalizzate per l'archiviazione di una raccolta di oggetti. È possibile ottenere queste definizioni di risorse eseguendo `kubectl get crds` .

Le definizioni di risorse personalizzate (CRD) e i metadati degli oggetti associati vengono archiviati da Kubernetes nel suo archivio metadati. In questo modo si elimina la necessità di un archivio separato per Trident.

Usi Trident `CustomResourceDefinition` oggetti per preservare l'identità degli oggetti Trident , come i backend Trident , le classi di archiviazione Trident e i volumi Trident . Questi oggetti sono gestiti da Trident. Inoltre, il framework degli snapshot del volume CSI introduce alcuni CRD necessari per definire gli snapshot del volume.

I CRD sono una struttura di Kubernetes. Gli oggetti delle risorse definite sopra vengono creati da Trident. Come semplice esempio, quando un backend viene creato utilizzando `tridentctl` , un corrispondente `tridentbackends` L'oggetto CRD viene creato per essere utilizzato da Kubernetes.

Ecco alcuni punti da tenere a mente sui CRD di Trident:

- Quando Trident viene installato, viene creato un set di CRD che può essere utilizzato come qualsiasi altro tipo di risorsa.
- Quando si disinstalla Trident utilizzando `tridentctl uninstall` comando, i pod Trident vengono eliminati ma i CRD creati non vengono ripuliti. Fare riferimento a"Disinstallare Trident" per capire come Trident può essere completamente rimosso e riconfigurato da zero.

Trident `StorageClass` oggetti

Trident crea classi di archiviazione corrispondenti per Kubernetes `StorageClass` oggetti che specificano `csi.trident.netapp.io` nel loro campo di fornitura. Il nome della classe di archiviazione corrisponde a quello di Kubernetes `StorageClass` oggetto che rappresenta.

> Con Kubernetes, questi oggetti vengono creati automaticamente quando un Kubernetes `StorageClass` che utilizza Trident come fornitore è registrato.

Le classi di archiviazione comprendono un insieme di requisiti per i volumi. Trident confronta questi requisiti con gli attributi presenti in ciascun pool di archiviazione; se corrispondono, quel pool di archiviazione è una destinazione valida per il provisioning dei volumi utilizzando quella classe di archiviazione.

È possibile creare configurazioni di classi di archiviazione per definire direttamente le classi di archiviazione

utilizzando l'API REST. Tuttavia, per le distribuzioni Kubernetes, ci aspettiamo che vengano create durante la registrazione di nuovi Kubernetes `StorageClass` oggetti.

Oggetti backend Trident

I backend rappresentano i provider di storage su cui Trident effettua il provisioning dei volumi; una singola istanza Trident può gestire un numero qualsiasi di backend.

> Questo è uno dei due tipi di oggetti che puoi creare e gestire autonomamente. L'altro è Kubernetes `StorageClass` oggetto.

Per maggiori informazioni su come costruire questi oggetti, fare riferimento a"configurazione dei backend" .

Trident `StoragePool` oggetti

I pool di archiviazione rappresentano le diverse posizioni disponibili per il provisioning su ciascun backend. Per ONTAP, questi corrispondono agli aggregati nelle SVM. Per NetApp HCI/ SolidFire, questi corrispondono alle bande QoS specificate dall'amministratore. Per il Cloud Volumes Service, questi corrispondono alle regioni del provider cloud. Ogni pool di archiviazione ha un set di attributi di archiviazione distinti, che ne definiscono le caratteristiche prestazionali e di protezione dei dati.

A differenza degli altri oggetti qui, i candidati al pool di archiviazione vengono sempre rilevati e gestiti automaticamente.

Trident `Volume` oggetti

I volumi sono l'unità di base del provisioning e comprendono endpoint back-end, come condivisioni NFS e LUN iSCSI e FC. In Kubernetes, questi corrispondono direttamente a `PersistentVolumes` . Quando si crea un volume, assicurarsi che disponga di una classe di archiviazione, che determina dove può essere eseguito il provisioning del volume, insieme a una dimensione.

