



Utilice Astra Trident

Astra Trident

NetApp
January 14, 2026

Tabla de contenidos

Utilice Astra Trident	1
Prepare el nodo de trabajo	1
Seleccionar las herramientas adecuadas	1
Detección del servicio de nodos	1
Volúmenes NFS	2
Volúmenes iSCSI	2
Volúmenes NVMe/TCP	6
Configurar y gestionar back-ends	7
Configurar los back-ends	7
Azure NetApp Files	8
NetApp Volumes para Google Cloud	26
Configure un back-end de Cloud Volumes Service para Google Cloud	38
Configure un back-end de NetApp HCI o SolidFire	50
Controladores para SAN de ONTAP	56
Controladores para NAS de ONTAP	80
Amazon FSX para ONTAP de NetApp	108
Cree back-ends con kubectl	141
Gestionar back-ends	148
Crear y gestionar clases de almacenamiento	157
Cree una clase de almacenamiento	157
Gestione las clases de almacenamiento	160
Aprovisione y gestione volúmenes	162
Aprovisione un volumen	162
Expanda los volúmenes	167
Importar volúmenes	175
Personalizar nombres y etiquetas de volúmenes	182
Comparta un volumen NFS en espacios de nombres	185
Replicar volúmenes mediante SnapMirror	189
Utilice Topología CSI	205
Trabajar con instantáneas	213

Utilice Astra Trident

Prepare el nodo de trabajo

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

Seleccionar las herramientas adecuadas

Si está utilizando una combinación de controladores, debe instalar todas las herramientas necesarias para sus controladores. Las versiones recientes de RedHat CoreOS tienen las herramientas instaladas de forma predeterminada.

Herramientas de NFS

"[Instale las herramientas NFS](#)" si utiliza: `ontap-nas`, `ontap-nas-economy`, `ontap-nas-flexgroup`, `azure-netapp-files`, `gcp-cvs`.

Herramientas iSCSI

"[Instale las herramientas iSCSI](#)" si está utilizando `ontap-san`: `ontap-san-economy`, `solidfire-san`.

Herramientas de NVMe

"[Instale las herramientas NVMe](#)" Si utiliza `ontap-san` para el protocolo de memoria no volátil rápida (NVMe) sobre TCP (NVMe/TCP).



Recomendamos ONTAP 9,12 o posterior para NVMe/TCP.

Detección del servicio de nodos

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



La detección de servicios de nodo identifica los servicios detectados, pero no garantiza que los servicios se configuren correctamente. Por el contrario, la ausencia de un servicio detectado no garantiza que se produzca un error en el montaje del volumen.

Revisar los eventos

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

```
kubect1 get event -A --field-selector involvedObject.name=<Kubernetes node name>
```

Revisar los servicios detectados

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

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

Volúmenes NFS

Instale las herramientas de NFS mediante los comandos del sistema operativo. Asegúrese de que el servicio NFS se haya iniciado durante el arranque.

RHEL 8 O POSTERIOR

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

Ubuntu

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



Reinicie los nodos de trabajo después de instalar las herramientas NFS para evitar que se produzcan fallos cuando conecte volúmenes a los contenedores.

Volúmenes iSCSI

Astra Trident puede establecer automáticamente una sesión iSCSI, analizar LUN y detectar dispositivos multivía, darles formato y montarlos en un pod.

Funcionalidades de reparación automática de iSCSI

En el caso de los sistemas ONTAP, Astra Trident ejecuta la reparación automática de iSCSI cada cinco minutos para:

1. **Identifique** el estado de sesión iSCSI deseado y el estado actual de la sesión iSCSI.
2. **Compare** el estado deseado al estado actual para identificar las reparaciones necesarias. Astra Trident determina las prioridades de reparación y cuándo deben anticiparse a las reparaciones.
3. **Realice las reparaciones** necesarias para devolver el estado actual de la sesión iSCSI al estado deseado de la sesión iSCSI.



Los registros de la actividad de autorrecuperación se encuentran en `trident-main` el contenedor del pod Daemonset correspondiente. Para ver los registros, debe haber establecido `debug` el valor «true» durante la instalación de Astra Trident.

Las funcionalidades de reparación automática de iSCSI de Astra Trident pueden ayudar a prevenir:

- Sesiones iSCSI obsoletas o poco saludables que podrían producirse después de un problema de conectividad de red. En caso de una sesión obsoleta, Astra Trident espera siete minutos antes de cerrar la sesión para restablecer la conexión con un portal.



Por ejemplo, si los secretos CHAP se rotaban en la controladora de almacenamiento y la red pierde la conectividad, podrían persistir los secretos CHAP antiguos (*obsoleta*). La reparación automática puede reconocer esto y restablecer automáticamente la sesión para aplicar los secretos CHAP actualizados.

- Faltan sesiones iSCSI
- Faltan LUN

Puntos a tener en cuenta antes de actualizar Trident

- Si solo se utilizan iGroups por nodo (introducidos en 23,04+), la reparación automática de iSCSI iniciará los análisis de SCSI para todos los dispositivos del bus SCSI.
- Si solo se utilizan iGroups de ámbito back-end (obsoletos a partir de 23,04), la reparación automática de iSCSI iniciará los nuevos análisis SCSI de los ID exactos de LUN en el bus SCSI.
- Si se utiliza una combinación de iGroups por nodo y iGroups de ámbito back-end, la reparación automática de iSCSI iniciará los análisis SCSI de los ID exactos de LUN en el bus SCSI.

Instale las herramientas iSCSI

Instale las herramientas iSCSI mediante los comandos del 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 `solidfire-san` el controlador y Element OS 12,5 o anterior, asegúrese de que el algoritmo de autenticación CHAP se haya configurado en MD5 en `/etc/iscsi/iscsid.conf`. Los algoritmos CHAP seguros 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 se utilicen los nodos de trabajo que ejecutan RHEL/RedHat CoreOS con VP iSCSI, especifique `discard mountOption` en `StorageClass` para realizar la recuperación de espacio en línea. Consulte ["Documentación de redhat"](#).

RHEL 8 O POSTERIOR

1. Instale los siguientes paquetes del sistema:

```
sudo yum install -y lsscsi iscsi-initiator-utils sg3_utils device-  
mapper-multipath
```

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

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

3. Configure el escaneo en manual:

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

4. Activar accesos múltiples:

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



Asegúrese de etc/multipath.conf que contiene find_multipaths no en defaults.

5. Asegúrese de que iscsid y multipathd están en ejecución:

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

6. Activar 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 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. Configure el escaneo en manual:

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

4. Activar accesos 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
```



Asegúrese de `etc/multipath.conf` que contiene `find_multipaths no` en `defaults`.

5. Asegúrese de que `open-iscsi` y `multipath-tools` están activados y en ejecución:

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



Para Ubuntu 18,04, debe detectar los puertos de destino con `iscsiadm` antes de iniciar `open-iscsi` el daemon iSCSI. También puede modificar el `iscsi` servicio para que se inicie `iscsid` automáticamente.

Configure o deshabilite la reparación automática de iSCSI

Puede configurar los siguientes ajustes de reparación automática de iSCSI de Astra Trident para corregir sesiones obsoletas:

- **Intervalo de autorrecuperación iSCSI:** Determina la frecuencia a la que se invoca la autorrecuperación iSCSI (valor predeterminado: 5 minutos). Puede configurarlo para que se ejecute con más frecuencia estableciendo un número menor o con menos frecuencia estableciendo un número mayor.



Si se configura el intervalo de reparación automática de iSCSI en 0, se detiene por completo la reparación automática de iSCSI. No recomendamos deshabilitar la reparación automática de iSCSI; solo debe deshabilitarse en ciertos casos cuando la reparación automática de iSCSI no funciona como se esperaba o con fines de depuración.

- **Tiempo de espera de autorrecuperación iSCSI:** Determina la duración de las esperas de autorrecuperación iSCSI antes de cerrar sesión en una sesión en mal estado e intentar iniciar sesión de nuevo (por defecto: 7 minutos). Puede configurarlo a un número mayor para que las sesiones identificadas como en mal estado tengan que esperar más tiempo antes de cerrar la sesión y, a continuación, se intente volver a iniciar sesión, o un número menor para cerrar la sesión e iniciar sesión anteriormente.

Timón

Para configurar o cambiar los ajustes de reparación automática de iSCSI, pase los `iscsiSelfHealingInterval` parámetros y `iscsiSelfHealingWaitTime` durante la instalación del timón o la actualización del timón.

En el siguiente ejemplo, se establece el intervalo de reparación automática de iSCSI en 3 minutos y el tiempo de espera de reparación automática en 6 minutos:

```
helm install trident trident-operator-100.2406.0.tgz --set
iscsiSelfHealingInterval=3m0s --set iscsiSelfHealingWaitTime=6m0s -n
trident
```

tridentctl

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

En el siguiente ejemplo, se establece el intervalo de reparación automática de iSCSI en 3 minutos y el tiempo de espera de reparación automática en 6 minutos:

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

Volúmenes NVMe/TCP

Instale las herramientas NVMe mediante los comandos de su 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 la versión de kernel, es posible que deba actualizar la versión del kernel del nodo a una con 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
```

Verifique la instalación

Después de la instalación, compruebe que cada nodo del clúster de Kubernetes tenga un NQN único mediante el comando:

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



Astra Trident modifica `ctrl_device_tmo` el valor para garantizar que NVMe no se rinde en el camino si cae. No cambie esta configuración.

