Usar Trident

Trident

NetApp
January 15, 2026

Tabla de contenidos

Usar Trident

Preparar el nodo de trabajo

Todos los nodos de trabajo del clúster de Kubernetes deben poder montar los volúmenes que has aprovisionado para tus pods. Para preparar los nodos de trabajo, debe instalar las herramientas NFS, iSCSI, NVMe/TCP o FC según el controlador que haya seleccionado.

Seleccionar las herramientas adecuadas

Si utiliza una combinación de controladores, deberá instalar todas las herramientas necesarias para sus controladores. Las versiones recientes de Red Hat Enterprise Linux CoreOS (RHCOS) tienen las herramientas instaladas por defecto.

Herramientas NFS

"Instala las herramientas NFS"Si estás usando: `ontap-nas` , `ontap-nas-economy` , `ontap-nas-flexgroup` , `azure-netapp-files` , `gcp-cvs` .

herramientas iSCSI

"Instala las herramientas iSCSI"Si estás usando: `ontap-san` , `ontap-san-economy` , `solidfire-san` .

Herramientas NVMe

"Instala las herramientas NVMe"Si estas usando `ontap-san` para el protocolo de memoria no volátil express (NVMe) sobre TCP (NVMe/TCP).

> NetApp recomienda ONTAP 9.12 o posterior para NVMe/TCP.

Herramientas SCSI sobre FC

Referirse a"Formas de configurar hosts SAN FC y FC-NVMe" Para obtener más información sobre la configuración de sus hosts SAN FC y FC-NVMe.

"Instala las herramientas FC"Si estas usando `ontap-san` con sanType `fcp` (SCSI sobre FC).

Puntos a considerar: * SCSI sobre FC es compatible con entornos OpenShift y KubeVirt. * SCSI sobre FC no es compatible con Docker. * La autorreparación de iSCSI no es aplicable a SCSI sobre FC.

Descubrimiento de servicios de nodo

Trident intenta detectar automáticamente si el nodo puede ejecutar servicios iSCSI o NFS.

> La detección de servicios de nodo identifica los servicios descubiertos, pero no garantiza que los servicios estén configurados correctamente. Por el contrario, la ausencia de un servicio detectado no garantiza que el montaje del volumen vaya a fallar.

Revisar eventos

Trident crea eventos para que el nodo identifique los servicios descubiertos. Para revisar estos eventos, ejecute:

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

Revisión de servicios descubiertos

Trident identifica los servicios habilitados para cada nodo en el nodo CR de Trident . Para ver los servicios detectados, ejecute:

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

Volúmenes de NFS

Instala las herramientas NFS utilizando los comandos correspondientes a tu sistema operativo. Asegúrese de que el servicio NFS se inicie durante el arranque del sistema.

RHEL 8+

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

Ubuntu

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

Reinicie sus nodos de trabajo después de instalar las herramientas NFS para evitar fallos al conectar volúmenes a los contenedores.

volúmenes iSCSI

Trident puede establecer automáticamente una sesión iSCSI, escanear LUN, descubrir dispositivos multipath, formatearlos y montarlos en un pod.

capacidades de autorreparación de iSCSI

Para los sistemas ONTAP , Trident ejecuta la autorreparación iSCSI cada cinco minutos para:

1. **Identificar** el estado de sesión iSCSI deseado y el estado de sesión iSCSI actual.
2. **Compara** el estado deseado con el estado actual para identificar las reparaciones necesarias. Trident determina las prioridades de reparación y cuándo anticiparse a las reparaciones.
3. **Realizar las reparaciones** necesarias para devolver el estado actual de la sesión iSCSI al estado deseado.

Los registros de actividad de autocuración se encuentran en el `trident-main` contenedor en el pod Daemonset correspondiente. Para ver los registros, debe haber configurado `debug` a "verdadero" durante la instalación de Trident .

Las capacidades de autorreparación de Trident iSCSI pueden ayudar a prevenir:

- Sesiones iSCSI obsoletas o defectuosas que podrían producirse tras un problema de conectividad de red. En caso de una sesión inactiva, Trident espera siete minutos antes de cerrar sesión para restablecer la conexión con un portal.

> Por ejemplo, si los secretos CHAP se rotaran en el controlador de almacenamiento y la red perdiera la conectividad, los secretos CHAP antiguos (*obsoletos*) podrían persistir. La autocuración puede reconocer esto y restablecer automáticamente la sesión para aplicar los secretos CHAP actualizados.

- Faltan sesiones iSCSI
- LUN faltantes

Puntos a considerar antes de actualizar Trident

- Si solo se utilizan grupos de información por nodo (introducidos en la versión 23.04+), la autocuración de iSCSI iniciará nuevos escaneos SCSI para todos los dispositivos en el bus SCSI.
- Si solo se utilizan igroups con ámbito de backend (obsoletos a partir de la versión 23.04), la autocuración de iSCSI iniciará nuevos escaneos SCSI para los ID de LUN exactos en el bus SCSI.
- Si se utiliza una combinación de igroups por nodo e igroups con ámbito de backend, la autorreparación de iSCSI iniciará nuevos escaneos SCSI para los ID de LUN exactos en el bus SCSI.

Instala las herramientas iSCSI

Instale las herramientas iSCSI utilizando los comandos correspondientes a su sistema operativo.

Antes de empezar

- Cada nodo del clúster de Kubernetes debe tener un IQN único. **Este es un requisito previo necesario**.
- Si utiliza RHCOS versión 4.5 o posterior, u otra distribución de Linux compatible con RHEL, con el `solidfire-san` En el controlador y Element OS 12.5 o anterior, asegúrese de que el algoritmo de autenticación CHAP esté configurado en MD5 en `/etc/iscsi/iscsid.conf` Los algoritmos CHAP seguros y compatibles con FIPS SHA1, SHA-256 y SHA3-256 están disponibles con Element 12.7.

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

- Cuando utilice nodos de trabajo que ejecuten RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) con volúmenes persistentes iSCSI, especifique el `discard` mountOption en la StorageClass para realizar la recuperación de espacio en línea. Referirse a "Documentación de Red Hat" .
- Asegúrese de haber actualizado a la última versión de `multipath-tools` .

RHEL 8+

1. Instale los siguientes paquetes del sistema:

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

2. Compruebe que la versión de iscsi-initiator-utils sea la 6.2.0.874-2.el7 o posterior:

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

3. Configurar el escaneo en modo manual:

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

4. Habilitar rutas múltiples:

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

> Asegurar `/etc/multipath.conf` contiene `find_multipaths no` bajo `defaults` .

5. Asegúrese de que `iscsid` y `multipathd` están en funcionamiento:

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

6. Habilitar e iniciar `iscsi` :

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

Ubuntu

1. Instale los siguientes paquetes del sistema:

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

2. Compruebe que la versión de open-iscsi sea 2.0.874-5ubuntu2.10 o posterior (para bionic) o 2.0.874-7.1ubuntu6.1 o posterior (para focal):

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

3. Configurar el escaneo en modo manual:

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

4. Habilitar rutas múltiples:

```
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
```

> Asegurar `/etc/multipath.conf` contiene `find_multipaths no` bajo `defaults` .

5. Asegúrese de que `open-iscsi` y `multipath-tools` están habilitados y en funcionamiento:

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

> Para Ubuntu 18.04, debe descubrir los puertos de destino con `iscsiadm` antes de comenzar `open-iscsi` para que se inicie el demonio iSCSI. También puedes modificar el `iscsi` servicio para comenzar `iscsid` automáticamente.

Configurar o deshabilitar la autorreparación de iSCSI

Puede configurar los siguientes ajustes de autorreparación de Trident iSCSI para corregir sesiones obsoletas:

- **Intervalo de autocuración iSCSI**: Determina la frecuencia con la que se invoca la autocuración iSCSI (predeterminado: 5 minutos). Puedes configurarlo para que se ejecute con mayor frecuencia estableciendo un número menor o con menor frecuencia estableciendo un número mayor.

Configurar el intervalo de autocuración de iSCSI en 0 detiene por completo la autocuración de iSCSI. No recomendamos deshabilitar la autorreparación de iSCSI; solo debe deshabilitarse en ciertos escenarios cuando la autorreparación de iSCSI no funciona como se espera o con fines de depuración.

- **Tiempo de espera de autocuración de iSCSI**: Determina la duración que la autocuración de iSCSI espera antes de cerrar la sesión de una sesión defectuosa e intentar iniciar sesión de nuevo (predeterminado: 7 minutos). Puede configurarlo con un número mayor para que las sesiones identificadas como no saludables tengan que esperar más tiempo antes de cerrarse y luego se intente volver a iniciar sesión, o con un número menor para cerrar sesión e iniciarla antes.

Timón

Para configurar o cambiar los ajustes de autorreparación de iSCSI, pase el `iscsiSelfHealingInterval` y `iscsiSelfHealingWaitTime` parámetros durante la instalación o actualización de Helm.

El siguiente ejemplo establece el intervalo de autorreparación de iSCSI en 3 minutos y el tiempo de espera de autorreparación en 6 minutos:

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

tridentctl

Para configurar o cambiar los ajustes de autorreparación de iSCSI, pase el `iscsi-self-healing-interval` y `iscsi-self-healing-wait-time` parámetros durante la instalación o actualización de tridentctl.

El siguiente ejemplo establece el intervalo de autorreparación de iSCSI en 3 minutos y el tiempo de espera de autorreparación en 6 minutos:

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

volúmenes NVMe/TCP

Instala las herramientas NVMe utilizando los comandos correspondientes a tu sistema operativo.

- NVMe requiere RHEL 9 o posterior.
- Si la versión del kernel de su nodo de Kubernetes es demasiado antigua o si el paquete NVMe no está disponible para su versión del kernel, es posible que deba actualizar la versión del kernel de su nodo a una que incluya el paquete 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
```

Verificar la instalación

Tras la instalación, verifique que cada nodo del clúster de Kubernetes tenga un NQN único mediante el siguiente comando:

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

Trident modifica el `ctrl_device_tmo` Valor para garantizar que NVMe no abandone la ruta si falla. No cambie esta configuración.

SCSI sobre volúmenes FC

Ahora puede utilizar el protocolo Fibre Channel (FC) con Trident para aprovisionar y administrar recursos de almacenamiento en el sistema ONTAP .

Prerrequisitos

Configure los ajustes de red y nodo necesarios para FC.

Configuración de red

1. Obtén el WWPN de las interfaces de destino. Referirse a "Mostrar interfaz de red" Para más información.
2. Obtenga el WWPN para las interfaces en el iniciador (Host).

   Consulte las utilidades correspondientes del sistema operativo anfitrión.

3. Configure el zonificado en el switch FC utilizando los WWPN del host y del destino.

   Consulte la documentación del proveedor del conmutador correspondiente para obtener información.

   Consulte la siguiente documentación de ONTAP para obtener más detalles:

   - "Descripción general de la zonificación de Fibre Channel y FCoE"
   - "Formas de configurar hosts SAN FC y FC-NVMe"

Instala las herramientas FC

Instala las herramientas FC utilizando los comandos correspondientes a tu sistema operativo.

- Cuando utilice nodos de trabajo que ejecuten RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) con PV FC, especifique el `discard` mountOption en la StorageClass para realizar la recuperación de espacio en línea. Referirse a "Documentación de Red Hat" .

RHEL 8+

1. Instale los siguientes paquetes del sistema:

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

2. Habilitar rutas múltiples:

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

> Asegurar `/etc/multipath.conf` contiene `find_multipaths no` bajo `defaults` .

3. Asegúrese de que `multipathd` está en funcionamiento:

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

Ubuntu

1. Instale los siguientes paquetes del sistema:

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

2. Habilitar rutas múltiples:

```
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
```

> Asegurar `/etc/multipath.conf` contiene `find_multipaths no` bajo `defaults` .

3. Asegúrese de que `multipath-tools` está habilitado y en funcionamiento:

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

Configurar y administrar backends

Configurar backends

Un backend define la relación entre Trident y un sistema de almacenamiento. Le indica a Trident cómo comunicarse con ese sistema de almacenamiento y cómo Trident aprovisionar volúmenes desde él.

Trident ofrece automáticamente pools de almacenamiento de backends que coinciden con los requisitos definidos por una clase de almacenamiento. Aprenda cómo configurar el backend de su sistema de almacenamiento.

- "Configurar un backend de Azure NetApp Files"
- "Configurar un backend de Google Cloud NetApp Volumes"
- "Configurar un Cloud Volumes Service para el backend de Google Cloud Platform"
- "Configure un backend de NetApp HCI o SolidFire"
- "Configure un backend con ONTAP o Cloud Volumes ONTAP NAS drivers"
- "Configure un backend con controladores SAN ONTAP o Cloud Volumes ONTAP."
- "Utilice Trident con Amazon FSx for NetApp ONTAP"

Azure NetApp Files

Configurar un backend de Azure NetApp Files

Puede configurar Azure NetApp Files como backend para Trident. Puede conectar volúmenes NFS y SMB utilizando un backend de Azure NetApp Files . Trident también admite la administración de credenciales mediante identidades administradas para clústeres de Azure Kubernetes Services (AKS).

Detalles del controlador de Azure NetApp Files

Trident proporciona los siguientes controladores de almacenamiento de Azure NetApp Files para comunicarse con el clúster. Los modos de acceso compatibles son: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Conductor	Protocolo	modo de volumen	Modos de acceso compatibles	Sistemas de archivos compatibles
`azure-netapp-files`	NFS SMB	Sistema de archivos	RWO, ROX, RWX, RWOP	`nfs`, `smb`

Consideraciones

- El servicio Azure NetApp Files no admite volúmenes inferiores a 50 GiB. Trident crea automáticamente volúmenes de 50 GiB si se solicita un volumen más pequeño.
- Trident solo admite volúmenes SMB montados en pods que se ejecutan en nodos Windows.

Identidades administradas para AKS

Trident apoya"identidades gestionadas" para clústeres de Azure Kubernetes Services. Para aprovechar la gestión simplificada de credenciales que ofrecen las identidades gestionadas, debe tener:

- Un clúster de Kubernetes implementado mediante AKS
- Identidades administradas configuradas en el clúster de Kubernetes de AKS
- Trident instalado que incluye el `cloudProvider` para especificar `"Azure"` .

Operador de Trident

Para instalar Trident usando el operador Trident , edite `tridentorchestrator_cr.yaml` para establecer `cloudProvider` a `"Azure"` . Por ejemplo:

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

Timón

El siguiente ejemplo instala conjuntos Trident `cloudProvider` a Azure usando la variable de entorno `$CP` :

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

<code>tridentctl</code>

El siguiente ejemplo instala Trident y configura `cloudProvider` bandera a `Azure` :

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

Identidad en la nube para AKS

La identidad en la nube permite que los pods de Kubernetes accedan a los recursos de Azure autenticándose como una identidad de carga de trabajo en lugar de proporcionar credenciales explícitas de Azure.

Para aprovechar la identidad en la nube en Azure, debe tener:

- Un clúster de Kubernetes implementado mediante AKS

- La identidad de la carga de trabajo y el emisor OIDC están configurados en el clúster de Kubernetes de AKS.
- Trident instalado que incluye el `cloudProvider` para especificar `"Azure"` y `cloudIdentity` especificar la identidad de la carga de trabajo

Operador de Trident

Para instalar Trident usando el operador Trident , edite `tridentorchestrator_cr.yaml` para establecer `cloudProvider` a `"Azure"` y establecer `cloudIdentity` a `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx` .

Por ejemplo:

```
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
```

Timón

Establezca los valores para las marcas **cloud-provider (CP)** y **cloud-identity (CI)** utilizando las siguientes variables de entorno:

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

El siguiente ejemplo instala Trident y lo configura. `cloudProvider` a Azure usando la variable de entorno `$CP` y establece el `cloudIdentity` utilizando la variable de entorno `$CI` :

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

<code>tridentctl</code>

Configure los valores para las marcas **proveedor de nube** e **identidad de nube** utilizando las siguientes variables de entorno:

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

El siguiente ejemplo instala Trident y configura `cloud-provider` bandera a `$CP` , y `cloud-identity` a `$CI` :

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

Prepárese para configurar un backend de Azure NetApp Files.

Antes de poder configurar su backend de Azure NetApp Files , debe asegurarse de que se cumplen los siguientes requisitos.

Requisitos previos para volúmenes NFS y SMB

Si utiliza Azure NetApp Files por primera vez o en una nueva ubicación, se requiere una configuración inicial para configurar Azure NetApp Files y crear un volumen NFS. Referirse a "Azure: Configurar Azure NetApp Files y crear un volumen NFS" .

Para configurar y usar un "Azure NetApp Files" Para el backend, necesitas lo siguiente:

- `subscriptionID`, `tenantID` , `clientID` , `location` , y `clientSecret` son opcionales cuando se utilizan identidades administradas en un clúster de AKS.
- `tenantID`, `clientID` , y `clientSecret` son opcionales cuando se utiliza una identidad en la nube en un clúster de AKS.

- Un fondo común de capacidad. Referirse a"Microsoft: Crear un grupo de capacidad para Azure NetApp Files" .
- Una subred delegada a Azure NetApp Files. Referirse a"Microsoft: Delegar una subred a Azure NetApp Files" .
- `subscriptionID`desde una suscripción de Azure con Azure NetApp Files habilitado.
- `tenantID`, `clientID` , y `clientSecret` de un"Registro de la aplicación" en Azure Active Directory con permisos suficientes para el servicio Azure NetApp Files . El registro de la aplicación debe utilizar una de las siguientes opciones:
  - El rol de propietario o colaborador"predefinido por Azure" .
  - A"rol de colaborador personalizado" a nivel de suscripción(`assignableScopes`) con los siguientes permisos, que se limitan únicamente a lo que Trident requiere. Tras crear el rol personalizado,"Asigna el rol mediante el portal de Azure." .

Rol de colaborador personalizado

```
{
  "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": []
        }
      ]
    }
}
```

- El Azure `location` que contiene al menos uno "subred delegada" . A partir de Trident 22.01, el `location` El parámetro es un campo obligatorio en el nivel superior del archivo de configuración del backend. Los valores de ubicación especificados en los grupos virtuales se ignoran.
- Para usar `Cloud Identity` , obtén el `client ID` de un "identidad gestionada asignada por el usuario" y especifique ese ID en `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx` .

Requisitos adicionales para volúmenes de PYMES

Para crear un volumen SMB, debe tener:

- Active Directory configurado y conectado a Azure NetApp Files. Referirse a"Microsoft: Crear y administrar conexiones de Active Directory para Azure NetApp Files" .
- Un clúster de Kubernetes con un nodo controlador Linux y al menos un nodo de trabajo Windows que ejecuta Windows Server 2022. Trident solo admite volúmenes SMB montados en pods que se ejecutan en nodos Windows.
- Al menos un secreto de Trident que contenga sus credenciales de Active Directory para que Azure NetApp Files pueda autenticarse en Active Directory. Para generar secretos `smbcreds` :

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

- Un proxy CSI configurado como servicio de Windows. Para configurar un `csi-proxy` , consulte a"GitHub: Proxy CSI" o"GitHub: Proxy CSI para Windows" para nodos de Kubernetes que se ejecutan en Windows.

Opciones y ejemplos de configuración del backend de Azure NetApp Files

Obtenga información sobre las opciones de configuración de backend NFS y SMB para

Azure NetApp Files y revise los ejemplos de configuración.

Opciones de configuración del backend

Trident utiliza su configuración de backend (subred, red virtual, nivel de servicio y ubicación) para crear volúmenes de Azure NetApp Files en grupos de capacidad que estén disponibles en la ubicación solicitada y que coincidan con el nivel de servicio y la subred solicitados.

* A partir de la versión 25.06 de NetApp Trident , los grupos de capacidad de QoS manuales se admiten como versión preliminar técnica.*

Los backends de Azure NetApp Files proporcionan estas opciones de configuración.

Parámetro	Descripción	Por defecto
`version`		Siempre 1
`storageDriverName`	Nombre del controlador de almacenamiento	"archivos de azure-netapp"
`backendName`	Nombre personalizado o el backend de almacenamiento	Nombre del conductor + "_" + caracteres aleatorios
`subscriptionID`	El ID de suscripción de su suscripción de Azure. Opcional cuando las identidades administradas están habilitadas en un clúster de AKS.	
`tenantID`	El ID de inquilino de un registro de aplicación es opcional cuando se utilizan identidades administradas o identidades en la nube en un clúster de AKS.	
`clientID`	El ID de cliente de un registro de aplicación es opcional cuando se utilizan identidades administradas o identidades en la nube en un clúster de AKS.	
`clientSecret`	El secreto del cliente de un registro de aplicación es opcional cuando se utilizan identidades administradas o identidades en la nube en un clúster de AKS.	
`serviceLevel`	Uno de `Standard` , `Premium` , o `Ultra`	"" (aleatorio)
`location`	Nombre de la ubicación de Azure donde se crearán los nuevos volúmenes. Opcional cuando las identidades administradas están habilitadas en un clúster de AKS.	

Parámetro	Descripción	Por defecto
`resourceGroups`	Lista de grupos de recursos para filtrar los recursos descubiertos	"[]" (sin filtro)
`netappAccounts`	Lista de cuentas de NetApp para filtrar los recursos detectados	"[]" (sin filtro)
`capacityPools`	Lista de grupos de capacidad para filtrar los recursos descubiertos	"[]" (sin filtro, aleatorio)
`virtualNetwork`	Nombre de una red virtual con una subred delegada	""
`subnet`	Nombre de una subred delegada a `Microsoft.Netapp/volumes`	""
`networkFeatures`	Conjunto de características de VNet para un volumen, puede ser `Basic` o `Standard` . La función de red no está disponible en todas las regiones y puede que sea necesario habilitarla mediante una suscripción. Especificar `networkFeatures` Cuando la funcionalidad no está habilitada, se produce un fallo en el aprovisionamiento de volúmenes.	""
`nfsMountOptions`	Control preciso de las opciones de montaje NFS. Se ignora para volúmenes SMB. Para montar volúmenes utilizando NFS versión 4.1, incluya `nfsvers=4` en la lista de opciones de montaje delimitadas por comas, seleccione NFS v4.1. Las opciones de montaje establecidas en la definición de una clase de almacenamiento anulan las opciones de montaje establecidas en la configuración del backend.	"nfsvers=3"
`limitVolumeSize`	Fallará el aprovisionamiento si el tamaño del volumen solicitado supera este valor.	" (no se aplica por defecto)
`debugTraceFlags`	Indicadores de depuración para usar al solucionar problemas. Ejemplo, `\{"api": false, "method": true, "discovery": true}` . No utilice esta función a menos que esté solucionando problemas y necesite un registro detallado.	nulo

Parámetro	Descripción	Por defecto
`nasType`	Configure la creación de volúmenes NFS o SMB. Las opciones son `nfs` , `smb` o nulo. Si se establece en nulo, se utilizarán volúmenes NFS por defecto.	`nfs`
`supportedTopologies`	Representa una lista de regiones y zonas compatibles con este backend. Para obtener más información, consulte"Utilizar la topología CSI" .	
`qosType`	Representa el tipo de QoS: Automático o Manual. **Vista previa técnica de Trident 25.06**	Auto
`maxThroughput`	Establece el rendimiento máximo permitido en MiB/seg. Compatible solo con grupos de capacidad de QoS manuales. **Vista previa técnica de Trident 25.06**	`4 MiB/sec`

Para obtener más información sobre las funciones de red, consulte"Configurar las características de red para un volumen de Azure NetApp Files" .

Permisos y recursos necesarios

Si recibe un error de "No se encontraron grupos de capacidad" al crear un PVC, es probable que el registro de su aplicación no tenga asociados los permisos y recursos necesarios (subred, red virtual, grupo de capacidad). Si la depuración está habilitada, Trident registrará los recursos de Azure detectados cuando se cree el backend. Verifique que se esté utilizando el rol adecuado.

Los valores para `resourceGroups` , `netappAccounts` , `capacityPools` , `virtualNetwork` , y `subnet` se pueden especificar utilizando nombres cortos o nombres completos. En la mayoría de los casos se recomienda utilizar nombres completos, ya que los nombres cortos pueden coincidir con varios recursos que tengan el mismo nombre.

El `resourceGroups` , `netappAccounts` , y `capacityPools` Los valores son filtros que restringen el conjunto de recursos descubiertos a aquellos disponibles para este backend de almacenamiento y pueden especificarse en cualquier combinación. Los nombres completos siguen este formato:

Tipo	Formato
Grupo de recursos	<grupo de recursos>
cuenta de NetApp	<grupo de recursos>/<cuenta de NetApp>
reserva de capacidad	<grupo de recursos>/<cuenta de NetApp>/<grupo de capacidad>
Red virtual	<grupo de recursos>/<red virtual>
Subred	<grupo de recursos>/<red virtual>/<subred>

Aprovisionamiento de volumen

Puede controlar el aprovisionamiento de volúmenes predeterminado especificando las siguientes opciones en una sección especial del archivo de configuración. Referirse a Configuraciones de ejemplo Para más detalles.

Parámetro	Descripción	Por defecto
`exportRule`	Reglas de exportación para nuevos volúmenes. `exportRule` Debe ser una lista separada por comas de cualquier combinación de direcciones IPv4 o subredes IPv4 en notación CIDR. Se ignora para volúmenes SMB.	"0.0.0.0/0"
`snapshotDir`	Controla la visibilidad del directorio .snapshot	"verdadero" para NFSv4, "falso" para NFSv3
`size`	El tamaño predeterminado de los nuevos volúmenes	"100G"
`unixPermissions`	Los permisos Unix de los nuevos volúmenes (4 dígitos octales). Se ignora para volúmenes SMB.	" (función de vista previa, requiere inclusión en la lista blanca de la suscripción)

Configuraciones de ejemplo

Los siguientes ejemplos muestran configuraciones básicas que dejan la mayoría de los parámetros con sus valores predeterminados. Esta es la forma más sencilla de definir un backend.

Configuración mínima

Esta es la configuración mínima absoluta del backend. Con esta configuración, Trident descubre todas sus cuentas de NetApp , grupos de capacidad y subredes delegadas a Azure NetApp Files en la ubicación configurada, y coloca nuevos volúmenes en uno de esos grupos y subredes de forma aleatoria. Porque `nasType` se omite, el `nfs` Se aplicará la configuración predeterminada y el backend aprovisionará los volúmenes NFS.

Esta configuración es ideal cuando estás empezando a usar Azure NetApp Files y haciendo pruebas, pero en la práctica querrás proporcionar un alcance adicional para los volúmenes que aprovisiones.

```
---
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
```

Identidades administradas para AKS

Esta configuración de backend omite `subscriptionID`, `tenantID`, `clientID`, y `clientSecret`, que son opcionales cuando se utilizan identidades administradas.

```
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
```

Identidad en la nube para AKS

Esta configuración de backend omite `tenantID` , `clientID` , y `clientSecret` , que son opcionales al usar una identidad en la nube.

```
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
```

Configuración específica del nivel de servicio con filtros de capacidad.

Esta configuración de backend coloca volúmenes en Azure `eastus` ubicación en un `Ultra` reserva de capacidad. Trident descubre automáticamente todas las subredes delegadas a Azure NetApp Files en esa ubicación y coloca un nuevo volumen en una de ellas de forma aleatoria.

```
---
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
```

Ejemplo de backend con grupos de capacidad de QoS manuales

Esta configuración de backend coloca volúmenes en Azure `eastus` Ubicación con pools de capacidad QoS manuales. **Vista previa técnica en 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
```

Configuración avanzada

Esta configuración de backend reduce aún más el alcance de la ubicación de volúmenes a una sola subred y también modifica algunos valores predeterminados de aprovisionamiento de volúmenes.

```
---
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"
```

Configuración de grupo virtual

Esta configuración de backend define múltiples grupos de almacenamiento en un solo archivo. Esto resulta útil cuando se tienen varios grupos de capacidad que admiten diferentes niveles de servicio y se desea crear clases de almacenamiento en Kubernetes que los representen. Se utilizaron etiquetas de piscinas virtuales para diferenciar las piscinas en función de `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
```

Configuración de topologías admitidas

Trident facilita el aprovisionamiento de volúmenes para cargas de trabajo en función de regiones y zonas de disponibilidad. El `supportedTopologies` Este bloque en la configuración del backend se utiliza para proporcionar una lista de regiones y zonas por backend. Los valores de región y zona especificados aquí deben coincidir con los valores de región y zona de las etiquetas de cada nodo del clúster de Kubernetes. Estas regiones y zonas representan la lista de valores permitidos que se pueden proporcionar en una clase de almacenamiento. Para las clases de almacenamiento que contienen un subconjunto de las regiones y zonas proporcionadas en un backend, Trident crea volúmenes en la región y zona mencionadas. Para obtener más información, consulte"Utilizar la topología 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
```

Definiciones de clases de almacenamiento

La siguiente `StorageClass` Las definiciones se refieren a los grupos de almacenamiento mencionados anteriormente.

Definiciones de ejemplo que utilizan `parameter.selector` campo

Usando `parameter.selector` Puedes especificarlo para cada uno. `StorageClass` el grupo virtual que se utiliza para alojar un volumen. El volumen tendrá los aspectos definidos en el pool elegido.

```
---
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
```

Definiciones de ejemplo para volúmenes SMB

Usando `nasType` , `node-stage-secret-name` , y `node-stage-secret-namespace` , puede especificar un volumen SMB y proporcionar las credenciales de Active Directory necesarias.

Configuración básica en el espacio de nombres predeterminado

```
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"
```

Utilizar diferentes secretos por espacio de nombres

```
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}
```

Utilizando diferentes secretos por volumen

```
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`Filtros para pools que admiten volúmenes SMB. `nasType: nfs` o `nasType: null` Filtros para pools NFS.

Crea el backend

Después de crear el archivo de configuración del backend, ejecute el siguiente comando:

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

Si falla la creación del backend, algo falla en la configuración del backend. Puedes consultar los registros para determinar la causa ejecutando el siguiente comando:

```
tridentctl logs
```

Después de identificar y corregir el problema con el archivo de configuración, puede volver a ejecutar el comando de creación.

Google Cloud NetApp Volumes

Configurar un backend de Google Cloud NetApp Volumes

Ahora puedes configurar Google Cloud NetApp Volumes como backend para Trident. Puedes conectar volúmenes NFS y SMB utilizando un backend de Google Cloud NetApp Volumes .

Detalles del controlador de Google Cloud NetApp Volumes

Trident proporciona el `google-cloud-netapp-volumes` controlador para comunicarse con el clúster. Los modos de acceso compatibles son: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Conductor	Protocolo	modo de volumen	Modos de acceso compatibles	Sistemas de archivos compatibles
`google-cloud-netapp-volumes`	NFS SMB	Sistema de archivos	RWO, ROX, RWX, RWOP	`nfs`, `smb`

Identidad en la nube para GKE

La identidad en la nube permite que los pods de Kubernetes accedan a los recursos de Google Cloud autenticándose como una identidad de carga de trabajo en lugar de proporcionar credenciales explícitas de Google Cloud.

Para aprovechar la identidad en la nube de Google Cloud, debes tener:

- Un clúster de Kubernetes desplegado utilizando GKE.
- La identidad de la carga de trabajo está configurada en el clúster de GKE y el servidor de metadatos de GKE está configurado en los grupos de nodos.

- Una cuenta de servicio de GCP con el rol de administrador de Google Cloud NetApp Volumes (roles/netapp.admin) o un rol personalizado.
- Trident instalado incluye cloudProvider para especificar "GCP" y cloudIdentity para especificar la nueva cuenta de servicio de GCP. A continuación se muestra un ejemplo.

Operador de Trident

Para instalar Trident usando el operador Trident , edite `tridentorchestrator_cr.yaml` para establecer `cloudProvider` a `"GCP"` y establecer `cloudIdentity` a `iam.gke.io/gcp-service-account: cloudvolumes-admin-sa@mygcpproject.iam.gserviceaccount.com` .

Por ejemplo:

```
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'
```

Timón

Establezca los valores para las marcas **cloud-provider (CP)** y **cloud-identity (CI)** utilizando las siguientes variables de entorno:

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

El siguiente ejemplo instala Trident y lo configura. `cloudProvider` a GCP utilizando la variable de entorno `$CP` y establece el `cloudIdentity` utilizando la variable de entorno `$ANNOTATION` :

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

<code>tridentctl</code>

Configure los valores para las marcas **proveedor de nube** e **identidad de nube** utilizando las siguientes variables de entorno:

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

El siguiente ejemplo instala Trident y configura `cloud-provider` bandera a `$CP` , y `cloud-identity` a `$ANNOTATION` :

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

Prepárate para configurar un backend de Google Cloud NetApp Volumes.

Antes de poder configurar su backend de Google Cloud NetApp Volumes , debe asegurarse de que se cumplen los siguientes requisitos.

Requisitos previos para volúmenes NFS

Si utiliza Google Cloud NetApp Volumes por primera vez o en una nueva ubicación, se requiere una configuración inicial para configurar Google Cloud NetApp Volumes y crear un volumen NFS. Referirse a"Antes de empezar" .

Asegúrese de tener lo siguiente antes de configurar el backend de Google Cloud NetApp Volumes :

- Una cuenta de Google Cloud configurada con el servicio Google Cloud NetApp Volumes . Referirse a"Google Cloud NetApp Volumes" .
- Número de proyecto de tu cuenta de Google Cloud. Referirse a"Identificación de proyectos" .
- Una cuenta de servicio de Google Cloud con el administrador de volúmenes de NetApp(`roles/netapp.admin`) role. Referirse a"Roles y permisos de gestión de identidades y accesos" .
- Archivo de clave API para su cuenta de GCNV. Referirse a"Cree una clave de cuenta de servicio"
- Un grupo de almacenamiento. Referirse a"Descripción general de los grupos de almacenamiento" .

Para obtener más información sobre cómo configurar el acceso a Google Cloud NetApp Volumes, consulte"Configurar el acceso a Google Cloud NetApp Volumes" .

Opciones y ejemplos de configuración del backend de Google Cloud NetApp Volumes

Aprenda sobre las opciones de configuración de backend para Google Cloud NetApp Volumes y revise ejemplos de configuración.

Opciones de configuración del backend

Cada backend aprovisiona volúmenes en una única región de Google Cloud. Para crear volúmenes en otras regiones, puede definir backends adicionales.

Parámetro	Descripción	Por defecto
`version`		Siempre 1
`storageDriverName`	Nombre del controlador de almacenamiento	El valor de `storageDriverName` debe especificarse como "google-cloud-netapp-volumes".

Parámetro	Descripción	Por defecto
`backendName`	(Opcional) Nombre personalizado del backend de almacenamiento	Nombre del controlador + "_" + parte de la clave API
`storagePools`	Parámetro opcional utilizado para especificar los grupos de almacenamiento para la creación de volúmenes.	
`projectNumber`	Número de proyecto de la cuenta de Google Cloud. El valor se encuentra en la página principal del portal de Google Cloud.	
`location`	La ubicación de Google Cloud donde Trident crea volúmenes GCNV. Al crear clústeres de Kubernetes entre regiones, los volúmenes creados en un `location` Puede utilizarse en cargas de trabajo programadas en nodos de varias regiones de Google Cloud. El tráfico interregional conlleva un coste adicional.	
`apiKey`	clave API para la cuenta de servicio de Google Cloud con la `netapp.admin` role. Incluye el contenido en formato JSON del archivo de clave privada de una cuenta de servicio de Google Cloud (copiado textualmente en el archivo de configuración del backend). El `apiKey` Debe incluir pares clave-valor para las siguientes claves: `type` , `project_id` , `client_email` , `client_id` , `auth_uri` , `token_uri` , `auth_provider_x509_cert_url` , y `client_x509_cert_url` .	
`nfsMountOptions`	Control preciso de las opciones de montaje NFS.	"nfsvers=3"
`limitVolumeSize`	Fallará el aprovisionamiento si el tamaño de volumen solicitado supera este valor.	" (no se aplica por defecto)
`serviceLevel`	El nivel de servicio de un grupo de almacenamiento y sus volúmenes. Los valores son `flex` , `standard` , `premium` , o `extreme` .	
`labels`	Conjunto de etiquetas arbitrarias con formato JSON para aplicar a los volúmenes	""
`network`	Red de Google Cloud utilizada para los volúmenes de GCNV.	
`debugTraceFlags`	Indicadores de depuración para usar al solucionar problemas. Ejemplo, `{"api":false, "method":true}` . No utilice esta función a menos que esté solucionando problemas y necesite un registro detallado.	nulo
`nasType`	Configure la creación de volúmenes NFS o SMB. Las opciones son `nfs` , `smb` o nulo. Si se establece en nulo, se utilizarán volúmenes NFS por defecto.	`nfs`

Parámetro	Descripción	Por defecto
`supportedTopologies`	Representa una lista de regiones y zonas compatibles con este backend. Para obtener más información, consulte"Utilizar la topología CSI" . Por ejemplo: `supportedTopologies: - topology.kubernetes.io/region: asia-east1 topology.kubernetes.io/zone: asia-east1-a`	

Opciones de aprovisionamiento de volumen

Puedes controlar el aprovisionamiento de volúmenes predeterminado en el `defaults` sección del archivo de configuración.

Parámetro	Descripción	Por defecto
`exportRule`	Las reglas de exportación para nuevos volúmenes. Debe ser una lista separada por comas de cualquier combinación de direcciones IPv4.	"0.0.0.0/0"
`snapshotDir`	Acceso a la `.snapshot` directorio	"verdadero" para NFSv4, "falso" para NFSv3
`snapshotReserve`	Porcentaje de volumen reservado para instantáneas	"" (aceptar valor predeterminado de 0)
`unixPermissions`	Los permisos Unix de los nuevos volúmenes (4 dígitos octales).	""

Configuraciones de ejemplo

Los siguientes ejemplos muestran configuraciones básicas que dejan la mayoría de los parámetros con sus valores predeterminados. Esta es la forma más sencilla de definir un backend.

Configuración mínima

Esta es la configuración mínima absoluta del backend. Con esta configuración, Trident descubre todos los grupos de almacenamiento delegados a Google Cloud NetApp Volumes en la ubicación configurada y coloca nuevos volúmenes en uno de esos grupos de forma aleatoria. Porque `nasType` se omite, el `nfs` Se aplicará la configuración predeterminada y el backend aprovisionará los volúmenes NFS.

Esta configuración es ideal cuando estás empezando a usar Google Cloud NetApp Volumes y haciendo pruebas, pero en la práctica lo más probable es que necesites proporcionar un alcance adicional para los volúmenes que aprovisiones.