> - In Kubernetes, questi oggetti vengono gestiti automaticamente. Puoi visualizzarli per vedere cosa ha fornito Trident .
> - Quando si elimina un PV con snapshot associati, il volume Trident corrispondente viene aggiornato allo stato **Eliminazione**. Per eliminare il volume Trident , è necessario rimuovere gli snapshot del volume.

Una configurazione del volume definisce le proprietà che un volume fornito dovrebbe avere.

Attributo	Tipo	Necessario	Descrizione
versione	corda	NO	Versione dell'API Trident ("1")
nome	corda	SÌ	Nome del volume da creare
classe di archiviazione	corda	SÌ	Classe di archiviazione da utilizzare durante il provisioning del volume
misurare	corda	SÌ	Dimensione del volume da fornire in byte

Attributo	Tipo	Necessario	Descrizione
protocollo	corda	NO	Tipo di protocollo da utilizzare; "file" o "blocco"
Nome interno	corda	NO	Nome dell'oggetto sul sistema di archiviazione; generato da Trident
cloneSourceVolume	corda	NO	ontap (nas, san) e solidfire-*: Nome del volume da cui clonare
splitOnClone	corda	NO	ontap (nas, san): divide il clone dal suo genitore
snapshotPolicy	corda	NO	ontap-*: criterio di snapshot da utilizzare
snapshotReserve	corda	NO	ontap-*: percentuale di volume riservata per gli snapshot
Politica di esportazione	corda	NO	ontap-nas*: politica di esportazione da utilizzare
directoryistantanea	bool	NO	ontap-nas*: se la directory snapshot è visibile
Permessi unix	corda	NO	ontap-nas*: permessi UNIX iniziali
dimensione del blocco	corda	NO	solidfire-*: dimensione del blocco/settore
file System	corda	NO	Tipo di file system

Trident genera `internalName` durante la creazione del volume. Si compone di due fasi. Innanzitutto, antepone il prefisso di archiviazione (predefinito `trident` o il prefisso nella configurazione del backend) al nome del volume, risultando in un nome del tipo `<prefix>-<volume-name>` . Procede quindi a ripulire il nome, sostituendo i caratteri non consentiti nel backend. Per i backend ONTAP , sostituisce i trattini con i caratteri di sottolineatura (quindi, il nome interno diventa `<prefix>_<volume-name>`). Per i backend Element, sostituisce i trattini bassi con i trattini.

È possibile utilizzare le configurazioni del volume per effettuare il provisioning diretto dei volumi tramite l'API REST, ma nelle distribuzioni Kubernetes ci aspettiamo che la maggior parte degli utenti utilizzi la configurazione standard di Kubernetes. `PersistentVolumeClaim` metodo. Trident crea automaticamente questo oggetto volume come parte del processo di provisioning.

Trident `Snapshot` oggetti

Gli snapshot sono copie di volumi effettuate in un dato momento, che possono essere utilizzate per predisporre nuovi volumi o ripristinarne lo stato. In Kubernetes, questi corrispondono direttamente a `VolumeSnapshotContent` oggetti. Ogni snapshot è associato a un volume, che è l'origine dei dati per lo snapshot.

Ogni `Snapshot` l'oggetto include le proprietà elencate di seguito:

Attributo	Tipo	Necessario	Descrizione
versione	Corda	SÌ	Versione dell'API Trident ("1")
nome	Corda	SÌ	Nome dell'oggetto snapshot Trident
Nome interno	Corda	SÌ	Nome dell'oggetto snapshot Trident sul sistema di archiviazione
NomeVolume	Corda	SÌ	Nome del volume persistente per il quale viene creato lo snapshot
volumeInternalName	Corda	SÌ	Nome dell'oggetto volume Trident associato sul sistema di archiviazione

In Kubernetes, questi oggetti vengono gestiti automaticamente. Puoi visualizzarli per vedere cosa ha fornito Trident .

Quando un Kubernetes `VolumeSnapshot` Una volta creata la richiesta dell'oggetto, Trident funziona creando un oggetto snapshot sul sistema di archiviazione di supporto. IL `internalName` di questo oggetto snapshot viene generato combinando il prefisso `snapshot-` con il `UID` del `VolumeSnapshot` oggetto (ad esempio, `snapshot-e8d8a0ca-9826-11e9-9807-525400f3f660`). `volumeName` E `volumeInternalName` vengono popolati ottenendo i dettagli del volume di supporto.