Configurar y gestionar back-ends

Configurar los back-ends

Un back-end define la relación entre Astra Trident y un sistema de almacenamiento. Le indica a Astra Trident cómo se comunica con ese sistema de almacenamiento y cómo debe aprovisionar volúmenes a partir de él.

Astra Trident ofrece automáticamente pools de almacenamiento a partir de los back-ends que cumplan los requisitos definidos por una clase de almacenamiento. Aprenda a configurar el back-end para el sistema de almacenamiento.

- ["Configure un back-end de Azure NetApp Files"](#)
- ["Configure un back-end de Cloud Volumes Service para Google Cloud Platform"](#)
- ["Configure un back-end de NetApp HCI o SolidFire"](#)
- ["Configure un back-end con controladores NAS ONTAP o Cloud Volumes ONTAP"](#)
- ["Configure un back-end con controladores SAN ONTAP o Cloud Volumes ONTAP"](#)
- ["Utilice Astra Trident con Amazon FSX para ONTAP de NetApp"](#)

Azure NetApp Files

Configure un back-end de Azure NetApp Files

Puede configurar Azure NetApp Files como back-end de Astra Trident. Puede asociar volúmenes NFS y SMB con un back-end de Azure NetApp Files. Astra Trident también es compatible con la gestión de credenciales mediante identidades gestionadas para clústeres de Azure Kubernetes Services (AKS).

Información del controlador de Azure NetApp Files

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

Controlador	Protocolo	VolumeMo de	Modos de acceso compatibles	Sistemas de archivos compatibles
azure-netapp-files	BLOQUE DE MENSAJES DEL SERVIDOR NFS	Sistema de archivos	RWO, ROX, RWX, RWOP	nfs, smb

Consideraciones

- El servicio Azure NetApp Files no admite volúmenes de menos de 100 GB. Astra Trident crea automáticamente volúmenes de 100 GiB si se solicita un volumen más pequeño.
- Astra Trident admite volúmenes de SMB montados en pods que se ejecutan solo en nodos de Windows.

Identidades administradas para AKS

Astra Trident es compatible con "[identidades administradas](#)" los clústeres de Azure Kubernetes Services. Para aprovechar la gestión de credenciales optimizada que ofrecen las identidades gestionadas, debe tener:

- Un clúster de Kubernetes puesto en marcha mediante AKS
- Identidades gestionadas configuradas en el clúster de kubernetes de AKS
- Astra Trident instalado que incluye el `cloudProvider` para especificar "Azure".

Operador de Trident

Para instalar Astra Trident con el operador Trident, edite `tridentorchestrator_cr.yaml` en Establecer `cloudProvider` en "Azure". Por ejemplo:

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

Timón

En el siguiente ejemplo se instalan conjuntos de Astra Trident `cloudProvider` en Azure con la variable de entorno `$CP`:

```
helm install trident trident-operator-100.2406.0.tgz --create
--namespace --namespace <trident-namespace> --set cloudProvider=$CP
```

`tridentctl`

En el siguiente ejemplo se instala Astra Trident y se establece `cloudProvider` la marca en Azure:

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

Identidad de nube para AKS

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

Para aprovechar la identidad de la nube en Azure, debes tener:

- Un clúster de Kubernetes puesto en marcha mediante AKS
- Identidad de carga de trabajo y emisor de oidc configurados en el clúster de Kubernetes de AKS
- Astra Trident instalado que incluye el `cloudProvider` para "Azure" especificar y `cloudIdentity` especificar la identidad de la carga de trabajo

Operador de Trident

Para instalar Astra Trident mediante el operador Trident, edite `tridentorchestrator_cr.yaml` en Establecer `cloudProvider` en "Azure" y establezca `cloudIdentity` en `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx`.

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-xxxxxxxxxxxx' *
```

Timón

Establezca los valores para los indicadores **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-
xxxxxxxxxxxx"
```

En el siguiente ejemplo se instala Astra Trident y se establece `cloudProvider` en Azure mediante la variable de entorno `$CP` y se establece el `cloudIdentity` mediante la variable de entorno `$CI`:

```
helm install trident trident-operator-100.2406.0.tgz --set
cloudProvider=$CP --set cloudIdentity=$CI
```

<code>tridentctl</code>

Establezca los valores para los indicadores **cloud provider** y **cloud identity** utilizando las siguientes variables de entorno:

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

En el siguiente ejemplo, se instala Astra Trident y se establece `cloud-provider` la marca en `$CP`, y `cloud-identity` en `$CI`:

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

Prepárese para configurar un back-end de Azure NetApp Files

Antes de configurar el back-end de Azure NetApp Files, debe asegurarse de que se cumplan los siguientes requisitos.

Requisitos previos para volúmenes NFS y SMB

Si utiliza Azure NetApp Files por primera vez o en una ubicación nueva, es necesario realizar alguna configuración inicial para configurar Azure NetApp Files y crear un volumen NFS. Consulte ["Azure: Configure Azure NetApp Files y cree un volumen NFS"](#).

Para configurar y utilizar un ["Azure NetApp Files"](#) backend, necesita lo siguiente:



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

- Un pool de capacidad. Consulte ["Microsoft: Cree un pool de capacidad para Azure NetApp Files"](#).
- Una subred delegada en Azure NetApp Files. Consulte ["Microsoft: Delege una subred en Azure NetApp Files"](#).
- `subscriptionID` Desde una suscripción a Azure con Azure NetApp Files habilitado.
- `tenantID`, `clientID` Y `clientSecret` de un ["Registro de aplicaciones"](#) en Azure Active Directory con permisos suficientes para el servicio Azure NetApp Files. El registro de aplicaciones debe usar:
 - Rol de Propietario o Contribuyente ["Predefinidos por Azure"](#).
 - A ["Rol Colaborador personalizado"](#) nivel de suscripción (`assignableScopes`) con los siguientes permisos que están limitados solo a lo que Astra Trident requiere. Después de crear el rol personalizado, ["Asigne el rol mediante el portal de Azure"](#).

```
{
  "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/MountTargets/read",

          "Microsoft.Network/virtualNetworks/read",

"Microsoft.Network/virtualNetworks/subnets/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrations/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": []
  }
]
}
}

```

- La Azure location que contiene al menos una ["subred delegada"](#). A partir de Trident 22,01, el location parámetro es un campo obligatorio en el nivel superior del archivo de configuración de backend. Los valores de ubicación especificados en los pools virtuales se ignoran.
- Para utilizar Cloud Identity, obtenga el client ID de a ["identidad gestionada asignada por el usuario"](#) y especifique ese ID en azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.

Requisitos adicionales para volúmenes SMB

Para crear un volumen de SMB, debe tener lo siguiente:

- Active Directory configurado y conectado a Azure NetApp Files. Consulte ["Microsoft: Cree y gestione conexiones de Active Directory para Azure NetApp Files"](#).
- Un clúster de Kubernetes con un nodo de controladora Linux y al menos un nodo de trabajo de Windows que ejecuta Windows Server 2022. Astra Trident admite volúmenes de SMB montados en pods que se ejecutan solo en nodos de Windows.
- Al menos un secreto de Astra Trident que contiene sus credenciales de Active Directory para que Azure NetApp Files pueda autenticarse en Active Directory. Para generar secreto smbcreds:

```

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

```

- Proxy CSI configurado como servicio de Windows. Para configurar un csi-proxy, consulte ["GitHub"](#):

[Proxy CSI](#) o ["GitHub: Proxy CSI para Windows"](#) para los nodos de Kubernetes que se ejecutan en Windows.