```
---
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
```

Configuración para volúmenes 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
```

Configuración 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
```

Configuración de grupo virtual

Esta configuración de backend define múltiples pools virtuales en un solo archivo. Los grupos virtuales se definen en el `storage` sección. Son útiles cuando se tienen varios grupos de almacenamiento que admiten diferentes niveles de servicio y se desea crear clases de almacenamiento en Kubernetes que los representen. Las etiquetas de los grupos virtuales se utilizan para diferenciar los grupos. Por ejemplo, en el siguiente ejemplo. `performance` etiqueta y `serviceLevel` El tipo se utiliza para diferenciar los grupos virtuales.

También puede establecer algunos valores predeterminados que se aplicarán a todos los grupos virtuales y sobrescribir los valores predeterminados para grupos virtuales individuales. En el siguiente ejemplo, `snapshotReserve` y `exportRule` sirven como valores predeterminados para todos los grupos virtuales.

Para obtener más información, consulte"piscinas virtuales" .

```
---
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
```

Identidad en la nube para 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
```

Configuración de topologías admitidas

Trident facilita el aprovisionamiento de volúmenes para cargas de trabajo en función de regiones y zonas de disponibilidad. El `supportedTopologies` Este bloque en la configuración del backend se utiliza para proporcionar una lista de regiones y zonas por backend. Los valores de región y zona especificados aquí deben coincidir con los valores de región y zona de las etiquetas de cada nodo del clúster de Kubernetes. Estas regiones y zonas representan la lista de valores permitidos que se pueden proporcionar en una clase de almacenamiento. Para las clases de almacenamiento que contienen un subconjunto de las regiones y zonas proporcionadas en un backend, Trident crea volúmenes en la región y zona mencionadas. Para obtener más información, consulte"Utilizar la topología 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
```

¿Que sigue?

Después de crear el archivo de configuración del backend, ejecute el siguiente comando:

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

Para verificar que el backend se ha creado correctamente, ejecute el siguiente comando:

```
kubectl get tridentbackendconfig

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

Si falla la creación del backend, algo falla en la configuración del backend. Puedes describir el backend usando el `kubectl get tridentbackendconfig <backend-name>` Ejecute el siguiente comando o consulte los registros para determinar la causa:

```
tridentctl logs
```

Una vez que haya identificado y corregido el problema con el archivo de configuración, puede eliminar el backend y volver a ejecutar el comando de creación.

Definiciones de clases de almacenamiento

Lo siguiente es un esquema básico `StorageClass` Definición que hace referencia al backend mencionado anteriormente.

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

Ejemplos de definiciones utilizando el `parameter.selector` campo:

Usando `parameter.selector` Puedes especificarlo para cada uno. `StorageClass` el"piscina virtual" que se utiliza para alojar un volumen. El volumen tendrá los aspectos definidos en el pool elegido.

```
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
```

Para obtener más detalles sobre las clases de almacenamiento, consulte"Crear una clase de almacenamiento" .

Definiciones de ejemplo para volúmenes SMB

Usando `nasType` , `node-stage-secret-name` , y `node-stage-secret-namespace` , puede especificar un volumen SMB y proporcionar las credenciales de Active Directory necesarias. Se puede utilizar cualquier usuario/contraseña de Active Directory con cualquier permiso o sin permisos para el secreto de etapa del nodo.

Configuración básica en el espacio de nombres predeterminado

```
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"
```

Utilizar diferentes secretos por espacio de nombres

```
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}
```

Utilizando diferentes secretos por volumen

```
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`Filtros para pools que admiten volúmenes SMB. `nasType: nfs` o `nasType: null` Filtros para pools NFS.

Ejemplo de definición de PVC

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

Para verificar si el PVC está enlazado, ejecute el siguiente 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
```

Configurar un Cloud Volumes Service para el backend de Google Cloud

Aprenda a configurar NetApp Cloud Volumes Service para Google Cloud como backend para su instalación de Trident utilizando las configuraciones de ejemplo proporcionadas.

Detalles del controlador de Google Cloud

Trident proporciona el `gcp-cvs` controlador para comunicarse con el clúster. Los modos de acceso compatibles son: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Conductor	Protocolo	modo de volumen	Modos de acceso compatibles	Sistemas de archivos compatibles
`gcp-cvs`	Sistema Nacional de Archivos	Sistema de archivos	RWO, ROX, RWX, RWOP	`nfs`

Obtén más información sobre la compatibilidad de Trident con Cloud Volumes Service para Google Cloud.

Trident puede crear volúmenes de Cloud Volumes Service en uno de dos"tipos de servicio" :

- **CVS-Performance**: El tipo de servicio Trident predeterminado. Este tipo de servicio optimizado para el rendimiento es el más adecuado para cargas de trabajo de producción que valoran el rendimiento. El tipo de servicio CVS-Performance es una opción de hardware que admite volúmenes con un tamaño mínimo de 100 GiB. Puedes elegir uno de"tres niveles de servicio" :
  - `standard`
  - `premium`
  - `extreme`
- **CVS**: El tipo de servicio CVS proporciona una alta disponibilidad zonal con niveles de rendimiento limitados a moderados. El tipo de servicio CVS es una opción de software que utiliza grupos de almacenamiento para admitir volúmenes de tan solo 1 GiB. El grupo de almacenamiento puede contener hasta 50 volúmenes, donde todos los volúmenes comparten la capacidad y el rendimiento del grupo. Puedes elegir uno de"dos niveles de servicio" :
  - `standardsw`
  - `zoneredundantstandardsw`

Lo que necesitarás

Para configurar y usar el "Cloud Volumes Service para Google Cloud" Para el backend, necesitas lo siguiente:

- Una cuenta de Google Cloud configurada con el servicio NetApp Cloud Volumes Service.
- Número de proyecto de tu cuenta de Google Cloud
- cuenta de servicio de Google Cloud con la `netappcloudvolumes.admin` role
- Archivo de clave API para su cuenta de Cloud Volumes Service

Opciones de configuración del backend

Cada backend aprovisiona volúmenes en una única región de Google Cloud. Para crear volúmenes en otras regiones, puede definir backends adicionales.

Parámetro	Descripción	Por defecto
`version`		Siempre 1
`storageDriverName`	Nombre del controlador de almacenamiento	"gcp-cvs"
`backendName`	Nombre personalizado o el backend de almacenamiento	Nombre del controlador + "_" + parte de la clave API
`storageClass`	Parámetro opcional utilizado para especificar el tipo de servicio CVS. Usar `software` para seleccionar el tipo de servicio CVS. De lo contrario, Trident asume el tipo de servicio CVS-Performance.(`hardware`).	
`storagePools`	Solo servicio de tipo CVS. Parámetro opcional utilizado para especificar los grupos de almacenamiento para la creación de volúmenes.	

Parámetro	Descripción	Por defecto
`projectNumber`	Número de proyecto de la cuenta de Google Cloud. El valor se encuentra en la página principal del portal de Google Cloud.	
`hostProjectNumber`	Requerido si se utiliza una red VPC compartida. En este escenario, `projectNumber` es el proyecto de servicio, y `hostProjectNumber` es el proyecto anfitrión.	
`apiRegion`	La región de Google Cloud donde Trident crea volúmenes de Cloud Volumes Service . Al crear clústeres de Kubernetes entre regiones, los volúmenes creados en un `apiRegion` Puede utilizarse en cargas de trabajo programadas en nodos de varias regiones de Google Cloud. El tráfico interregional conlleva un coste adicional.	
`apiKey`	clave API para la cuenta de servicio de Google Cloud con la `netappcloudvolumes.admin` role. Incluye el contenido en formato JSON del archivo de clave privada de una cuenta de servicio de Google Cloud (copiado textualmente en el archivo de configuración del backend).	
`proxyURL`	URL del proxy si se requiere un servidor proxy para conectarse a la cuenta de CVS. El servidor proxy puede ser un proxy HTTP o un proxy HTTPS. En el caso de un proxy HTTPS, se omite la validación del certificado para permitir el uso de certificados autofirmados en el servidor proxy. No se admiten servidores proxy con autenticación habilitada.	
`nfsMountOptions`	Control preciso de las opciones de montaje NFS.	"nfsvers=3"
`limitVolumeSize`	Fallará el aprovisionamiento si el tamaño de volumen solicitado supera este valor.	" (no se aplica por defecto)
`serviceLevel`	El nivel de servicio CVS-Performance o CVS para nuevos volúmenes. Los valores de rendimiento de CVS son `standard` , `premium` , o `extreme` . Los valores CVS son `standardsw` o `zoneredundantstandardsw` .	El valor predeterminado de CVS-Performance es "estándar". El valor predeterminado de CVS es "standardsw".
`network`	Red de Google Cloud utilizada para los volúmenes del Cloud Volumes Service .	"por defecto"
`debugTraceFlags`	Indicadores de depuración para usar al solucionar problemas. Ejemplo, `\{"api":false, "method":true}` . No utilice esta función a menos que esté solucionando problemas y necesite un registro detallado.	nulo

Parámetro	Descripción	Por defecto

`allowedTopologies`	Para habilitar el acceso entre regiones, su definición de StorageClass para `allowedTopologies` Debe incluir todas las regiones. Por ejemplo:
	`- key: topology.kubernetes.io/region`
	`values:`
	`- us-east1`
	`- europe-west1`	

Opciones de aprovisionamiento de volumen

Puedes controlar el aprovisionamiento de volúmenes predeterminado en el `defaults` sección del archivo de configuración.

Parámetro	Descripción	Por defecto
`exportRule`	Las reglas de exportación para nuevos volúmenes. Debe ser una lista separada por comas de cualquier combinación de direcciones IPv4 o subredes IPv4 en notación CIDR.	"0.0.0.0/0"
`snapshotDir`	Acceso a la `.snapshot` directorio	"FALSO"
`snapshotReserve`	Porcentaje de volumen reservado para instantáneas	"" (aceptar el valor predeterminado de CVS de 0)
`size`	El tamaño de los nuevos volúmenes. El rendimiento mínimo de CVS es de 100 GiB. El tamaño mínimo de CVS es de 1 GiB.	El tipo de servicio CVS-Performance tiene como valor predeterminado "100 GiB". El tipo de servicio CVS no establece un valor predeterminado, pero requiere un mínimo de 1 GiB.

Ejemplos de tipos de servicio de CVS-Performance

Los siguientes ejemplos proporcionan configuraciones de muestra para el tipo de servicio CVS-Performance.

Ejemplo 1: Configuración mínima

Esta es la configuración mínima de backend utilizando el tipo de servicio CVS-Performance predeterminado con el nivel de servicio "estándar" predeterminado.

```
---
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
```

Ejemplo 2: Configuración del nivel de servicio

Este ejemplo ilustra las opciones de configuración del backend, incluyendo el nivel de servicio y los valores predeterminados de volumen.

```
---
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
```

Ejemplo 3: Configuración de grupo virtual

Esta muestra utiliza `storage` para configurar grupos virtuales y el `StorageClasses` que hacen referencia a ellos. Referirse aDefiniciones de clases de almacenamiento para ver cómo se definieron las clases de almacenamiento.

Aquí se establecen valores predeterminados específicos para todos los grupos virtuales, que definen `snapshotReserve` al 5% y el `exportRule` a 0.0.0.0/0. Los grupos virtuales se definen en el `storage` sección. Cada grupo virtual individual define su propio `serviceLevel` , y algunos grupos sobrescriben los valores predeterminados. Se utilizaron etiquetas de piscinas virtuales para diferenciar las piscinas en función de `performance` y `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
```

Definiciones de clases de almacenamiento

Las siguientes definiciones de StorageClass se aplican al ejemplo de configuración de grupo virtual. Usando `parameters.selector` , puede especificar para cada StorageClass el grupo virtual utilizado para alojar un volumen. El volumen tendrá los aspectos definidos en el pool elegido.

Ejemplo de clase de almacenamiento

```
---
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
```

- La primera clase de almacenamiento(`cvs-extreme-extra-protection`) se asigna al primer grupo virtual. Esta es la única piscina que ofrece un rendimiento extremo con una reserva instantánea del 10%.
- La última clase de almacenamiento(`cvs-extra-protection`) menciona cualquier grupo de almacenamiento que proporcione una reserva de instantáneas del 10%. Trident decide qué grupo virtual se selecciona y garantiza que se cumpla el requisito de reserva de instantáneas.

Ejemplos de tipos de servicio de CVS

Los siguientes ejemplos proporcionan configuraciones de muestra para el tipo de servicio CVS.

Ejemplo 1: Configuración mínima

Esta es la configuración mínima de backend que utiliza `storageClass` para especificar el tipo de servicio CVS y el valor predeterminado `standardsw` nivel de servicio.

```
---
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
```

Ejemplo 2: Configuración del grupo de almacenamiento

Esta configuración de backend de ejemplo utiliza `storagePools` para configurar un grupo de almacenamiento.

```
---
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
```

¿Que sigue?

Después de crear el archivo de configuración del backend, ejecute el siguiente comando:

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

Si falla la creación del backend, algo falla en la configuración del backend. Puedes consultar los registros para determinar la causa ejecutando el siguiente comando:

```
tridentctl logs
```

Después de identificar y corregir el problema con el archivo de configuración, puede volver a ejecutar el comando de creación.

Configure un backend de NetApp HCI o SolidFire

Aprende cómo crear y usar un backend de Element con tu instalación de Trident .

Detalles del controlador Element

Trident proporciona el `solidfire-san` Controlador de almacenamiento para comunicarse con el clúster. Los modos de acceso compatibles son: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

El `solidfire-san` El controlador de almacenamiento admite los modos de volumen *archivo* y *bloque*. Para el `Filesystem` En volumeMode, Trident crea un volumen y un sistema de archivos. El tipo de sistema de archivos se especifica mediante la StorageClass.

Conductor	Protocolo	Modo de volumen	Modos de acceso compatibles	Sistemas de archivos compatibles
`solidfire-san`	iSCSI	Bloquear	RWO, ROX, RWX, RWOP	Sin sistema de archivos. Dispositivo de bloque sin procesar.
`solidfire-san`	iSCSI	Sistema de archivos	RWO, RWOP	`xfs`, `ext3` , `ext4`

Antes de empezar

Necesitarás lo siguiente antes de crear un backend de Element.

- Un sistema de almacenamiento compatible que ejecuta el software Element.
- Credenciales de un administrador de clúster NetApp HCI/ SolidFire o de un usuario inquilino que pueda administrar volúmenes.
- Todos tus nodos de trabajo de Kubernetes deben tener instaladas las herramientas iSCSI apropiadas. Referirse a"Información de preparación del nodo de trabajo" .

Opciones de configuración del backend

Consulte la siguiente tabla para ver las opciones de configuración del backend:

Parámetro	Descripción	Por defecto
`version`		Siempre 1
`storageDriverName`	Nombre del controlador de almacenamiento	Siempre "solidfire-san"

Parámetro	Descripción	Por defecto
`backendName`	Nombre personalizado o el backend de almacenamiento	"solidfire_" + dirección IP de almacenamiento (iSCSI)
`Endpoint`	MVIP para el clúster SolidFire con credenciales de inquilino	
`SVIP`	Dirección IP y puerto de almacenamiento (iSCSI)	
`labels`	Conjunto de etiquetas arbitrarias con formato JSON para aplicar a los volúmenes.	""
`TenantName`	Nombre del inquilino a utilizar (se creará si no se encuentra)	
`InitiatorIFace`	Restringir el tráfico iSCSI a una interfaz de host específica	"por defecto"
`UseCHAP`	Utilice CHAP para autenticar iSCSI. Trident utiliza CHAP.	verdadero
`AccessGroups`	Lista de identificadores de grupos de acceso para usar	Encuentra el ID de un grupo de acceso llamado "trident".
`Types`	Especificaciones de QoS	
`limitVolumeSize`	Fallará el aprovisionamiento si el tamaño del volumen solicitado supera este valor.	" (no se aplica por defecto)
`debugTraceFlags`	Indicadores de depuración para usar al solucionar problemas. Ejemplo: {"api":false, "method":true}	nulo

No utilizar `debugTraceFlags` a menos que esté solucionando problemas y necesite un registro detallado.

Ejemplo 1: Configuración del backend para `solidfire-san` Controlador con tres tipos de volumen

Este ejemplo muestra un archivo de backend que utiliza la autenticación CHAP y modela tres tipos de volúmenes con garantías de QoS específicas. Lo más probable es que luego definas clases de almacenamiento para consumir cada una de ellas utilizando el `IOPS` Parámetro de clase de almacenamiento.

```
---
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
```

Ejemplo 2: Configuración de la clase de almacenamiento y del backend para `solidfire-san` controlador con grupos virtuales

Este ejemplo muestra el archivo de definición del backend configurado con pools virtuales junto con las StorageClasses que hacen referencia a ellos.

Trident copia las etiquetas presentes en un grupo de almacenamiento al LUN de almacenamiento backend durante el aprovisionamiento. Para mayor comodidad, los administradores de almacenamiento pueden definir etiquetas por grupo virtual y agrupar volúmenes por etiqueta.

En el archivo de definición de backend de ejemplo que se muestra a continuación, se establecen valores predeterminados específicos para todos los grupos de almacenamiento, que establecen el `type` en Plata. Los grupos virtuales se definen en el `storage` sección. En este ejemplo, algunos de los grupos de almacenamiento establecen su propio tipo, y algunos grupos anulan los valores predeterminados establecidos anteriormente.

```
---
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
```

Las siguientes definiciones de StorageClass hacen referencia a los grupos virtuales mencionados

anteriormente. Utilizando el `parameters.selector` En cada campo, cada StorageClass especifica qué grupo o grupos virtuales se pueden usar para alojar un volumen. El volumen tendrá los aspectos definidos en el pool virtual elegido.

La primera clase de almacenamiento(`solidfire-gold-four`) se asignará al primer grupo virtual. Esta es la única piscina que ofrece rendimiento oro con un `Volume Type QoS` de oro. La última clase de almacenamiento(`solidfire-silver`) menciona cualquier grupo de almacenamiento que ofrezca un rendimiento plateado. Trident decidirá qué grupo virtual se selecciona y garantiza que se cumplan los requisitos de almacenamiento.

```
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
```

Encuentra más información

- "Grupos de acceso por volumen"

Controladores SAN de ONTAP

Descripción general del controlador SAN de ONTAP

Aprenda a configurar un backend de ONTAP con ONTAP y los controladores SAN de Cloud Volumes ONTAP .

Detalles del controlador SAN de ONTAP

Trident proporciona los siguientes controladores de almacenamiento SAN para comunicarse con el clúster ONTAP . Los modos de acceso compatibles son: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Conductor	Protocolo	modo de volumen	Modos de acceso compatibles	Sistemas de archivos compatibles
`ontap-san`	iSCSI SCSI sobre FC	Bloquear	RWO, ROX, RWX, RWOP	Sin sistema de archivos; dispositivo de bloques sin formato
`ontap-san`	iSCSI SCSI sobre FC	Sistema de archivos	RWO, RWOP Los parámetros ROX y RWX no están disponibles en el modo de volumen del sistema de archivos.	`xfs`, `ext3` , `ext4`

Conductor	Protocolo	modo de volumen	Modos de acceso compatibles	Sistemas de archivos compatibles
`ontap-san`	NVMe/TCP Referirse aConsideraciones adicionales para NVMe/TCP.	Bloquear	RWO, ROX, RWX, RWOP	Sin sistema de archivos; dispositivo de bloques sin formato
`ontap-san`	NVMe/TCP Referirse aConsideraciones adicionales para NVMe/TCP.	Sistema de archivos	RWO, RWOP Los parámetros ROX y RWX no están disponibles en el modo de volumen del sistema de archivos.	`xfs`, `ext3` , `ext4`
`ontap-san-economy`	iSCSI	Bloquear	RWO, ROX, RWX, RWOP	Sin sistema de archivos; dispositivo de bloques sin formato
`ontap-san-economy`	iSCSI	Sistema de archivos	RWO, RWOP Los parámetros ROX y RWX no están disponibles en el modo de volumen del sistema de archivos.	`xfs`, `ext3` , `ext4`

- Usar `ontap-san-economy` solo si se espera que el recuento de uso de volumen persistente sea superior a"límites de volumen ONTAP compatibles" .
- Usar `ontap-nas-economy` solo si se espera que el recuento de uso de volumen persistente sea superior a"límites de volumen ONTAP compatibles" y el `ontap-san-economy` El controlador no se puede utilizar.
- No usar `ontap-nas-economy` Si prevé la necesidad de protección de datos, recuperación ante desastres o movilidad.
- NetApp no recomienda usar el crecimiento automático de Flexvol en todos los controladores ONTAP , excepto en ontap-san. Como solución alternativa, Trident admite el uso de reserva de instantáneas y escala los volúmenes Flexvol en consecuencia.

Permisos de usuario

Trident espera ejecutarse como administrador de ONTAP o SVM, normalmente utilizando el `admin` usuario del clúster o un `vsadmin` Usuario de SVM, o un usuario con un nombre diferente que tenga la misma función.

Para implementaciones de Amazon FSx for NetApp ONTAP , Trident requiere ejecutarse como administrador de ONTAP o SVM, utilizando el clúster. `fsxadmin` usuario o un `vsadmin` Usuario de SVM, o un usuario con un nombre diferente que tenga la misma función. El `fsxadmin` El usuario es un reemplazo limitado para el usuario administrador del clúster.

> Si utiliza el `limitAggregateUsage` Se requieren permisos de administrador de clúster para este parámetro. Al utilizar Amazon FSx for NetApp ONTAP con Trident, `limitAggregateUsage` El parámetro no funcionará con el `vsadmin` y `fsxadmin` cuentas de usuario. La operación de configuración fallará si especifica este parámetro.

Si bien es posible crear un rol más restrictivo dentro de ONTAP que pueda usar un controlador Trident , no lo recomendamos. La mayoría de las nuevas versiones de Trident utilizarán API adicionales que habría que tener en cuenta, lo que dificultaría las actualizaciones y las haría propensas a errores.

Consideraciones adicionales para NVMe/TCP

Trident admite el protocolo de memoria no volátil express (NVMe) mediante el `ontap-san` controlador que incluye:

- IPv6
- Instantáneas y clones de volúmenes NVMe
- Redimensionar un volumen NVMe
- Importar un volumen NVMe creado fuera de Trident para que su ciclo de vida pueda ser gestionado por Trident.
- Multiruta nativa de NVMe
- Apagado correcto o incorrecto de los nodos K8s (24.06)

Trident no admite:

- DH-HMAC-CHAP compatible de forma nativa con NVMe
- Multiruta del mapeador de dispositivos (DM)
- Cifrado LUKS

> NVMe solo es compatible con las API REST de ONTAP y no con ONTAPI (ZAPI).

Prepárese para configurar el backend con los controladores SAN de ONTAP.

Comprenda los requisitos y las opciones de autenticación para configurar un backend ONTAP con controladores ONTAP SAN.

Requisitos

Para todos los backends de ONTAP , Trident requiere que se asigne al menos un agregado al SVM.

> "Sistemas ASA r2"Se diferencian de otros sistemas ONTAP (ASA, AFF y FAS) en la implementación de su capa de almacenamiento. En los sistemas ASA r2, se utilizan zonas de disponibilidad de almacenamiento en lugar de agregados. Referirse a"este" Artículo de la base de conocimientos sobre cómo asignar agregados a SVM en sistemas ASA r2.

Recuerda que también puedes ejecutar más de un controlador y crear clases de almacenamiento que apunten a uno u otro. Por ejemplo, podrías configurar un `san-dev` clase que utiliza la `ontap-san` conductor y un `san-default` clase que utiliza la `ontap-san-economy` uno.

Todos los nodos de trabajo de Kubernetes deben tener instaladas las herramientas iSCSI adecuadas. Referirse a "Preparar el nodo de trabajo" Para más detalles.

Autenticar el backend de ONTAP

Trident ofrece dos modos de autenticación de un backend ONTAP .

- Basado en credenciales: El nombre de usuario y la contraseña de un usuario de ONTAP con los permisos necesarios. Se recomienda utilizar un rol de inicio de sesión de seguridad predefinido, como por ejemplo: `admin` o `vsadmin` para garantizar la máxima compatibilidad con las versiones de ONTAP .
- Basado en certificados: Trident también puede comunicarse con un clúster ONTAP utilizando un certificado instalado en el backend. Aquí, la definición del backend debe contener valores codificados en Base64 del certificado del cliente, la clave y el certificado de CA de confianza si se utiliza (recomendado).

Puedes actualizar los sistemas backend existentes para alternar entre métodos basados en credenciales y métodos basados en certificados. Sin embargo, solo se admite un método de autenticación a la vez. Para cambiar a un método de autenticación diferente, debe eliminar el método existente de la configuración del backend.

> Si intenta proporcionar **tanto credenciales como certificados**, la creación del backend fallará con un error que indica que se proporcionó más de un método de autenticación en el archivo de configuración.

Habilitar la autenticación basada en credenciales

Trident requiere las credenciales de un administrador con ámbito SVM/ámbito de clúster para comunicarse con el backend de ONTAP . Se recomienda utilizar roles estándar predefinidos, tales como: `admin` o `vsadmin` . Esto garantiza la compatibilidad con versiones futuras de ONTAP que podrían exponer API de funciones para ser utilizadas por futuras versiones de Trident . Se puede crear y usar un rol de inicio de sesión de seguridad personalizado con Trident, pero no se recomienda.

Un ejemplo de definición de backend se vería así:

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"
}
```

Tenga en cuenta que la definición del backend es el único lugar donde las credenciales se almacenan en texto plano. Una vez creado el backend, los nombres de usuario y las contraseñas se codifican con Base64 y se almacenan como secretos de Kubernetes. La creación o actualización de un backend es el único paso que requiere conocer las credenciales. Por lo tanto, se trata de una operación exclusiva para administradores, que debe ser realizada por el administrador de Kubernetes/almacenamiento.

Habilitar la autenticación basada en certificados

Los backends nuevos y existentes pueden usar un certificado y comunicarse con el backend de ONTAP . Se requieren tres parámetros en la definición del backend.

- clientCertificate: Valor codificado en Base64 del certificado del cliente.
- clientPrivateKey: Valor codificado en Base64 de la clave privada asociada.
- trustedCACertificate: Valor codificado en Base64 del certificado de CA de confianza. Si se utiliza una CA de confianza, este parámetro debe proporcionarse. Esto puede ignorarse si no se utiliza ninguna CA de confianza.

Un flujo de trabajo típico comprende los siguientes pasos.

Pasos

1. Generar un certificado y una clave de cliente. Al generar, configure el Nombre Común (CN) con el usuario ONTAP con el que se autenticará.

```
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. Agregar certificado de CA de confianza al clúster ONTAP . Es posible que esto ya lo gestione el administrador de almacenamiento. Ignorar si no se utiliza ninguna CA de confianza.

```
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. Instale el certificado y la clave del cliente (del paso 1) en el clúster ONTAP .

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

4. Confirme que el rol de inicio de sesión de seguridad de ONTAP es compatible. `cert` Método de autenticación.

```
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. Prueba de autenticación utilizando el certificado generado. Reemplace < ONTAP Management LIF> y <vserver name> con la IP de Management LIF y el nombre de 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. Codifique el certificado, la clave y el certificado de CA de confianza con Base64.

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

7. Crea el backend utilizando los valores obtenidos en el paso anterior.

```
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 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Actualizar los métodos de autenticación o rotar las credenciales

Puedes actualizar un backend existente para usar un método de autenticación diferente o para rotar sus credenciales. Esto funciona en ambos sentidos: los sistemas de gestión de backends que utilizan nombre de usuario/contraseña pueden actualizarse para usar certificados; los sistemas de gestión de backends que utilizan certificados pueden actualizarse para basarse en nombre de usuario/contraseña. Para ello, debe eliminar el método de autenticación existente y agregar el nuevo método de autenticación. A continuación, utilice el archivo backend.json actualizado que contiene los parámetros necesarios para ejecutar `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 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Al rotar las contraseñas, el administrador de almacenamiento primero debe actualizar la contraseña del usuario en ONTAP. A continuación se realiza una actualización del servidor. Al rotar los certificados, se pueden agregar varios certificados al usuario. Posteriormente, se actualiza el sistema backend para utilizar el nuevo certificado, tras lo cual se puede eliminar el certificado antiguo del clúster ONTAP .

La actualización de un backend no interrumpe el acceso a los volúmenes que ya se han creado, ni afecta a las conexiones de volumen realizadas posteriormente. Una actualización exitosa del backend indica que Trident puede comunicarse con el backend de ONTAP y gestionar futuras operaciones de volumen.

Cree un rol ONTAP personalizado para Trident.

Puede crear un rol de clúster ONTAP con privilegios mínimos para que no tenga que usar el rol de administrador de ONTAP para realizar operaciones en Trident. Cuando incluyes el nombre de usuario en una configuración de backend de Trident , Trident utiliza el rol de clúster ONTAP que creaste para realizar las operaciones.

Referirse a"Generador de roles personalizados de Trident" Para obtener más información sobre la creación de roles personalizados de Trident .

Uso de la CLI de ONTAP

1. Crea un nuevo rol utilizando el siguiente comando:

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

2. Crea un nombre de usuario para el usuario de 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. Asigna el rol al usuario:

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

Usando el Administrador del sistema

Realice los siguientes pasos en ONTAP System Manager:

1. **Crea un rol personalizado**:
   a. Para crear un rol personalizado a nivel de clúster, seleccione **Clúster > Configuración**.