Trident `ResourceQuota` oggetto

Il set di demoni Trident consuma un `system-node-critical` Classe di priorità: la classe di priorità più elevata disponibile in Kubernetes, per garantire che Trident possa identificare e ripulire i volumi durante l'arresto regolare dei nodi e consentire ai pod daemonset Trident di anticipare i carichi di lavoro con una priorità inferiore nei cluster in cui vi è un'elevata pressione sulle risorse.

Per raggiungere questo obiettivo, Trident impiega un `ResourceQuota` oggetto per garantire che venga soddisfatta una classe di priorità "system-node-critical" sul daemonset Trident . Prima della distribuzione e della creazione del daemonset, Trident cerca `ResourceQuota` oggetto e, se non viene scoperto, lo applica.

Se hai bisogno di un maggiore controllo sulla quota di risorse predefinita e sulla classe di priorità, puoi generare un `custom.yaml` o configurare il `ResourceQuota` oggetto utilizzando il grafico Helm.

Di seguito è riportato un esempio di un oggetto `ResourceQuota` che assegna la priorità al daemonset Trident .

```
apiVersion: <version>
kind: ResourceQuota
metadata:
  name: trident-csi
  labels:
    app: node.csi.trident.netapp.io
spec:
  scopeSelector:
    matchExpressions:
      - operator: In
        scopeName: PriorityClass
        values:
          - system-node-critical
```

Per ulteriori informazioni sulle quote di risorse, fare riferimento a"Kubernetes: quote di risorse" .

Ripulire `ResourceQuota` se l'installazione fallisce

Nel raro caso in cui l'installazione fallisca dopo l' `ResourceQuota` l'oggetto è stato creato, primo tentativo"disinstallazione" e poi reinstallarlo.

Se ciò non funziona, rimuovere manualmente il `ResourceQuota` oggetto.

Rimuovere `ResourceQuota`

Se preferisci controllare la tua allocazione delle risorse, puoi rimuovere il Trident `ResourceQuota` oggetto utilizzando il comando:

```
kubectl delete quota trident-csi -n trident
```

Standard di sicurezza dei pod (PSS) e vincoli di contesto di sicurezza (SCC)

Gli standard di sicurezza dei pod (PSS) e le policy di sicurezza dei pod (PSP) di Kubernetes definiscono i livelli di autorizzazione e limitano il comportamento dei pod. Allo stesso modo, i vincoli di contesto di sicurezza (SCC) di OpenShift definiscono le restrizioni dei pod specifiche per OpenShift Kubernetes Engine. Per fornire questa personalizzazione, Trident abilita determinate autorizzazioni durante l'installazione. Le sezioni seguenti descrivono in dettaglio le autorizzazioni impostate da Trident.

PSS sostituisce Pod Security Policies (PSP). PSP è stato deprecato in Kubernetes v1.21 e verrà rimosso nella v1.25. Per maggiori informazioni, fare riferimento a"Kubernetes: sicurezza" .

Contesto di sicurezza Kubernetes obbligatorio e campi correlati

Permesso	Descrizione
Privilegiato	CSI richiede che i punti di montaggio siano bidirezionali, il che significa che il pod del nodo Trident deve eseguire un contenitore privilegiato. Per maggiori informazioni, fare riferimento a"Kubernetes: propagazione del mount" .
Rete di host	Necessario per il demone iSCSI. `iscsiadm` gestisce i mount iSCSI e utilizza la rete host per comunicare con il demone iSCSI.
IPC host	NFS utilizza la comunicazione interprocesso (IPC) per comunicare con NFSD.
PID host	Necessario per iniziare `rpc-statd` per NFS. Trident interroga i processi host per determinare se `rpc-statd` è in esecuzione prima di montare i volumi NFS.