Opciones y ejemplos de configuración del back-end de Azure NetApp Files

Obtenga más información sobre las opciones de configuración de back-end NFS y SMB para Azure NetApp Files y revise los ejemplos de configuración.

Opciones de configuración del back-end

Astra Trident utiliza la configuración de back-end (subred, red virtual, nivel de servicio y ubicación) para crear volúmenes de Azure NetApp Files en los pools de capacidad que están disponibles en la ubicación solicitada y que coincidan con el nivel de servicio y la subred solicitados.



Astra Trident no admite pools de capacidad de calidad de servicio manual.

Los back-ends de Azure NetApp Files proporcionan estas opciones de configuración.

Parámetro	Descripción	Predeterminado
version		Siempre 1
storageDriverName	Nombre del controlador de almacenamiento	"azure-netapp-files"
backendName	Nombre personalizado o el back-end de almacenamiento	Nombre del controlador + "_" + caracteres aleatorios
subscriptionID	El ID de suscripción de la suscripción de Azure Opcional cuando se habilitan identidades administradas en un clúster de AKS.	
tenantID	ID de inquilino de un registro de aplicaciones Opcional cuando se utilizan identidades gestionadas o identidad de nube en un clúster de AKS.	
clientID	El ID de cliente de un registro de aplicaciones Opcional cuando se utilizan identidades gestionadas o identidad de nube en un clúster de AKS.	
clientSecret	El secreto de cliente de un registro de aplicaciones Opcional cuando se utilizan identidades gestionadas o identidad de nube en un clúster de AKS.	
serviceLevel	Uno de Standard Premium , o. Ultra	"" (aleatorio)

Parámetro	Descripción	Predeterminado
location	Nombre de la ubicación de Azure donde se crearán los nuevos volúmenes Opcional cuando se habiliten identidades gestionadas en un clúster de AKS.	
resourceGroups	Lista de grupos de recursos para filtrar los recursos detectados	"" (sin filtro)
netappAccounts	Lista de cuentas de NetApp para filtrar los recursos detectados	"" (sin filtro)
capacityPools	Lista de pools de capacidad para filtrar los recursos detectados	"" (sin filtro, aleatorio)
virtualNetwork	Nombre de una red virtual con una subred delegada	""
subnet	Nombre de una subred delegada en Microsoft.Netapp/volumes	""
networkFeatures	El conjunto de funciones vnet para un volumen puede ser Basic o. Standard Las funciones de red no están disponibles en todas las regiones y es posible que tengan que activarse en una suscripción. Si se especifica networkFeatures cuando la funcionalidad no está habilitada, se produce un error en el aprovisionamiento del volumen.	""
nfsMountOptions	Control preciso de las opciones de montaje NFS. Ignorada para volúmenes de SMB. Para montar volúmenes con NFS versión 4,1, incluya nfsvers=4 en la lista de opciones de montaje delimitadas por comas para elegir NFS v4,1. Las opciones de montaje establecidas en una definición de clase de almacenamiento anulan las opciones de montaje establecidas en la configuración de back-end.	"nfsvers=3"
limitVolumeSize	No se puede aprovisionar si el tamaño del volumen solicitado es superior a este valor	"" (no se aplica de forma predeterminada)

Parámetro	Descripción	Predeterminado
debugTraceFlags	Indicadores de depuración que se deben usar para la solución de problemas. Ejemplo, <code>\{"api": false, "method": true, "discovery": true\}</code> . No lo utilice a menos que esté solucionando problemas y necesite un volcado de registro detallado.	nulo
nasType	Configure la creación de volúmenes NFS o SMB. Las opciones son <code>nfs smb</code> o nulas. El valor predeterminado es nulo en volúmenes de NFS.	nfs
supportedTopologies	Representa una lista de regiones y zonas soportadas por este backend. Para obtener más información, consulte "Utilice Topología CSI" .	



Para obtener más información sobre las funciones de red, consulte ["Configure las funciones de red para un volumen de Azure NetApp Files"](#).

Permisos y recursos necesarios

Si recibes un error “No se han encontrado pools de capacidad” al crear una RVP, es probable que el registro de tu aplicación no tenga los permisos y recursos necesarios (subred, red virtual, pool de capacidad) asociados. Si la depuración está habilitada, Astra Trident registrará los recursos de Azure detectados cuando se cree el back-end. Compruebe que se está utilizando un rol adecuado.

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

Los `resourceGroups` `netappAccounts` valores, y `capacityPools` son filtros que restringen el juego de recursos detectados a los disponibles para este backend de almacenamiento y se pueden especificar en cualquier combinación. Los nombres completos siguen este formato:

Tipo	Formato
Grupo de recursos	<code><resource group></code>
Cuenta de NetApp	<code><resource group>/<netapp account></code>
Pool de capacidad	<code><resource group>/<netapp account>/<capacity pool></code>
Red virtual	<code><resource group>/<virtual network></code>
Subred	<code><resource group>/<virtual network>/<subnet></code>

Aprovisionamiento de volúmenes

Puede controlar el aprovisionamiento de volúmenes predeterminado especificando las siguientes opciones en una sección especial del archivo de configuración. Consulte [Configuraciones de ejemplo](#) para obtener más información.

Parámetro	Descripción	Predeterminado
exportRule	Reglas de exportación de volúmenes nuevos. exportRule Debe ser una lista separada por comas de cualquier combinación de direcciones IPv4 o subredes IPv4 en notación CIDR. Ignorada para volúmenes de SMB.	"0.0.0.0/0"
snapshotDir	Controla la visibilidad del directorio .snapshot	"falso"
size	El tamaño predeterminado de los volúmenes nuevos	"100 G"
unixPermissions	Los permisos unix de nuevos volúmenes (4 dígitos octal). Ignorada para volúmenes de SMB.	"" (función de vista previa, requiere incluir en la lista blanca de suscripciones)

Configuraciones de ejemplo

Los ejemplos siguientes muestran configuraciones básicas que dejan la mayoría de los parámetros en los valores predeterminados. Esta es la forma más sencilla de definir un back-end.

Configuración mínima

Ésta es la configuración mínima absoluta del back-end. Con esta configuración, Astra Trident detecta todas sus cuentas de NetApp, pools de capacidad y subredes delegadas en Azure NetApp Files en la ubicación configurada, y coloca volúmenes nuevos en uno de esos pools y subredes de forma aleatoria. Dado que `nasType` se omite, `nfs` se aplica el valor predeterminado y el back-end se aprovisionará para los volúmenes de NFS.

Esta configuración es ideal cuando solo se está empezando a usar Azure NetApp Files y probando cosas, pero en la práctica va a querer proporcionar un ámbito adicional para los volúmenes que aprovisione.

```
---
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
```

Identities administradas para AKS

Esta configuración de backend omite `subscriptionID`, `tenantID`, `clientID` y `clientSecret`, que son opcionales al utilizar identidades gestionadas.

```
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 de nube para AKS

Esta configuración de backend omite `tenantID`, `clientID`, y `clientSecret`, que son opcionales cuando se utiliza una identidad de 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 de niveles de servicio específica con filtros de pools de capacidad

Esta configuración de backend coloca los volúmenes en la ubicación de Azure `eastus` en un `Ultra` pool de capacidad. Astra Trident detecta automáticamente todas las subredes delegadas en Azure NetApp Files en esa ubicación y coloca un volumen nuevo 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
```

Configuración avanzada

Esta configuración de back-end reduce aún más el alcance de la ubicación de volúmenes en una única 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 pool virtual

Esta configuración back-end define varios pools de almacenamiento en un único archivo. Esto resulta útil cuando hay varios pools de capacidad que admiten diferentes niveles de servicio y desea crear clases de almacenamiento en Kubernetes que representan estos. Las etiquetas de pool virtual se utilizaron para diferenciar los pools 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

Astra Trident facilita el aprovisionamiento de volúmenes para cargas de trabajo según regiones y zonas de disponibilidad. El `supportedTopologies` bloque en esta configuración de 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 de 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 contengan un subconjunto de las regiones y zonas proporcionadas en un back-end, Astra Trident creará volúmenes en la región y la zona mencionadas. Para obtener más información, consulte ["Utilice 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

Las siguientes `StorageClass` definiciones hacen referencia a los pools de almacenamiento anteriores.