      (O) Para crear un rol personalizado a nivel de SVM, seleccione **Almacenamiento > Máquinas virtuales de almacenamiento > `required SVM` > Configuración > Usuarios y roles**.

   b. Seleccione el icono de flecha (→) junto a **Usuarios y roles**.
   c. Seleccione ****Agregar** en **Roles**.
   d. Define las reglas para el rol y haz clic en **Guardar**.
2. **Asigna el rol al usuario de Trident *: + Realiza los siguientes pasos en la página *Usuarios y roles**:
   a. Seleccione el icono Agregar ***+** debajo de **Usuarios**.
   b. Seleccione el nombre de usuario requerido y seleccione un rol en el menú desplegable para **Rol**.
   c. Haga clic en **Guardar**.

Para obtener más información, consulte las siguientes páginas:

- "Roles personalizados para la administración de ONTAP"o"Definir roles personalizados"
- "Trabajar con roles y usuarios"

Autenticar conexiones con CHAP bidireccional

Trident puede autenticar sesiones iSCSI con CHAP bidireccional para la `ontap-san` y `ontap-san-economy` conductores. Esto requiere habilitar el `useCHAP` opción en la definición de tu backend. Cuando se configura para `true` Trident configura la seguridad del iniciador predeterminado de la SVM en CHAP bidireccional y establece el nombre de usuario y los secretos desde el archivo de backend. NetApp recomienda utilizar CHAP bidireccional para autenticar las conexiones. Consulte la siguiente configuración de ejemplo:

```
---
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
```

El `useCHAP` El parámetro es una opción booleana que solo se puede configurar una vez. Está configurado como falso por defecto. Una vez que lo hayas configurado como verdadero, no podrás configurarlo como falso.

Además de `useCHAP=true` , el `chapInitiatorSecret` , `chapTargetInitiatorSecret` , `chapTargetUsername` , y `chapUsername` Los campos deben incluirse en la definición del backend. Los secretos se pueden cambiar después de crear un backend ejecutando `tridentctl update` .

Cómo funciona

Mediante la configuración `useCHAP` Para que sea verdadero, el administrador de almacenamiento le indica a Trident que configure CHAP en el backend de almacenamiento. Esto incluye lo siguiente:

- Configuración de CHAP en la SVM:
  - Si el tipo de seguridad del iniciador predeterminado de la SVM es ninguno (establecido por defecto) **y** no hay LUN preexistentes en el volumen, Trident establecerá el tipo de seguridad predeterminado en `CHAP` y proceda a configurar el iniciador CHAP y el nombre de usuario y las claves de destino.
  - Si la SVM contiene LUN, Trident no habilitará CHAP en la SVM. Esto garantiza que el acceso a las LUN que ya están presentes en la SVM no se vea restringido.
- Configurar el nombre de usuario y las claves secretas del iniciador y del destino CHAP; estas opciones deben especificarse en la configuración del backend (como se muestra arriba).

Una vez creado el backend, Trident crea un correspondiente `tridentbackend` CRD y almacena los secretos CHAP y los nombres de usuario como secretos de Kubernetes. Todos los PV que Trident cree en este backend se montarán y conectarán a través de CHAP.

Rotar credenciales y actualizar backends

Puede actualizar las credenciales CHAP actualizando los parámetros CHAP en el archivo `backend.json` archivo. Esto requerirá actualizar los secretos CHAP y utilizar el `tridentctl update` orden para reflejar estos cambios.

Al actualizar los secretos CHAP para un backend, debe usar `tridentctl` para actualizar el backend. No actualice las credenciales en el clúster de almacenamiento mediante ONTAP CLI o ONTAP System Manager, ya que Trident no podrá detectar estos cambios.

```
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 |
+----------------+----------------+--------------------------------------
+--------+---------+
```

Las conexiones existentes no se verán afectadas; seguirán activas si Trident actualiza las credenciales en la SVM. Las nuevas conexiones utilizan las credenciales actualizadas y las conexiones existentes permanecen activas. Desconectar y volver a conectar los PV antiguos hará que utilicen las credenciales actualizadas.

Opciones y ejemplos de configuración de SAN de ONTAP

Aprenda cómo crear y utilizar controladores ONTAP SAN con su instalación de Trident . Esta sección proporciona ejemplos de configuración de backend y detalles para mapear backends a StorageClasses.

"Sistemas ASA r2"Se diferencia de otros sistemas ONTAP (ASA, AFF y FAS) en la implementación de su capa de almacenamiento. Estas variaciones afectan al uso de ciertos parámetros, tal como se indica. "Obtenga más información sobre las diferencias entre los sistemas ASA r2 y otros sistemas ONTAP .".

Sólo el `ontap-san` El controlador (con protocolos iSCSI y NVMe/TCP) es compatible con los sistemas ASA r2.

En la configuración del backend de Trident , no es necesario especificar que su sistema sea ASA r2. Cuando seleccionas `ontap-san` como el `storageDriverName` Trident detecta automáticamente el ASA r2 o el sistema ONTAP tradicional. Algunos parámetros de configuración del backend no son aplicables a los sistemas ASA r2, como se indica en la tabla siguiente.

Opciones de configuración del backend

Consulte la siguiente tabla para ver las opciones de configuración del backend:

Parámetro	Descripción	Por defecto
`version`		Siempre 1
`storageDriverName`	Nombre del controlador de almacenamiento	`ontap-san`o `ontap-san-economy`
`backendName`	Nombre personalizado o el backend de almacenamiento	Nombre del controlador + "_" + dataLIF
`managementLIF`	Dirección IP de un clúster o LIF de gestión de SVM. Se puede especificar un nombre de dominio completo (FQDN). Se puede configurar para usar direcciones IPv6 si Trident se instaló usando la bandera IPv6. Las direcciones IPv6 deben definirse entre corchetes, como por ejemplo: `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . Para una transición fluida a MetroCluster , consulteEjemplo de MetroCluster . Si está utilizando credenciales "vsadmin", `managementLIF` debe ser la del SVM; si se utilizan credenciales de "administrador", `managementLIF` debe ser el del grupo.	"10.0.0.1", "[2001:1234:abcd::fefe]"
`dataLIF`	Dirección IP del protocolo LIF. Se puede configurar para usar direcciones IPv6 si Trident se instaló usando la bandera IPv6. Las direcciones IPv6 deben definirse entre corchetes, como por ejemplo: `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . **No especificar para iSCSI.** Trident utiliza"Mapa selectivo de LUN de ONTAP" para descubrir los LIF iSCSI necesarios para establecer una sesión de múltiples rutas. Se genera una advertencia si `dataLIF` está definido explícitamente. **Omitir para Metrocluster.** Ver elEjemplo de MetroCluster .	Derivado por la SVM

Parámetro	Descripción	Por defecto
`svm`	Máquina virtual de almacenamiento a utilizar **Omitir para Metrocluster.** Ver elEjemplo de MetroCluster .	Derivado si se trata de una SVM `managementLIF` se especifica
`useCHAP`	Utilice CHAP para autenticar iSCSI para controladores SAN ONTAP [Booleano]. Empezar a `true` para que Trident configure y utilice CHAP bidireccional como autenticación predeterminada para el SVM proporcionado en el backend. Referirse a "Prepárese para configurar el backend con los controladores SAN de ONTAP." Para más detalles. **No compatible con FCP ni NVMe/TCP.**	`false`
`chapInitiatorSecret`	Secreto del iniciador de CHAP. Requerido si `useCHAP=true`	""
`labels`	Conjunto de etiquetas arbitrarias con formato JSON para aplicar a los volúmenes	""
`chapTargetInitiatorSecret`	Secreto del iniciador del objetivo CHAP. Requerido si `useCHAP=true`	""
`chapUsername`	Nombre de usuario entrante. Requerido si `useCHAP=true`	""
`chapTargetUsername`	Nombre de usuario objetivo. Requerido si `useCHAP=true`	""
`clientCertificate`	Valor codificado en Base64 del certificado del cliente. Se utiliza para la autenticación basada en certificados.	""
`clientPrivateKey`	Valor codificado en Base64 de la clave privada del cliente. Se utiliza para la autenticación basada en certificados.	""
`trustedCACertificate`	Valor codificado en Base64 del certificado de CA de confianza. Opcional. Se utiliza para la autenticación basada en certificados.	""
`username`	Nombre de usuario necesario para comunicarse con el clúster ONTAP . Se utiliza para la autenticación basada en credenciales. Para la autenticación de Active Directory, consulte "Autenticar Trident en un SVM backend mediante credenciales de Active Directory".	""
`password`	Contraseña necesaria para comunicarse con el clúster ONTAP . Se utiliza para la autenticación basada en credenciales. Para la autenticación de Active Directory, consulte "Autenticar Trident en un SVM backend mediante credenciales de Active Directory".	""
`svm`	máquina virtual de almacenamiento a utilizar	Derivado si se trata de una SVM `managementLIF` se especifica

Parámetro	Descripción	Por defecto
`storagePrefix`	Prefijo utilizado al aprovisionar nuevos volúmenes en la SVM. No se puede modificar posteriormente. Para actualizar este parámetro, deberá crear un nuevo backend.	`trident`
`aggregate`	Agregado para aprovisionamiento (opcional; si se establece, debe asignarse a la SVM). Para el `ontap-nas-flexgroup` conductor, esta opción se ignora. Si no se asigna, cualquiera de los agregados disponibles se puede utilizar para aprovisionar un volumen FlexGroup . Cuando se actualiza el agregado en SVM, se actualiza automáticamente en Trident mediante sondeos a SVM sin necesidad de reiniciar el controlador Trident . Cuando se ha configurado un agregado específico en Trident para aprovisionar volúmenes, si el agregado se renombra o se mueve fuera del SVM, el backend pasará a un estado de error en Trident mientras consulta el agregado del SVM. Debe cambiar el agregado por uno que esté presente en la SVM o eliminarlo por completo para volver a poner en línea el backend. **No especificar para sistemas ASA r2**.	""
`limitAggregateUsage`	Fallará el aprovisionamiento si el uso supera este porcentaje. Si utiliza un backend de Amazon FSx for NetApp ONTAP , no especifique `limitAggregateUsage` . El proporcionado `fsxadmin` y `vsadmin` No contienen los permisos necesarios para recuperar el uso agregado y limitarlo mediante Trident. **No especificar para sistemas ASA r2**.	" (no se aplica por defecto)
`limitVolumeSize`	Fallará el aprovisionamiento si el tamaño de volumen solicitado supera este valor. También restringe el tamaño máximo de los volúmenes que administra para las LUN.	" (no se aplica por defecto)
`lunsPerFlexvol`	Número máximo de LUN por Flexvol, debe estar en el rango [50, 200]	`100`
`debugTraceFlags`	Indicadores de depuración para usar al solucionar problemas. Ejemplo: {"api":false, "method":true} No lo utilice a menos que esté solucionando problemas y necesite un volcado de registro detallado.	`null`

Parámetro	Descripción	Por defecto
`useREST`	Parámetro booleano para utilizar las API REST de ONTAP . `useREST`Cuando se configura para `true` Trident utiliza las API REST de ONTAP para comunicarse con el backend; cuando se configura en `false` Trident utiliza llamadas ONTAPI (ZAPI) para comunicarse con el backend. Esta función requiere ONTAP 9.11.1 y versiones posteriores. Además, el rol de inicio de sesión de ONTAP utilizado debe tener acceso a `ontapi` solicitud. Esto se satisface mediante lo predefinido. `vsadmin` y `cluster-admin` roles. A partir de la versión Trident 24.06 y ONTAP 9.15.1 o posterior, `useREST` está configurado para `true` por defecto; cambiar `useREST` a `false` para utilizar llamadas ONTAPI (ZAPI). `useREST`está totalmente cualificado para NVMe/TCP. NVMe solo es compatible con las API REST de ONTAP y no con ONTAPI (ZAPI). **Si se especifica, siempre se establecerá en `true` para sistemas ASA r2**.	`true`para ONTAP 9.15.1 o posterior, de lo contrario `false`.
`sanType`	Utilice para seleccionar `iscsi` para iSCSI, `nvme` para NVMe/TCP o `fcp` para SCSI sobre Fibre Channel (FC).	`iscsi`si está en blanco

Parámetro	Descripción	Por defecto
`formatOptions`	Usar `formatOptions` para especificar los argumentos de la línea de comandos para el `mkfs` comando que se aplicará siempre que se formatee un volumen. Esto le permite formatear el volumen según sus preferencias. Asegúrese de especificar las opciones de formato de forma similar a las opciones del comando mkfs, excluyendo la ruta del dispositivo. Ejemplo: "-E no descartar" **Compatible con `ontap-san` y `ontap-san-economy` controladores con protocolo iSCSI. Además, se admite para sistemas ASA r2 cuando se utilizan los protocolos iSCSI y NVMe/TCP.**	
`limitVolumePoolSize`	Tamaño máximo de FlexVol solicitable al usar LUN en el backend ontap-san-economy.	" (no se aplica por defecto)
`denyNewVolumePools`	Restringe `ontap-san-economy` Los sistemas backend crean nuevos volúmenes FlexVol para contener sus LUN. Solo se utilizan Flexvols preexistentes para el aprovisionamiento de nuevos PV.	

Recomendaciones para el uso de formatOptions

Trident recomienda la siguiente opción para agilizar el proceso de formateo:

-E no descartar:

- Conservar, no intentar descartar bloques en el momento de mkfs (descartar bloques inicialmente es útil en dispositivos de estado sólido y almacenamiento disperso/de aprovisionamiento ligero). Esto reemplaza la opción obsoleta "-K" y es aplicable a todos los sistemas de archivos (xfs, ext3 y ext4).

Autenticar Trident en un SVM backend mediante credenciales de Active Directory

Puede configurar Trident para autenticarse en un SVM de backend usando credenciales de Active Directory (AD). Antes de que una cuenta de AD pueda acceder a la SVM, debe configurar el acceso del controlador de dominio de AD al clúster o SVM. Para la administración de un clúster con una cuenta de AD, debe crear un túnel de dominio. Referirse a "Configurar el acceso al controlador de dominio de Active Directory en ONTAP" Para más detalles.

pasos

1. Configurar los ajustes del Sistema de nombres de dominio (DNS) para un SVM de backend:

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

2. Ejecute el siguiente comando para crear una cuenta de computadora para la SVM en Active Directory:

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

3. Utilice este comando para crear un usuario o grupo de AD para administrar el clúster o SVM

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

4. En el archivo de configuración del backend de Trident , configure el `username` y `password` parámetros al nombre de usuario o grupo de AD y la contraseña, respectivamente.

Opciones de configuración de backend para el aprovisionamiento de volúmenes

Puedes controlar el aprovisionamiento predeterminado utilizando estas opciones en el `defaults` sección de la configuración. Para ver un ejemplo, consulte los ejemplos de configuración a continuación.

Parámetro	Descripción	Por defecto
`spaceAllocation`	Asignación de espacio para LUN	"verdadero" **Si se especifica, establecer en `true` para sistemas ASA r2**.
`spaceReserve`	Modo de reserva de espacio: "ninguno" (delgado) o "volumen" (grueso). **Empezar a `none` para sistemas ASA r2**.	"ninguno"
`snapshotPolicy`	Política de instantáneas a utilizar. **Empezar a `none` para sistemas ASA r2**.	"ninguno"
`qosPolicy`	Grupo de políticas QoS que se asignará a los volúmenes creados. Elija una de las opciones qosPolicy o adaptiveQosPolicy por grupo de almacenamiento/backend. El uso de grupos de políticas QoS con Trident requiere ONTAP 9.8 o posterior. Debe utilizar un grupo de políticas QoS no compartido y asegurarse de que el grupo de políticas se aplique a cada componente individualmente. Un grupo de políticas QoS compartidas impone un límite máximo al rendimiento total de todas las cargas de trabajo.	""
`adaptiveQosPolicy`	Grupo de políticas QoS adaptativas para asignar a los volúmenes creados. Elija una de las opciones qosPolicy o adaptiveQosPolicy por grupo de almacenamiento/backend.	""
`snapshotReserve`	Porcentaje de volumen reservado para instantáneas. **No especificar para sistemas ASA r2**.	"0" si `snapshotPolicy` es "ninguno", de lo contrario ""
`splitOnClone`	Separar un clon de su progenitor al crearlo	"FALSO"
`encryption`	Habilite el cifrado de volumen de NetApp (NVE) en el nuevo volumen; el valor predeterminado es `false` . Para utilizar esta opción, NVE debe estar licenciado y habilitado en el clúster. Si NAE está habilitado en el backend, cualquier volumen aprovisionado en Trident tendrá NAE habilitado. Para obtener más información, consulte:"Cómo funciona Trident con NVE y NAE" .	"falso" **Si se especifica, establecer en `true` para sistemas ASA r2**.

Parámetro	Descripción	Por defecto
`luksEncryption`	Habilitar el cifrado LUKS. Referirse a"Utilice la configuración de clave unificada de Linux (LUKS)." .	" **Configurado a `false` para sistemas ASA r2**.
`tieringPolicy`	Política de jerarquización para usar "ninguna" **No especificar para sistemas ASA r2** .	
`nameTemplate`	Plantilla para crear nombres de volumen personalizados.	""

Ejemplos de aprovisionamiento por volumen

Aquí tenéis un ejemplo con valores predeterminados definidos:

```
---
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'
```

> Para todos los volúmenes creados utilizando el `ontap-san` El controlador Trident agrega un 10 por ciento de capacidad adicional al FlexVol para dar cabida a los metadatos de LUN. La LUN se aprovisionará con el tamaño exacto que el usuario solicite en el PVC. Trident añade un 10 por ciento al FlexVol (se muestra como Tamaño disponible en ONTAP). Los usuarios ahora obtendrán la cantidad de capacidad utilizable que solicitaron. Este cambio también evita que las LUN se conviertan en de solo lectura a menos que se utilice todo el espacio disponible. Esto no se aplica a ontap-san-economy.

Para backends que definen `snapshotReserve` Trident calcula el tamaño de los volúmenes de la siguiente manera:

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

El 1.1 es el 10 por ciento adicional que Trident agrega al FlexVol para acomodar los metadatos del LUN. Para `snapshotReserve` = 5%, y solicitud de PVC = 5 GiB, el tamaño total del volumen es 5,79 GiB y el tamaño disponible es 5,5 GiB. El `volume show` El comando debería mostrar resultados similares a este ejemplo:

Actualmente, la única forma de utilizar el nuevo cálculo para un volumen existente es redimensionándolo.

Ejemplos de configuración mínima

Los siguientes ejemplos muestran configuraciones básicas que dejan la mayoría de los parámetros con sus valores predeterminados. Esta es la forma más sencilla de definir un backend.

> Si utiliza Amazon FSx en NetApp ONTAP con Trident, NetApp recomienda que especifique nombres DNS para las LIF en lugar de direcciones IP.

Ejemplo de SAN de ONTAP

Esta es una configuración básica que utiliza `ontap-san` conductor.

```
---
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>
```

Ejemplo de MetroCluster

Puede configurar el backend para evitar tener que actualizar manualmente la definición del backend después del cambio de estado y el cambio de estado durante"replicación y recuperación de SVM" .

Para una conmutación y recuperación sin interrupciones, especifique el SVM utilizando `managementLIF` y omitir el `svm` parámetros. Por ejemplo:

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

Ejemplo económico de ONTAP SAN

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

Ejemplo de autenticación basada en certificados

En este ejemplo de configuración básica `clientCertificate` , `clientPrivateKey` , y `trustedCACertificate` (opcional, si se utiliza una CA de confianza) se rellenan en `backend.json` y tome los valores codificados en base64 del certificado del cliente, la clave privada y el certificado de CA de confianza, respectivamente.

```
---
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
```

Ejemplos de CHAP bidireccionales

Estos ejemplos crean un backend con `useCHAP` empezar a `true` .

Ejemplo de 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>
```

Ejemplo de economía ONTAP SAN CHAP

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

Ejemplo de NVMe/TCP

Debe tener una SVM configurada con NVMe en su backend ONTAP . Esta es una configuración básica de backend para 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
```

Ejemplo de SCSI sobre FC (FCP)

Debe tener un SVM configurado con FC en su backend ONTAP . Esta es una configuración básica de backend para 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
```

Ejemplo de configuración de backend con plantilla de nombre

```
---
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}}"
```

Ejemplo de opciones de formato para el controlador 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
```

Ejemplos de backends con pools virtuales

En estos archivos de definición de backend de ejemplo, se establecen valores predeterminados específicos para todos los grupos de almacenamiento, tales como: `spaceReserve` en ninguno, `spaceAllocation` en falso, y `encryption` en falso. Los grupos virtuales se definen en la sección de almacenamiento.

Trident establece las etiquetas de aprovisionamiento en el campo "Comentarios". Los comentarios se configuran en el FlexVol volume. Trident copia todas las etiquetas presentes en un grupo virtual al volumen de almacenamiento durante el aprovisionamiento. Para mayor comodidad, los administradores de almacenamiento pueden definir etiquetas por grupo virtual y agrupar volúmenes por etiqueta.

En estos ejemplos, algunos de los grupos de almacenamiento establecen sus propias configuraciones. `spaceReserve` , `spaceAllocation` , y `encryption` valores, y algunos pools anulan los valores predeterminados.

Ejemplo de SAN de ONTAP

```
---
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"
```

Ejemplo económico de 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"
```

Ejemplo de 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"
```

Asignar backends a StorageClasses

Las siguientes definiciones de StorageClass hacen referencia aEjemplos de backends con pools virtuales . Utilizando el `parameters.selector` En cada campo, cada StorageClass especifica qué grupos virtuales se pueden usar para alojar un volumen. El volumen tendrá los aspectos definidos en el pool virtual elegido.

- El `protection-gold` StorageClass se asignará al primer grupo virtual en el `ontap-san` backend. Esta es la única piscina que ofrece protección de nivel oro.

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

- El `protection-not-gold` StorageClass se asignará al segundo y tercer grupo virtual en `ontap-san` backend. Estas son las únicas piscinas que ofrecen un nivel de protección distinto al oro.

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

- El `app-mysqldb` StorageClass se asignará al tercer grupo virtual en `ontap-san-economy` backend. Este es el único pool que ofrece configuración de pool de almacenamiento para aplicaciones de tipo mysqldb.

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

- El `protection-silver-creditpoints-20k` StorageClass se asignará al segundo grupo virtual en `ontap-san` backend. Este es el único fondo que ofrece protección de nivel plata y 20000 puntos de crédito.

```
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"
```

- El `creditpoints-5k` StorageClass se asignará al tercer grupo virtual en `ontap-san` backend y el cuarto grupo virtual en el `ontap-san-economy` backend. Estas son las únicas ofertas de pool con 5000 puntos de crédito.

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

- El `my-test-app-sc` StorageClass se asignará a `testAPP` piscina virtual en la `ontap-san` conductor con `sanType: nvme` . Esta es la única oferta de piscina `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 decidirá qué grupo virtual se selecciona y garantiza que se cumplan los requisitos de almacenamiento.

Controladores NAS ONTAP

Descripción general del controlador NAS de ONTAP

Aprenda a configurar un backend de ONTAP con ONTAP y los controladores NAS de Cloud Volumes ONTAP .

Detalles del controlador NAS de ONTAP

Trident proporciona los siguientes controladores de almacenamiento NAS para comunicarse con el clúster ONTAP . Los modos de acceso compatibles son: *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX), *ReadWriteOncePod* (RWOP).

Conductor	Protocolo	modo de volumen	Modos de acceso compatibles	Sistemas de archivos compatibles
`ontap-nas`	NFS SMB	Sistema de archivos	RWO, ROX, RWX, RWOP	"", `nfs` , `smb`
`ontap-nas-economy`	NFS SMB	Sistema de archivos	RWO, ROX, RWX, RWOP	"", `nfs` , `smb`

Conductor	Protocolo	modo de volumen	Modos de acceso compatibles	Sistemas de archivos compatibles
`ontap-nas-flexgroup`	NFS SMB	Sistema de archivos	RWO, ROX, RWX, RWOP	"", `nfs` , `smb`

- Usar `ontap-san-economy` solo si se espera que el recuento de uso de volumen persistente sea superior a"límites de volumen ONTAP compatibles" .
- Usar `ontap-nas-economy` solo si se espera que el recuento de uso de volumen persistente sea superior a"límites de volumen ONTAP compatibles" y el `ontap-san-economy` El controlador no se puede utilizar.
- No usar `ontap-nas-economy` Si prevé la necesidad de protección de datos, recuperación ante desastres o movilidad.
- NetApp no recomienda usar el crecimiento automático de Flexvol en todos los controladores ONTAP , excepto en ontap-san. Como solución alternativa, Trident admite el uso de reserva de instantáneas y escala los volúmenes Flexvol en consecuencia.

Permisos de usuario

Trident espera ejecutarse como administrador de ONTAP o SVM, normalmente utilizando el `admin` usuario del clúster o un `vsadmin` Usuario de SVM, o un usuario con un nombre diferente que tenga la misma función.

Para implementaciones de Amazon FSx for NetApp ONTAP , Trident requiere ejecutarse como administrador de ONTAP o SVM, utilizando el clúster. `fsxadmin` usuario o un `vsadmin` Usuario de SVM, o un usuario con un nombre diferente que tenga la misma función. El `fsxadmin` El usuario es un reemplazo limitado para el usuario administrador del clúster.

Si utiliza el `limitAggregateUsage` Se requieren permisos de administrador de clúster para este parámetro. Al utilizar Amazon FSx for NetApp ONTAP con Trident, `limitAggregateUsage` El parámetro no funcionará con el `vsadmin` y `fsxadmin` cuentas de usuario. La operación de configuración fallará si especifica este parámetro.

Si bien es posible crear un rol más restrictivo dentro de ONTAP que pueda usar un controlador Trident , no lo recomendamos. La mayoría de las nuevas versiones de Trident utilizarán API adicionales que habría que tener en cuenta, lo que dificultaría las actualizaciones y las haría propensas a errores.

Prepárese para configurar un backend con controladores NAS ONTAP .

Comprenda los requisitos, las opciones de autenticación y las políticas de exportación para configurar un backend ONTAP con controladores ONTAP NAS.

Requisitos

- Para todos los backends de ONTAP , Trident requiere que se asigne al menos un agregado al SVM.
- Puedes ejecutar más de un controlador y crear clases de almacenamiento que apunten a uno u otro. Por ejemplo, podrías configurar una clase Gold que utilice la `ontap-nas` conductor y una clase Bronce que utiliza el `ontap-nas-economy` uno.
- Todos tus nodos de trabajo de Kubernetes deben tener instaladas las herramientas NFS apropiadas. Referirse a"aquí" Para más detalles.

- Trident solo admite volúmenes SMB montados en pods que se ejecutan en nodos Windows. Referirse a Preparar el aprovisionamiento de volúmenes SMB Para más detalles.

Autenticar el backend de ONTAP

Trident ofrece dos modos de autenticación de un backend ONTAP .

- Basado en credenciales: Este modo requiere permisos suficientes para el backend de ONTAP . Se recomienda utilizar una cuenta asociada a un rol de inicio de sesión de seguridad predefinido, como por ejemplo: `admin` o `vsadmin` para garantizar la máxima compatibilidad con las versiones de ONTAP .
- Basado en certificados: Este modo requiere un certificado instalado en el backend para que Trident se comunique con un clúster ONTAP . Aquí, la definición del backend debe contener valores codificados en Base64 del certificado del cliente, la clave y el certificado de CA de confianza si se utiliza (recomendado).

Puedes actualizar los sistemas backend existentes para alternar entre métodos basados en credenciales y métodos basados en certificados. Sin embargo, solo se admite un método de autenticación a la vez. Para cambiar a un método de autenticación diferente, debe eliminar el método existente de la configuración del backend.

> Si intenta proporcionar **tanto credenciales como certificados**, la creación del backend fallará con un error que indica que se proporcionó más de un método de autenticación en el archivo de configuración.

Habilitar la autenticación basada en credenciales

Trident requiere las credenciales de un administrador con ámbito SVM/ámbito de clúster para comunicarse con el backend de ONTAP . Se recomienda utilizar roles estándar predefinidos, tales como: `admin` o `vsadmin` . Esto garantiza la compatibilidad con versiones futuras de ONTAP que podrían exponer API de funciones para ser utilizadas por futuras versiones de Trident . Se puede crear y usar un rol de inicio de sesión de seguridad personalizado con Trident, pero no se recomienda.

Un ejemplo de definición de backend se vería así:

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"
    }
}
```

Tenga en cuenta que la definición del backend es el único lugar donde las credenciales se almacenan en texto plano. Una vez creado el backend, los nombres de usuario y las contraseñas se codifican con Base64 y se almacenan como secretos de Kubernetes. La creación/actualización de un backend es el único paso que requiere conocer las credenciales. Por lo tanto, se trata de una operación exclusiva para administradores, que debe ser realizada por el administrador de Kubernetes/almacenamiento.

Habilitar la autenticación basada en certificados

Los backends nuevos y existentes pueden usar un certificado y comunicarse con el backend de ONTAP . Se requieren tres parámetros en la definición del backend.

- clientCertificate: Valor codificado en Base64 del certificado del cliente.
- clientPrivateKey: Valor codificado en Base64 de la clave privada asociada.
- trustedCACertificate: Valor codificado en Base64 del certificado de CA de confianza. Si se utiliza una CA de confianza, este parámetro debe proporcionarse. Esto puede ignorarse si no se utiliza ninguna CA de confianza.

Un flujo de trabajo típico comprende los siguientes pasos.

Pasos

1. Generar un certificado y una clave de cliente. Al generar, configure el Nombre Común (CN) con el usuario ONTAP con el que se autenticará.

```
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. Agregar certificado de CA de confianza al clúster ONTAP . Es posible que esto ya lo gestione el administrador de almacenamiento. Ignorar si no se utiliza ninguna CA de confianza.

```
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. Instale el certificado y la clave del cliente (del paso 1) en el clúster ONTAP .

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

4. Confirme que el rol de inicio de sesión de seguridad de ONTAP es compatible. `cert` Método de autenticación.

```
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. Prueba de autenticación utilizando el certificado generado. Reemplace < ONTAP Management LIF> y <vserver name> con la IP de Management LIF y el nombre de SVM. Debe asegurarse de que la LIF tenga configurada su política de servicio para `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. Codifique el certificado, la clave y el certificado de CA de confianza con Base64.

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

7. Crea el backend utilizando los valores obtenidos en el paso anterior.

```
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 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Actualizar los métodos de autenticación o rotar las credenciales

Puedes actualizar un backend existente para usar un método de autenticación diferente o para rotar sus credenciales. Esto funciona en ambos sentidos: los sistemas de gestión de backends que utilizan nombre de usuario/contraseña pueden actualizarse para usar certificados; los sistemas de gestión de backends que utilizan certificados pueden actualizarse para basarse en nombre de usuario/contraseña. Para ello, debe eliminar el método de autenticación existente y agregar el nuevo método de autenticación. A continuación, utilice el archivo backend.json actualizado que contiene los parámetros necesarios para ejecutar `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 |
+------------+----------------+--------------------------------------
+--------+---------+
```

Al rotar las contraseñas, el administrador de almacenamiento primero debe actualizar la contraseña del usuario en ONTAP. A continuación se realiza una actualización del servidor. Al rotar los certificados, se pueden agregar varios certificados al usuario. Posteriormente, se actualiza el sistema backend para utilizar el nuevo certificado, tras lo cual se puede eliminar el certificado antiguo del clúster ONTAP .

La actualización de un backend no interrumpe el acceso a los volúmenes que ya se han creado, ni afecta a las conexiones de volumen realizadas posteriormente. Una actualización exitosa del backend indica que Trident puede comunicarse con el backend de ONTAP y gestionar futuras operaciones de volumen.

Cree un rol ONTAP personalizado para Trident.

Puede crear un rol de clúster ONTAP con privilegios mínimos para que no tenga que usar el rol de administrador de ONTAP para realizar operaciones en Trident. Cuando incluyes el nombre de usuario en una configuración de backend de Trident , Trident utiliza el rol de clúster ONTAP que creaste para realizar las operaciones.

Referirse a"Generador de roles personalizados de Trident" Para obtener más información sobre la creación de roles personalizados de Trident .

Uso de la CLI de ONTAP

1. Crea un nuevo rol utilizando el siguiente comando:

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

2. Crea un nombre de usuario para el usuario de 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. Asigna el rol al usuario:

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

Usando el Administrador del sistema

Realice los siguientes pasos en ONTAP System Manager:

1. **Crea un rol personalizado**:
   a. Para crear un rol personalizado a nivel de clúster, seleccione **Clúster > Configuración**.

      (O) Para crear un rol personalizado a nivel de SVM, seleccione **Almacenamiento > Máquinas virtuales de almacenamiento > `required SVM` > Configuración > Usuarios y roles**.

   b. Seleccione el icono de flecha (→) junto a **Usuarios y roles**.
   c. Seleccione ****Agregar** en **Roles**.
   d. Define las reglas para el rol y haz clic en **Guardar**.
2. **Asigna el rol al usuario de Trident *: + Realiza los siguientes pasos en la página *Usuarios y roles**:
   a. Seleccione el icono Agregar ***+** debajo de **Usuarios**.
   b. Seleccione el nombre de usuario requerido y seleccione un rol en el menú desplegable para **Rol**.
   c. Haga clic en **Guardar**.

Para obtener más información, consulte las siguientes páginas:

- "Roles personalizados para la administración de ONTAP"o"Definir roles personalizados"
- "Trabajar con roles y usuarios"

Gestionar las políticas de exportación NFS

Trident utiliza políticas de exportación NFS para controlar el acceso a los volúmenes que aprovisiona.

Trident ofrece dos opciones al trabajar con políticas de exportación:

- Trident puede gestionar dinámicamente la propia política de exportación; en este modo de funcionamiento, el administrador de almacenamiento especifica una lista de bloques CIDR que representan direcciones IP admisibles. Trident agrega automáticamente a la política de exportación, en el momento de la publicación, las direcciones IP de los nodos aplicables que se encuentren dentro de estos rangos. Alternativamente, cuando no se especifican CIDR, todas las IP de unidifusión de ámbito global que se encuentren en el nodo al que se publica el volumen se agregarán a la política de exportación.
- Los administradores de almacenamiento pueden crear una política de exportación y agregar reglas manualmente. Trident utiliza la política de exportación predeterminada a menos que se especifique un nombre de política de exportación diferente en la configuración.