Capacità	IL `SYS_ADMIN` la funzionalità è fornita come parte delle funzionalità predefinite per i contenitori privilegiati. Ad esempio, Docker imposta queste funzionalità per i contenitori privilegiati:
	`CapPrm: 0000003fffffffff`
	`CapEff: 0000003fffffffff`

Seccomp	Il profilo Seccomp è sempre "Non vincolato" nei contenitori privilegiati; pertanto, non può essere abilitato in Trident.
SELinux	Su OpenShift, i contenitori privilegiati vengono eseguiti in `spc_t` ("Super Privileged Container") dominio e i contenitori non privilegiati vengono eseguiti nel `container_t` dominio. SU `containerd` , con `container-selinux` installato, tutti i contenitori vengono eseguiti in `spc_t` dominio, che di fatto disabilita SELinux. Pertanto, Trident non aggiunge `seLinuxOptions` ai contenitori.
DAC	I contenitori privilegiati devono essere eseguiti come root. I contenitori non privilegiati vengono eseguiti come root per accedere ai socket Unix richiesti da CSI.

Standard di sicurezza del pod (PSS)

Etichetta	Descrizione	Predefinito
`pod-security.kubernetes.io/enforce` `pod-security.kubernetes.io/enforce-version`	Consente l'ammissione del controller Trident e dei nodi nello spazio dei nomi di installazione. Non modificare l'etichetta dello spazio dei nomi.	`enforce: privileged enforce-version: <version of the current cluster or highest version of PSS tested.>`

La modifica delle etichette dello spazio dei nomi può comportare la mancata pianificazione dei pod, un messaggio di errore "Errore durante la creazione: …" o "Avviso: trident-csi-…". Se ciò accade, controllare se l'etichetta dello spazio dei nomi per `privileged` è stato cambiato. In tal caso, reinstallare Trident.

Criteri di sicurezza dei pod (PSP)

Campo	Descrizione	Predefinito
`allowPrivilegeEscalation`	I contenitori privilegiati devono consentire l'escalation dei privilegi.	`true`
`allowedCSIDrivers`	Trident non utilizza volumi effimeri CSI in linea.	Vuoto
`allowedCapabilities`	I contenitori Trident non privilegiati non richiedono più capacità rispetto al set predefinito e ai contenitori privilegiati vengono concesse tutte le capacità possibili.	Vuoto
`allowedFlexVolumes`	Trident non utilizza un"Driver FlexVolume" , pertanto non sono inclusi nell'elenco dei volumi consentiti.	Vuoto
`allowedHostPaths`	Il pod del nodo Trident monta il file system radice del nodo, pertanto non vi è alcun vantaggio nell'impostare questo elenco.	Vuoto
`allowedProcMountTypes`	Trident non usa nessuno `ProcMountTypes` .	Vuoto
`allowedUnsafeSysctls`	Trident non richiede alcuna sicurezza `sysctls` .	Vuoto
`defaultAddCapabilities`	Non è necessario aggiungere alcuna funzionalità ai contenitori privilegiati.	Vuoto
`defaultAllowPrivilegeEscalation`	L'autorizzazione all'escalation dei privilegi viene gestita in ogni pod Trident .	`false`
`forbiddenSysctls`	NO `sysctls` sono consentiti.	Vuoto
`fsGroup`	I contenitori Trident vengono eseguiti come root.	`RunAsAny`
`hostIPC`	Il montaggio dei volumi NFS richiede che l'host IPC comunichi con `nfsd`	`true`
`hostNetwork`	iscsiadm richiede che la rete host comunichi con il demone iSCSI.	`true`

Campo	Descrizione	Predefinito
`hostPID`	È necessario il PID host per verificare se `rpc-statd` è in esecuzione sul nodo.	`true`
`hostPorts`	Trident non utilizza alcuna porta host.	Vuoto
`privileged`	I pod del nodo Trident devono eseguire un contenitore privilegiato per poter montare i volumi.	`true`
`readOnlyRootFilesystem`	I pod del nodo Trident devono scrivere sul file system del nodo.	`false`
`requiredDropCapabilities`	I pod del nodo Trident eseguono un contenitore privilegiato e non possono rilasciare capacità.	`none`
`runAsGroup`	I contenitori Trident vengono eseguiti come root.	`RunAsAny`
`runAsUser`	I contenitori Trident vengono eseguiti come root.	`runAsAny`
`runtimeClass`	Trident non usa `RuntimeClasses` .	Vuoto
`seLinux`	Il Trident non tramonta `seLinuxOptions` perché attualmente ci sono differenze nel modo in cui i runtime dei container e le distribuzioni di Kubernetes gestiscono SELinux.	Vuoto
`supplementalGroups`	I contenitori Trident vengono eseguiti come root.	`RunAsAny`
`volumes`	I pod Trident richiedono questi plugin di volume.	`hostPath, projected, emptyDir`