Ejemplo de definiciones utilizando `parameter.selector` el campo

Mediante `parameter.selector` una posible especificación para cada `StorageClass` pool virtual que se utilizará para alojar un volumen. Los aspectos definidos en el pool elegido serán el volumen.

```

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 de volúmenes SMB

Con `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"
```

Uso de 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}
```

Uso de 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.

Cree el back-end

Después de crear el archivo de configuración del back-end, ejecute el siguiente comando:

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

Si la creación del back-end falla, algo está mal con la configuración del back-end. Puede ver 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 ejecutar de nuevo el comando `create`.

NetApp Volumes para Google Cloud

Configura un back-end de Google Cloud NetApp Volumes

Ahora puede configurar Google Cloud NetApp Volumes como back-end para Astra Trident. Puede adjuntar volúmenes de NFS con un back-end de Google Cloud NetApp Volumes.

```
Google Cloud NetApp Volumes is a tech preview feature in Astra Trident 24.06.
```

Detalles del controlador de Google Cloud NetApp Volumes

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

Controlador	Protocolo	VolumeMo de	Modos de acceso compatibles	Sistemas de archivos compatibles
<code>google-cloud-netapp-volumes</code>	NFS	Sistema de archivos	RWO, ROX, RWX, RWOP	<code>nfs</code>

Prepárate para configurar un back-end de Google Cloud NetApp Volumes

Para poder configurar el back-end de Google Cloud NetApp Volumes, debe asegurarse de que se cumplan los siguientes requisitos.

Requisitos previos para volúmenes de NFS

Si utiliza Google Cloud NetApp Volumes por primera vez o en una ubicación nueva, es necesario tener alguna configuración inicial para configurar volúmenes de Google Cloud NetApp y crear un volumen NFS. Consulte ["Antes de empezar"](#).

Asegúrate de disponer de lo siguiente antes de configurar el back-end de Google Cloud NetApp Volumes:

- Una cuenta de Google Cloud configurada con el servicio NetApp Volumes de Google Cloud. Consulte ["NetApp Volumes para Google Cloud"](#).
- Número de proyecto de tu cuenta de Google Cloud. Consulte ["Identificación de proyectos"](#).
- Una cuenta de servicio de Google Cloud con el rol Administrador de volúmenes de NetApp (`netappcloudvolumes.admin`). Consulte ["Funciones y permisos de Identity and Access Management"](#).
- Archivo de claves de API para tu cuenta de GCNV. Consulte ["Autentica mediante claves de API"](#)
- Un pool de almacenamiento. Consulte ["Información general sobre pools de almacenamiento"](#).

Para obtener más información acerca de cómo configurar el acceso a volúmenes de Google Cloud NetApp, consulte ["Configure el acceso a Google Cloud NetApp Volumes"](#).

Opciones de configuración y ejemplos de back-end de Google Cloud NetApp Volumes

Obtén más información sobre las opciones de configuración del back-end de NFS para Google Cloud NetApp Volumes y revisa los ejemplos de configuración.

Opciones de configuración del back-end

Cada back-end aprovisiona volúmenes en una única región de Google Cloud. Para crear volúmenes en otras regiones, se pueden definir back-ends adicionales.

Parámetro	Descripción	Predeterminado
<code>version</code>		Siempre 1
<code>storageDriverName</code>	Nombre del controlador de almacenamiento	El valor de <code>storageDriverName</code> debe especificarse como «google-cloud-netapp-Volumes».
<code>backendName</code>	(Opcional) Nombre personalizado del back-end de almacenamiento	Nombre de controlador + "_ " + parte de la clave de API
<code>storagePools</code>	Parámetro opcional que se utiliza para especificar pools de almacenamiento para la creación del volumen.	
<code>projectNumber</code>	Número de proyecto de cuenta de Google Cloud. El valor está disponible en la página de inicio del portal de Google Cloud.	

Parámetro	Descripción	Predeterminado
location	La ubicación de Google Cloud, donde Astra Trident crea volúmenes de GCNV. Al crear clústeres de Kubernetes entre regiones, los volúmenes creados en un location se pueden usar en cargas de trabajo programadas en nodos de varias regiones de Google Cloud. El tráfico entre regiones conlleva un coste adicional.	
apiKey	La clave de la API para la cuenta de servicio de Google Cloud con <code>netappcloudvolumes.admin</code> el rol. Incluye el contenido en formato JSON del archivo de clave privada de una cuenta de servicio de Google Cloud (copiado literal en el archivo de configuración de back-end). <code>apiKey`</code> Debe incluir pares clave-valor para las siguientes claves <code>`type:,,,project_idclient_emailclient_id,,auth_uritoken_uriauth_provider_x509_cert_url,yclient_x509_cert_url.</code>	
nfsMountOptions	Control preciso de las opciones de montaje NFS.	"nfsvers=3"
limitVolumeSize	No se puede aprovisionar si el tamaño del volumen solicitado es superior a este valor.	"" (no se aplica de forma predeterminada)
serviceLevel	El nivel de servicio de un pool de almacenamiento y sus volúmenes. Los valores son <code>flex</code> , <code>standard</code> , <code>premium</code> , o <code>extreme</code> .	
network	La red de Google Cloud utilizada para los volúmenes GCNV.	
debugTraceFlags	Indicadores de depuración que se deben usar para la solución de problemas. Ejemplo, <code>{"api":false,"method":true}</code> . No lo utilice a menos que esté solucionando problemas y necesite un volcado de registro detallado.	nulo
supportedTopologies	Representa una lista de regiones y zonas soportadas por este backend. Para obtener más información, consulte "Utilice Topología CSI" . Por ejemplo: supportedTopologies: - topology.kubernetes.io/region: europe-west6 topology.kubernetes.io/zone: europe-west6-b	

Opciones de aprovisionamiento de volúmenes

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

Parámetro	Descripción	Predeterminado
exportRule	Las reglas de exportación de nuevos volúmenes. Debe ser una lista separada por comas de cualquier combinación de direcciones IPv4.	"0.0.0.0/0"
snapshotDir	Acceso al .snapshot directorio	"falso"
snapshotReserve	Porcentaje de volumen reservado para las Snapshot	" (aceptar valor por defecto de 0)
unixPermissions	Los permisos unix de nuevos volúmenes (4 dígitos octal).	""

Configuraciones de ejemplo

Los ejemplos siguientes muestran configuraciones básicas que dejan la mayoría de los parámetros en los valores predeterminados. Esta es la forma más sencilla de definir un back-end.

Configuración mínima

Ésta es la configuración mínima absoluta del back-end. Con esta configuración, Astra Trident detecta todos tus pools de almacenamiento delegados en volúmenes de Google Cloud NetApp en la ubicación configurada y coloca volúmenes nuevos en uno de esos pools de forma aleatoria. Dado que `nasType` se omite, `nfs` se aplica el valor predeterminado y el back-end se aprovisionará para los volúmenes de NFS.

Esta configuración es ideal cuando solo vas a empezar a usar Google Cloud NetApp Volumes e intentarlo, pero en la práctica probablemente necesites proporcionar un ámbito adicional para los volúmenes que aprovisionas.

[illegible]


```
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
```

```
  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 con filtro StoragePools

[illegible]

```
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 pool virtual

Esta configuración de backend define varios pools virtuales en un único archivo. Los pools virtuales se definen en `storage` la sección. Son útiles cuando tienes varios pools de almacenamiento que admiten diferentes niveles de servicio y deseas crear clases de almacenamiento en Kubernetes que las representen. Las etiquetas de pools virtuales se utilizan para diferenciar los pools. Por ejemplo, en el ejemplo que aparece a continuación `performance`, se utiliza la etiqueta `serviceLevel` el tipo para diferenciar los pools virtuales.

También puede configurar algunos valores predeterminados para que sean aplicables a todos los pools virtuales y sobrescribir los valores predeterminados de los pools virtuales individuales. En el siguiente ejemplo, `snapshotReserve` y `exportRule` sirven como valores predeterminados para todos los pools virtuales.

Para obtener más información, consulte ["Pools virtuales"](#).

[illegible]