Gestionar dinámicamente las políticas de exportación

Trident ofrece la capacidad de gestionar dinámicamente las políticas de exportación para los sistemas backend de ONTAP . Esto proporciona al administrador de almacenamiento la capacidad de especificar un espacio de direcciones permitido para las IP de los nodos de trabajo, en lugar de definir reglas explícitas manualmente. Simplifica enormemente la gestión de la política de exportación; las modificaciones a la política de exportación ya no requieren intervención manual en el clúster de almacenamiento. Además, esto ayuda a restringir el acceso al clúster de almacenamiento únicamente a los nodos de trabajo que están montando volúmenes y tienen direcciones IP dentro del rango especificado, lo que permite una gestión automatizada y de grano fino.

> No utilice la traducción de direcciones de red (NAT) cuando utilice políticas de exportación dinámicas. Con NAT, el controlador de almacenamiento ve la dirección NAT de front-end y no la dirección IP real del host, por lo que el acceso se denegará cuando no se encuentre ninguna coincidencia en las reglas de exportación.

Ejemplo

Existen dos opciones de configuración que deben utilizarse. Aquí tenéis un ejemplo de definición de 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
```

> Al utilizar esta función, debe asegurarse de que la unión raíz en su SVM tenga una política de exportación creada previamente con una regla de exportación que permita el bloque CIDR del nodo (como la política de exportación predeterminada). Siga siempre las mejores prácticas recomendadas NetApp para dedicar una SVM a Trident.

Aquí tienes una explicación de cómo funciona esta función utilizando el ejemplo anterior:

- `autoExportPolicy`está configurado para `true`. Esto indica que Trident crea una política de exportación para cada volumen aprovisionado con este backend para el `svm1` SVM y gestionar la adición y eliminación de reglas utilizando `autoexportCIDRs` bloques de direcciones. Hasta que un volumen se conecta a un nodo, el volumen utiliza una política de exportación vacía sin reglas para evitar el acceso no deseado a ese volumen. Cuando se publica un volumen en un nodo, Trident crea una política de exportación con el mismo nombre que el qtree subyacente que contiene la IP del nodo dentro del bloque CIDR especificado. Estas direcciones IP también se añadirán a la política de exportación utilizada por el FlexVol volume principal.
  - Por ejemplo:
    - UUID del backend 403b5326-8482-40db-96d0-d83fb3f4daec
    - `autoExportPolicy`empezar a `true`
    - prefijo de almacenamiento `trident`
    - UUID de PVC a79bcf5f-7b6d-4a40-9876-e2551f159c1c
    - El qtree denominado trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c crea una política de exportación para el FlexVol denominado `trident-403b5326-8482-40db96d0-d83fb3f4daec`, una política de exportación para el qtree llamado `trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c` y una política de exportación vacía llamada `trident_empty` en la SVM. Las reglas para la política de exportación de FlexVol serán un superconjunto de cualquier regla contenida en las políticas de exportación de qtree. La política de exportación vacía será reutilizada por cualquier volumen que no esté adjunto.
- `autoExportCIDRs`Contiene una lista de bloques de direcciones. Este campo es opcional y por defecto es ["0.0.0.0/0", "::/0"]. Si no se define, Trident agrega todas las direcciones unicast de ámbito global que se encuentren en los nodos de trabajo con publicaciones.

En este ejemplo, el `192.168.0.0/24` Se proporciona espacio de direcciones. Esto indica que las direcciones IP de los nodos de Kubernetes que se encuentren dentro de este rango de direcciones con publicaciones se agregarán a la política de exportación que crea Trident . Cuando Trident registra un nodo en el que se ejecuta, recupera las direcciones IP del nodo y las compara con los bloques de direcciones proporcionados en `autoExportCIDRs` En el momento de la publicación, después de filtrar las direcciones IP, Trident crea las reglas de política de exportación para las direcciones IP de los clientes del nodo al que está publicando.

Puedes actualizar `autoExportPolicy` y `autoExportCIDRs` para los backends después de crearlos. Puede agregar nuevos CIDR para un backend que se administra automáticamente o eliminar los CIDR existentes. Tenga cuidado al eliminar CIDR para asegurarse de que no se pierdan las conexiones existentes. También puedes optar por desactivar `autoExportPolicy` para un backend y recurrir a una política de exportación creada manualmente. Esto requerirá configurar el `exportPolicy` parámetro en la configuración de tu backend.

Después de que Trident crea o actualiza un backend, puede comprobar el backend mediante `tridentctl` o el correspondiente `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
```

Cuando se elimina un nodo, Trident revisa todas las políticas de exportación para eliminar las reglas de acceso correspondientes al nodo. Al eliminar la IP de este nodo de las políticas de exportación de los backends administrados, Trident evita montajes no autorizados, a menos que esta IP sea reutilizada por un nuevo nodo en el clúster.

Para los backends existentes, actualizar el backend con `tridentctl update backend` garantiza que Trident gestione automáticamente las políticas de exportación. Esto crea dos nuevas políticas de exportación que reciben el nombre del UUID del backend y del nombre del qtree cuando sea necesario. Los volúmenes que se encuentren en el backend utilizarán las políticas de exportación recién creadas después de desmontarlos y volverlos a montar.

> Eliminar un backend con políticas de exportación autogestionadas eliminará la política de exportación creada dinámicamente. Si se vuelve a crear el backend, se tratará como un backend nuevo y dará lugar a la creación de una nueva política de exportación.

Si se actualiza la dirección IP de un nodo activo, debe reiniciar el pod de Trident en el nodo. A continuación, Trident actualizará la política de exportación para los backends que administra para reflejar este cambio de IP.

Preparar el aprovisionamiento de volúmenes SMB

Con un poco de preparación adicional, puede aprovisionar volúmenes SMB usando `ontap-nas` conductores.

> Debes configurar los protocolos NFS y SMB/CIFS en la SVM para crear una `ontap-nas-economy` Volumen SMB para clústeres ONTAP locales. Si no se configura alguno de estos protocolos, la creación del volumen SMB fallará.

`autoExportPolicy`No es compatible con volúmenes SMB.

Antes de empezar

Antes de poder aprovisionar volúmenes SMB, debe tener lo siguiente.

- Un clúster de Kubernetes con un nodo controlador Linux y al menos un nodo de trabajo Windows que ejecuta Windows Server 2022. Trident solo admite volúmenes SMB montados en pods que se ejecutan en nodos Windows.
- Al menos un secreto de Trident que contenga sus credenciales de Active Directory. Para generar secretos `smbcreds` :

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

- Un proxy CSI configurado como servicio de Windows. Para configurar un `csi-proxy` , consulte a"GitHub: Proxy CSI" o"GitHub: Proxy CSI para Windows" para nodos de Kubernetes que se ejecutan en Windows.

Pasos

1. Para ONTAP local, opcionalmente puede crear un recurso compartido SMB o Trident puede crearlo por usted.

   Se requieren recursos compartidos SMB para Amazon FSx para ONTAP.

   Puedes crear los recursos compartidos de administración SMB de dos maneras: utilizando…"Consola de administración de Microsoft" Complemento de carpetas compartidas o mediante la CLI de ONTAP . Para crear los recursos compartidos SMB mediante la CLI de ONTAP :

   a. Si es necesario, cree la estructura de rutas de directorio para el recurso compartido.

      El `vserver cifs share create` El comando verifica la ruta especificada en la opción -path durante la creación del recurso compartido. Si la ruta especificada no existe, el comando falla.

   b. Cree un recurso compartido SMB asociado con la SVM especificada:

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

   c. Verifique que se haya creado el recurso compartido:

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

   Referirse a"Crear un recurso compartido SMB" Para más detalles.

2. Al crear el backend, debe configurar lo siguiente para especificar los volúmenes SMB. Para conocer todas las opciones de configuración del backend de FSx para ONTAP , consulte"Opciones de configuración y

ejemplos de FSx para ONTAP" .

Parámetro	Descripción	Ejemplo
`smbShare`	Puede especificar una de las siguientes opciones: el nombre de un recurso compartido SMB creado mediante la Consola de administración de Microsoft o la CLI de ONTAP ; un nombre para permitir que Trident cree el recurso compartido SMB; o puede dejar el parámetro en blanco para evitar el acceso compartido común a los volúmenes. Este parámetro es opcional para ONTAP local. Este parámetro es obligatorio para los backends de Amazon FSx para ONTAP y no puede estar en blanco.	`smb-share`
`nasType`	**Debe configurarse en `smb` .** Si es nulo, el valor predeterminado es `nfs` .	`smb`
`securityStyle`	Estilo de seguridad para nuevos volúmenes. **Debe configurarse en `ntfs` o `mixed` para volúmenes SMB.**	`ntfs`o `mixed` para volúmenes SMB
`unixPermissions`	Modo para nuevos volúmenes. **Debe dejarse vacío para volúmenes SMB.**	""

Habilitar SMB seguro

A partir de la versión 25.06, NetApp Trident admite el aprovisionamiento seguro de volúmenes SMB creados mediante `ontap-nas` y `ontap-nas-economy` backends. Cuando SMB seguro está habilitado, puede proporcionar acceso controlado a los recursos compartidos SMB para usuarios y grupos de usuarios de Active Directory (AD) mediante listas de control de acceso (ACL).

Puntos para recordar

- Importador `ontap-nas-economy` No se admiten volúmenes.
- Solo se admiten clones de solo lectura para `ontap-nas-economy` volúmenes.
- Si Secure SMB está habilitado, Trident ignorará el recurso compartido SMB mencionado en el backend.
- La actualización de la anotación PVC, la anotación de la clase de almacenamiento y el campo backend no actualiza la ACL del recurso compartido SMB.
- Las ACL de recursos compartidos SMB especificadas en la anotación del PVC clonado tendrán prioridad sobre las del PVC de origen.
- Asegúrese de proporcionar usuarios de AD válidos al habilitar SMB seguro. Los usuarios no válidos no se agregarán a la ACL.
- Si se proporciona el mismo usuario de AD en el backend, la clase de almacenamiento y el PVC con diferentes permisos, la prioridad de permisos será: PVC, clase de almacenamiento y, por último, backend.
- Se admite Secure SMB para `ontap-nas` Se aplica a las importaciones de volumen gestionadas y no a las importaciones de volumen no gestionadas.

Pasos

1. Especifique adAdminUser en TridentBackendConfig como se muestra en el siguiente ejemplo:

```
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. Agregue la anotación en la clase de almacenamiento.

   Añade el `trident.netapp.io/smbShareAdUser` Anotación a la clase de almacenamiento para habilitar SMB seguro sin fallos. El valor de usuario especificado para la anotación `trident.netapp.io/smbShareAdUser` debe ser el mismo que el nombre de usuario especificado en el `smbcreds` secreto. Puedes elegir una de las siguientes opciones para `smbShareAdUserPermission` : `full_control` , `change` , o `read` . El permiso predeterminado es `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. Crea un tubo de PVC.

   El siguiente ejemplo 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
```

Opciones y ejemplos de configuración de ONTAP NAS

Aprenda a crear y utilizar controladores NAS ONTAP con su instalación de Trident . Esta sección proporciona ejemplos de configuración de backend y detalles para mapear backends a StorageClasses.

Opciones de configuración del backend

Consulte la siguiente tabla para ver las opciones de configuración del backend:

Parámetro	Descripción	Por defecto
`version`		Siempre 1
`storageDriverName`	Nombre del controlador de almacenamiento	`ontap-nas`, `ontap-nas-economy` , o `ontap-nas-flexgroup`
`backendName`	Nombre personalizado o el backend de almacenamiento	Nombre del controlador + "_" + dataLIF
`managementLIF`	Dirección IP de un clúster o LIF de administración de SVM Se puede especificar un nombre de dominio completo (FQDN). Se puede configurar para usar direcciones IPv6 si Trident se instaló usando la bandera IPv6. Las direcciones IPv6 deben definirse entre corchetes, como por ejemplo: `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . Para una transición fluida a MetroCluster , consulteEjemplo de MetroCluster .	"10.0.0.1", "[2001:1234:abcd::fefe]"

Parámetro	Descripción	Por defecto
`dataLIF`	Dirección IP del protocolo LIF. NetApp recomienda especificar `dataLIF` . Si no se proporcionan, Trident obtiene los dataLIF del SVM. Puede especificar un nombre de dominio completo (FQDN) para usarlo en las operaciones de montaje NFS, lo que le permite crear un DNS round-robin para equilibrar la carga entre varios dataLIF. Puede modificarse después de la configuración inicial. Referirse a . Se puede configurar para usar direcciones IPv6 si Trident se instaló usando la bandera IPv6. Las direcciones IPv6 deben definirse entre corchetes, como por ejemplo: `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` . **Omitir para Metrocluster.** Ver elEjemplo de MetroCluster .	Dirección especificada o derivada de SVM, si no se especifica (no recomendado).
`svm`	Máquina virtual de almacenamiento a utilizar **Omitir para Metrocluster.** Ver elEjemplo de MetroCluster .	Derivado si se trata de una SVM `managementLIF` se especifica
`autoExportPolicy`	Habilitar la creación y actualización automática de políticas de exportación [Booleano]. Utilizando el `autoExportPolicy` y `autoExportCIDRs` Con algunas opciones, Trident puede gestionar las políticas de exportación automáticamente.	FALSO
`autoExportCIDRs`	Lista de CIDR para filtrar las direcciones IP de los nodos de Kubernetes cuando `autoExportPolicy` está habilitado. Utilizando el `autoExportPolicy` y `autoExportCIDRs` Con algunas opciones, Trident puede gestionar las políticas de exportación automáticamente.	["0.0.0.0/0", "::/0"]`
`labels`	Conjunto de etiquetas arbitrarias con formato JSON para aplicar a los volúmenes	""
`clientCertificate`	Valor codificado en Base64 del certificado del cliente. Se utiliza para la autenticación basada en certificados.	""
`clientPrivateKey`	Valor codificado en Base64 de la clave privada del cliente. Se utiliza para la autenticación basada en certificados.	""
`trustedCACertificate`	Valor codificado en Base64 del certificado de CA de confianza. Opcional. Se utiliza para la autenticación basada en certificados.	""
`username`	Nombre de usuario para conectarse al clúster/SVM. Se utiliza para la autenticación basada en credenciales. Para la autenticación de Active Directory, consulte "Autenticar Trident en un SVM backend mediante credenciales de Active Directory".	

Parámetro	Descripción	Por defecto
`password`	Contraseña para conectarse al cluster/SVM. Se utiliza para la autenticación basada en credenciales. Para la autenticación de Active Directory, consulte "Autenticar Trident en un SVM backend mediante credenciales de Active Directory".	
`storagePrefix`	Prefijo utilizado al aprovisionar nuevos volúmenes en la SVM. No se puede actualizar después de configurarlo. Cuando se utiliza ontap-nas-economy y un prefijo de almacenamiento de 24 caracteres o más, los qtrees no tendrán el prefijo de almacenamiento incrustado, aunque sí estará en el nombre del volumen.	"tridente"
`aggregate`	Agregado para aprovisionamiento (opcional; si se establece, debe asignarse a la SVM). Para el `ontap-nas-flexgroup` conductor, esta opción se ignora. Si no se asigna, cualquiera de los agregados disponibles se puede utilizar para aprovisionar un volumen FlexGroup . Cuando se actualiza el agregado en SVM, se actualiza automáticamente en Trident mediante sondeos a SVM sin necesidad de reiniciar el controlador Trident . Cuando se ha configurado un agregado específico en Trident para aprovisionar volúmenes, si el agregado se renombra o se mueve fuera del SVM, el backend pasará a un estado de error en Trident mientras consulta el agregado del SVM. Debe cambiar el agregado por uno que esté presente en la SVM o eliminarlo por completo para volver a poner en línea el backend.	""
`limitAggregateUsage`	Fallará el aprovisionamiento si el uso supera este porcentaje. **No se aplica a Amazon FSx para ONTAP**.	" (no se aplica por defecto)

Parámetro	Descripción	Por defecto
Lista agregada de flexgroup	Lista de agregados para el aprovisionamiento (opcional; si se establece, debe asignarse a la SVM). Todos los agregados asignados al SVM se utilizan para aprovisionar un volumen FlexGroup . Compatible con el controlador de almacenamiento **ontap-nas-flexgroup**. Cuando se actualiza la lista agregada en SVM, la lista se actualiza automáticamente en Trident mediante sondeos a SVM sin necesidad de reiniciar el controlador Trident . Cuando se ha configurado una lista agregada específica en Trident para aprovisionar volúmenes, si la lista agregada se renombra o se mueve fuera de SVM, el backend pasará a un estado de error en Trident mientras consulta el agregado de SVM. Debe cambiar la lista agregada por una que esté presente en la SVM o eliminarla por completo para volver a poner en línea el backend.	""
`limitVolumeSize`	Fallará el aprovisionamiento si el tamaño de volumen solicitado supera este valor. También restringe el tamaño máximo de los volúmenes que administra para los qtrees, y el `qtreesPerFlexvol` Esta opción permite personalizar el número máximo de qtrees por FlexVol volume.	" (no se aplica por defecto)
`debugTraceFlags`	Indicadores de depuración para usar al solucionar problemas. Ejemplo: {"api":false, "method":true} No usar `debugTraceFlags` a menos que esté solucionando problemas y necesite un registro detallado.	nulo
`nasType`	Configure la creación de volúmenes NFS o SMB. Las opciones son `nfs` , `smb` o nulo. Si se establece en nulo, se utilizarán volúmenes NFS por defecto.	`nfs`

Parámetro	Descripción	Por defecto
`nfsMountOptions`	Lista de opciones de montaje NFS separadas por comas. Las opciones de montaje para volúmenes persistentes de Kubernetes normalmente se especifican en las clases de almacenamiento, pero si no se especifican opciones de montaje en una clase de almacenamiento, Trident recurrirá a las opciones de montaje especificadas en el archivo de configuración del backend de almacenamiento. Si no se especifican opciones de montaje en la clase de almacenamiento o en el archivo de configuración, Trident no establecerá ninguna opción de montaje en un volumen persistente asociado.	""
`qtreesPerFlexvol`	Número máximo de Qtrees por FlexVol, debe estar en el rango [50, 300]	"200"
`smbShare`	Puede especificar una de las siguientes opciones: el nombre de un recurso compartido SMB creado mediante la Consola de administración de Microsoft o la CLI de ONTAP ; un nombre para permitir que Trident cree el recurso compartido SMB; o puede dejar el parámetro en blanco para evitar el acceso compartido común a los volúmenes. Este parámetro es opcional para ONTAP local. Este parámetro es obligatorio para los backends de Amazon FSx para ONTAP y no puede estar en blanco.	`smb-share`
`useREST`	Parámetro booleano para utilizar las API REST de ONTAP . `useREST`Cuando se configura para `true` Trident utiliza las API REST de ONTAP para comunicarse con el backend; cuando se configura en `false` Trident utiliza llamadas ONTAPI (ZAPI) para comunicarse con el backend. Esta función requiere ONTAP 9.11.1 y versiones posteriores. Además, el rol de inicio de sesión de ONTAP utilizado debe tener acceso a `ontapi` solicitud. Esto se satisface mediante lo predefinido. `vsadmin` y `cluster-admin` roles. A partir de la versión Trident 24.06 y ONTAP 9.15.1 o posterior, `useREST` está configurado para `true` por defecto; cambiar `useREST` a `false` para utilizar llamadas ONTAPI (ZAPI).	`true`para ONTAP 9.15.1 o posterior, de lo contrario `false`.
`limitVolumePoolSize`	Tamaño máximo de FlexVol que se puede solicitar al usar Qtrees en el backend ontap-nas-economy.	" (no se aplica por defecto)
`denyNewVolumePools`	Restringe `ontap-nas-economy` backends que crean nuevos volúmenes FlexVol para contener sus Qtrees. Solo se utilizan Flexvols preexistentes para el aprovisionamiento de nuevos PV.	

Parámetro	Descripción	Por defecto
adAdminUser	Usuario o grupo de usuarios administradores de Active Directory con acceso completo a los recursos compartidos SMB. Utilice este parámetro para otorgar derechos de administrador al recurso compartido SMB con control total.	

Opciones de configuración de backend para el aprovisionamiento de volúmenes

Puedes controlar el aprovisionamiento predeterminado utilizando estas opciones en el `defaults` sección de la configuración. Para ver un ejemplo, consulte los ejemplos de configuración a continuación.

Parámetro	Descripción	Por defecto
spaceAllocation	Asignación de espacio para Qtrees	"verdadero"
spaceReserve	Modo de reserva de espacio: "ninguno" (delgado) o "volumen" (grueso).	"ninguno"
snapshotPolicy	Política de instantáneas a utilizar	"ninguno"
qosPolicy	Grupo de políticas QoS que se asignará a los volúmenes creados. Elija una de las opciones qosPolicy o adaptiveQosPolicy por grupo de almacenamiento/backend.	""
adaptiveQosPolicy	Grupo de políticas QoS adaptativas para asignar a los volúmenes creados. Elija una de las opciones qosPolicy o adaptiveQosPolicy por grupo de almacenamiento/backend. No compatible con ontap-nas-economy.	""
snapshotReserve	Porcentaje de volumen reservado para instantáneas	"0" si `snapshotPolicy` es "ninguno", de lo contrario ""
splitOnClone	Separar un clon de su progenitor al crearlo	"FALSO"
encryption	Habilite el cifrado de volumen de NetApp (NVE) en el nuevo volumen; el valor predeterminado es `false` . Para utilizar esta opción, NVE debe estar licenciado y habilitado en el clúster. Si NAE está habilitado en el backend, cualquier volumen aprovisionado en Trident tendrá NAE habilitado. Para obtener más información, consulte:"Cómo funciona Trident con NVE y NAE" .	"FALSO"
tieringPolicy	Política de niveles para usar "ninguno"	
unixPermissions	Modo para nuevos volúmenes	"777" para volúmenes NFS; vacío (no aplicable) para volúmenes SMB
snapshotDir	Controla el acceso a `.snapshot` directorio	"verdadero" para NFSv4, "falso" para NFSv3
exportPolicy	Política de exportación a utilizar	"por defecto"

Parámetro	Descripción	Por defecto
`securityStyle`	Estilo de seguridad para nuevos volúmenes. NFS admite `mixed` y `unix` Estilos de seguridad. Las PYMES son compatibles con el soporte. `mixed` y `ntfs` Estilos de seguridad.	El valor predeterminado de NFS es `unix` . El valor predeterminado de SMB es `ntfs` .
`nameTemplate`	Plantilla para crear nombres de volumen personalizados.	""

El uso de grupos de políticas QoS con Trident requiere ONTAP 9.8 o posterior. Debe utilizar un grupo de políticas QoS no compartido y asegurarse de que el grupo de políticas se aplique a cada componente individualmente. Un grupo de políticas QoS compartidas impone un límite máximo al rendimiento total de todas las cargas de trabajo.

Ejemplos de aprovisionamiento por volumen

Aquí tenéis un ejemplo con valores predeterminados definidos:

```
---
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"
```

Para `ontap-nas` y `ontap-nas-flexgroups` Trident ahora utiliza un nuevo cálculo para garantizar que el FlexVol tenga el tamaño correcto con el porcentaje de snapshotReserve y el PVC. Cuando el usuario solicita un PVC, Trident crea el FlexVol original con más espacio mediante el nuevo cálculo. Este cálculo garantiza que el usuario reciba el espacio de escritura solicitado en el PVC, y no menos espacio del solicitado. Antes de

la versión v21.07, cuando el usuario solicitaba un PVC (por ejemplo, 5 GiB), con el snapshotReserve al 50 %, obtenía solo 2,5 GiB de espacio de escritura. Esto se debe a que lo que el usuario solicitó fue el volumen completo y `snapshotReserve` es un porcentaje de eso. Con Trident 21.07, lo que el usuario solicita es el espacio de escritura y Trident define el `snapshotReserve` número como porcentaje del volumen total. Esto no se aplica a `ontap-nas-economy` . Vea el siguiente ejemplo para ver cómo funciona

El cálculo es el siguiente:

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

Para snapshotReserve = 50% y la solicitud de PVC = 5 GiB, el tamaño total del volumen es 5/.5 = 10 GiB y el tamaño disponible es 5 GiB, que es lo que el usuario solicitó en la solicitud de PVC El `volume show` El comando debería mostrar resultados similares a este ejemplo:

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

Los backends existentes de instalaciones anteriores aprovisionarán volúmenes como se explicó anteriormente al actualizar Trident. Para los volúmenes creados antes de la actualización, debe redimensionarlos para que se observe el cambio. Por ejemplo, un PVC de 2 GiB con `snapshotReserve=50` Anteriormente se obtuvo un volumen que proporciona 1 GiB de espacio de escritura. Por ejemplo, al redimensionar el volumen a 3 GiB, la aplicación obtiene 3 GiB de espacio de escritura en un volumen de 6 GiB.

Ejemplos de configuración mínima

Los siguientes ejemplos muestran configuraciones básicas que dejan la mayoría de los parámetros con sus valores predeterminados. Esta es la forma más sencilla de definir un backend.

> Si está utilizando Amazon FSx en NetApp ONTAP con Trident, se recomienda especificar nombres DNS para las LIF en lugar de direcciones IP.

Ejemplo de economía NAS de ONTAP

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

Ejemplo de 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
```

Ejemplo de MetroCluster

Puede configurar el backend para evitar tener que actualizar manualmente la definición del backend después del cambio de estado y el cambio de estado durante"replicación y recuperación de SVM" .

Para una conmutación y recuperación sin interrupciones, especifique el SVM utilizando `managementLIF` y omitir el `dataLIF` y `svm` parámetros. Por ejemplo:

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

Ejemplo de volúmenes 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
```

Ejemplo de autenticación basada en certificados

Este es un ejemplo mínimo de configuración de backend. `clientCertificate` , `clientPrivateKey` , y `trustedCACertificate` (opcional, si se utiliza una CA de confianza) se rellenan en `backend.json` y tome los valores codificados en base64 del certificado del cliente, la clave privada y el certificado de CA de confianza, respectivamente.

```
---
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_
```

Ejemplo de política de exportación automática

Este ejemplo muestra cómo puede configurar Trident para que utilice políticas de exportación dinámicas para crear y gestionar automáticamente la política de exportación. Esto funciona igual para el `ontap-nas-economy` y `ontap-nas-flexgroup` conductores.

```
---
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
```

Ejemplo de direcciones IPv6

Este ejemplo muestra `managementLIF` utilizando una dirección 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
```

Ejemplo de Amazon FSx para ONTAP con volúmenes SMB

El `smbShare` Este parámetro es necesario para FSx para ONTAP que utiliza volúmenes 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_
```

Ejemplo de configuración de backend con plantilla de nombre

```
---
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}}"
```

Ejemplos de backends con pools virtuales

En los archivos de definición de backend de ejemplo que se muestran a continuación, se establecen valores predeterminados específicos para todos los grupos de almacenamiento, tales como: `spaceReserve` en ninguno, `spaceAllocation` en falso, y `encryption` en falso. Los grupos virtuales se definen en la sección de almacenamiento.

Trident establece las etiquetas de aprovisionamiento en el campo "Comentarios". Los comentarios están configurados en FlexVol para `ontap-nas` o FlexGroup para `ontap-nas-flexgroup` . Trident copia todas las etiquetas presentes en un grupo virtual al volumen de almacenamiento durante el aprovisionamiento. Para mayor comodidad, los administradores de almacenamiento pueden definir etiquetas por grupo virtual y agrupar volúmenes por etiqueta.

En estos ejemplos, algunos de los grupos de almacenamiento establecen sus propias configuraciones. `spaceReserve` , `spaceAllocation` , y `encryption` valores, y algunos pools anulan los valores predeterminados.

Ejemplo de 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"
```

Ejemplo de 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"
```

Ejemplo de economía NAS de 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"
```

Asignar backends a StorageClasses

Las siguientes definiciones de StorageClass hacen referencia aEjemplos de backends con pools virtuales . Utilizando el `parameters.selector` En cada campo, cada StorageClass especifica qué grupos virtuales se pueden usar para alojar un volumen. El volumen tendrá los aspectos definidos en el pool virtual elegido.

- El `protection-gold` StorageClass se asignará al primer y segundo grupo virtual en el `ontap-nas-flexgroup` backend. Estas son las únicas piscinas que ofrecen protección de nivel oro.

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

- El `protection-not-gold` StorageClass se asignará al tercer y cuarto grupo virtual en el `ontap-nas-flexgroup` backend. Estas son las únicas piscinas que ofrecen un nivel de protección distinto al oro.

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

- El `app-mysqldb` StorageClass se asignará al cuarto grupo virtual en el `ontap-nas` backend. Este es el único pool que ofrece configuración de pool de almacenamiento para aplicaciones de tipo mysqldb.

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

- El `protection-silver-creditpoints-20k` StorageClass se asignará al tercer grupo virtual en el `ontap-nas-flexgroup` backend. Este es el único fondo que ofrece protección de nivel plata y 20000 puntos de crédito.

```
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"
```

- El `creditpoints-5k` StorageClass se asignará al tercer grupo virtual en el `ontap-nas` backend y el segundo grupo virtual en el `ontap-nas-economy` backend. Estas son las únicas ofertas de pool con 5000 puntos de crédito.

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

Trident decidirá qué grupo virtual se selecciona y garantiza que se cumplan los requisitos de almacenamiento.

Actualizar `dataLIF` después de la configuración inicial

Puede cambiar el dataLIF después de la configuración inicial ejecutando el siguiente comando para proporcionar el nuevo archivo JSON de backend con el dataLIF actualizado.

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

Si los PVC están conectados a uno o varios pods, debe desconectar todos los pods correspondientes y luego volver a conectarlos para que el nuevo dataLIF surta efecto.

Ejemplos de seguridad para pymes

Configuración del backend con el controlador 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
```

Configuración de backend con el controlador 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
```

Configuración del backend con grupo de almacenamiento

```
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
```

Ejemplo de clase de almacenamiento con controlador 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
```

Asegúrese de agregar `annotations` para habilitar SMB seguro. El protocolo SMB seguro no funciona sin las anotaciones, independientemente de las configuraciones establecidas en el backend o en el PVC.

Ejemplo de clase de almacenamiento con controlador 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
```

Ejemplo de PVC con un solo usuario de 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
```

Ejemplo de PVC con múltiples usuarios de 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

Utilice Trident con Amazon FSx for NetApp ONTAP

"Amazon FSx for NetApp ONTAP"Es un servicio de AWS totalmente administrado que permite a los clientes lanzar y ejecutar sistemas de archivos basados en el sistema operativo de almacenamiento NetApp ONTAP . FSx para ONTAP le permite aprovechar las características, el rendimiento y las capacidades administrativas de NetApp con las que ya está familiarizado, al tiempo que se beneficia de la simplicidad, la agilidad, la seguridad y la escalabilidad del almacenamiento de datos en AWS. FSx para ONTAP admite las funciones del sistema de archivos y las API de administración de ONTAP .

Puede integrar su sistema de archivos Amazon FSx for NetApp ONTAP con Trident para garantizar que los clústeres de Kubernetes que se ejecutan en Amazon Elastic Kubernetes Service (EKS) puedan aprovisionar volúmenes persistentes de bloques y archivos respaldados por ONTAP.

En Amazon FSx, el sistema de archivos es el recurso principal, análogo a un clúster ONTAP local. Dentro de cada SVM puedes crear uno o varios volúmenes, que son contenedores de datos que almacenan los archivos

y carpetas de tu sistema de archivos. Con Amazon FSx for NetApp ONTAP se proporcionará como un sistema de archivos administrado en la nube. El nuevo tipo de sistema de archivos se llama * NetApp ONTAP*.

Al usar Trident con Amazon FSx for NetApp ONTAP, puede garantizar que los clústeres de Kubernetes que se ejecutan en Amazon Elastic Kubernetes Service (EKS) puedan aprovisionar volúmenes persistentes de bloques y archivos respaldados por ONTAP.

Requisitos

Además de"Requisitos de Trident" Para integrar FSx para ONTAP con Trident, necesitas:

- Un clúster de Amazon EKS existente o un clúster de Kubernetes autogestionado con `kubectl` instalado.
- Una máquina virtual de almacenamiento y sistema de archivos (SVM) Amazon FSx for NetApp ONTAP existente a la que se pueda acceder desde los nodos de trabajo de su clúster.
- Nodos de trabajo que están preparados para"NFS o iSCSI" .

> Asegúrese de seguir los pasos de preparación de nodos necesarios para Amazon Linux y Ubuntu. "Imágenes de máquinas de Amazon" (AMI) dependiendo de su tipo de AMI EKS.

Consideraciones

- Volúmenes SMB:
  - Los volúmenes SMB son compatibles mediante el uso de `ontap-nas` Solo el conductor.
  - Los volúmenes SMB no son compatibles con el complemento Trident EKS.
  - Trident solo admite volúmenes SMB montados en pods que se ejecutan en nodos Windows. Referirse a "Preparar el aprovisionamiento de volúmenes SMB" Para más detalles.
- Antes de Trident 24.02, los volúmenes creados en sistemas de archivos Amazon FSx que tenían habilitadas las copias de seguridad automáticas no podían ser eliminados por Trident. Para evitar este problema en Trident 24.02 o posterior, especifique el `fsxFilesystemID` AWS `apiRegion` AWS `apikey` y AWS `secretKey` en el archivo de configuración de backend para AWS FSx para ONTAP.

> Si especificas un rol de IAM para Trident, puedes omitir la especificación del `apiRegion` , `apiKey` , y `secretKey` campos a Trident explícitamente. Para obtener más información, consulte"Opciones de configuración y ejemplos de FSx para ONTAP" .