Vincoli di contesto di sicurezza (SCC)

Etichette	Descrizione	Predefinito
`allowHostDirVolumePlugin`	I pod del nodo Trident montano il file system radice del nodo.	`true`
`allowHostIPC`	Il montaggio dei volumi NFS richiede che l'host IPC comunichi con `nfsd` .	`true`
`allowHostNetwork`	iscsiadm richiede che la rete host comunichi con il demone iSCSI.	`true`
`allowHostPID`	È necessario il PID host per verificare se `rpc-statd` è in esecuzione sul nodo.	`true`

Etichette	Descrizione	Predefinito
`allowHostPorts`	Trident non utilizza alcuna porta host.	`false`
`allowPrivilegeEscalation`	I contenitori privilegiati devono consentire l'escalation dei privilegi.	`true`
`allowPrivilegedContainer`	I pod del nodo Trident devono eseguire un contenitore privilegiato per poter montare i volumi.	`true`
`allowedUnsafeSysctls`	Trident non richiede alcuna sicurezza `sysctls` .	`none`
`allowedCapabilities`	I contenitori Trident non privilegiati non richiedono più capacità rispetto al set predefinito e ai contenitori privilegiati vengono concesse tutte le capacità possibili.	Vuoto
`defaultAddCapabilities`	Non è necessario aggiungere alcuna funzionalità ai contenitori privilegiati.	Vuoto
`fsGroup`	I contenitori Trident vengono eseguiti come root.	`RunAsAny`
`groups`	Il presente SCC è specifico per Trident ed è vincolante per il suo utente.	Vuoto
`readOnlyRootFilesystem`	I pod del nodo Trident devono scrivere sul file system del nodo.	`false`
`requiredDropCapabilities`	I pod del nodo Trident eseguono un contenitore privilegiato e non possono rilasciare capacità.	`none`
`runAsUser`	I contenitori Trident vengono eseguiti come root.	`RunAsAny`
`seLinuxContext`	Il Trident non tramonta `seLinuxOptions` perché attualmente ci sono differenze nel modo in cui i runtime dei container e le distribuzioni di Kubernetes gestiscono SELinux.	Vuoto
`seccompProfiles`	I contenitori privilegiati vengono sempre eseguiti in modalità "Unconfined".	Vuoto
`supplementalGroups`	I contenitori Trident vengono eseguiti come root.	`RunAsAny`
`users`	Viene fornita una voce per associare questo SCC all'utente Trident nello spazio dei nomi Trident .	n / a

Etichette	Descrizione	Predefinito
`volumes`	I pod Trident richiedono questi plugin di volume.	`hostPath, downwardAPI, projected, emptyDir`

Note legali

Le note legali forniscono accesso a dichiarazioni di copyright, marchi commerciali, brevetti e altro ancora.

Copyright

"https://www.netapp.com/company/legal/copyright/"

Marchi

NETAPP, il logo NETAPP e i marchi elencati nella pagina Marchi NetApp sono marchi di NetApp, Inc. Altri nomi di aziende e prodotti possono essere marchi dei rispettivi proprietari.

"https://www.netapp.com/company/legal/trademarks/"

Brevetti

Un elenco aggiornato dei brevetti di proprietà di NetApp è disponibile all'indirizzo:

https://www.netapp.com/pdf.html?item=/media/11887-patentspage.pdf

Politica sulla riservatezza

"https://www.netapp.com/company/legal/privacy-policy/"

Open source

È possibile rivedere i diritti d'autore e le licenze di terze parti utilizzati nel software NetApp per Trident nel file di avvisi per ogni versione all'indirizzo https://github.com/NetApp/trident/ .

Informazioni sul copyright

Informazioni sul marchio commerciale