```
znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m
znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m
znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m
znHczZsrtrHisIsAbOguSaPIKeyAZNchRAGz1zZE4jK3bl/qp8B4Kws8zX5ojY9m
XsYg6gyxy4zq70lwWgLwGa==
-----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
```

El futuro

Después de crear el archivo de configuración del back-end, 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	PHASE	STATUS	BACKEND NAME	BACKEND UUID
backend-tbc-gcnv	Bound	Success	backend-tbc-gcnv	b2fd1ff9-b234-477e-88fd-713913294f65

Si la creación del back-end falla, algo está mal con la configuración del back-end. Puede describir el backend con el `kubectl get tridentbackendconfig <backend-name>` comando o ver los logs para determinar la causa ejecutando el siguiente comando:

```
tridentctl logs
```

Después de identificar y corregir el problema con el archivo de configuración, puede suprimir el backend y ejecutar el comando create de nuevo.

Más ejemplos

Ejemplos de definición de la clase de almacenamiento

La siguiente es una definición básica `StorageClass` que hace referencia al backend anterior.

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

Ejemplo de definiciones usando el `parameter.selector` campo:

Mediante el uso `parameter.selector` se puede especificar para cada uno `StorageClass` de los "pool virtual" que se utiliza para alojar un volumen. Los aspectos definidos en el pool elegido serán el volumen.

```
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 información sobre las clases de almacenamiento, consulte ["Cree una clase de almacenamiento"](#).

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 la RVP está vinculada, 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	

Configure un back-end de Cloud Volumes Service para Google Cloud

Descubra cómo configurar Cloud Volumes Service de NetApp para Google Cloud como back-end para su instalación de Astra Trident con las configuraciones de ejemplo proporcionadas.

Detalles del controlador de Google Cloud

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

Controlador	Protocolo	VolumeMode	Modos de acceso compatibles	Sistemas de archivos compatibles
gcp-cvs	NFS	Sistema de archivos	RWO, ROX, RWX, RWOP	nfs

Obtenga más información sobre la compatibilidad de Astra Trident con Cloud Volumes Service para Google Cloud

Astra Trident puede crear volúmenes de Cloud Volumes Service en uno de estos dos [tipos de servicio](#):

- **CVS-Performance:** El tipo de servicio predeterminado Astra Trident. Este tipo de servicio optimizado para el rendimiento es 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. Puede elegir una de "tres niveles de servicio" las opciones:

- 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 usa pools de almacenamiento para admitir volúmenes de solo 1 GiB. El pool de almacenamiento puede contener hasta 50 volúmenes en los que todos los volúmenes comparten la capacidad y el rendimiento del pool. Puede elegir una de "dos niveles de servicio" las opciones:

- standardsw
- zoneredundantstandardsw

Lo que necesitará

Para configurar y utilizar el "Cloud Volumes Service para Google Cloud" backend, necesita lo siguiente:

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

Opciones de configuración del back-end

Cada back-end aprovisiona volúmenes en una única región de Google Cloud. Para crear volúmenes en otras regiones, se pueden definir back-ends adicionales.

Parámetro	Descripción	Predeterminado
version		Siempre 1
storageDriverName	Nombre del controlador de almacenamiento	"gcp-cvs"
backendName	Nombre personalizado o el back-end de almacenamiento	Nombre de controlador + "_ " + parte de la clave de API
storageClass	Parámetro opcional utilizado para especificar el tipo de servicio CVS. Se utiliza <code>software</code> para seleccionar el tipo de servicio CVS. De lo contrario, Astra Trident asume el tipo de servicio CVS-Performance (<code>hardware</code>).	
storagePools	Solo tipo de servicio CVS. Parámetro opcional que se utiliza para especificar pools de almacenamiento para la creación del volumen.	
projectNumber	Número de proyecto de cuenta de Google Cloud. El valor está disponible en la página de inicio del portal de Google Cloud.	

Parámetro	Descripción	Predeterminado
hostProjectNumber	Se requiere si se utiliza una red VPC compartida. En este escenario, <code>projectNumber</code> es el proyecto de servicio y <code>hostProjectNumber</code> es el proyecto host.	
apiRegion	Región de Google Cloud en la que Astra Trident crea volúmenes de Cloud Volumes Service. Al crear clústeres de Kubernetes entre regiones, los volúmenes creados en un <code>apiRegion</code> se pueden usar en cargas de trabajo programadas en nodos de varias regiones de Google Cloud. El tráfico entre regiones conlleva un coste adicional.	
apiKey	La clave de la API para la cuenta de servicio de Google Cloud con <code>netappcloudvolumes.admin</code> el rol. Incluye el contenido en formato JSON del archivo de clave privada de una cuenta de servicio de Google Cloud (copiado literal en el archivo de configuración de back-end).	
proxyURL	URL de proxy si se requiere servidor proxy para conectarse a la cuenta CVS. El servidor proxy puede ser un proxy HTTP o HTTPS. En el caso de un proxy HTTPS, se omite la validación de certificados para permitir el uso de certificados autofirmados en el servidor proxy. No se admiten los servidores proxy con autenticación habilitada.	
nfsMountOptions	Control preciso de las opciones de montaje NFS.	"nfsvers=3"
limitVolumeSize	No se puede aprovisionar si el tamaño del volumen solicitado es superior a este valor.	"" (no se aplica de forma predeterminada)
serviceLevel	El nivel de servicio CVS-Performance o CVS para nuevos volúmenes. Los valores de rendimiento de CVS son <code>standard premium</code> , o <code>extreme</code> . Los valores de CVS son <code>standardsw</code> o <code>zoneredundantstandardsw</code> .	El valor predeterminado de CVS-Performance es "estándar". El valor predeterminado de CVS es "standardsw".
network	Se utiliza la red de Google Cloud para Cloud Volumes Service Volumes.	"predeterminado"
debugTraceFlags	Indicadores de depuración que se deben usar para la solución de problemas. Ejemplo, <code>\{"api": false, "method": true\}</code> . No lo utilice a menos que esté solucionando problemas y necesite un volcado de registro detallado.	nulo
allowedTopologies	Para habilitar el acceso entre regiones, su definición de <code>StorageClass</code> para <code>allowedTopologies</code> debe incluir todas las regiones. Por ejemplo: <ul style="list-style-type: none"> - <code>key: topology.kubernetes.io/region</code> <code>values:</code> - <code>us-east1</code> - <code>europa-west1</code> 	

Opciones de aprovisionamiento de volúmenes

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

Parámetro	Descripción	Predeterminado
<code>exportRule</code>	Las reglas de exportación de nuevos volúmenes. Debe ser una lista separada por comas con cualquier combinación de direcciones IPv4 o subredes IPv4 en notación CIDR.	"0.0.0.0/0"
<code>snapshotDir</code>	Acceso al <code>.snapshot</code> directorio	"falso"
<code>snapshotReserve</code>	Porcentaje de volumen reservado para las Snapshot	"" (Aceptar CVS por defecto de 0)
<code>size</code>	El tamaño de los volúmenes nuevos. CVS-Performance mínimo es 100 GIB. El mínimo de CVS es 1 GIB.	El tipo de servicio CVS-Performance se establece de manera predeterminada en "100GIB". El tipo de servicio CVS no establece un valor predeterminado, pero requiere un mínimo de 1 GIB.

Ejemplos de tipo de servicio CVS-Performance

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

Ejemplo 1: Configuración mínima

Esta es la configuración de back-end mínima usando 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 de nivel de servicio

Este ejemplo muestra las opciones de configuración del back-end, incluidos 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 pool virtual

Este ejemplo utiliza `storage` para configurar pools virtuales y los `StorageClasses` que hacen referencia a ellos. Consulte [Definiciones de clases de almacenamiento](#) para ver cómo se definieron las clases de almacenamiento.

Aquí, los valores predeterminados específicos se establecen para todos los pools virtuales, que establecen `snapshotReserve` el valor en 5% y el `exportRule` en 0,0.0,0/0. Los pools virtuales se definen en la `storage` sección. Cada pool virtual individual define su propio `serviceLevel` y algunos pools sobrescriben los valores por defecto. Las etiquetas de pool virtual se utilizaron para diferenciar los pools 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 pool virtual. Con `parameters.selector`, puede especificar para cada clase de almacenamiento el pool virtual utilizado para alojar un volumen. Los aspectos definidos en el pool elegido serán el volumen.