Uso simultáneo del controlador Trident SAN/iSCSI y EBS-CSI

Si planea usar controladores ontap-san (por ejemplo, iSCSI) con AWS (EKS, ROSA, EC2 o cualquier otra instancia), la configuración de múltiples rutas requerida en los nodos podría entrar en conflicto con el controlador CSI de Amazon Elastic Block Store (EBS). Para garantizar que las funciones de rutas múltiples funcionen sin interferir con los discos EBS en el mismo nodo, debe excluir EBS en su configuración de rutas múltiples. Este ejemplo muestra un `multipath.conf` Archivo que incluye la configuración necesaria de Trident excluyendo los discos EBS del multipathing:

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

Autenticación

Trident ofrece dos modos de autenticación.

- Basado en credenciales (recomendado): Almacena las credenciales de forma segura en AWS Secrets Manager. Puedes utilizar el `fsxadmin` usuario para su sistema de archivos o el `vsadmin` Usuario configurado para su SVM.

> Trident espera ser gestionado como un `vsadmin` Usuario de SVM o como usuario con un nombre diferente que tenga el mismo rol. Amazon FSx for NetApp ONTAP tiene un `fsxadmin` usuario que es un reemplazo limitado del ONTAP `admin` Usuario del clúster. Recomendamos encarecidamente utilizar `vsadmin` con Trident.

- Basado en certificados: Trident se comunicará con la SVM en su sistema de archivos FSx utilizando un certificado instalado en su SVM.

Para obtener detalles sobre cómo habilitar la autenticación, consulte la autenticación correspondiente a su tipo de controlador:

- "Autenticación NAS de ONTAP"
- "Autenticación SAN de ONTAP"

Imágenes de máquina de Amazon (AMI) probadas

El clúster EKS admite varios sistemas operativos, pero AWS ha optimizado ciertas imágenes de máquina de Amazon (AMI) para contenedores y EKS. Las siguientes AMI se han probado con NetApp Trident 25.02.

IAM	NAS	Economía NAS	iSCSI	Economía iSCSI
AL2023_x86_64_STANDARD	Sí	Sí	Sí	Sí
AL2_x86_64	Sí	Sí	Sí*	Sí*
BOTTLEROCKET_x86_64	Sí**	Sí	N/A	N/A
AL2023_ARM_64_STANDARD	Sí	Sí	Sí	Sí
AL2_ARM_64	Sí	Sí	Sí*	Sí*

BOTTLEROCKET_ARM_64	Sí**	Sí	N/A	N/A

- * No se puede eliminar el PV sin reiniciar el nodo
- ** No funciona con NFSv3 con Trident versión 25.02.

Si la AMI que desea no aparece en esta lista, no significa que no sea compatible; simplemente significa que no se ha probado. Esta lista sirve de guía para las AMI que se sabe que funcionan.

Pruebas realizadas con:

- Versión de EKS: 1.32
- Método de instalación: Helm 25.06 y como complemento de AWS 25.06
- Para NAS se probaron tanto NFSv3 como NFSv4.1.
- Para SAN solo se probó iSCSI, no NVMe-oF.

Pruebas realizadas:

- Crear: Clase de almacenamiento, PVC, cápsula
- Eliminar: pod, pvc (normal, qtree/lun – económico, NAS con copia de seguridad de AWS)

Encuentra más información

- "Documentación de Amazon FSx for NetApp ONTAP"
- "Artículo de blog sobre Amazon FSx for NetApp ONTAP"

Crea un rol de IAM y un secreto de AWS

Puedes configurar los pods de Kubernetes para que accedan a los recursos de AWS autenticándose como un rol de AWS IAM en lugar de proporcionar credenciales explícitas de AWS.

Para autenticarse utilizando un rol de AWS IAM, debe tener un clúster de Kubernetes implementado utilizando EKS.

Crear secreto de AWS Secrets Manager

Dado que Trident emitirá API contra un servidor virtual FSx para administrar el almacenamiento por usted, necesitará credenciales para hacerlo. La forma segura de transmitir esas credenciales es mediante un secreto de AWS Secrets Manager. Por lo tanto, si aún no tiene uno, deberá crear un secreto de AWS Secrets Manager que contenga las credenciales de la cuenta vsadmin.

Este ejemplo crea un secreto de AWS Secrets Manager para almacenar las credenciales de Trident CSI:

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

Crear política de IAM

Trident también necesita permisos de AWS para funcionar correctamente. Por lo tanto, debe crear una política que otorgue a Trident los permisos que necesita.

Los siguientes ejemplos crean una política de IAM utilizando la CLI de AWS:

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

Ejemplo de JSON de política:

```
{
  "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"
}
```

Crear identidad de pod o rol de IAM para la asociación de cuenta de servicio (IRSA)

Puede configurar una cuenta de servicio de Kubernetes para que asuma un rol de AWS Identity and Access Management (IAM) con EKS Pod Identity o IAM role for Service account association (IRSA). Cualquier Pod que esté configurado para usar la cuenta de servicio podrá acceder a cualquier servicio de AWS al que el rol tenga permisos de acceso.

Identidad de pod

Las asociaciones de identidad de pods de Amazon EKS ofrecen la capacidad de administrar las credenciales de sus aplicaciones, de forma similar a como los perfiles de instancias de Amazon EC2 proporcionan credenciales a las instancias de Amazon EC2.

Instale Pod Identity en su clúster EKS:

Puede crear la identidad del Pod a través de la consola de AWS o utilizando el siguiente comando de la CLI de AWS:

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

Para obtener más información, consulte"Configurar el agente de identidad de pod de Amazon EKS" .

Crear trust-relationship.json:

Cree el archivo trust-relationship.json para permitir que la entidad de servicio de EKS asuma este rol para la identidad del pod. A continuación, cree un rol con esta política de confianza:

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

archivo trust-relationship.json:

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

Adjunte la política de rol al rol de IAM:

Asocie la política de rol del paso anterior al rol de IAM que se creó:

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

Crear una asociación de identidad de pod:

Cree una asociación de identidad de pod entre el rol de IAM y la cuenta de servicio de 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
```

Rol de IAM para la asociación de cuentas de servicio (IRSA)

Utilizando la CLI de AWS:

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

Archivo 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"
        }
      }
    }
  ]
}
```

Actualiza los siguientes valores en el `trust-relationship.json` archivo:

- **<account_id>** - Tu ID de cuenta de AWS
- **<oidc_provider>** - El OIDC de su clúster EKS. Puede obtener el proveedor oidc ejecutando:

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

Asocia el rol de IAM con la política de IAM:

Una vez creado el rol, adjúntele la política (creada en el paso anterior) mediante este comando:

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

Verificar que el proveedor de OICD esté asociado:

Verifique que su proveedor OIDC esté asociado con su clúster. Puedes verificarlo usando este comando:

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

Si la salida está vacía, utilice el siguiente comando para asociar IAM OIDC a su clúster:

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

Si utiliza eksctl, siga el siguiente ejemplo para crear un rol de IAM para la cuenta de servicio en 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
```

Instalar Trident

Trident simplifica la gestión del almacenamiento de Amazon FSx for NetApp ONTAP en Kubernetes para que sus desarrolladores y administradores puedan centrarse en el despliegue de aplicaciones.

Puedes instalar Trident utilizando uno de los siguientes métodos:

- Timón
- Complemento EKS

Si desea utilizar la funcionalidad de instantáneas, instale el complemento CSI snapshot controller. Referirse a"Habilitar la funcionalidad de instantáneas para volúmenes CSI" Para más información.

Instala Trident mediante Helm.

Identidad de pod

1. Añadir el repositorio Trident Helm:

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

2. Instala Trident siguiendo el siguiente ejemplo:

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

Puedes utilizar el `helm list` comando para revisar los detalles de la instalación, como nombre, espacio de nombres, gráfico, estado, versión de la aplicación y número de revisión.

```
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
```

Asociación de cuentas de servicio (IRSA)

1. Añadir el repositorio Trident Helm:

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

2. Establezca los valores para **proveedor de nube** e **identidad de nube**:

```
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
```

Puedes utilizar el `helm list` comando para revisar los detalles de la instalación, como nombre, espacio de nombres, gráfico, estado, versión de la aplicación y número de revisión.

```
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
```

Si planea utilizar iSCSI, asegúrese de que iSCSI esté habilitado en su máquina cliente. Si utiliza el sistema operativo del nodo de trabajo AL2023, puede automatizar la instalación del cliente iSCSI agregando el parámetro node prep en la instalación de helm:

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

Instala Trident mediante el complemento EKS.

El complemento Trident EKS incluye los últimos parches de seguridad, correcciones de errores y está validado por AWS para funcionar con Amazon EKS. El complemento EKS le permite garantizar de forma consistente que sus clústeres de Amazon EKS sean seguros y estables, y reduce la cantidad de trabajo que necesita realizar para instalar, configurar y actualizar complementos.

Prerrequisitos

Asegúrese de tener lo siguiente antes de configurar el complemento Trident para AWS EKS:

- Una cuenta de clúster de Amazon EKS con suscripción adicional
- Permisos de AWS para el mercado de AWS:
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- Tipo de AMI: Amazon Linux 2 (AL2_x86_64) o Amazon Linux 2 Arm (AL2_ARM_64)
- Tipo de nodo: AMD o ARM
- Un sistema de archivos Amazon FSx for NetApp ONTAP existente

Habilita el complemento Trident para AWS

Consola de administración

1. Abra la consola de Amazon EKS en https://console.aws.amazon.com/eks/home#/clusters .
2. En el panel de navegación izquierdo, seleccione **Clústeres**.
3. Seleccione el nombre del clúster para el que desea configurar el complemento NetApp Trident CSI.
4. Seleccione **Complementos** y luego seleccione **Obtener más complementos**.
5. Siga estos pasos para seleccionar el complemento:
   a. Desplácese hacia abajo hasta la sección **Complementos de AWS Marketplace** y escriba **"Trident"** en el cuadro de búsqueda.
   b. Seleccione la casilla de verificación en la esquina superior derecha del cuadro Trident by NetApp .
   c. Seleccione **Siguiente**.
6. En la página de configuración **Configurar complementos seleccionados**, haga lo siguiente:

   Omite estos pasos si estás usando la asociación de identidad de pod.

   a. Seleccione la **Versión** que desea utilizar.
   b. Si utiliza la autenticación IRSA, asegúrese de configurar los valores disponibles en la configuración opcional:
      - Seleccione la **Versión** que desea utilizar.
      - Siga el **Esquema de configuración del complemento** y configure el parámetro **configurationValues** en la sección **Valores de configuración** con el ARN del rol que creó en el paso anterior (el valor debe tener el siguiente formato):

```
{

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

}
```

   +
   Si selecciona Anular como método de resolución de conflictos, una o más de las configuraciones del complemento existente pueden sobrescribirse con la configuración del complemento de Amazon EKS. Si no habilita esta opción y hay un conflicto con su configuración existente, la operación fallará. Puede utilizar el mensaje de error resultante para solucionar el conflicto. Antes de seleccionar esta opción, asegúrese de que el complemento Amazon EKS no gestione configuraciones que usted deba administrar manualmente.

7. Seleccione **Siguiente**.
8. En la página **Revisar y agregar**, seleccione **Crear**.

   Una vez finalizada la instalación del complemento, verá el complemento instalado.

CLI de AWS

1. Crea el `add-on.json` archivo:

Para la identidad del pod, utilice el siguiente formato:

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

Para la autenticación IRSA, utilice el siguiente 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"
  }
}
```

Reemplazar `<role ARN>` con el ARN del rol que se creó en el paso anterior.

2. Instala el complemento Trident EKS.

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

eksctl

El siguiente comando de ejemplo instala el complemento Trident EKS:

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

Actualizar el complemento Trident EKS

Consola de administración

1. Abra la consola de Amazon EKS https://console.aws.amazon.com/eks/home#/clusters .
2. En el panel de navegación izquierdo, seleccione **Clústeres**.
3. Seleccione el nombre del clúster para el que desea actualizar el complemento NetApp Trident CSI.
4. Seleccione la pestaña **Complementos**.
5. Seleccione * Trident by NetApp* y luego seleccione **Editar**.
6. En la página **Configurar Trident de NetApp**, haga lo siguiente:
   a. Seleccione la **Versión** que desea utilizar.
   b. Amplíe la sección **Ajustes de configuración opcionales** y modifíquela según sea necesario.
   c. Seleccione **Guardar cambios**.

CLI de AWS

El siguiente ejemplo actualiza el complemento 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

- Comprueba la versión actual de tu complemento FSxN Trident CSI. Reemplazar `my-cluster` con el nombre de su clúster.

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

Ejemplo de salida:

```
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'"}
```

- Actualiza el complemento a la versión que se muestra en ACTUALIZACIÓN DISPONIBLE en el resultado del paso anterior.

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

Si eliminas el `--force` Si alguna de las opciones y alguna de las configuraciones del complemento de Amazon EKS entra en conflicto con su configuración existente, la actualización del complemento de Amazon EKS fallará; recibirá un mensaje de error para ayudarle a resolver el conflicto. Antes de especificar esta opción, asegúrese de que el complemento Amazon EKS no gestione configuraciones que usted necesite administrar, ya que estas configuraciones se sobrescriben con esta opción. Para obtener más información sobre otras opciones para esta configuración, consulte"Complementos" . Para obtener más información sobre la administración de campos de Amazon EKS Kubernetes, consulte"Gestión de campos de Kubernetes" .

Desinstala/elimina el complemento Trident EKS.

Tienes dos opciones para eliminar un complemento de Amazon EKS:

- **Conservar el software complementario en su clúster** – Esta opción elimina la administración de Amazon EKS de cualquier configuración. También elimina la capacidad de Amazon EKS para notificarle sobre actualizaciones y actualizar automáticamente el complemento de Amazon EKS después de que usted inicie una actualización. Sin embargo, conserva el software adicional en su clúster. Esta opción convierte el complemento en una instalación autogestionada, en lugar de un complemento de Amazon EKS. Con esta opción, no hay tiempo de inactividad para el complemento. Conservar el `--preserve` opción en el comando para conservar el complemento.
- **Elimine por completo el software complementario de su clúster** – NetApp recomienda que elimine el complemento Amazon EKS de su clúster solo si no hay recursos en su clúster que dependan de él. Quitar el `--preserve` opción de la `delete` comando para eliminar el complemento.

Si el complemento tiene una cuenta IAM asociada, dicha cuenta no se eliminará.

Consola de administración

1. Abra la consola de Amazon EKS en https://console.aws.amazon.com/eks/home#/clusters .
2. En el panel de navegación izquierdo, seleccione **Clústeres**.
3. Seleccione el nombre del clúster para el que desea eliminar el complemento NetApp Trident CSI.
4. Seleccione la pestaña **Complementos** y luego seleccione * Trident de NetApp*.
5. Seleccione **Eliminar**.
6. En el cuadro de diálogo **Eliminar confirmación de netapp_trident-operator**, haga lo siguiente:
   a. Si desea que Amazon EKS deje de administrar la configuración del complemento, seleccione **Conservar en el clúster**. Haz esto si quieres conservar el software adicional en tu clúster para poder gestionar tú mismo todas las configuraciones del complemento.
   b. Introduzca **netapp_trident-operator**.
   c. Seleccione **Eliminar**.

CLI de AWS

Reemplazar `my-cluster` con el nombre de su clúster y, a continuación, ejecute el siguiente comando.

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

eksctl

El siguiente comando desinstala el complemento Trident EKS:

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

Configurar el backend de almacenamiento

Integración de controladores SAN y NAS de ONTAP

Para crear un backend de almacenamiento, necesitas crear un archivo de configuración en formato JSON o YAML. El archivo debe especificar el tipo de almacenamiento que desea (NAS o SAN), el sistema de archivos, la SVM de la que se obtendrá y cómo autenticarse con ella. El siguiente ejemplo muestra cómo definir el almacenamiento basado en NAS y cómo usar un secreto de AWS para almacenar las credenciales de la SVM que desea usar:

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"
    }
  }
}
```

Ejecute los siguientes comandos para crear y validar la configuración del backend de Trident (TBC):

- Crea la configuración del backend de Trident (TBC) a partir del archivo YAML y ejecuta el siguiente comando:

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

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

- Validar que la configuración del backend de Trident (TBC) se creó correctamente:

```
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
```

Detalles del controlador FSx para ONTAP

Puede integrar Trident con Amazon FSx for NetApp ONTAP utilizando los siguientes controladores:

- `ontap-san`Cada PV aprovisionado es un LUN dentro de su propio volumen Amazon FSx for NetApp ONTAP . Recomendado para almacenamiento en bloque.
- `ontap-nas`Cada PV aprovisionado es un volumen completo de Amazon FSx for NetApp ONTAP . Recomendado para NFS y SMB.
- `ontap-san-economy`Cada PV aprovisionado es un LUN con un número configurable de LUN por volumen de Amazon FSx for NetApp ONTAP .
- `ontap-nas-economy`Cada PV aprovisionado es un qtree, con un número configurable de qtrees por volumen de Amazon FSx for NetApp ONTAP .
- `ontap-nas-flexgroup`Cada PV aprovisionado es un volumen completo de Amazon FSx for NetApp ONTAP FlexGroup .

Para obtener detalles sobre el conductor, consulte"Controladores NAS" y"Controladores SAN" .

Una vez creado el archivo de configuración, ejecute este comando para crearlo en su EKS:

```
kubectl create -f configuration_file
```

Para verificar el estado, ejecute este 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
```

Configuración avanzada del backend y ejemplos

Consulte la siguiente tabla para ver las opciones de configuración del backend:

Parámetro	Descripción	Ejemplo
`version`		Siempre 1
`storageDriverName`	Nombre del controlador de almacenamiento	`ontap-nas`, `ontap-nas-economy` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy`
`backendName`	Nombre personalizado o el backend de almacenamiento	Nombre del controlador + "_" + dataLIF
`managementLIF`	Dirección IP de un clúster o LIF de administración de SVM Se puede especificar un nombre de dominio completo (FQDN). Se puede configurar para usar direcciones IPv6 si Trident se instaló usando la bandera IPv6. Las direcciones IPv6 deben definirse entre corchetes, como por ejemplo [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]. Si usted proporciona el `fsxFilesystemID` bajo el `aws` campo, no es necesario que proporcione el `managementLIF` porque Trident recupera la SVM `managementLIF` Información de AWS. Por lo tanto, debe proporcionar las credenciales de un usuario en la SVM (por ejemplo: vsadmin) y el usuario debe tener `vsadmin` role.	"10.0.0.1", "[2001:1234:abcd::fefe]"

Parámetro	Descripción	Ejemplo
`dataLIF`	Dirección IP del protocolo LIF. * Controladores NAS ONTAP **: NetApp recomienda especificar dataLIF. Si no se proporcionan, Trident obtiene los dataLIF del SVM. Puede especificar un nombre de dominio completo (FQDN) para usarlo en las operaciones de montaje NFS, lo que le permite crear un DNS round-robin para equilibrar la carga en múltiples dataLIF. Puede modificarse después de la configuración inicial. Referirse a . * Controladores ONTAP SAN**: No especificar para iSCSI. Trident utiliza ONTAP Selective LUN Map para descubrir los LIF iSCI necesarios para establecer una sesión de múltiples rutas. Se genera una advertencia si dataLIF se define explícitamente. Se puede configurar para usar direcciones IPv6 si Trident se instaló usando la bandera IPv6. Las direcciones IPv6 deben definirse entre corchetes, como por ejemplo [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	
`autoExportPolicy`	Habilitar la creación y actualización automática de políticas de exportación [Booleano]. Utilizando el `autoExportPolicy` y `autoExportCIDRs` Con algunas opciones, Trident puede gestionar las políticas de exportación automáticamente.	`false`
`autoExportCIDRs`	Lista de CIDR para filtrar las direcciones IP de los nodos de Kubernetes cuando `autoExportPolicy` está habilitado. Utilizando el `autoExportPolicy` y `autoExportCIDRs` Con algunas opciones, Trident puede gestionar las políticas de exportación automáticamente.	"["0.0.0.0/0", "::/0"]"
`labels`	Conjunto de etiquetas arbitrarias con formato JSON para aplicar a los volúmenes	""

Parámetro	Descripción	Ejemplo
`clientCertificate`	Valor codificado en Base64 del certificado del cliente. Se utiliza para la autenticación basada en certificados.	""
`clientPrivateKey`	Valor codificado en Base64 de la clave privada del cliente. Se utiliza para la autenticación basada en certificados.	""
`trustedCACertificate`	Valor codificado en Base64 del certificado de CA de confianza. Opcional. Se utiliza para la autenticación basada en certificados.	""
`username`	Nombre de usuario para conectarse al clúster o SVM. Se utiliza para la autenticación basada en credenciales. Por ejemplo, vsadmin.	
`password`	Contraseña para conectarse al clúster o SVM. Se utiliza para la autenticación basada en credenciales.	
`svm`	máquina virtual de almacenamiento a utilizar	Se deriva si se especifica un SVM managementLIF.
`storagePrefix`	Prefijo utilizado al aprovisionar nuevos volúmenes en la SVM. No se puede modificar después de su creación. Para actualizar este parámetro, deberá crear un nuevo backend.	`trident`
`limitAggregateUsage`	**No especificar para Amazon FSx for NetApp ONTAP.** El proporcionado `fsxadmin` y `vsadmin` No contienen los permisos necesarios para recuperar el uso agregado y limitarlo mediante Trident.	No utilizar.
`limitVolumeSize`	Fallará el aprovisionamiento si el tamaño de volumen solicitado supera este valor. También restringe el tamaño máximo de los volúmenes que administra para qtrees y LUN, y el `qtreesPerFlexvol` Esta opción permite personalizar el número máximo de qtrees por FlexVol volume.	" (no se aplica por defecto)

Parámetro	Descripción	Ejemplo
`lunsPerFlexvol`	El número máximo de LUN por volumen de Flexvol debe estar en el rango [50, 200]. Solo SAN.	“100”
`debugTraceFlags`	Indicadores de depuración para usar al solucionar problemas. Ejemplo: {"api":false, "method":true} No usar `debugTraceFlags` a menos que esté solucionando problemas y necesite un registro detallado.	nulo
`nfsMountOptions`	Lista de opciones de montaje NFS separadas por comas. Las opciones de montaje para volúmenes persistentes de Kubernetes normalmente se especifican en las clases de almacenamiento, pero si no se especifican opciones de montaje en una clase de almacenamiento, Trident recurrirá a las opciones de montaje especificadas en el archivo de configuración del backend de almacenamiento. Si no se especifican opciones de montaje en la clase de almacenamiento o en el archivo de configuración, Trident no establecerá ninguna opción de montaje en un volumen persistente asociado.	""
`nasType`	Configure la creación de volúmenes NFS o SMB. Las opciones son `nfs` , `smb` , o nulo. **Debe configurarse en `smb` para volúmenes SMB.** Si se establece en nulo, se utilizarán volúmenes NFS por defecto.	`nfs`
`qtreesPerFlexvol`	Número máximo de Qtrees por FlexVol volume, debe estar en el rango [50, 300]	`"200"`
`smbShare`	Puede especificar una de las siguientes opciones: el nombre de un recurso compartido SMB creado mediante la Consola de administración de Microsoft o la CLI de ONTAP , o un nombre para permitir que Trident cree el recurso compartido SMB. Este parámetro es necesario para los backends de Amazon FSx para ONTAP .	`smb-share`

Parámetro	Descripción	Ejemplo
`useREST`	Parámetro booleano para utilizar las API REST de ONTAP . Cuando se configura para `true` Trident utilizará las API REST de ONTAP para comunicarse con el backend. Esta función requiere ONTAP 9.11.1 y versiones posteriores. Además, el rol de inicio de sesión de ONTAP utilizado debe tener acceso a `ontap` solicitud. Esto se satisface mediante lo predefinido. `vsadmin` y `cluster-admin` roles.	`false`
`aws`	En el archivo de configuración de AWS FSx para ONTAP puede especificar lo siguiente: `fsxFilesystemID` : Especifique el ID del sistema de archivos AWS FSx. - `apiRegion` Nombre de la región de la API de AWS. - `apikey` Clave de API de AWS. - `secretKey` Clave secreta de AWS.	`""` `""` `""`
`credentials`	Especifique las credenciales de FSx SVM que se almacenarán en AWS Secrets Manager. - `name` : Nombre de recurso de Amazon (ARN) del secreto, que contiene las credenciales de SVM. - `type` : Configurado a `awsarn` . Referirse a"Crea un secreto de AWS Secrets Manager" Para más información.	

Opciones de configuración de backend para el aprovisionamiento de volúmenes

Puedes controlar el aprovisionamiento predeterminado utilizando estas opciones en el `defaults` sección de la configuración. Para ver un ejemplo, consulte los ejemplos de configuración a continuación.

Parámetro	Descripción	Por defecto
`spaceAllocation`	Asignación de espacio para LUN	`true`
`spaceReserve`	Modo de reserva de espacio: "ninguno" (delgado) o "volumen" (grueso).	`none`
`snapshotPolicy`	Política de instantáneas a utilizar	`none`

Parámetro	Descripción	Por defecto
`qosPolicy`	Grupo de políticas QoS que se asignará a los volúmenes creados. Elija una de las opciones qosPolicy o adaptiveQosPolicy por grupo de almacenamiento o backend. El uso de grupos de políticas QoS con Trident requiere ONTAP 9.8 o posterior. Debe utilizar un grupo de políticas QoS no compartido y asegurarse de que el grupo de políticas se aplique a cada componente individualmente. Un grupo de políticas QoS compartidas impone un límite máximo al rendimiento total de todas las cargas de trabajo.	""
`adaptiveQosPolicy`	Grupo de políticas QoS adaptativas para asignar a los volúmenes creados. Elija una de las opciones qosPolicy o adaptiveQosPolicy por grupo de almacenamiento o backend. No compatible con ontap-nas-economy.	""
`snapshotReserve`	Porcentaje de volumen reservado para instantáneas "0"	Si `snapshotPolicy` es `none` , `else` ""
`splitOnClone`	Separar un clon de su progenitor al crearlo	`false`
`encryption`	Habilite el cifrado de volumen de NetApp (NVE) en el nuevo volumen; el valor predeterminado es `false` . Para utilizar esta opción, NVE debe estar licenciado y habilitado en el clúster. Si NAE está habilitado en el backend, cualquier volumen aprovisionado en Trident tendrá NAE habilitado. Para obtener más información, consulte:"Cómo funciona Trident con NVE y NAE" .	`false`
`luksEncryption`	Habilitar el cifrado LUKS. Referirse a"Utilice la configuración de clave unificada de Linux (LUKS)." . Solo SAN.	""
`tieringPolicy`	Política de niveles a utilizar `none`	
`unixPermissions`	Modo para nuevos volúmenes. **Dejar en blanco para volúmenes SMB.**	""

Parámetro	Descripción	Por defecto
`securityStyle`	Estilo de seguridad para nuevos volúmenes. NFS admite `mixed` y `unix` Estilos de seguridad. Las PYMES son compatibles con el soporte. `mixed` y `ntfs` Estilos de seguridad.	El valor predeterminado de NFS es `unix` . El valor predeterminado de SMB es `ntfs` .

Preparar el aprovisionamiento de volúmenes SMB

Puede aprovisionar volúmenes SMB utilizando `ontap-nas` conductor. Antes de completarIntegración de controladores SAN y NAS de ONTAP Complete los siguientes pasos.

Antes de empezar

Antes de poder aprovisionar volúmenes SMB utilizando el `ontap-nas` Conductor, usted debe tener lo siguiente.

- Un clúster de Kubernetes con un nodo controlador Linux y al menos un nodo de trabajo Windows que ejecuta Windows Server 2019. Trident solo admite volúmenes SMB montados en pods que se ejecutan en nodos Windows.
- Al menos un secreto de Trident que contenga sus credenciales de Active Directory. Para generar secretos `smbcreds` :

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

- Un proxy CSI configurado como servicio de Windows. Para configurar un `csi-proxy` , consulte a"GitHub: Proxy CSI" o"GitHub: Proxy CSI para Windows" para nodos de Kubernetes que se ejecutan en Windows.

Pasos

1. Crear recursos compartidos SMB. Puedes crear los recursos compartidos de administración SMB de dos maneras: utilizando…"Consola de administración de Microsoft" Complemento de carpetas compartidas o mediante la CLI de ONTAP . Para crear los recursos compartidos SMB mediante la CLI de ONTAP :
   a. Si es necesario, cree la estructura de rutas de directorio para el recurso compartido.

      El `vserver cifs share create` El comando verifica la ruta especificada en la opción -path durante la creación del recurso compartido. Si la ruta especificada no existe, el comando falla.

   b. Cree un recurso compartido SMB asociado con la SVM especificada:

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

   c. Verifique que se haya creado el recurso compartido:

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

Referirse a"Crear un recurso compartido SMB" Para más detalles.

2. Al crear el backend, debe configurar lo siguiente para especificar los volúmenes SMB. Para conocer todas las opciones de configuración del backend de FSx para ONTAP , consulte"Opciones de configuración y ejemplos de FSx para ONTAP" .

Parámetro	Descripción	Ejemplo
`smbShare`	Puede especificar una de las siguientes opciones: el nombre de un recurso compartido SMB creado mediante la Consola de administración de Microsoft o la CLI de ONTAP , o un nombre para permitir que Trident cree el recurso compartido SMB. Este parámetro es necesario para los backends de Amazon FSx para ONTAP .	`smb-share`
`nasType`	**Debe configurarse en `smb` .** Si es nulo, el valor predeterminado es `nfs` .	`smb`
`securityStyle`	Estilo de seguridad para nuevos volúmenes. **Debe configurarse en `ntfs` o `mixed` para volúmenes SMB.**	`ntfs`o `mixed` para volúmenes SMB
`unixPermissions`	Modo para nuevos volúmenes. **Debe dejarse vacío para volúmenes SMB.**	""

Configure una clase de almacenamiento y un PVC

Configure un objeto StorageClass de Kubernetes y cree la clase de almacenamiento para indicar a Trident cómo aprovisionar volúmenes. Cree un PersistentVolumeClaim (PVC) que utilice la StorageClass de Kubernetes configurada para solicitar acceso al PV. Luego puedes montar el panel fotovoltaico en un soporte.

Crear una clase de almacenamiento

Configurar un objeto StorageClass de Kubernetes

El "Objeto StorageClass de Kubernetes" El objeto identifica a Trident como el proveedor que se utiliza para esa clase e indica a Trident cómo aprovisionar un volumen. Utilice este ejemplo para configurar Storageclass para volúmenes que utilizan NFS (consulte la sección Atributos de Trident a continuación para obtener la lista completa de atributos):

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

Utilice este ejemplo para configurar Storageclass para volúmenes que utilizan 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"
```

Para aprovisionar volúmenes NFSv3 en AWS Bottlerocket, agregue los necesarios. `mountOptions` a la clase de almacenamiento:

```
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
```

Referirse a"Objetos de Kubernetes y Trident" Para obtener detalles sobre cómo interactúan las clases de almacenamiento con `PersistentVolumeClaim` y parámetros para controlar cómo Trident gestiona los volúmenes.

Crear una clase de almacenamiento

Pasos

1. Este es un objeto de Kubernetes, así que use `kubectl` para crearlo en Kubernetes.

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

2. Ahora debería ver una clase de almacenamiento **basic-csi** tanto en Kubernetes como en Trident, y Trident debería haber detectado los pools en el backend.

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

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

Crea el PVC

A "*Reclamación de volumen persistente*" (PVC) es una solicitud de acceso al PersistentVolume en el clúster.

El PVC se puede configurar para solicitar el almacenamiento de un tamaño o modo de acceso determinado. Mediante la StorageClass asociada, el administrador del clúster puede controlar más que el tamaño del PersistentVolume y el modo de acceso, como el rendimiento o el nivel de servicio.

Una vez creado el PVC, puede montar el volumen en un soporte.

Ejemplos de manifiestos

Muestra de PersistentVolumeClaim

Estos ejemplos muestran opciones básicas de configuración de PVC.

PVC con acceso RWX

Este ejemplo muestra un PVC básico con acceso RWX que está asociado a una StorageClass llamada `basic-csi` .

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

Ejemplo de PVC usando iSCSI

Este ejemplo muestra un PVC básico para iSCSI con acceso RWO que está asociado con una StorageClass llamada `protection-gold` .

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

Crear PVC

Pasos

1. Crear el PVC.

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

2. Verifique el estado del PVC.

```
kubectl get pvc
```

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

Referirse a"Objetos de Kubernetes y Trident" Para obtener detalles sobre cómo interactúan las clases de almacenamiento con `PersistentVolumeClaim` y parámetros para controlar cómo Trident gestiona los volúmenes.

Atributos del Trident

Estos parámetros determinan qué pools de almacenamiento gestionados por Trident deben utilizarse para aprovisionar volúmenes de un tipo determinado.

Atributo	Tipo	Valores	Oferta	Pedido	Con el apoyo de
medios[1]	cadena	disco duro, híbrido, SSD	La piscina contiene medios de este tipo; híbrido significa ambos	Tipo de medio especificado	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san
tipo de aprovisionamiento	cadena	delgado, grueso	Pool admite este método de aprovisionamiento.	Método de aprovisionamiento especificado	Espeso: todo Ontap; fino: todo Ontap y Solidfire-san
Tipo de backend	cadena	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san, gcp-cvs, azure-netapp-files, ontap-san-economy	Pool pertenece a este tipo de backend.	Backend especificado	Todos los conductores
instantáneas	bool	verdadero, falso	El pool admite volúmenes con instantáneas.	Volumen con instantáneas habilitadas	ontap-nas, ontap-san, solidfire-san, gcp-cvs
clones	bool	verdadero, falso	Pool admite la clonación de volúmenes.	Volumen con clones habilitado	ontap-nas, ontap-san, solidfire-san, gcp-cvs

Atributo	Tipo	Valores	Oferta	Pedido	Con el apoyo de
cifrado	bool	verdadero, falso	Pool admite volúmenes cifrados	Volumen con cifrado habilitado	ontap-nas, ontap-nas-economy, ontap-nas-flexgroups, ontap-san
IOPS	int	entero positivo	Pool es capaz de garantizar IOPS en este rango.	El volumen garantizaba estas IOPS	solidfire-san

[1]: No compatible con los sistemas ONTAP Select

Implementar aplicación de muestra

Una vez creadas la clase de almacenamiento y el PVC, puede montar el PV en un pod. Esta sección enumera el comando de ejemplo y la configuración para adjuntar el PV a un pod.

Pasos

1. Monte el volumen en una cápsula.

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

Estos ejemplos muestran configuraciones básicas para conectar el PVC a un pod: **Configuración básica**:

```
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
```

Puedes supervisar el progreso usando `kubectl get pod --watch`.

2. Verifique que el volumen esté montado en `/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
```

Ahora puedes eliminar el Pod. La aplicación Pod dejará de existir, pero el volumen se mantendrá.