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
```

- El primer StorageClass (`cvs-extreme-extra-protection`) se asigna al primer pool virtual. Se trata del único pool que ofrece un rendimiento extremo con una reserva Snapshot del 10%.
- The Last StorageClass (`cvs-extra-protection`) llama a cualquier pool de almacenamiento que proporciona una reserva de instantáneas del 10%. Astra Trident decide qué pool virtual se selecciona y garantiza que se cumpla el requisito de reserva de Snapshot.

Ejemplos de tipo de servicio CVS

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

Ejemplo 1: Configuración mínima

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

```
---
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 pool de almacenamiento

Esta configuración de backend de ejemplo utiliza `storagePools` para configurar un pool 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
```

El futuro

Después de crear el archivo de configuración del back-end, ejecute el siguiente comando:

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

Si la creación del back-end falla, algo está mal con la configuración del back-end. Puede ver 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 ejecutar de nuevo el comando `create`.

Configure un back-end de NetApp HCI o SolidFire

Descubre cómo crear y utilizar un back-end de Element con tu instalación de Astra Trident.

Detalles del controlador de elementos

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

```
`solidfire-san`El controlador de almacenamiento admite los modos de volumen _FILE_ y _BLOCK_. Para `Filesystem` el volumeMode, Astra Trident crea un volumen y crea un sistema de archivos. El tipo de sistema de archivos se especifica mediante StorageClass.
```

Controlador	Protocolo	Modo VolumeMode	Modos de acceso compatibles	Sistemas de archivos compatibles
solidfire-san	ISCSI	Bloque	RWO, ROX, RWX, RWOP	No hay sistema de archivos. Dispositivo de bloque RAW.
solidfire-san	ISCSI	Sistema de archivos	RWO, RWOP	xfs, ext3, , ext4

Antes de empezar

Necesitarás lo siguiente antes de crear un backend de elemento.

- Es un sistema de almacenamiento compatible que ejecuta el software Element.
- Credenciales a un usuario administrador del clúster o inquilino de HCI de NetApp/SolidFire que puede gestionar volúmenes.
- Todos sus nodos de trabajo de Kubernetes deben tener instaladas las herramientas iSCSI adecuadas. Consulte ["información de preparación del nodo de trabajo"](#).

Opciones de configuración del back-end

Consulte la siguiente tabla para ver las opciones de configuración del back-end:

Parámetro	Descripción	Predeterminado
version		Siempre 1
storageDriverName	Nombre del controlador de almacenamiento	Siempre "solidfire-san"
backendName	Nombre personalizado o el back-end de almacenamiento	Dirección IP "SolidFire_" + almacenamiento (iSCSI)
Endpoint	MVIP para el clúster de SolidFire con credenciales de inquilino	
SVIP	La dirección IP y el puerto de almacenamiento (iSCSI)	
labels	Conjunto de etiquetas con formato JSON arbitrario que se aplica en los volúmenes.	""
TenantName	Nombre de inquilino que se va a usar (creado si no se encuentra)	
InitiatorIFace	Restringir el tráfico de iSCSI a una interfaz de host específica	"predeterminado"
UseCHAP	Utilice CHAP para autenticar iSCSI. Astra Trident utiliza CHAP.	verdadero
AccessGroups	Lista de ID de grupos de acceso que se van a usar	Busca el código de un grupo de acceso denominado "trident".
Types	Especificaciones de calidad de servicio	
limitVolumeSize	Error en el aprovisionamiento si el tamaño del volumen solicitado es superior a este valor	"" (no se aplica de forma predeterminada)
debugTraceFlags	Indicadores de depuración que se deben usar para la solución de problemas. Ejemplo, {"api":false, "method":true}	nulo



No lo utilice `debugTraceFlags` a menos que esté solucionando problemas y necesite un volcado de log detallado.

Ejemplo 1: Configuración back-end para `solidfire-san` el controlador con tres tipos de volumen

Este ejemplo muestra un archivo de back-end mediante autenticación CHAP y modelado de tres tipos de volúmenes con garantías de calidad de servicio específicas. Lo más probable es que a continuación defina las clases de almacenamiento para consumir cada una de ellas mediante `IOPS` el parámetro de clase `storage`.

```

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

```

Ejemplo 2: Configuración de back-end y clase de almacenamiento para solidfire-san controlador con pools virtuales

En este ejemplo, se muestra el archivo de definición del back-end configurado con pools virtuales junto con StorageClasses que les devuelve referencia.

Astra Trident copia las etiquetas presentes en un pool de almacenamiento a la LUN de almacenamiento del entorno de administración al aprovisionar. Para mayor comodidad, los administradores de almacenamiento pueden definir etiquetas por pool 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 pools de almacenamiento, que establecen `type` AT Silver. Los pools virtuales se definen en la `storage` sección. En este ejemplo, algunos pools de almacenamiento establecen su propio tipo, y algunos pools anulan los valores predeterminados definidos 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 se refieren a los pools virtuales anteriores. En

`parameters.selector` el campo, cada `StorageClass` indica qué pools virtuales pueden utilizarse para alojar un volumen. El volumen tendrá los aspectos definidos en el pool virtual elegido.

El primer `StorageClass` (`solidfire-gold-four`) se asignará al primer pool virtual. Este es el único pool que ofrece rendimiento de oro con `Volume Type QoS` de Oro. The Last `StorageClass` (`solidfire-silver`) llama a cualquier pool de almacenamiento que ofrece un rendimiento óptimo. Astra Trident decidirá qué pool virtual se selecciona y garantizará 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"

```

Obtenga más información

- ["Los grupos de acceso de volúmenes"](#)


Controladores para SAN de ONTAP

Información general del controlador de SAN de ONTAP

Obtenga más información sobre la configuración de un entorno de administración de ONTAP con controladores SAN de ONTAP y Cloud Volumes ONTAP.

Información sobre el controlador de SAN de ONTAP

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



Si utiliza Astra Control para protección, recuperación y movilidad, lea [Compatibilidad de controladores Astra Control](#).

Controlador	Protocolo	VolumeMo de	Modos de acceso compatibles	Sistemas de archivos compatibles
ontap-san	ISCSI	Bloque	RWO, ROX, RWX, RWOP	Sin sistema de archivos; dispositivo de bloque sin procesar
ontap-san	ISCSI	Sistema de archivos	RWO, RWOP ROX y RWX no están disponibles en el modo de volumen del sistema de archivos.	xfs, ext3, , ext4
ontap-san	NVMe/TCP Consulte Consideraciones adicionales para NVMe/TCP .	Bloque	RWO, ROX, RWX, RWOP	Sin sistema de archivos; dispositivo de bloque sin procesar
ontap-san	NVMe/TCP Consulte Consideraciones adicionales para NVMe/TCP .	Sistema de archivos	RWO, RWOP ROX y RWX no están disponibles en el modo de volumen del sistema de archivos.	xfs, ext3, , ext4

Controlador	Protocolo	VolumeMo de	Modos de acceso compatibles	Sistemas de archivos compatibles
ontap-san-economy	ISCSI	Bloque	RWO, ROX, RWX, RWOP	Sin sistema de archivos; dispositivo de bloque sin procesar
ontap-san-economy	ISCSI	Sistema de archivos	RWO, RWOP ROX y RWX no están disponibles en el modo de volumen del sistema de archivos.	xfs, ext3, , ext4

Compatibilidad de controladores Astra Control

Astra Control proporciona protección fluida, recuperación ante desastres y movilidad (mover volúmenes entre clústeres de Kubernetes) para volúmenes creados con los ontap-nas controladores , ontap-nas-flexgroup y. ontap-san Consulte ["Requisitos previos de replicación de Astra Control"](#) para obtener más información.



- Utilice ontap-san-economy solo si se espera que el recuento de uso de volúmenes persistentes sea superior a ["Límites de volumen ONTAP compatibles"](#).
- Utilice ontap-nas-economy solo si se espera que el recuento de uso de volúmenes persistentes sea superior a ["Límites de volumen ONTAP compatibles"](#) y ontap-san-economy no se puede utilizar el controlador.
- No utilice ontap-nas-economy si anticipa la necesidad de protección de datos, recuperación ante desastres o movilidad.