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

Configure el complemento Trident EKS en un clúster EKS.

NetApp Trident simplifica la gestión del almacenamiento de Amazon FSx for NetApp ONTAP en Kubernetes para que sus desarrolladores y administradores puedan centrarse en el despliegue de aplicaciones. El complemento NetApp Trident EKS incluye los últimos parches de seguridad, correcciones de errores y está validado por AWS para funcionar con Amazon EKS. El complemento EKS le permite garantizar de forma consistente que sus clústeres de Amazon EKS sean seguros y estables, y reduce la cantidad de trabajo que necesita realizar para instalar, configurar y actualizar complementos.

Prerrequisitos

Asegúrese de tener lo siguiente antes de configurar el complemento Trident para AWS EKS:

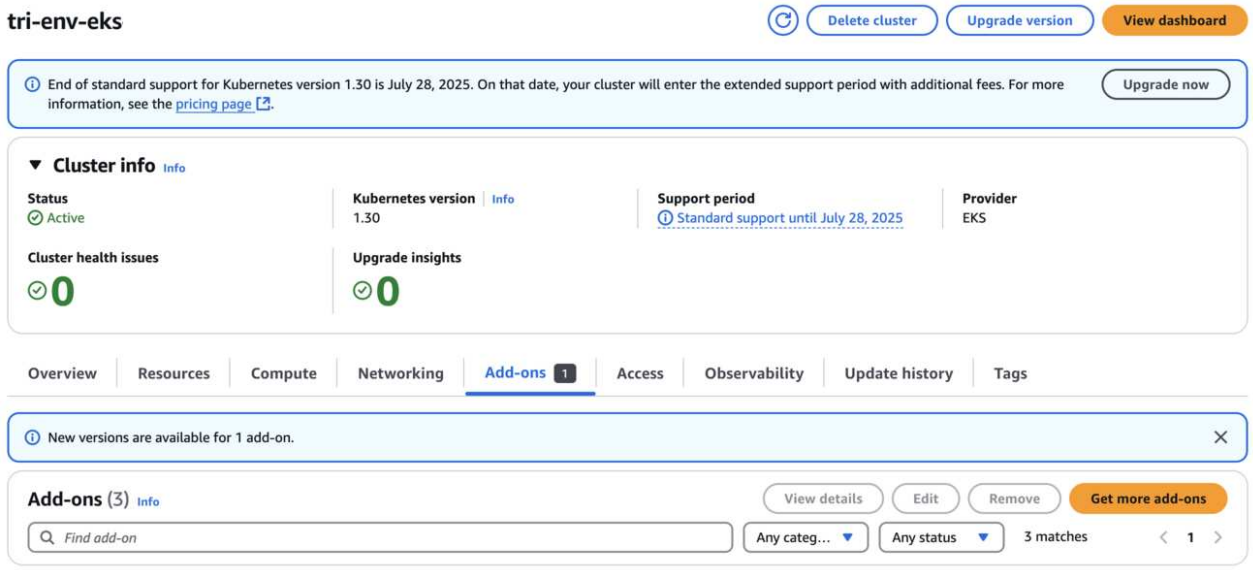
- Una cuenta de clúster de Amazon EKS con permisos para trabajar con complementos. Referirse a"Complementos de Amazon EKS" .
- Permisos de AWS para el mercado de AWS:
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- Tipo de AMI: Amazon Linux 2 (AL2_x86_64) o Amazon Linux 2 Arm (AL2_ARM_64)
- Tipo de nodo: AMD o ARM
- Un sistema de archivos Amazon FSx for NetApp ONTAP existente

Pasos

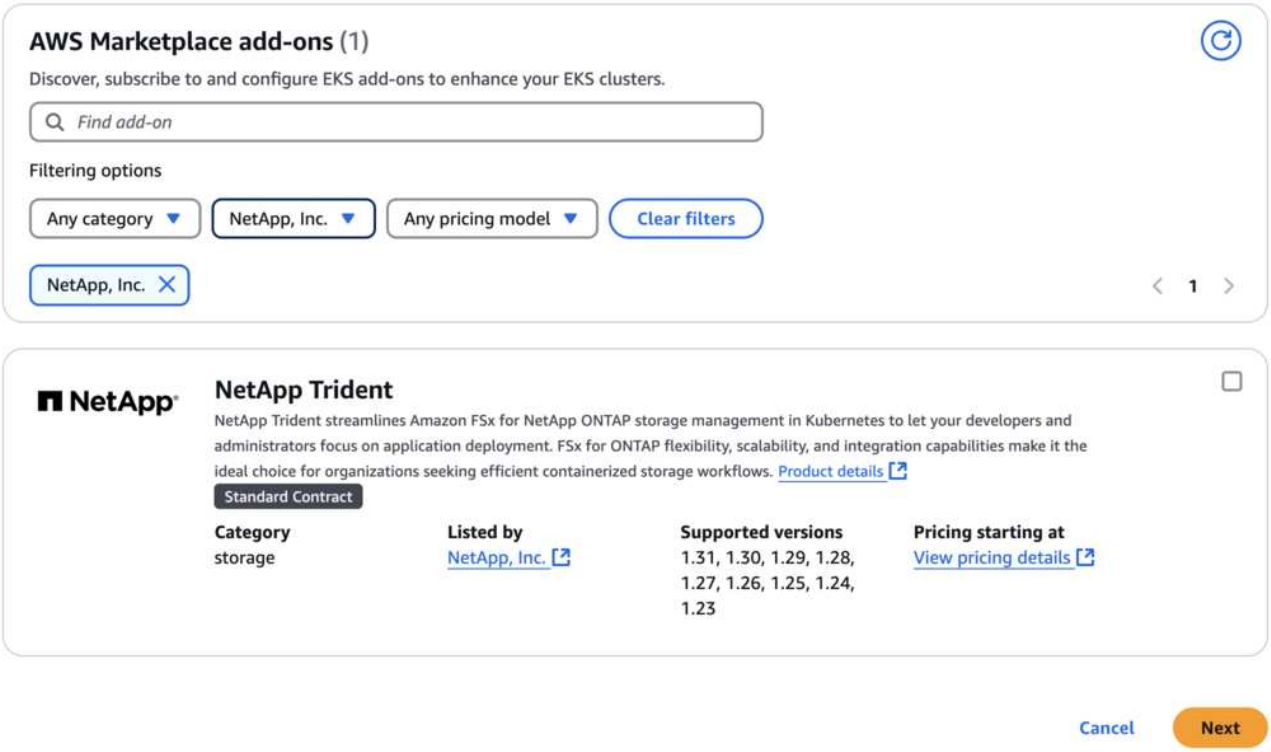
1. Asegúrese de crear un rol de IAM y un secreto de AWS para permitir que los pods de EKS accedan a los

recursos de AWS. Para obtener instrucciones, consulte"Crea un rol de IAM y un secreto de AWS" .

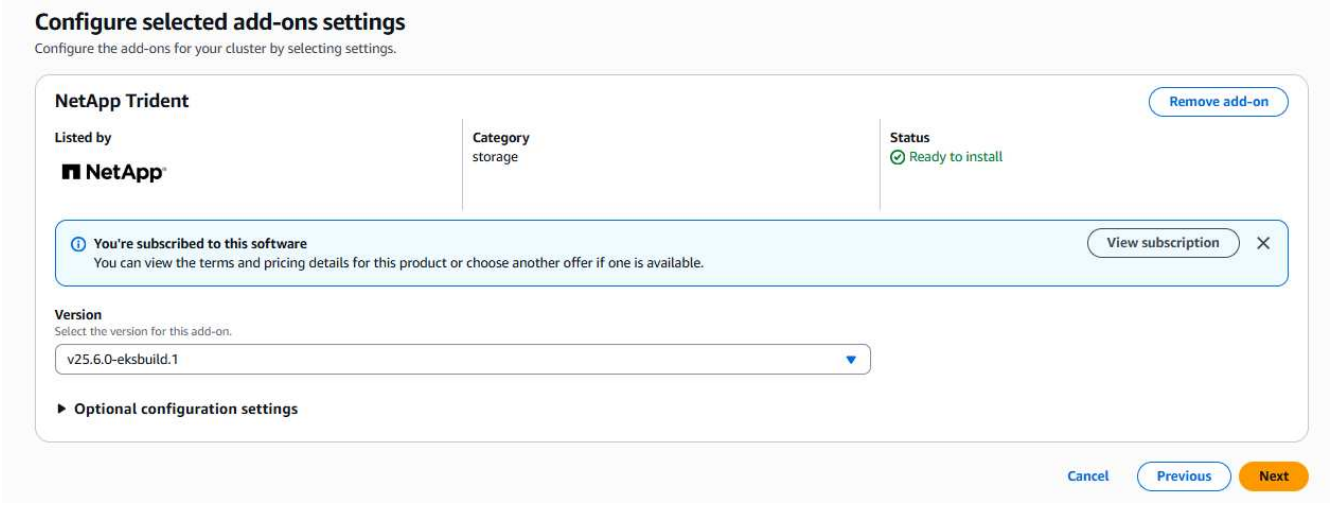
2. En su clúster de Kubernetes EKS, vaya a la pestaña **Complementos**.



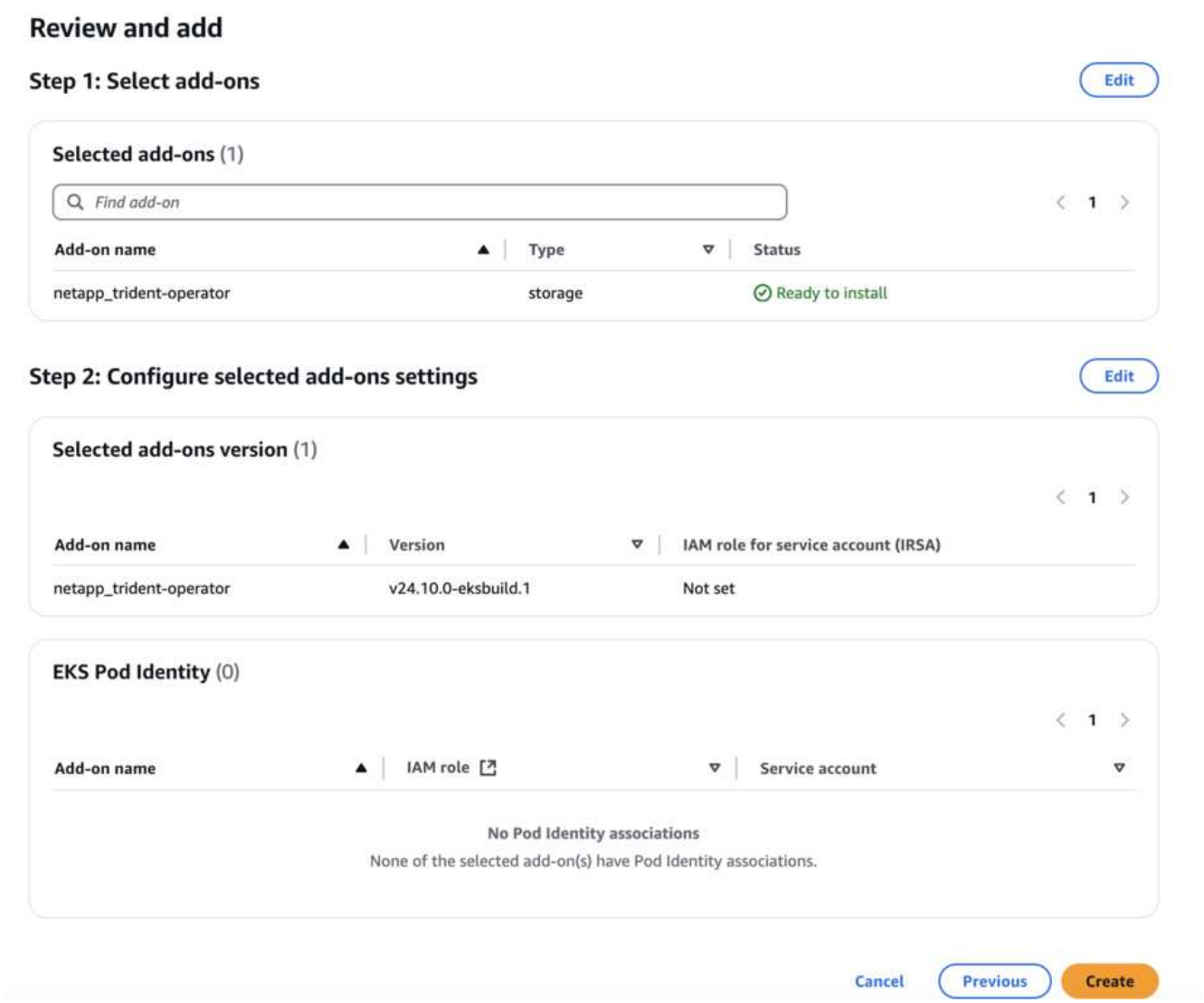
3. Vaya a **Complementos de AWS Marketplace** y elija la categoría *almacenamiento*.



4. Localice * NetApp Trident* y seleccione la casilla de verificación del complemento Trident , y haga clic en **Siguiente**.
5. Elige la versión deseada del complemento.

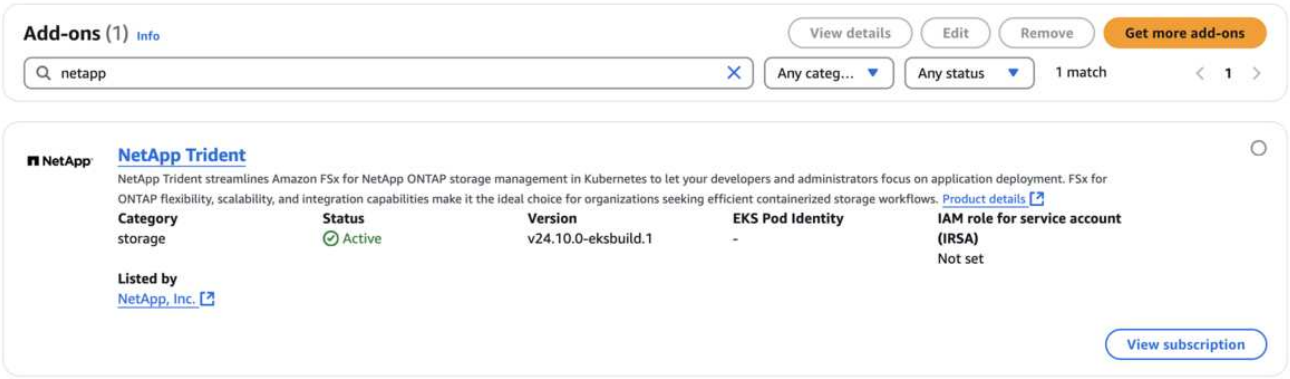


6. Configure los ajustes complementarios necesarios.



7. Si utiliza IRSA (roles de IAM para cuentas de servicio), consulte los pasos de configuración adicionales."aquí" .
8. Seleccione **Crear**.

9. Verifique que el estado del complemento sea *Activo*.



10. Ejecute el siguiente comando para verificar que Trident esté correctamente instalado en el clúster:

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

11. Continúe con la configuración y configure el backend de almacenamiento. Para obtener más información, consulte"Configurar el backend de almacenamiento" .

Instalar/desinstalar el complemento Trident EKS mediante la CLI

Instale el complemento NetApp Trident EKS mediante la CLI:

El siguiente comando de ejemplo instala el complemento Trident EKS: `eksctl create addon --cluster clusterName --name netapp_trident-operator --version v25.6.0-eksbuild.1` (con una versión dedicada)

Desinstale el complemento NetApp Trident EKS mediante la CLI:

El siguiente comando desinstala el complemento Trident EKS:

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

Crea backends con kubectl

Un backend define la relación entre Trident y un sistema de almacenamiento. Le indica a Trident cómo comunicarse con ese sistema de almacenamiento y cómo Trident aprovisionar volúmenes desde él. Una vez instalado Trident , el siguiente paso es crear un backend. El `TridentBackendConfig` La definición de recursos personalizados (CRD) le permite crear y administrar backends de Trident directamente a través de la interfaz de Kubernetes. Puedes hacerlo utilizando `kubectl` o la herramienta CLI equivalente para su distribución de Kubernetes.

`TridentBackendConfig`

`TridentBackendConfig` (`tbc`, `tbconfig` , `tbackendconfig`) es un CRD frontend con espacios de nombres que le permite administrar backends de Trident usando `kubectl` . Los administradores de Kubernetes y almacenamiento ahora pueden crear y administrar backends directamente a través de la CLI de

Kubernetes sin necesidad de una utilidad de línea de comandos dedicada.(`tridentctl`).

Tras la creación de un `TridentBackendConfig` objeto, sucede lo siguiente:

- Trident crea automáticamente un backend basándose en la configuración que proporciones. Esto se representa internamente como un `TridentBackend` (`tbe` , `tridentbackend`) CR.
- El `TridentBackendConfig` está singularmente ligado a un `TridentBackend` que fue creada por Trident.

Cada `TridentBackendConfig` mantiene una correspondencia uno a uno con un `TridentBackend` La primera es la interfaz que se proporciona al usuario para diseñar y configurar backends; la segunda es la forma en que Trident representa el objeto backend real.

> `TridentBackend`Trident crea automáticamente los CR. No debes modificarlos. Si deseas realizar actualizaciones en los backends, hazlo modificando el `TridentBackendConfig` objeto.

Consulte el siguiente ejemplo para ver el formato de `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
```

También puedes consultar los ejemplos en el "instalador de trident" Directorio con configuraciones de ejemplo para la plataforma/servicio de almacenamiento deseado.

El `spec` Toma parámetros de configuración específicos del backend. En este ejemplo, el backend utiliza el `ontap-san` El controlador de almacenamiento utiliza los parámetros de configuración que se tabulan aquí. Para obtener la lista de opciones de configuración para el controlador de almacenamiento que desee, consulte el"Información de configuración de backend para su controlador de almacenamiento" .

El `spec` Esta sección también incluye `credentials` y `deletionPolicy` campos, que se introducen recientemente en el `TridentBackendConfig` CR:

- `credentials`Este parámetro es un campo obligatorio y contiene las credenciales utilizadas para autenticarse con el sistema/servicio de almacenamiento. Esto se configura con un secreto de Kubernetes creado por el usuario. Las credenciales no se pueden pasar en texto plano y se producirá un error.
- `deletionPolicy`Este campo define qué debe suceder cuando el

`TridentBackendConfig se elimina. Puede tomar uno de dos valores posibles:

- delete`Esto da como resultado la eliminación de ambos. `TridentBackendConfig CR y el backend asociado. Este es el valor predeterminado.
- retain`Cuando un `TridentBackendConfig Si se elimina el CR, la definición del backend seguirá presente y se podrá gestionar con `tridentctl` . Configurar la política de eliminación a `retain` permite a los usuarios regresar a una versión anterior (anterior a la 21.04) y conservar los backends creados. El valor de este campo se puede actualizar después de un `TridentBackendConfig` se crea

El nombre de un backend se establece mediante `spec.backendName` . Si no se especifica, el nombre del backend se establece como el nombre del `TridentBackendConfig` objeto (metadata.nombre). Se recomienda establecer explícitamente los nombres de los backends utilizando `spec.backendName` .

Backends que fueron creados con `tridentctl` no tienen un asociado `TridentBackendConfig` objeto. Puedes optar por gestionar dichos backends con `kubectl` al crear un `TridentBackendConfig` CR. Se debe tener cuidado al especificar parámetros de configuración idénticos (como por ejemplo `spec.backendName` , `spec.storagePrefix` , `spec.storageDriverName` , etcétera). Trident enlazará automáticamente el recién creado `TridentBackendConfig` con el backend preexistente.

Resumen de pasos

Para crear un nuevo backend utilizando `kubectl` , debes hacer lo siguiente:

1. Crear una "Secreto de Kubernetes" El secreto contiene las credenciales que Trident necesita para comunicarse con el clúster/servicio de almacenamiento.
2. Crear una `TridentBackendConfig` objeto. Esto contiene detalles sobre el clúster/servicio de almacenamiento y hace referencia al secreto creado en el paso anterior.

Después de crear un backend, puede observar su estado mediante el uso de `kubectl get tbc <tbc-name> -n <trident-namespace>` y recabar detalles adicionales.

Paso 1: Crear un secreto de Kubernetes

Crea un secreto que contenga las credenciales de acceso para el backend. Esto es específico de cada servicio/plataforma de almacenamiento. He aquí un ejemplo:

```
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
```

Esta tabla resume los campos que deben incluirse en el secreto para cada plataforma de almacenamiento:

Descripción de los campos secretos de la plataforma de almacenamiento	Secreto	Descripción de los campos
Azure NetApp Files	ID de cliente	El ID de cliente de un registro de aplicación
Cloud Volumes Service para GCP	ID de clave privada	Identificador de la clave privada. Parte de la clave API para la cuenta de servicio de GCP con rol de administrador de CVS
Cloud Volumes Service para GCP	clave_privada	Clave privada. Parte de la clave API para la cuenta de servicio de GCP con rol de administrador de CVS
Elemento (NetApp HCI/ SolidFire)	Punto final	MVIP para el clúster SolidFire con credenciales de inquilino
ONTAP	nombre de usuario	Nombre de usuario para conectarse al clúster/SVM. Se utiliza para la autenticación basada en credenciales.
ONTAP	contraseña	Contraseña para conectarse al cluster/SVM. Se utiliza para la autenticación basada en credenciales.
ONTAP	clave privada del cliente	Valor codificado en Base64 de la clave privada del cliente. Se utiliza para la autenticación basada en certificados.

Descripción de los campos secretos de la plataforma de almacenamiento	Secreto	Descripción de los campos
ONTAP	chapNombre de usuario	Nombre de usuario entrante. Requerido si useCHAP=true. Para `ontap-san` y `ontap-san-economy`
ONTAP	Secreto del iniciador del capítulo	Secreto del iniciador de CHAP. Requerido si useCHAP=true. Para `ontap-san` y `ontap-san-economy`
ONTAP	chapTargetUsername	Nombre de usuario objetivo. Requerido si useCHAP=true. Para `ontap-san` y `ontap-san-economy`
ONTAP	chapTargetInitiatorSecret	Secreto del iniciador del objetivo CHAP. Requerido si useCHAP=true. Para `ontap-san` y `ontap-san-economy`

El secreto creado en este paso se utilizará como referencia en el `spec.credentials` campo de la `TridentBackendConfig` objeto que se crea en el siguiente paso.

Paso 2: Crear el `TridentBackendConfig` CR

Ahora estás listo para crear tu `TridentBackendConfig` CR. En este ejemplo, un backend que utiliza el `ontap-san` El controlador se crea utilizando el `TridentBackendConfig` objeto que se muestra a continuación:

```
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
```

Paso 3: Verificar el estado del `TridentBackendConfig` CR

Ahora que has creado el `TridentBackendConfig` CR, puede verificar el estado. Vea el siguiente ejemplo:

```
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
```

Se creó correctamente un backend y se vinculó al `TridentBackendConfig` CR.

La fase puede tomar uno de los siguientes valores:

- `Bound`: El `TridentBackendConfig` CR está asociado a un backend, y ese backend contiene `configRef` puesto a `TridentBackendConfig` UID de CR.
- `Unbound`Representado mediante `"" . El `TridentBackendConfig` El objeto no está vinculado a un backend. Todo recién creado `TridentBackendConfig` Los CR se encuentran en esta fase por defecto. Tras el cambio de fase, no puede volver a estar sin ataduras.
- `Deleting`: El `TridentBackendConfig` CR `deletionPolicy` estaba configurado para eliminar. Cuando el `TridentBackendConfig` Cuando se elimina el CR, pasa al estado de Eliminación.
  - Si no existen reclamaciones de volumen persistentes (PVC) en el backend, eliminar el `TridentBackendConfig` Esto provocará que Trident elimine tanto el backend como el `TridentBackendConfig` CR.
  - Si hay uno o más PVC presentes en el backend, pasa a un estado de eliminación. El `TridentBackendConfig` Posteriormente, CR también entra en la fase de eliminación. El backend y `TridentBackendConfig` se eliminan solo después de que se hayan eliminado todos los PVC.
- `Lost`El backend asociado con el `TridentBackendConfig` CR fue borrado accidental o deliberadamente y el `TridentBackendConfig` CR aún conserva una referencia al backend eliminado. El `TridentBackendConfig` CR aún puede eliminarse independientemente de `deletionPolicy` valor.

- `Unknown`Trident no puede determinar el estado o la existencia del backend asociado con el `TridentBackendConfig` CR. Por ejemplo, si el servidor API no responde o si `tridentbackends.trident.netapp.io` Falta el CRD. Esto podría requerir intervención.

¡En esta etapa, el backend se ha creado con éxito! Existen varias operaciones que también se pueden realizar, como por ejemplo:"actualizaciones y eliminaciones de backend" .

(Opcional) Paso 4: Obtenga más detalles

Puedes ejecutar el siguiente comando para obtener más información sobre tu 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
```

Además, también puede obtener un volcado YAML/JSON de `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 el `backendName` y el `backendUUID` del backend que se creó en respuesta a `TridentBackendConfig` CR. El `lastOperationStatus` El campo representa el estado de la última operación de `TridentBackendConfig` CR, que puede ser activado por el usuario (por ejemplo, el usuario cambió algo en `spec`) o activado por Trident (por ejemplo, durante los reinicios de Trident). Puede ser un éxito o un fracaso. `phase` representa el estado de la relación entre el `TridentBackendConfig` CR y el backend. En el ejemplo anterior, `phase` tiene el valor Bound, lo que significa que el `TridentBackendConfig` CR está asociado con el backend.

Puedes ejecutar el `kubectl -n trident describe tbc <tbc-cr-name>` comando para obtener detalles de los registros de eventos.

> No se puede actualizar ni eliminar un backend que contenga un asociado `TridentBackendConfig` objeto usando `tridentctl` . Para comprender los pasos que implica el cambio entre `tridentctl` y `TridentBackendConfig` ,"ver aquí" .

Gestionar backends

Realizar la gestión del backend con kubectl

Aprenda cómo realizar operaciones de administración de backend utilizando `kubectl` .

Eliminar un backend

Al eliminar un `TridentBackendConfig` , le indicas a Trident que elimine/conserve los backends (según `deletionPolicy`). Para eliminar un backend, asegúrese de que `deletionPolicy` está configurado para eliminar. Para eliminar solo el `TridentBackendConfig` , asegúrese de que `deletionPolicy` está previsto que se mantenga. Esto garantiza que el backend siga presente y pueda gestionarse mediante `tridentctl` .

Ejecute el siguiente comando:

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

Trident no elimina los secretos de Kubernetes que estaban en uso por `TridentBackendConfig` . El usuario de Kubernetes es responsable de la limpieza de secretos. Hay que tener cuidado al borrar secretos. Solo debes eliminar los secretos si los backends no los están utilizando.

Ver los backends existentes

Ejecute el siguiente comando:

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

También puedes correr `tridentctl get backend -n trident` o `tridentctl get backend -o yaml -n trident` para obtener una lista de todos los backends existentes. Esta lista también incluirá backends que fueron creados con `tridentctl` .

Actualizar un backend

Existen múltiples razones para actualizar un backend:

- Las credenciales del sistema de almacenamiento han cambiado. Para actualizar las credenciales, el secreto de Kubernetes que se utiliza en el `TridentBackendConfig` El objeto debe actualizarse. Trident actualizará automáticamente el sistema backend con las últimas credenciales proporcionadas. Ejecute el siguiente comando para actualizar el secreto de Kubernetes:

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

- Es necesario actualizar los parámetros (como el nombre de la SVM de ONTAP que se está utilizando).
  - Puedes actualizar `TridentBackendConfig` objetos directamente a través de Kubernetes usando el siguiente comando:

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

- Alternativamente, puede realizar cambios en el sistema existente. `TridentBackendConfig` CR usando el siguiente comando:

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

- Si falla una actualización del backend, este permanecerá en su última configuración conocida. Puedes ver los registros para determinar la causa ejecutando `kubectl get tbc <tbc-name> -o yaml -n trident` o `kubectl describe tbc <tbc-name> -n trident` .
- Después de identificar y corregir el problema con el archivo de configuración, puede volver a ejecutar el comando de actualización.

Realizar la gestión del backend con tridentctl

Aprenda cómo realizar operaciones de administración de backend utilizando `tridentctl` .

Crear un backend

Después de crear un"archivo de configuración del backend" , ejecute el siguiente comando:

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

Si falla la creación del backend, algo falló en la configuración del backend. Puedes consultar los registros para determinar la causa ejecutando el siguiente comando:

```
tridentctl logs -n trident
```

Una vez que haya identificado y corregido el problema con el archivo de configuración, simplemente puede ejecutar el programa. `create` Orden de nuevo.

Eliminar un backend

Para eliminar un backend de Trident, siga estos pasos:

1. Recuperar el nombre del backend:

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

2. Eliminar el backend:

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

Si Trident ha aprovisionado volúmenes e instantáneas desde este backend que aún existen, eliminar el backend impide que se aprovisionen nuevos volúmenes desde él. El backend seguirá existiendo en estado "Eliminando".

Ver los backends existentes

Para ver los backends que Trident conoce, haga lo siguiente:

- Para obtener un resumen, ejecute el siguiente comando:

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

- Para obtener todos los detalles, ejecute el siguiente comando:

```
tridentctl get backend -o json -n trident
```

Actualizar un backend

Después de crear un nuevo archivo de configuración de backend, ejecute el siguiente comando:

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

Si falla la actualización del backend, algo falló en la configuración del backend o intentaste una actualización no válida. Puedes consultar los registros para determinar la causa ejecutando el siguiente comando:

```
tridentctl logs -n trident
```

Una vez que haya identificado y corregido el problema con el archivo de configuración, simplemente puede ejecutar el programa. `update` Orden de nuevo.

Identifique las clases de almacenamiento que utilizan un backend.

Este es un ejemplo del tipo de preguntas que puedes responder con el JSON que `tridentctl` Salidas para objetos de backend. Esto utiliza el `jq` utilidad que necesitas instalar.

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

Esto también se aplica a los backends que se crearon utilizando `TridentBackendConfig` .

Cambiar entre las opciones de administración de backend

Aprende sobre las diferentes formas de gestionar los backends en Trident.

Opciones para la gestión de backends

Con la introducción de `TridentBackendConfig` Ahora, los administradores disponen de dos formas únicas de gestionar los sistemas backend. Esto plantea las siguientes preguntas:

- ¿Se pueden crear backends usando `tridentctl` ser gestionado con `TridentBackendConfig` ?
- ¿Se pueden crear backends usando `TridentBackendConfig` ser gestionado mediante `tridentctl` ?

Administrar `tridentctl` backends que utilizan `TridentBackendConfig`

Esta sección abarca los pasos necesarios para gestionar los backends que se crearon utilizando `tridentctl` directamente a través de la interfaz de Kubernetes creando `TridentBackendConfig` objetos.

Esto se aplicará a los siguientes escenarios:

- Sistemas backend preexistentes que no tienen un `TridentBackendConfig` porque fueron creados con `tridentctl` .
- Nuevos backends que se crearon con `tridentctl` , mientras que otros `TridentBackendConfig` Los objetos existen.

En ambos escenarios, los backends seguirán presentes, con Trident programando volúmenes y operando sobre ellos. Los administradores tienen dos opciones:

- Continuar usando `tridentctl` para gestionar los backends que se crearon utilizándolo.
- Enlace de backends creados usando `tridentctl` a un nuevo `TridentBackendConfig` objeto. Hacerlo significaría que los backends se gestionarían utilizando `kubectl` y no `tridentctl` .

Para gestionar un backend preexistente utilizando `kubectl` , necesitarás crear un `TridentBackendConfig` que se vincula al backend existente. Aquí tenéis una descripción general de cómo funciona:

1. Crea un secreto de Kubernetes. El secreto contiene las credenciales que Trident necesita para comunicarse con el clúster/servicio de almacenamiento.
2. Crear una `TridentBackendConfig` objeto. Esto contiene detalles sobre el clúster/servicio de almacenamiento y hace referencia al secreto creado en el paso anterior. Se debe tener cuidado al especificar parámetros de configuración idénticos (como por ejemplo `spec.backendName` , `spec.storagePrefix` , `spec.storageDriverName` , etcétera). `spec.backendName` Debe configurarse con el nombre del backend existente.

Paso 0: Identificar el backend

Para crear una `TridentBackendConfig` Si se vincula a un backend existente, deberá obtener la configuración del backend. En este ejemplo, supongamos que se creó un backend utilizando la siguiente definición 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"
      }
    }
  ]
}
```

Paso 1: Crear un secreto de Kubernetes

Crea un secreto que contenga las credenciales para el backend, como se muestra en este ejemplo:

```
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
```

Paso 2: Crear un `TridentBackendConfig` CR

El siguiente paso es crear un `TridentBackendConfig` CR que se vinculará automáticamente al preexistente `ontap-nas-backend` (como en este ejemplo). Asegúrese de que se cumplan los siguientes requisitos:

- El mismo nombre de backend se define en `spec.backendName` .
- Los parámetros de configuración son idénticos a los del backend original.
- Los pools virtuales (si existen) deben mantener el mismo orden que en el backend original.
- Las credenciales se proporcionan a través de un secreto de Kubernetes y no en texto plano.

En este caso, el `TridentBackendConfig` Se verá así:

```
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
```

Paso 3: Verificar el estado del `TridentBackendConfig` CR

Después de `TridentBackendConfig` Se ha creado, su fase debe ser `Bound` . También debe reflejar el mismo nombre de backend y UUID que el backend existente.