Permisos de usuario

Astra Trident espera ejecutarse como administrador de ONTAP o SVM, normalmente usando el usuario del clúster o vsadmin un usuario de SVM, admin o bien como un usuario con un nombre distinto que tenga el mismo rol. Para puestas en marcha de Amazon FSx para NetApp ONTAP, Astra Trident espera ejecutarse como administrador de ONTAP o SVM, usando el usuario del clúster fsxadmin o vsadmin un usuario de SVM, o como un usuario con un nombre distinto que tenga el mismo rol. `fsxadmin` El usuario es un sustituto limitado para el usuario administrador del clúster.



Si se usa limitAggregateUsage el parámetro, se requieren permisos de administrador del clúster. Cuando se utiliza Amazon FSx para NetApp ONTAP con Astra Trident, limitAggregateUsage el parámetro no funcionará con vsadmin las cuentas de usuario y. fsxadmin La operación de configuración generará un error si se especifica este parámetro.

Si bien es posible crear un rol más restrictivo dentro de ONTAP que puede utilizar un controlador Trident, no lo recomendamos. La mayoría de las nuevas versiones de Trident denominan API adicionales que se tendrían que tener en cuenta, por lo que las actualizaciones son complejas y propensas a errores.

Consideraciones adicionales para NVMe/TCP

Astra Trident admite el protocolo expres de memoria no volátil (NVMe) mediante ontap-san el controlador

que se incluye:

- IPv6
- Snapshots y clones de volúmenes NVMe
- Cambiar el tamaño de un volumen NVMe
- Se importa un volumen NVMe que se creó fuera de Astra Trident para que Astra Trident gestione su ciclo de vida
- Multivía nativa de NVMe
- Cierre correcto o sin complicaciones de los K8s nodos (24,06)

Astra Trident no es compatible:

- DH-HMAC-CHAP que es compatible con NVMe de forma nativa
- Rutas múltiples del asignador de dispositivos (DM)
- Cifrado LUKS

Prepárese para configurar el back-end con los controladores SAN de ONTAP

Conozca los requisitos y las opciones de autenticación para configurar un back-end de ONTAP con controladores SAN de ONTAP.

Requisitos

Para todos los back-ends de ONTAP, Astra Trident requiere al menos un agregado asignado a la SVM.

Recuerde que también puede ejecutar más de un controlador y crear clases de almacenamiento que señalen a uno o a otro. Por ejemplo, puede configurar una `san-dev` clase que utilice `ontap-san` el controlador y una clase que utilice el `ontap-san-economy` controlador `san-default`.

Todos sus nodos de trabajo de Kubernetes deben tener instaladas las herramientas iSCSI adecuadas. Consulte "[Prepare el nodo de trabajo](#)" para obtener más información.

Autentique el backend de ONTAP

Astra Trident ofrece dos modos de autenticación de un back-end de ONTAP.

- Basado en credenciales: El nombre de usuario y la contraseña de un usuario ONTAP con los permisos requeridos. 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: Astra Trident también puede comunicarse con un clúster de ONTAP mediante un certificado instalado en el back-end. Aquí, la definición de backend debe contener valores codificados en Base64 del certificado de cliente, la clave y el certificado de CA de confianza si se utiliza (recomendado).

Puede actualizar los back-ends existentes para moverse entre métodos basados en credenciales y 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 back-end.



Si intenta proporcionar **tanto credenciales como certificados**, la creación de backend fallará y se producirá un error en el que se haya proporcionado más de un método de autenticación en el archivo de configuración.

Habilite la autenticación basada en credenciales

Astra Trident requiere las credenciales a un administrador con ámbito de SVM o clúster para comunicarse con el back-end de ONTAP. Se recomienda hacer uso de roles estándar, predefinidos como `admin` o `vsadmin`. De este modo se garantiza la compatibilidad con futuras versiones de ONTAP que puedan dar a conocer API de funciones que podrán utilizarse en futuras versiones de Astra Trident. Se puede crear y utilizar una función de inicio de sesión de seguridad personalizada con Astra Trident, pero no es recomendable.

Una definición de backend de ejemplo tendrá este aspecto:

YAML

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

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 de backend es el único lugar en el que las credenciales se almacenan en texto sin formato. Una vez creado el back-end, 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 conocimiento de las credenciales. Por tanto, es una operación de solo administración que deberá realizar el administrador de Kubernetes o almacenamiento.

Habilite la autenticación basada en certificados

Los back-ends nuevos y existentes pueden utilizar un certificado y comunicarse con el back-end de ONTAP. Se necesitan tres parámetros en la definición de backend.

- `ClientCertificate`: Valor codificado en base64 del certificado de 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, se debe proporcionar este parámetro. Esto se puede ignorar si no se utiliza ninguna CA de confianza.

Un flujo de trabajo típico implica los pasos siguientes.

Pasos

1. Genere una clave y un certificado de cliente. Al generar, establezca el nombre común (CN) en el usuario de ONTAP para autenticarse como.

```
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. Añada un certificado de CA de confianza al clúster ONTAP. Es posible que ya sea gestionado por el administrador de almacenamiento. Ignore 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 de cliente (desde el 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 admite `cert` el 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. Probar la autenticación mediante un certificado generado. Reemplace <LIF de gestión de ONTAP> y <vserver name> por la IP de LIF de gestión 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 certificados, claves y certificados 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. Cree un backend utilizando los valores obtenidos del paso anterior.

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

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

Actualice los métodos de autenticación o gire las credenciales

Puede actualizar un back-end existente para utilizar un método de autenticación diferente o para rotar sus credenciales. Esto funciona de las dos maneras: Los back-ends que utilizan nombre de usuario/contraseña se pueden actualizar para usar certificados. Los back-ends que utilizan certificados pueden actualizarse a 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 |
+-----+-----+-----+
+-----+-----+

```



Cuando gira contraseñas, el administrador de almacenamiento debe actualizar primero la contraseña del usuario en ONTAP. A esto le sigue una actualización de back-end. Al rotar certificados, se pueden agregar varios certificados al usuario. A continuación, el back-end se actualiza para usar el nuevo certificado, siguiendo el cual se puede eliminar el certificado antiguo del clúster de ONTAP.

La actualización de un back-end no interrumpe el acceso a los volúmenes que se han creado ni afecta a las conexiones de volúmenes realizadas después. Una actualización de back-end correcta indica que Astra Trident puede comunicarse con el back-end de ONTAP y gestionar futuras operaciones de volúmenes.

Autentica conexiones con CHAP bidireccional

Astra Trident puede autenticar sesiones iSCSI con CHAP bidireccional para `ontap-san` los controladores y `ontap-san-economy`. Esto requiere la activación de `useCHAP` la opción en la definición de backend. Cuando se establece en `true`, Astra Trident configura la seguridad del iniciador predeterminado de la máquina virtual de almacenamiento como CHAP bidireccional y establece el nombre de usuario y los secretos del archivo 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` parámetro es una opción booleana que puede configurarse una sola vez. De forma predeterminada, se establece en `FALSE`. Después de configurarlo en `true`, no puede establecerlo en `false`.

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

Cómo funciona

Al configurarse `useCHAP` en `true`, el administrador de almacenamiento le ordena a Astra Trident que configure CHAP en el back-end de almacenamiento. Esto incluye lo siguiente:

- Configuración de CHAP en la SVM:
 - Si el tipo de seguridad de iniciador predeterminado de la SVM es `none` (establecido de forma predeterminada) y no hay LUN preexistentes ya presentes en el volumen, Astra Trident establecerá el tipo de seguridad predeterminado en `CHAP` y procederá a configurar el iniciador CHAP y el nombre de usuario y los secretos de destino.
 - Si la SVM contiene LUN, Astra Trident no habilitará CHAP en la SVM. De este modo se garantiza que no se restrinja el acceso a las LUN que ya están presentes en la SVM.
- Configurar el iniciador de CHAP, el nombre de usuario y los secretos de destino; estas opciones deben especificarse en la configuración del back-end (como se muestra más arriba).

Después de crear el back-end, Astra Trident crea un CRD correspondiente `tridentbackend` y almacena los secretos CHAP y los nombres de usuario como secretos de Kubernetes. Todos los VP creados por Astra Trident en este back-end se montarán y se conectan mediante CHAP.