```
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 |
+---------------------+----------------
+--------------------------------------+--------+---------+
```

El backend ahora se gestionará completamente utilizando `tbc-ontap-nas-backend` `TridentBackendConfig` objeto.

Administrar `TridentBackendConfig` backends que utilizan `tridentctl`

```
`tridentctl`se puede utilizar para listar los backends que se crearon
utilizando `TridentBackendConfig` .  Además, los administradores también
pueden optar por gestionar completamente dichos sistemas backend mediante
`tridentctl` al eliminar `TridentBackendConfig` y asegurándose
`spec.deletionPolicy` está configurado para `retain` .
```

Paso 0: Identificar el backend

Por ejemplo, supongamos que el siguiente backend se creó utilizando `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 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

De los resultados se observa que `TridentBackendConfig` Se creó correctamente y está vinculado a un backend [observe el UUID del backend].

Paso 1: Confirmar `deletionPolicy` **está configurado para** `retain`

Analicemos el valor de `deletionPolicy` . Esto debe configurarse a `retain` . Esto garantiza que cuando un `TridentBackendConfig` Si se elimina el CR, la definición del backend seguirá presente y se podrá gestionar 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
```

No continúe con el siguiente paso a menos que `deletionPolicy` está configurado para `retain` .

Paso 2: Eliminar el `TridentBackendConfig` CR

El último paso es eliminar el `TridentBackendConfig` CR. Tras confirmar el `deletionPolicy` está configurado para `retain` , puedes proceder con la eliminación:

```
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 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

Tras la eliminación de `TridentBackendConfig` Trident simplemente elimina el objeto sin borrar realmente el backend.

Crear y gestionar clases de almacenamiento

Crear una clase de almacenamiento

Configure un objeto StorageClass de Kubernetes y cree la clase de almacenamiento para indicar a Trident cómo aprovisionar volúmenes.

Configurar un objeto StorageClass de Kubernetes

El "Objeto StorageClass de Kubernetes" identifica a Trident como el proveedor que se utiliza para esa clase e indica a Trident cómo aprovisionar un volumen. Por ejemplo:

```
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
```

Referirse a"Objetos de Kubernetes y Trident" Para obtener detalles sobre cómo interactúan las clases de almacenamiento con `PersistentVolumeClaim` y parámetros para controlar cómo Trident gestiona los volúmenes.

Crear una clase de almacenamiento

Después de crear el objeto StorageClass, puede crear la clase de almacenamiento. muestras de clase de almacenamiento proporciona algunos ejemplos básicos que puede usar o modificar.

Pasos

1. Este es un objeto de Kubernetes, así que use `kubectl` para crearlo en Kubernetes.

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

2. Ahora debería ver una clase de almacenamiento **basic-csi** tanto en Kubernetes como en Trident, y Trident debería haber detectado los pools en el 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"
        ]
      }
    }
  ]
}
```


muestras de clase de almacenamiento

Trident proporciona "Definiciones de clases de almacenamiento simples para backends específicos" .

Alternativamente, puedes editar `sample-input/storage-class-csi.yaml.templ` archivo que viene con el instalador y reemplazar *`BACKEND_TYPE`* con el nombre del controlador de almacenamiento.

```
./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
```

Gestionar clases de almacenamiento

Puede ver las clases de almacenamiento existentes, establecer una clase de almacenamiento predeterminada, identificar el backend de la clase de almacenamiento y eliminar clases de almacenamiento.

Vea las clases de almacenamiento existentes

- Para ver las clases de almacenamiento de Kubernetes existentes, ejecute el siguiente comando:

```
kubectl get storageclass
```

- Para ver los detalles de la clase de almacenamiento de Kubernetes, ejecute el siguiente comando:

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

- Para ver las clases de almacenamiento sincronizadas de Trident, ejecute el siguiente comando:

```
tridentctl get storageclass
```

- Para ver los detalles de la clase de almacenamiento sincronizado de Trident, ejecute el siguiente comando:

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

Establecer una clase de almacenamiento predeterminada

Kubernetes 1.6 añadió la capacidad de establecer una clase de almacenamiento predeterminada. Esta es la clase de almacenamiento que se utilizará para aprovisionar un volumen persistente si un usuario no especifica uno en una reclamación de volumen persistente (PVC).

- Define una clase de almacenamiento predeterminada configurando la anotación. `storageclass.kubernetes.io/is-default-class` a verdadero en la definición de la clase de almacenamiento. Según la especificación, cualquier otro valor o la ausencia de la anotación se interpreta como falso.
- Puede configurar una clase de almacenamiento existente como clase de almacenamiento predeterminada mediante el siguiente comando:

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

- De igual forma, puede eliminar la anotación de la clase de almacenamiento predeterminada mediante el siguiente comando:

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

También hay ejemplos en el paquete de instalación de Trident que incluyen esta anotación.

> Solo debe haber una clase de almacenamiento predeterminada en su clúster a la vez. Técnicamente, Kubernetes no impide tener más de una, pero se comportará como si no existiera ninguna clase de almacenamiento predeterminada.

Identificar el backend para una clase de almacenamiento

Este es un ejemplo del tipo de preguntas que puedes responder con el JSON que `tridentctl` Salidas para objetos backend de Trident . Esto utiliza el `jq` utilidad, que tal vez necesites instalar primero.

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

Eliminar una clase de almacenamiento

Para eliminar una clase de almacenamiento de Kubernetes, ejecute el siguiente comando:

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

`<storage-class>`debe reemplazarse con su clase de almacenamiento.

Los volúmenes persistentes que se hayan creado mediante esta clase de almacenamiento permanecerán intactos y Trident seguirá gestionándolos.

> Trident impone un espacio en blanco `fsType` por los volúmenes que crea. Para los backends iSCSI, se recomienda aplicar `parameters.fsType` en la clase de almacenamiento. Debes eliminar las StorageClasses existentes y volver a crearlas con `parameters.fsType` especificado.

Provisión y gestión de volúmenes

Provisión de un volumen

Cree un PersistentVolumeClaim (PVC) que utilice la StorageClass de Kubernetes configurada para solicitar acceso al PV. Luego puedes montar el panel fotovoltaico en un soporte.

Descripción general

A "*Reclamación de volumen persistente*" (PVC) es una solicitud de acceso al PersistentVolume en el clúster.

El PVC se puede configurar para solicitar el almacenamiento de un tamaño o modo de acceso determinado. Mediante la StorageClass asociada, el administrador del clúster puede controlar más que el tamaño del PersistentVolume y el modo de acceso, como el rendimiento o el nivel de servicio.

Una vez creado el PVC, puede montar el volumen en un soporte.

Crea el PVC

Pasos

1. Crear el PVC.

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

2. Verifique el estado del PVC.

```
kubectl get pvc
```

```
NAME          STATUS   VOLUME     CAPACITY   ACCESS MODES   STORAGECLASS AGE
pvc-storage   Bound    pv-name    1Gi        RWO                    5m
```

1. Monte el volumen en una cápsula.

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

Puedes supervisar el progreso usando `kubectl get pod --watch`.

2. Verifique que el volumen esté montado en `/my/mount/path`.

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

3. Ahora puedes eliminar el Pod. La aplicación Pod dejará de existir, pero el volumen se mantendrá.

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

Ejemplos de manifiestos

Muestra de PersistentVolumeClaim

Estos ejemplos muestran opciones básicas de configuración de PVC.

PVC con acceso RWO

Este ejemplo muestra un PVC básico con acceso RWO que está asociado a una StorageClass llamada `basic-csi` .

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

PVC con NVMe/TCP

Este ejemplo muestra un PVC básico para NVMe/TCP con acceso RWO que está asociado con una StorageClass llamada `protection-gold` .

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: pvc-san-nvme
spec:
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: protection-gold
```

Ejemplos de manifiesto de Pod

Estos ejemplos muestran configuraciones básicas para unir el PVC a un módulo.

Configuración básica

```
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
```

Configuración básica NVMe/TCP

```
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
```

Referirse a"Objetos de Kubernetes y Trident" Para obtener detalles sobre cómo interactúan las clases de almacenamiento con `PersistentVolumeClaim` y parámetros para controlar cómo Trident gestiona los volúmenes.

Ampliar volúmenes

Trident brinda a los usuarios de Kubernetes la capacidad de expandir sus volúmenes después de su creación. Encuentre información sobre las configuraciones necesarias para expandir volúmenes iSCSI, NFS, SMB, NVMe/TCP y FC.

Expandir un volumen iSCSI

Puede expandir un volumen persistente iSCSI (PV) utilizando el aprovisionador CSI.

> La expansión de volumen iSCSI es compatible con `ontap-san` , `ontap-san-economy` , `solidfire-san` controladores y requiere Kubernetes 1.16 y posterior.

Paso 1: Configure la StorageClass para que admita la expansión de volumen.

Edite la definición de StorageClass para configurar `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
```

Para una StorageClass ya existente, edítela para incluir la `allowVolumeExpansion` parámetro.

Paso 2: Cree un PVC con la StorageClass que creó.

Edite la definición de PVC y actualice la `spec.resources.requests.storage` para reflejar el nuevo tamaño deseado, que debe ser mayor que el tamaño original.

```
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 Volumen Persistente (PV) y lo asocia con esta Reclamación de Volumen Persistente (PVC).

```
kubectl get pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi
RWO            ontap-san      8s

kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM              STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi        RWO
Delete           Bound    default/san-pvc    ontap-san               10s
```

Paso 3: Defina un módulo que conecte el PVC.

Conecte el PV a una cápsula para poder cambiar su tamaño. Existen dos escenarios al redimensionar un PV iSCSI:

- Si el PV está conectado a un pod, Trident expande el volumen en el backend de almacenamiento, vuelve a escanear el dispositivo y redimensiona el sistema de archivos.
- Al intentar cambiar el tamaño de un PV no conectado, Trident expande el volumen en el backend de almacenamiento. Una vez que el PVC se vincula a un pod, Trident vuelve a escanear el dispositivo y redimensiona el sistema de archivos. Kubernetes actualiza entonces el tamaño del PVC una vez que la operación de expansión se ha completado correctamente.

En este ejemplo, se crea un pod que utiliza el `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
```

Paso 4: Ampliar el PV

Para cambiar el tamaño del PV creado de 1Gi a 2Gi, edite la definición del PVC y actualícela. `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
 # ...
```

Paso 5: Validar la expansión

Puedes comprobar que la expansión ha funcionado correctamente verificando el tamaño del PVC, el PV y el volumen del 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    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

Ampliar un volumen FC

Puede expandir un volumen persistente FC (PV) utilizando el aprovisionador CSI.

> La expansión del volumen FC está soportada por la `ontap-san` controlador y requiere Kubernetes 1.16 y posterior.

Paso 1: Configure la StorageClass para que admita la expansión de volumen.

Edite la definición de StorageClass para configurar `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
```

Para una StorageClass ya existente, edítela para incluir la `allowVolumeExpansion` parámetro.

Paso 2: Cree un PVC con la StorageClass que creó.

Edite la definición de PVC y actualice la `spec.resources.requests.storage` para reflejar el nuevo tamaño deseado, que debe ser mayor que el tamaño original.

```
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 Volumen Persistente (PV) y lo asocia con esta Reclamación de Volumen Persistente (PVC).

```
kubectl get pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi
RWO            ontap-san      8s

kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM              STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi        RWO
Delete           Bound    default/san-pvc    ontap-san               10s
```

Paso 3: Defina un módulo que conecte el PVC.

Conecte el PV a una cápsula para poder cambiar su tamaño. Existen dos escenarios al redimensionar un PV de FC:

- Si el PV está conectado a un pod, Trident expande el volumen en el backend de almacenamiento, vuelve a escanear el dispositivo y redimensiona el sistema de archivos.
- Al intentar cambiar el tamaño de un PV no conectado, Trident expande el volumen en el backend de almacenamiento. Una vez que el PVC se vincula a un pod, Trident vuelve a escanear el dispositivo y redimensiona el sistema de archivos. Kubernetes actualiza entonces el tamaño del PVC una vez que la

operación de expansión se ha completado correctamente.

En este ejemplo, se crea un pod que utiliza el `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
```

Paso 4: Ampliar el PV

Para cambiar el tamaño del PV creado de 1Gi a 2Gi, edite la definición del PVC y actualícela. `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
 # ...
```

Paso 5: Validar la expansión

Puedes comprobar que la expansión ha funcionado correctamente verificando el tamaño del PVC, el PV y el volumen del 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    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

Expandir un volumen NFS

Trident admite la expansión de volumen para PV NFS aprovisionados en `ontap-nas` , `ontap-nas-economy` , `ontap-nas-flexgroup` , `gcp-cvs` , y `azure-netapp-files` backends.

Paso 1: Configure la StorageClass para que admita la expansión de volumen.

Para cambiar el tamaño de un volumen físico NFS, el administrador primero debe configurar la clase de almacenamiento para permitir la expansión del volumen configurando la `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
```

Si ya ha creado una clase de almacenamiento sin esta opción, simplemente puede editar la clase de almacenamiento existente mediante el uso de `kubectl edit storageclass` para permitir la expansión de volumen.

Paso 2: Cree un PVC con la StorageClass que creó.

```
cat pvc-ontapnas.yaml
```

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: ontapnas20mb
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 20Mi
  storageClassName: ontapnas
```

Trident debería crear un PV NFS de 20 MiB para este 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
```

Paso 3: Ampliar el PV

Para cambiar el tamaño del PV recién creado de 20 MiB a 1 GiB, edite el PVC y configure `spec.resources.requests.storage` hasta 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
# ...
```

Paso 4: Validar la expansión

Puede comprobar que el cambio de tamaño se ha realizado correctamente verificando el tamaño del PVC, el PV y el volumen de 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    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

volúmenes de importación

Puedes importar volúmenes de almacenamiento existentes como un PV de Kubernetes usando `tridentctl import`.

Descripción general y consideraciones

Podrías importar un volumen a Trident para:

- Contenieriza una aplicación y reutiliza su conjunto de datos existente.
- Utilice un clon de un conjunto de datos para una aplicación efímera.
- Reconstruir un clúster de Kubernetes que ha fallado
- Migrar los datos de la aplicación durante la recuperación ante desastres

Consideraciones

Antes de importar un volumen, revise las siguientes consideraciones.

- Trident solo puede importar volúmenes ONTAP de tipo RW (lectura-escritura). Los volúmenes de tipo DP (protección de datos) son volúmenes de destino SnapMirror . Debes romper la relación de espejo antes de importar el volumen a Trident.

- Recomendamos importar volúmenes sin conexiones activas. Para importar un volumen que se esté utilizando activamente, clone el volumen y luego realice la importación.

  Esto es especialmente importante para los volúmenes en bloque, ya que Kubernetes desconocería la conexión anterior y podría fácilmente adjuntar un volumen activo a un pod. Esto puede provocar corrupción de datos.

- Aunque `StorageClass` Debe especificarse en un PVC; Trident no utiliza este parámetro durante la importación. Las clases de almacenamiento se utilizan durante la creación de volúmenes para seleccionar entre los grupos disponibles en función de las características de almacenamiento. Dado que el volumen ya existe, no es necesario seleccionar ningún pool durante la importación. Por lo tanto, la importación no fallará incluso si el volumen existe en un backend o pool que no coincide con la clase de almacenamiento especificada en el PVC.
- El volumen existente se determina y se establece en el PVC. Una vez que el controlador de almacenamiento importa el volumen, se crea el PV con una ClaimRef al PVC.
  - La política de reclamaciones está inicialmente establecida para `retain` en el PV. Una vez que Kubernetes enlaza correctamente el PVC y el PV, la política de recuperación se actualiza para que coincida con la política de recuperación de la clase de almacenamiento.
  - Si la política de recuperación de la Clase de Almacenamiento es `delete` El volumen de almacenamiento se eliminará cuando se elimine el PV.
- Por defecto, Trident gestiona el PVC y cambia el nombre del FlexVol volume y del LUN en el backend. Puedes pasar el `--no-manage` bandera para importar un volumen no administrado. Si utilizas `--no-manage` Trident no realiza ninguna operación adicional en el PVC o PV durante el ciclo de vida de los objetos. El volumen de almacenamiento no se elimina cuando se elimina el PV y otras operaciones como la clonación y el cambio de tamaño del volumen también se ignoran.

  Esta opción resulta útil si desea utilizar Kubernetes para cargas de trabajo en contenedores, pero por lo demás desea gestionar el ciclo de vida del volumen de almacenamiento fuera de Kubernetes.

- Se añade una anotación al PVC y al PV que cumple una doble función: indicar si el volumen fue importado y si el PVC y el PV están gestionados. Esta anotación no debe modificarse ni eliminarse.

Importar un volumen

Puedes utilizar `tridentctl import` para importar un volumen.

Pasos

1. Cree el archivo de reclamación de volumen persistente (PVC) (por ejemplo, `pvc.yaml`) que se utilizará para crear el PVC. El archivo de PVC debe incluir `name` , `namespace` , `accessModes` , y `storageClassName` . Opcionalmente, puede especificar `unixPermissions` en su definición de PVC.

   El siguiente es un ejemplo de una especificación mínima:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: my_claim
  namespace: my_namespace
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: my_storage_class
```

No incluya parámetros adicionales como el nombre del PV o el tamaño del volumen. Esto puede provocar que falle el comando de importación.

2. Utilice el `tridentctl import` comando para especificar el nombre del backend de Trident que contiene el volumen y el nombre que identifica de forma única el volumen en el almacenamiento (por ejemplo: ONTAP FlexVol, Element Volume, ruta del Cloud Volumes Service). El `-f` Se requiere un argumento para especificar la ruta al archivo PVC.

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

Ejemplos

Revise los siguientes ejemplos de importación de volumen para los controladores compatibles.

ONTAP NAS y ONTAP NAS FlexGroup

Trident admite la importación de volúmenes mediante el `ontap-nas` y `ontap-nas-flexgroup` conductores.

- Trident no admite la importación de volumen mediante el `ontap-nas-economy` conductor.
- El `ontap-nas` y `ontap-nas-flexgroup` Los controladores no permiten nombres de volumen duplicados.

Cada volumen creado con el `ontap-nas` El controlador es un FlexVol volume en el clúster ONTAP . Importación de volúmenes FlexVol con el `ontap-nas` El controlador funciona igual. Un volumen FlexVol que ya existe en un clúster ONTAP se puede importar como un `ontap-nas` CLORURO DE POLIVINILO. De forma similar, los volúmenes de FlexGroup se pueden importar como `ontap-nas-flexgroup` PVC.

Ejemplos de ONTAP NAS

A continuación se muestra un ejemplo de importación de un volumen administrado y un volumen no administrado.

Volumen gestionado

El siguiente ejemplo importa un volumen llamado `managed_volume` en un backend llamado `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    |
+------------------------------------------+---------+---------------
+----------+------------------------------------------+--------+---------+
```

Volumen no gestionado

Al usar el `--no-manage` argumento, Trident no cambia el nombre del volumen.

El siguiente ejemplo importa `unmanaged_volume` en el `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 admite la importación de volumen mediante el `ontap-san` (iSCSI, NVMe/TCP y FC) y `ontap-san-economy` Conductores.

Trident puede importar volúmenes ONTAP SAN FlexVol que contengan un solo LUN. Esto es coherente con la `ontap-san` controlador, que crea un FlexVol volume para cada PVC y un LUN dentro del FlexVol volume. Trident importa el FlexVol volume y lo asocia con la definición de PVC. Trident puede importar `ontap-san-`

`economy` volúmenes que contienen múltiples LUN.

Ejemplos de ONTAP SAN

A continuación se muestra un ejemplo de importación de un volumen administrado y un volumen no administrado.

Volumen gestionado

Para los volúmenes administrados, Trident cambia el nombre del FlexVol volume a `pvc-<uuid>` formato y el LUN dentro del FlexVol volume a `lun0` .

El siguiente ejemplo importa el `ontap-san-managed` FlexVol volume que está presente en el `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    |
+----------------------------------------+--------+---------------
+----------+----------------------------------------+--------+---------+
```

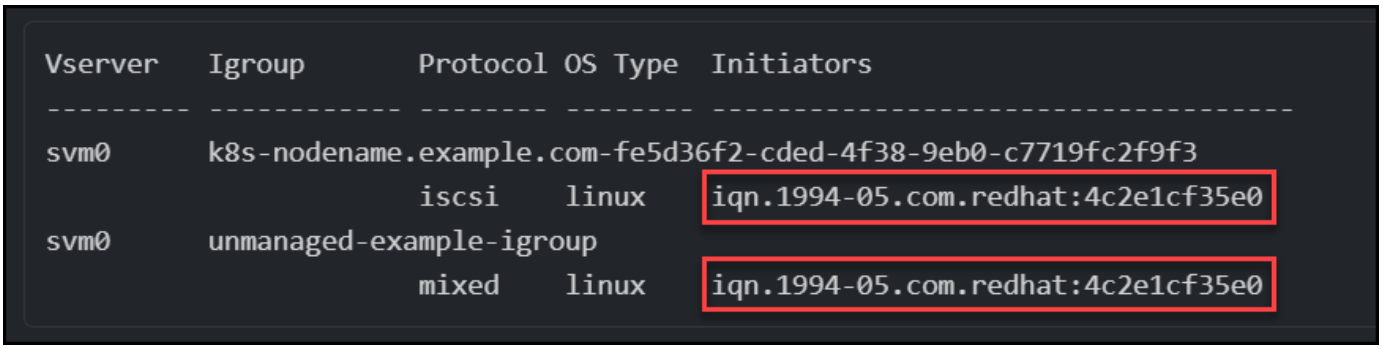
Volumen no gestionado

El siguiente ejemplo importa `unmanaged_example_volume` en el `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   |
+----------------------------------------+---------+---------------
+----------+----------------------------------------+--------+---------+
```

Si tiene LUNS asignados a igroups que comparten un IQN con un IQN de nodo de Kubernetes, como se

muestra en el siguiente ejemplo, recibirá el error: `LUN already mapped to initiator(s) in this group` . Deberá eliminar el iniciador o desasignar el LUN para importar el volumen.



Elemento

Trident admite el software NetApp Element y la importación de volúmenes NetApp HCI mediante `solidfire-san` conductor.

> El controlador Element admite nombres de volumen duplicados. Sin embargo, Trident devuelve un error si hay nombres de volumen duplicados. Como solución alternativa, clone el volumen, asígnele un nombre único e importe el volumen clonado.

Ejemplo de elemento

El siguiente ejemplo importa un `element-managed` volumen en el 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    |
+----------------------------------------+--------+---------------
+----------+----------------------------------------+--------+---------+
```

Plataforma de Google Cloud

Trident admite la importación de volúmenes mediante el `gcp-cvs` conductor.

Para importar un volumen respaldado por el Cloud Volumes Service NetApp Cloud Volumes en Google Cloud Platform, identifique el volumen por su ruta de acceso. La ruta del volumen es la porción de la ruta de exportación del volumen después de `:/` . Por ejemplo, si la ruta de exportación es `10.0.0.1:/adroit-jolly-swift` , la ruta del volumen es `adroit-jolly-swift` .

Ejemplo de Google Cloud Platform

El siguiente ejemplo importa un `gcp-cvs` volumen en el backend `gcpcvs_YEppr` con la ruta de volumen de `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 admite la importación de volúmenes mediante el `azure-netapp-files` conductor.

Para importar un volumen de Azure NetApp Files , identifique el volumen por su ruta de acceso. La ruta del volumen es la porción de la ruta de exportación del volumen después de `:/` . Por ejemplo, si la ruta de montaje es `10.0.0.2:/importvol1` , la ruta del volumen es `importvol1` .

Ejemplo de Azure NetApp Files

El siguiente ejemplo importa un `azure-netapp-files` volumen en el backend `azurenetappfiles_40517` con la ruta del volumen `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 admite la importación de volúmenes mediante el `google-cloud-netapp-volumes` conductor.

Ejemplo de Google Cloud NetApp Volumes

El siguiente ejemplo importa un `google-cloud-netapp-volumes` volumen en el backend `backend-tbc-gcnv1` con el volumen `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
|
+----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

El siguiente ejemplo importa un `google-cloud-netapp-volumes` volumen cuando hay dos volúmenes presentes en la misma región:

```
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
|
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

Personaliza los nombres y etiquetas de los volúmenes.

Con Trident, puedes asignar nombres y etiquetas descriptivos a los volúmenes que crees. Esto te ayuda a identificar y asignar fácilmente los volúmenes a sus respectivos recursos de Kubernetes (PVC). También puede definir plantillas a nivel de backend para crear nombres de volumen personalizados y etiquetas personalizadas; cualquier volumen que cree, importe o clone se ajustará a las plantillas.

Antes de empezar

Compatibilidad con nombres y etiquetas de volumen personalizables:

1. Operaciones de creación, importación y clonación de volúmenes.
2. En el caso del controlador ontap-nas-economy, solo el nombre del volumen Qtree cumple con la plantilla de nombre.
3. En el caso del controlador ontap-san-economy, solo el nombre del LUN cumple con la plantilla de nombre.

Limitaciones

1. Los nombres de volumen personalizables solo son compatibles con los controladores ONTAP locales.
2. Los nombres de volumen personalizables no se aplican a los volúmenes existentes.

Comportamientos clave de los nombres de volumen personalizables

1. Si se produce un fallo debido a una sintaxis no válida en una plantilla de nombre, falla la creación del backend. Sin embargo, si falla la aplicación de la plantilla, el volumen se nombrará de acuerdo con la convención de nomenclatura existente.

2. El prefijo de almacenamiento no es aplicable cuando un volumen se nombra utilizando una plantilla de nombre de la configuración del backend. Se puede añadir directamente a la plantilla cualquier valor de prefijo deseado.

Ejemplos de configuración de backend con plantilla de nombre y etiquetas

Se pueden definir plantillas de nombres personalizadas a nivel de raíz y/o de grupo.

Ejemplo de nivel raíz

```
{
  "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}}"
  }
}
```

Ejemplo de nivel de 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 }}"
      }
    }
  ]
}
```

Ejemplos de plantillas de nombres

Ejemplo 1:

```
"nameTemplate": "{{ .config.StoragePrefix }}_{{ .volume.Name }}_{{
.config.BackendName }}"
```

Ejemplo 2:

```
"nameTemplate": "pool_{{ .config.StoragePrefix }}_{{ .volume.Name }}_{{
slice .volume.RequestName 1 5 }}""
```

Puntos a considerar

1. En el caso de importaciones de volúmenes, las etiquetas se actualizan solo si el volumen existente tiene etiquetas en un formato específico. Por ejemplo: `{"provisioning":{"Cluster":"ClusterA", "PVC": "pvcname"}}`.
2. En el caso de importaciones de volúmenes gestionados, el nombre del volumen sigue la plantilla de nombre definida en el nivel raíz de la definición del backend.
3. Trident no admite el uso de un operador de segmentación con el prefijo de almacenamiento.
4. Si las plantillas no generan nombres de volumen únicos, Trident añadirá algunos caracteres aleatorios para crear nombres de volumen únicos.
5. Si el nombre personalizado de un volumen NAS economy supera los 64 caracteres, Trident nombrará los volúmenes según la convención de nomenclatura existente. Para todos los demás controladores ONTAP , si el nombre del volumen excede el límite de nombres, el proceso de creación del volumen falla.

Compartir un volumen NFS entre espacios de nombres

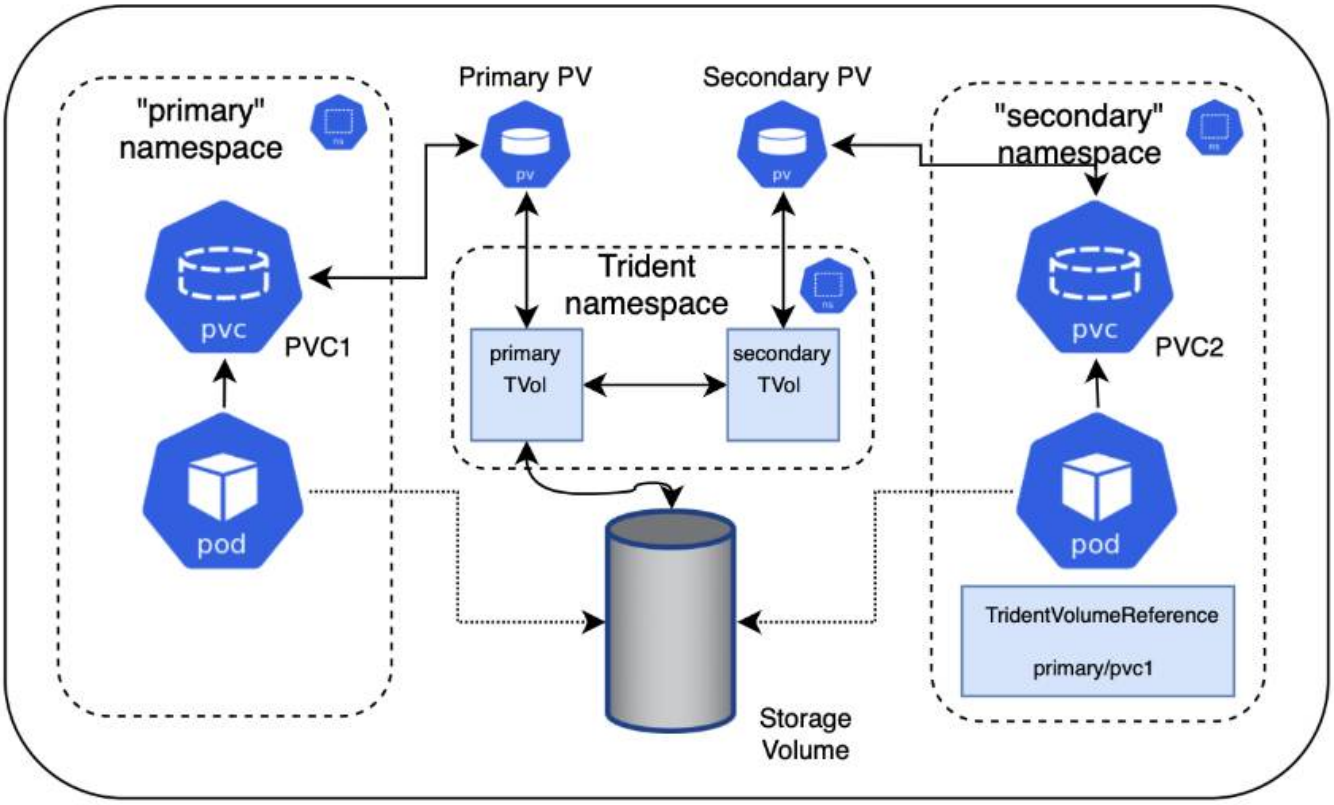
Con Trident, puedes crear un volumen en un espacio de nombres primario y compartirlo en uno o más espacios de nombres secundarios.

Funciones

El CR TridentVolumeReference le permite compartir de forma segura volúmenes NFS ReadWriteMany (RWX) a través de uno o más espacios de nombres de Kubernetes. Esta solución nativa de Kubernetes tiene las siguientes ventajas:

- Múltiples niveles de control de acceso para garantizar la seguridad
- Funciona con todos los controladores de volumen Trident NFS.
- No depende de tridentctl ni de ninguna otra función no nativa de Kubernetes.

Este diagrama ilustra el uso compartido de volúmenes NFS entre dos espacios de nombres de Kubernetes.



Inicio rápido

Puedes configurar el uso compartido de volúmenes NFS en tan solo unos pocos pasos.

1 **Configure el PVC de origen para compartir el volumen.**

El propietario del espacio de nombres de origen otorga permiso para acceder a los datos en el PVC de origen.

2 **Conceder permiso para crear un CR en el espacio de nombres de destino**

El administrador del clúster otorga permiso al propietario del espacio de nombres de destino para crear el CR TridentVolumeReference.

3 **Cree una referencia TridentVolumeReference en el espacio de nombres de destino.**

El propietario del espacio de nombres de destino crea el CR TridentVolumeReference para hacer referencia al PVC de origen.

4 **Cree el PVC subordinado en el espacio de nombres de destino.**

El propietario del espacio de nombres de destino crea el PVC subordinado para utilizar la fuente de datos del PVC de origen.

Configurar los espacios de nombres de origen y destino

Para garantizar la seguridad, el uso compartido entre espacios de nombres requiere la colaboración y la acción del propietario del espacio de nombres de origen, el administrador del clúster y el propietario del espacio de nombres de destino. El rol del usuario se designa en cada paso.

Pasos

1. **Propietario del espacio de nombres de origen:** Crear el PVC(`pvc1`) en el espacio de nombres de origen que otorga permiso para compartir con el espacio de nombres de destino(`namespace2`) usando el `shareToNamespace` anotación.

```
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 el PV y su volumen de almacenamiento NFS de backend.

   - Puedes compartir el PVC con múltiples espacios de nombres utilizando una lista delimitada por comas. Por ejemplo, `trident.netapp.io/shareToNamespace: namespace2,namespace3,namespace4` .
   - Puedes compartir con todos los espacios de nombres usando `*` . Por ejemplo, `trident.netapp.io/shareToNamespace: *`
   - Puedes actualizar el PVC para incluir el `shareToNamespace` anotaciones en cualquier momento.

2. **Administrador del clúster:** Asegúrese de que exista un RBAC adecuado para otorgar permiso al propietario del espacio de nombres de destino para crear la CR TridentVolumeReference en el espacio de nombres de destino.
3. **Propietario del espacio de nombres de destino:** Cree un CR TridentVolumeReference en el espacio de nombres de destino que haga referencia al espacio de nombres de origen. `pvc1` .

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. **Propietario del espacio de nombres de destino:** Crear un PVC(`pvc2`) en el espacio de nombres de destino(`namespace2`) usando el `shareFromPVC` Anotación para designar el PVC de origen.

```
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
```

El tamaño del tubo de PVC de destino debe ser menor o igual que el del tubo de PVC de origen.

Resultados

Trident lee el `shareFromPVC` Se realiza una anotación en el PVC de destino y se crea el PV de destino como un volumen subordinado sin recursos de almacenamiento propios que apunta al PV de origen y comparte los recursos de almacenamiento del PV de origen. Los valores de destino PVC y PV aparecen vinculados con normalidad.

Eliminar un volumen compartido

Puedes eliminar un volumen que se comparte entre varios espacios de nombres. Trident eliminará el acceso al volumen en el espacio de nombres de origen y mantendrá el acceso para otros espacios de nombres que compartan el volumen. Cuando se eliminan todos los espacios de nombres que hacen referencia al volumen, Trident borra el volumen.

Usar `tridentctl get` para consultar volúmenes subordinados

Utilizando el[`tridentctl` utilidad, puedes ejecutar la `get` comando para obtener volúmenes subordinados. Para obtener más información, consulte el siguiente enlace: ../trident-reference/tridentctl.html[`tridentctl` comandos y opciones].

```
Usage:
  tridentctl get [option]
```

Banderas:

- ``-h, --help`Ayuda para volúmenes.
- `--parentOfSubordinate string`Limitar la consulta al volumen de origen subordinado.
- `--subordinateOf string`Limitar la consulta a los subordinados del volumen.

Limitaciones

- Trident no puede impedir que los espacios de nombres de destino escriban en el volumen compartido. Debe utilizar el bloqueo de archivos u otros procesos para evitar la sobrescritura de datos de volúmenes compartidos.
- No se puede revocar el acceso al PVC de origen eliminando el `shareToNamespace` o `shareFromNamespace` anotaciones o eliminar las `TridentVolumeReference` CR. Para revocar el acceso, debe eliminar el PVC subordinado.
- Las instantáneas, los clones y la duplicación no son posibles en volúmenes subordinados.

Para más información

Para obtener más información sobre el acceso a volúmenes entre espacios de nombres:

- Visita"Compartir volúmenes entre espacios de nombres: Descubra el acceso a volúmenes entre espacios de nombres." .
- Vea la demostración en"NetAppTV" .

Clonar volúmenes entre espacios de nombres

Con Trident, puedes crear nuevos volúmenes utilizando volúmenes existentes o instantáneas de volúmenes de un espacio de nombres diferente dentro del mismo clúster de Kubernetes.

Prerrequisitos

Antes de clonar volúmenes, asegúrese de que los backends de origen y destino sean del mismo tipo y tengan la misma clase de almacenamiento.

> La clonación entre espacios de nombres solo se admite para `ontap-san` y `ontap-nas` Controladores de almacenamiento. No se admiten clones de solo lectura.

Inicio rápido

Puedes configurar la clonación de volúmenes en tan solo unos pasos.

Configure el PVC de origen para clonar el volumen.

El propietario del espacio de nombres de origen otorga permiso para acceder a los datos en el PVC de origen.

Conceder permiso para crear un CR en el espacio de nombres de destino

El administrador del clúster otorga permiso al propietario del espacio de nombres de destino para crear el CR TridentVolumeReference.

Cree una referencia TridentVolumeReference en el espacio de nombres de destino.

El propietario del espacio de nombres de destino crea el CR TridentVolumeReference para hacer referencia al PVC de origen.

Cree el PVC clonado en el espacio de nombres de destino.

El propietario del espacio de nombres de destino crea un PVC para clonar el PVC del espacio de nombres de origen.

Configurar los espacios de nombres de origen y destino

Para garantizar la seguridad, la clonación de volúmenes entre espacios de nombres requiere la colaboración y la acción del propietario del espacio de nombres de origen, el administrador del clúster y el propietario del espacio de nombres de destino. El rol del usuario se designa en cada paso.

Pasos

1. **Propietario del espacio de nombres de origen:** Crear el PVC(`pvc1`) en el espacio de nombres de origen(`namespace1`) que otorga permiso para compartir con el espacio de nombres de destino(`namespace2`) usando el `cloneToNamespace` anotación.

```
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 el PV y su volumen de almacenamiento backend.

- Puedes compartir el PVC con múltiples espacios de nombres utilizando una lista delimitada por comas. Por ejemplo, `trident.netapp.io/cloneToNamespace: namespace2,namespace3,namespace4` .
- Puedes compartir con todos los espacios de nombres usando `*` . Por ejemplo, `trident.netapp.io/cloneToNamespace: *`
- Puedes actualizar el PVC para incluir el `cloneToNamespace` anotaciones en cualquier momento.

2. **Administrador del clúster:** Asegúrese de que el control de acceso basado en roles (RBAC) esté configurado correctamente para otorgar permiso al propietario del espacio de nombres de destino para crear el recurso compartido TridentVolumeReference en el espacio de nombres de destino.(`namespace2`).
3. **Propietario del espacio de nombres de destino:** Cree un CR TridentVolumeReference en el espacio de nombres de destino que haga referencia al espacio de nombres de origen. `pvc1` .

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. **Propietario del espacio de nombres de destino:** Crear un PVC(`pvc2`) en el espacio de nombres de destino(`namespace2`) usando el `cloneFromPVC` o `cloneFromSnapshot` , y `cloneFromNamespace` Anotaciones para designar el PVC de origen.

```
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
```

Limitaciones

- Para los PVC aprovisionados mediante controladores ontap-nas-economy, no se admiten clones de solo lectura.

Replicar volúmenes usando SnapMirror

Trident admite relaciones de espejo entre un volumen de origen en un clúster y el volumen de destino en el clúster emparejado para replicar datos para la recuperación ante desastres. Puede utilizar una definición de recurso personalizado (CRD) con espacio de nombres, denominada relación de espejo de Trident (TMR), para realizar las siguientes operaciones:

- Crear relaciones de simetría entre volúmenes (PVC)
- Eliminar las relaciones de simetría entre volúmenes
- Rompe las relaciones espejo
- Promover el volumen secundario durante situaciones de desastre (conmutaciones por error).
- Realizar una transición sin pérdidas de aplicaciones de un clúster a otro (durante conmutaciones por error o migraciones planificadas).

Requisitos previos de replicación

Asegúrese de que se cumplen los siguientes requisitos previos antes de comenzar:

Clústeres ONTAP

- * Trident*: Debe existir la versión 22.10 o posterior de Trident tanto en los clústeres de Kubernetes de origen como de destino que utilizan ONTAP como backend.
- **Licencias**: Las licencias asíncronas de ONTAP SnapMirror que utilizan el paquete de protección de datos deben estar habilitadas tanto en el clúster ONTAP de origen como en el de destino. Referirse a "Información general sobre licencias de SnapMirror en ONTAP" Para más información.

A partir de ONTAP 9.10.1, todas las licencias se entregan como un archivo de licencia de NetApp (NLF), que es un único archivo que habilita múltiples funciones. Referirse a"Licencias incluidas con ONTAP One" Para más información.

> Solo se admite la protección asíncrona de SnapMirror .

Mirando

- **Clúster y SVM**: Los backends de almacenamiento ONTAP deben estar interconectados. Referirse a "Descripción general del emparejamiento de clústeres y SVM" Para más información.

> Asegúrese de que los nombres de SVM utilizados en la relación de replicación entre dos clústeres ONTAP sean únicos.

- * Trident y SVM*: Las SVM remotas emparejadas deben estar disponibles para Trident en el clúster de destino.

Controladores compatibles

NetApp Trident admite la replicación de volúmenes con la tecnología NetApp SnapMirror mediante clases de almacenamiento compatibles con los siguientes controladores: **`ontap-nas` : NFS** `ontap-san` : iSCSI **`ontap-san` : FC** `ontap-san` : NVMe/TCP (requiere como mínimo la versión 9.15.1 de ONTAP)

> La replicación de volúmenes mediante SnapMirror no es compatible con los sistemas ASA r2. Para obtener información sobre los sistemas ASA r2, consulte"Conozca los sistemas de almacenamiento ASA r2" .

Crea un PVC espejado

Siga estos pasos y utilice los ejemplos de CRD para crear una relación de simetría entre los volúmenes primario y secundario.

Pasos

1. Realice los siguientes pasos en el clúster principal de Kubernetes:
   a. Crea un objeto StorageClass con el `trident.netapp.io/replication: true` parámetro.

      Ejemplo

      ```
      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. Cree un PVC con la StorageClass creada previamente.

Ejemplo

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: csi-nas
```

c. Cree un CR MirrorRelationship con información local.

Ejemplo

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: promoted
  volumeMappings:
  - localPVCName: csi-nas
```

Trident recupera la información interna del volumen y el estado actual de protección de datos (DP) del volumen, y luego completa el campo de estado de MirrorRelationship.

d. Obtenga el CR TridentMirrorRelationship para obtener el nombre interno y el 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. Realice los siguientes pasos en el clúster secundario de Kubernetes:
   a. Cree una StorageClass con el parámetro trident.netapp.io/replication: true.

      Ejemplo

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

   b. Cree un CR MirrorRelationship con información de destino y origen.

      Ejemplo

```
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 creará una relación SnapMirror con el nombre de política de relación configurado (o el predeterminado para ONTAP) y la inicializará.

c. Cree un PVC con la StorageClass creada previamente para que actúe como destino secundario (destino SnapMirror).

Ejemplo

```
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 comprobará la existencia del CRD TridentMirrorRelationship y no podrá crear el volumen si la relación no existe. Si existe la relación, Trident se asegurará de que el nuevo FlexVol volume se coloque en una SVM que esté emparejada con la SVM remota definida en MirrorRelationship.

Estados de replicación de volumen

Una relación de espejo Trident (TMR) es un CRD que representa un extremo de una relación de replicación entre PVC. El TMR de destino tiene un estado, que le indica a Trident cuál es el estado deseado. El TMR de destino tiene los siguientes estados:

- **Establecido**: el PVC local es el volumen de destino de una relación de espejo, y esta es una nueva relación.
- **Promocionado**: el PVC local es de lectura/escritura y se puede montar, sin ninguna relación de espejo actualmente en vigor.
- **Restablecido**: el PVC local es el volumen de destino de una relación de espejo y también estuvo previamente en esa relación de espejo.
  - Debe utilizarse el estado restablecido si el volumen de destino alguna vez estuvo relacionado con el volumen de origen, ya que sobrescribe el contenido del volumen de destino.
  - El estado restablecido fallará si el volumen no estaba previamente relacionado con la fuente.

Promover la PVC secundaria durante una conmutación por error no planificada

Realice el siguiente paso en el clúster de Kubernetes secundario:

- Actualizar el campo *spec.state* de TridentMirrorRelationship a `promoted` .

Promover la PVC secundaria durante una conmutación por error planificada

Durante una conmutación por error planificada (migración), realice los siguientes pasos para promover el PVC secundario:

Pasos

1. En el clúster principal de Kubernetes, cree una instantánea del PVC y espere hasta que se cree la instantánea.
2. En el clúster principal de Kubernetes, cree el CR SnapshotInfo para obtener detalles internos.

   Ejemplo

   ```
   kind: SnapshotInfo
   apiVersion: trident.netapp.io/v1
   metadata:
     name: csi-nas
   spec:
     snapshot-name: csi-nas-snapshot
   ```

3. En el clúster secundario de Kubernetes, actualice el campo *spec.state* del CR *TridentMirrorRelationship* a *promoted* y *spec.promotedSnapshotHandle* para que sea el internalName de la instantánea.
4. En el clúster secundario de Kubernetes, confirme que el estado (campo status.state) de TridentMirrorRelationship sea promovido.

Restablecer una relación de espejo después de una conmutación por error

Antes de restablecer una relación de espejo, elige el lado que quieres convertir en el nuevo principal.

Pasos

1. En el clúster secundario de Kubernetes, asegúrese de que se actualicen los valores del campo *spec.remoteVolumeHandle* en TridentMirrorRelationship.
2. En el clúster secundario de Kubernetes, actualice el campo *spec.mirror* de TridentMirrorRelationship a `reestablished`.

Operaciones adicionales

Trident admite las siguientes operaciones en los volúmenes primario y secundario:

Replicar el PVC primario a un nuevo PVC secundario

Asegúrese de que ya dispone de un tubo de PVC primario y un tubo de PVC secundario.

Pasos

1. Elimine los CRD PersistentVolumeClaim y TridentMirrorRelationship del clúster secundario (de destino) establecido.
2. Elimine el CRD TridentMirrorRelationship del clúster primario (de origen).
3. Cree un nuevo CRD TridentMirrorRelationship en el clúster primario (origen) para el nuevo PVC secundario (destino) que desea establecer.

Cambiar el tamaño de un PVC reflejado, primario o secundario

El tamaño del PVC se puede cambiar como de costumbre; ONTAP expandirá automáticamente cualquier destino flevxols si la cantidad de datos excede el tamaño actual.

Eliminar la replicación de un PVC

Para eliminar la replicación, realice una de las siguientes operaciones en el volumen secundario actual:

- Elimine la relación MirrorRelationship en el PVC secundario. Esto rompe la relación de replicación.
- O bien, actualice el campo spec.state a *promoted*.

Eliminar un PVC (que previamente se había duplicado)

Trident comprueba si existen PVC replicados y libera la relación de replicación antes de intentar eliminar el volumen.

Eliminar un TMR

Eliminar un TMR en un lado de una relación reflejada provoca que el TMR restante pase al estado *promocionado* antes de que Trident complete la eliminación. Si el TMR seleccionado para su eliminación ya se encuentra en estado *promocionado*, no existe ninguna relación de réplica y el TMR será eliminado y Trident promoverá el PVC local a *Lectura/Escritura*. Esta eliminación libera los metadatos de SnapMirror para el volumen local en ONTAP. Si este volumen se utiliza en una relación de espejo en el futuro, deberá utilizar un nuevo TMR con un estado de replicación de volumen *establecido* al crear la nueva relación de espejo.

Actualizar las relaciones de espejo cuando ONTAP esté en línea

Las relaciones de espejo se pueden actualizar en cualquier momento después de que se hayan establecido. Puedes utilizar el `state: promoted` o `state: reestablished` campos para actualizar las relaciones. Al promover un volumen de destino a un volumen ReadWrite normal, puede usar *promotedSnapshotHandle* para especificar una instantánea específica a la que restaurar el volumen actual.

Actualizar las relaciones de réplica cuando ONTAP esté fuera de línea

Puede utilizar un CRD para realizar una actualización de SnapMirror sin que Trident tenga conectividad directa con el clúster ONTAP . Consulte el siguiente ejemplo de formato de TridentActionMirrorUpdate:

Ejemplo

```
apiVersion: trident.netapp.io/v1
kind: TridentActionMirrorUpdate
metadata:
  name: update-mirror-b
spec:
  snapshotHandle: "pvc-1234/snapshot-1234"
  tridentMirrorRelationshipName: mirror-b
```

`status.state`refleja el estado del CRD TridentActionMirrorUpdate. Puede tomar un valor de *Succeeded*, *In Progress* o *Failed*.

Utilizar la topología CSI

Trident puede crear y adjuntar volúmenes de forma selectiva a los nodos presentes en un clúster de Kubernetes mediante el uso de "Función de topología CSI" .

Descripción general

Al utilizar la función de topología CSI, el acceso a los volúmenes se puede limitar a un subconjunto de nodos, según las regiones y las zonas de disponibilidad. Actualmente, los proveedores de servicios en la nube permiten a los administradores de Kubernetes crear nodos basados en zonas. Los nodos pueden ubicarse en diferentes zonas de disponibilidad dentro de una región o en varias regiones. Para facilitar el aprovisionamiento de volúmenes para cargas de trabajo en una arquitectura multizona, Trident utiliza la topología CSI.

Obtenga más información sobre la función de topología CSI. "aquí" .

Kubernetes proporciona dos modos únicos de enlace de volúmenes:

- Con `VolumeBindingMode` empezar a `Immediate` Trident crea el volumen sin tener en cuenta la topología. La vinculación de volúmenes y el aprovisionamiento dinámico se gestionan cuando se crea el PVC. Este es el valor predeterminado `VolumeBindingMode` y es adecuado para clústeres que no imponen restricciones de topología. Los volúmenes persistentes se crean sin depender de los requisitos de programación del pod solicitante.
- Con `VolumeBindingMode` empezar a `WaitForFirstConsumer` La creación y el enlace de un volumen persistente para un PVC se retrasan hasta que se programa y se crea un pod que utiliza el PVC. De esta manera, se crean volúmenes para cumplir con las restricciones de programación impuestas por los requisitos de topología.

El `WaitForFirstConsumer` El modo de enlace no requiere etiquetas de topología. Esto se puede utilizar independientemente de la función de topología CSI.

Lo que necesitarás

Para utilizar la topología CSI, necesita lo siguiente:

- Un clúster de Kubernetes que ejecuta un"Versión de Kubernetes compatible"

```
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"}
```

- Los nodos del clúster deben tener etiquetas que indiquen la topología.(`topology.kubernetes.io/region` y `topology.kubernetes.io/zone`). Estas etiquetas **deben estar presentes en los nodos del clúster** antes de instalar Trident para que Trident tenga en cuenta la topología.

```
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"}]
```

Paso 1: Crear un backend que tenga en cuenta la topología

Los sistemas de almacenamiento Trident pueden diseñarse para aprovisionar volúmenes de forma selectiva en función de las zonas de disponibilidad. Cada backend puede incluir una opción. `supportedTopologies` bloque que representa una lista de zonas y regiones admitidas. Para las StorageClasses que utilizan dicho backend, solo se creará un volumen si lo solicita una aplicación programada en una zona o región compatible.

Aquí tenéis un ejemplo de definición de 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`Se utiliza para proporcionar una lista de regiones y zonas por backend. Estas regiones y zonas representan la lista de valores permitidos que se pueden proporcionar en una StorageClass. Para las StorageClasses que contienen un subconjunto de las regiones y zonas proporcionadas en un backend, Trident crea un volumen en el backend.

Puedes definir `supportedTopologies` por cada grupo de almacenamiento también. Vea el siguiente

ejemplo:

```
---
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
```

En este ejemplo, el `region` y `zone` Las etiquetas indican la ubicación del depósito de almacenamiento. `topology.kubernetes.io/region` y `topology.kubernetes.io/zone` dictar desde dónde se pueden consumir los grupos de almacenamiento.

Paso 2: Defina las StorageClasses que tengan en cuenta la topología.

En función de las etiquetas de topología que se proporcionan a los nodos del clúster, se pueden definir StorageClasses para que contengan información de topología. Esto determinará los grupos de almacenamiento que sirven como candidatos para las solicitudes de PVC realizadas, y el subconjunto de nodos que pueden utilizar los volúmenes aprovisionados por Trident.

Vea el siguiente ejemplo:

```
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
```

En la definición de StorageClass proporcionada anteriormente, `volumeBindingMode` está configurado para `WaitForFirstConsumer` . Los PVC que se soliciten con esta StorageClass no se procesarán hasta que se haga referencia a ellos en un pod. Y, `allowedTopologies` proporciona las zonas y la región que se utilizarán. El `netapp-san-us-east1` StorageClass crea PVC en el `san-backend-us-east1` Backend definido anteriormente.

Paso 3: Crear y usar un PVC

Una vez creada la StorageClass y asignada a un backend, ya puede crear PVC.

Vea el ejemplo `spec` abajo:

```
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata: null
name: pvc-san
spec: null
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: netapp-san-us-east1
```

La creación de un PVC utilizando este manifiesto daría como resultado lo siguiente:

```
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
```

Para que Trident cree un volumen y lo una al PVC, utilice el PVC en una cápsula. Vea el siguiente ejemplo:

```
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
```

Esta especificación de pod indica a Kubernetes que programe el pod en los nodos que están presentes en el `us-east1` región, y elige cualquier nodo que esté presente en la `us-east1-a` o `us-east1-b` zonas.

Vea el siguiente resultado:

```
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
```

Actualizar los backends para incluir `supportedTopologies`

Los sistemas backend preexistentes se pueden actualizar para incluir una lista de `supportedTopologies` usando `tridentctl backend update` . Esto no afectará a los volúmenes que ya se hayan aprovisionado y solo se utilizará para los PVC posteriores.

Encuentra más información

- "Gestionar recursos para contenedores"
- "selector de nodo"
- "Afinidad y antiafinidad"
- "Manchas y tolerancias"

Trabajar con instantáneas

Las instantáneas de volúmenes persistentes (PV) de Kubernetes permiten realizar copias puntuales de los volúmenes. Puede crear una instantánea de un volumen creado con Trident, importar una instantánea creada fuera de Trident, crear un nuevo volumen a partir de una instantánea existente y recuperar datos de volumen a partir de instantáneas.

Descripción general

La instantánea de volumen es compatible con `ontap-nas` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy` , `solidfire-san` , `gcp-cvs` , `azure-netapp-files` , y `google-cloud-netapp-volumes` conductores.

Antes de empezar

Para trabajar con instantáneas, debe disponer de un controlador de instantáneas externo y definiciones de recursos personalizados (CRD). Esta es responsabilidad del orquestador de Kubernetes (por ejemplo: Kubeadm, GKE, OpenShift).

Si su distribución de Kubernetes no incluye el controlador de instantáneas ni los CRD, consulteImplementar un controlador de instantáneas de volumen .

No cree un controlador de instantáneas si va a crear instantáneas de volumen bajo demanda en un entorno GKE. GKE utiliza un controlador de instantáneas integrado y oculto.

Cree una instantánea de volumen

Pasos

1. Crear una `VolumeSnapshotClass` Para obtener más información, consulte"Clase de instantánea de volumen" .
   - El `driver` señala al controlador Trident CSI.
   - `deletionPolicy`puede ser `Delete o `Retain` . Cuando se configura para `Retain` La instantánea física subyacente en el clúster de almacenamiento se conserva incluso cuando `VolumeSnapshot` El objeto ha sido eliminado.

     Ejemplo

     ```
     cat snap-sc.yaml
     ```

     ```
     apiVersion: snapshot.storage.k8s.io/v1
     kind: VolumeSnapshotClass
     metadata:
       name: csi-snapclass
     driver: csi.trident.netapp.io
     deletionPolicy: Delete
     ```

2. Cree una instantánea de un PVC existente.

   Ejemplos

   - Este ejemplo crea una instantánea de un PVC existente.

     ```
     cat snap.yaml
     ```

     ```
     apiVersion: snapshot.storage.k8s.io/v1
     kind: VolumeSnapshot
     metadata:
       name: pvc1-snap
     spec:
       volumeSnapshotClassName: csi-snapclass
       source:
         persistentVolumeClaimName: pvc1
     ```

   - Este ejemplo crea un objeto de instantánea de volumen para un PVC llamado `pvc1` y el nombre de la instantánea se establece en `pvc1-snap` . Un VolumeSnapshot es análogo a un PVC y está asociado con un `VolumeSnapshotContent` objeto que representa la instantánea real.

```
kubectl create -f snap.yaml
volumesnapshot.snapshot.storage.k8s.io/pvc1-snap created

kubectl get volumesnapshots
NAME                    AGE
pvc1-snap               50s
```

- Puedes identificar el `VolumeSnapshotContent` objeto para el `pvc1-snap` VolumeSnapshot describiéndolo. El `Snapshot Content Name` identifica el objeto VolumeSnapshotContent que sirve a esta instantánea. El `Ready To Use` El parámetro indica que la instantánea se puede utilizar para crear un nuevo 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
...
```

Cree un PVC a partir de una instantánea de volumen.

Puedes utilizar `dataSource` para crear un PVC utilizando un VolumeSnapshot llamado `<pvc-name>` como fuente de los datos. Una vez creado el PVC, se puede acoplar a una cápsula y utilizar como cualquier otro PVC.

> El PVC se creará en el mismo backend que el volumen fuente. Referirse a"KB: No se puede crear un PVC a partir de una instantánea de PVC de Trident en un backend alternativo." .

El siguiente ejemplo crea el PVC utilizando `pvc1-snap` como fuente de datos.

```
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
```

Importar una instantánea de volumen

Trident apoya a"Proceso de instantáneas preaprovisionadas de Kubernetes" para permitir que el administrador del clúster cree un `VolumeSnapshotContent` Objetos e instantáneas de importación creadas fuera de Trident.

Antes de empezar

Trident debe haber creado o importado el volumen principal de la instantánea.

Pasos

1. **Administrador del clúster:** Crear un `VolumeSnapshotContent` Objeto que hace referencia a la instantánea del backend. Esto inicia el flujo de trabajo de instantáneas en Trident.
   - Especifique el nombre de la instantánea del backend en `annotations` como `trident.netapp.io/internalSnapshotName: <"backend-snapshot-name">`.
   - Especificar `<name-of-parent-volume-in-trident>/<volume-snapshot-content-name>` en `snapshotHandle` Esta es la única información proporcionada a Trident por el capturador de instantáneas externo en el `ListSnapshots` llamar.

     > El `<volumeSnapshotContentName>` No siempre puede coincidir el nombre de la instantánea del backend debido a las restricciones de nomenclatura de CR.

     Ejemplo

     El siguiente ejemplo crea un `VolumeSnapshotContent` Objeto que hace referencia a la instantánea 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. **Administrador del clúster:** Crear el `VolumeSnapshot` CR que hace referencia al `VolumeSnapshotContent` objeto. Esta solicitud permite el uso de `VolumeSnapshot` en un espacio de nombres determinado.

   Ejemplo

   El siguiente ejemplo crea un `VolumeSnapshot` CR llamado `import-snap` que hace referencia a `VolumeSnapshotContent` llamado `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. **Procesamiento interno (no se requiere ninguna acción):** El generador de instantáneas externo reconoce la instantánea recién creada `VolumeSnapshotContent` y ejecuta el `ListSnapshots` llamar. Trident crea el `TridentSnapshot` .
   - El capturador externo establece el `VolumeSnapshotContent` a `readyToUse` y el `VolumeSnapshot` a `true` .
   - El Trident regresa `readyToUse=true` .
4. **Cualquier usuario:** Crea un `PersistentVolumeClaim` para hacer referencia al nuevo `VolumeSnapshot` , donde el `spec.dataSource` (o `spec.dataSourceRef`) el nombre es el

`VolumeSnapshot` nombre.

Ejemplo

El siguiente ejemplo crea un PVC que hace referencia al `VolumeSnapshot` llamado `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
```

Recuperar datos de volumen mediante instantáneas

El directorio de instantáneas está oculto de forma predeterminada para facilitar la máxima compatibilidad de los volúmenes aprovisionados mediante `ontap-nas` y `ontap-nas-economy` conductores. Habilitar el `.snapshot` directorio para recuperar datos directamente desde instantáneas.

Utilice la CLI de ONTAP para restaurar instantáneas de volumen para restaurar un volumen a un estado registrado en una instantánea anterior.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

> Al restaurar una copia de instantánea, se sobrescribe la configuración de volumen existente. Los cambios realizados en los datos del volumen después de la creación de la copia de instantánea se pierden.

Restauración de volumen in situ a partir de una instantánea

Trident proporciona una restauración de volumen rápida e in situ a partir de una instantánea utilizando la `TridentActionSnapshotRestore` (TASR) CR. Esta solicitud de cambio (CR) funciona como una acción imperativa de Kubernetes y no persiste una vez finalizada la operación.

Trident admite la restauración de instantáneas en el `ontap-san` , `ontap-san-economy` , `ontap-nas` , `ontap-nas-flexgroup` , `azure-netapp-files` , `gcp-cvs` , `google-cloud-netapp-volumes` , y `solidfire-san` conductores.

Antes de empezar

Debe tener un PVC vinculado y una instantánea de volumen disponible.

- Verifique que el estado del PVC esté vinculado.

```
kubectl get pvc
```

- Verifique que la instantánea del volumen esté lista para usar.

```
kubectl get vs
```

Pasos

1. Cree el TASR CR. Este ejemplo crea una solicitud de cambio para PVC. `pvc1` y instantánea de volumen `pvc1-snapshot` .

   El CR TASR debe estar en un espacio de nombres donde existan el PVC y el 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. Aplique el CR para restaurar desde la instantánea. Este ejemplo restaura desde una instantánea. `pvc1` .

```
kubectl create -f tasr-pvc1-snapshot.yaml
```

```
tridentactionsnapshotrestore.trident.netapp.io/trident-snap created
```

Resultados

Trident restaura los datos a partir de la instantánea. Puedes verificar el estado de restauración de la instantánea:

```
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: ""
```

- En la mayoría de los casos, Trident no reintentará automáticamente la operación en caso de fallo. Deberá repetir la operación.
- Los usuarios de Kubernetes sin acceso de administrador podrían necesitar que el administrador les otorgue permiso para crear un CR TASR en el espacio de nombres de su aplicación.

Eliminar un PV con instantáneas asociadas

Al eliminar un volumen persistente con instantáneas asociadas, el volumen Trident correspondiente se actualiza a un estado de "Eliminación". Elimine las instantáneas de volumen para borrar el volumen de Trident .

Implementar un controlador de instantáneas de volumen

Si tu distribución de Kubernetes no incluye el controlador de instantáneas y los CRD, puedes implementarlos de la siguiente manera.

Pasos

1. Crear CRD de instantáneas de volumen.

```
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. Crea el controlador de instantáneas.

```
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
```

Si es necesario, abra `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` y actualizar `namespace` a tu espacio de nombres.

Enlaces relacionados

- "Instantáneas de volumen"
- "Clase de instantánea de volumen"

Trabajar con instantáneas de grupos de volúmenes

Instantáneas de grupos de volúmenes de Kubernetes de volúmenes persistentes (PV) NetApp Trident ofrece la capacidad de crear instantáneas de múltiples volúmenes (un grupo de instantáneas de volumen). Esta instantánea del grupo de volúmenes representa copias de múltiples volúmenes tomadas en el mismo momento.

VolumeGroupSnapshot es una función beta en Kubernetes con API beta. Kubernetes 1.32 es la versión mínima requerida para VolumeGroupSnapshot.

Crear instantáneas de grupos de volúmenes

La instantánea de grupo de volúmenes es compatible con `ontap-san` Controlador, solo para el protocolo iSCSI, aún no compatible con Fibre Channel (FCP) ni NVMe/TCP. Antes de comenzar

- Asegúrese de que su versión de Kubernetes sea K8s 1.32 o superior.
- Para trabajar con instantáneas, debe disponer de un controlador de instantáneas externo y definiciones de recursos personalizados (CRD). Esta es responsabilidad del orquestador de Kubernetes (por ejemplo: Kubeadm, GKE, OpenShift).

  Si su distribución de Kubernetes no incluye el controlador de instantáneas externo ni los CRD, consulteImplementar un controlador de instantáneas de volumen .

  No cree un controlador de instantáneas si va a crear instantáneas de grupos de volúmenes bajo demanda en un entorno GKE. GKE utiliza un controlador de instantáneas integrado y oculto.

- En el archivo YAML del controlador de instantáneas, configure el `CSIVolumeGroupSnapshot` Establezca el valor de la función en 'true' para garantizar que la instantánea del grupo de volúmenes esté habilitada.
- Cree las clases de instantáneas de grupo de volúmenes necesarias antes de crear una instantánea de grupo de volúmenes.
- Asegúrese de que todos los PVC/volúmenes estén en el mismo SVM para poder crear VolumeGroupSnapshot.

Pasos

- Cree una clase VolumeGroupSnapshotClass antes de crear un objeto VolumeGroupSnapshot. Para obtener más información, consulte"Clase de instantánea de grupo de volumen" .

```
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
```

- Cree PVC con las etiquetas necesarias utilizando las clases de almacenamiento existentes, o agregue estas etiquetas a los PVC existentes.

  El siguiente ejemplo crea el PVC utilizando `pvc1-group-snap` como fuente de datos y etiqueta `consistentGroupSnapshot: groupA` . Define la clave y el valor de la etiqueta según tus requisitos.

```
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 misma etiqueta(`consistentGroupSnapshot: groupA`) especificado en el PVC.

  Este ejemplo crea una instantánea de grupo de volúmenes:

```
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshot
metadata:
  name: "vgs1"
  namespace: trident
spec:
  volumeGroupSnapshotClassName: csi-group-snap-class
  source:
    selector:
      matchLabels:
        consistentGroupSnapshot: groupA
```

Recuperar datos de volumen mediante una instantánea de grupo

Puede restaurar volúmenes persistentes individuales utilizando las instantáneas individuales que se han creado como parte de la instantánea del grupo de volúmenes. No se puede recuperar la instantánea del grupo de volúmenes como una unidad.

Utilice la CLI de ONTAP para restaurar instantáneas de volumen para restaurar un volumen a un estado registrado en una instantánea anterior.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

Al restaurar una copia de instantánea, se sobrescribe la configuración de volumen existente. Los cambios realizados en los datos del volumen después de la creación de la copia de instantánea se pierden.

Restauración de volumen in situ a partir de una instantánea

Trident proporciona una restauración de volumen rápida e in situ a partir de una instantánea utilizando la `TridentActionSnapshotRestore` (TASR) CR. Esta solicitud de cambio (CR) funciona como una acción imperativa de Kubernetes y no persiste una vez finalizada la operación.

Para obtener más información, consulte "Restauración de volumen in situ a partir de una instantánea".

Eliminar un PV con instantáneas de grupo asociadas

Al eliminar una instantánea de volumen de grupo:

- Puede eliminar VolumeGroupSnapshots en su totalidad, no instantáneas individuales dentro del grupo.
- Si se eliminan volúmenes persistentes mientras existe una instantánea para ese volumen persistente, Trident moverá ese volumen a un estado de "eliminación" porque la instantánea debe eliminarse antes de que el volumen pueda eliminarse de forma segura.
- Si se ha creado un clon utilizando una instantánea agrupada y luego se va a eliminar el grupo, se iniciará una operación de división en clon y el grupo no se podrá eliminar hasta que se complete la división.

Implementar un controlador de instantáneas de volumen

Si tu distribución de Kubernetes no incluye el controlador de instantáneas y los CRD, puedes implementarlos de la siguiente manera.

Pasos

1. Crear CRD de instantáneas de volumen.

```
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_volumegroupsnapshotclasses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotcontents.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. Crea el controlador de instantáneas.

```
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
```

Si es necesario, abra `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` y actualizar `namespace` a tu espacio de nombres.

Enlaces relacionados

- "Clase de instantánea de grupo de volumen"
- "Instantáneas de volumen"