Rotar las credenciales y actualizar los back-ends

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



Al actualizar los secretos CHAP para un backend, debe utilizar `tridentctl` para actualizar el backend. No actualice las credenciales en el clúster de almacenamiento a través de la interfaz de usuario de CLI/ONTAP, ya que Astra Trident no podrá recoger 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-4378-8a33-1a472ffbeeb5c |
online | 7 |
+-----+-----+-----+-----+
+-----+-----+
```

Las conexiones existentes no se verán afectadas; seguirán activas si Astra Trident actualiza las credenciales en la SVM. Las nuevas conexiones utilizarán las credenciales actualizadas y las conexiones existentes seguirán activas. Al desconectar y volver a conectar los VP antiguos, se utilizarán las credenciales actualizadas.

Opciones y ejemplos de configuración SAN de ONTAP

Descubre cómo crear y utilizar controladores SAN de ONTAP con tu instalación de Astra Trident. Esta sección proporciona ejemplos de configuración de backend y detalles para la asignación de back-ends a StorageClasses.

Opciones de configuración del back-end

Consulte la siguiente tabla para ver las opciones de configuración del back-end:

Parámetro	Descripción	Predeterminado
version		Siempre 1
storageDrive rName	Nombre del controlador de almacenamiento	ontap-nas,,, ontap-nas- economy ontap-nas-flexgroup ontap-san, ontap-san- economy
backendName	Nombre personalizado o el back-end de almacenamiento	Nombre de controlador + «_» + LIF de datos
managementLIF	La dirección IP de un clúster o una LIF de gestión de SVM. Se puede especificar un nombre de dominio completo (FQDN). Puede configurarse para que utilice direcciones IPv6 si Astra Trident se instaló mediante la marca IPv6. Las direcciones IPv6 deben definirse entre corchetes, [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555] como . Para una conmutación de sitios MetroCluster fluida, consulte [mcc-best] .	“10.0.0.1”, “[2001:1234:abcd::fefe]”
dataLIF	Dirección IP de LIF de protocolo. No especifique para iSCSI. Astra Trident utiliza "Asignación de LUN selectiva de ONTAP" para descubrir las LIF iSCSI necesarias para establecer una sesión multivía. Se genera una advertencia si dataLIF se define explícitamente. Omitir para MetroCluster. Consulte la [mcc-best] .	Derivado del SVM
svm	Máquina virtual de almacenamiento para usar Omitir para MetroCluster. Consulte la [mcc-best] .	Derivada si se especifica una SVM managementLIF
useCHAP	Use CHAP para autenticar iSCSI para los controladores SAN de ONTAP [Boolean]. Establezca en true para que Astra Trident configure y utilice CHAP bidireccional como la autenticación predeterminada para la SVM proporcionada en el back-end. Consulte "Prepárese para configurar el back-end con los controladores SAN de ONTAP" para obtener más información.	false
chapInitiatorSecret	Secreto CHAP del iniciador. Obligatorio si useCHAP=true	""
labels	Conjunto de etiquetas con formato JSON arbitrario que se aplica en los volúmenes	""
chapTargetInitiatorSecret	Secreto CHAP del iniciador de destino. Obligatorio si useCHAP=true	""
chapUsername	Nombre de usuario entrante. Obligatorio si useCHAP=true	""
chapTargetUsername	Nombre de usuario de destino. Obligatorio si useCHAP=true	""

Parámetro	Descripción	Predeterminado
clientCertificate	Valor codificado en base64 del certificado de cliente. Se utiliza para autenticación basada en certificados	""
clientPrivateKey	Valor codificado en base64 de la clave privada de cliente. Se utiliza para autenticación basada en certificados	""
trustedCACertificate	Valor codificado en base64 del certificado de CA de confianza. Opcional. Se utiliza para autenticación basada en certificados.	""
username	El nombre de usuario necesario para comunicarse con el clúster de ONTAP. Se utiliza para autenticación basada en credenciales.	""
password	La contraseña necesaria para comunicarse con el clúster de ONTAP. Se utiliza para autenticación basada en credenciales.	""
svm	Máquina virtual de almacenamiento que usar	Derivada si se especifica una SVM managementLIF
storagePrefix	El prefijo que se utiliza cuando se aprovisionan volúmenes nuevos en la SVM. No se puede modificar más adelante. Para actualizar este parámetro, deberá crear un nuevo backend.	trident
limitAggregateUsage	Error al aprovisionar si el uso supera este porcentaje. Si estás usando un backend de Amazon FSx for NetApp ONTAP, no especifiques limitAggregateUsage. El proporcionado fsxadmin y vsadmin no contiene los permisos necesarios para recuperar el uso de agregados y limitarlo mediante Astra Trident.	"" (no se aplica de forma predeterminada)
limitVolumeSize	Error en el aprovisionamiento si el tamaño del volumen solicitado es superior a este valor. También restringe el tamaño máximo de los volúmenes que gestiona para qtrees y LUN.	"" (no se aplica de forma predeterminada)
lunsPerFlexvol	El número máximo de LUN por FlexVol debe estar comprendido entre [50 y 200]	100
debugTraceFlags	Indicadores de depuración que se deben usar para la solución de problemas. Ejemplo, {"api":false, "method":true} no lo utilice a menos que esté solucionando problemas y requiera un volcado de log detallado.	null

Parámetro	Descripción	Predeterminado
useREST	<p>Parámetro booleano para usar las API DE REST de ONTAP.</p> <p>useREST Cuando se establece en <code>true</code>, Astra Trident utilizará las API REST DE ONTAP para comunicarse con el back-end. Cuando se establezca en <code>false</code>, Astra Trident utilizará las llamadas ZAPI de ONTAP para comunicarse con el back-end. Esta función requiere ONTAP 9.11.1 o posterior. Además, el rol de inicio de sesión de ONTAP utilizado debe tener acceso a <code>ontap</code> la aplicación. Esto se cumple con los roles predefinidos <code>vsadmin</code> y <code>cluster-admin</code>. A partir de la versión de Astra Trident 24,06 y ONTAP 9.15.1 o posterior, useREST se establece en <code>true</code> de forma predeterminada; cambie useREST a <code>false</code> para utilizar llamadas de ONTAP ZAPI.</p> <p>useREST Está totalmente cualificado para NVMe/TCP.</p>	<code>true</code> Para ONTAP 9.15.1 o posterior, de lo contrario <code>false</code> .
sanType	Utilice para seleccionar <code>iscsi</code> iSCSI o <code>nvme</code> NVMe/TCP.	<code>iscsi</code> si está en blanco

Opciones de configuración de back-end para el aprovisionamiento de volúmenes

Puede controlar el aprovisionamiento predeterminado mediante estas opciones en la `defaults` sección de la configuración. Para ver un ejemplo, vea los ejemplos de configuración siguientes.

Parámetro	Descripción	Predeterminado
spaceAllocation	Asignación de espacio para las LUN	verdadero
spaceReserve	Modo de reserva de espacio; «ninguno» (fino) o «volumen» (grueso)	ninguno
snapshotPolicy	Política de Snapshot que se debe usar	ninguno
qosPolicy	Grupo de políticas de calidad de servicio que se asignará a los volúmenes creados. Elija uno de <code>qosPolicy</code> o <code>adaptiveQosPolicy</code> por pool/back-end de almacenamiento. El uso de grupos de políticas de calidad de servicio con Astra Trident requiere ONTAP 9.8 o posterior. Recomendamos utilizar un grupo de políticas QoS no compartido y garantizar que el grupo de políticas se aplique a cada componente por separado. Un grupo de políticas de calidad de servicio compartido hará que se aplique el techo para el rendimiento total de todas las cargas de trabajo.	""

Parámetro	Descripción	Predeterminado
adaptiveQosPolicy	Grupo de políticas de calidad de servicio adaptativo que permite asignar los volúmenes creados. Elija uno de qosPolicy o adaptiveQosPolicy por pool/back-end de almacenamiento	""
snapshotReserve	Porcentaje de volumen reservado para las Snapshot	«0» si snapshotPolicy no es «ninguno», de lo contrario «
splitOnClone	Divida un clon de su elemento principal al crearlo	"falso"
encryption	Habilite el cifrado de volúmenes de NetApp (NVE) en el nuevo volumen; los valores predeterminados son false. Para usar esta opción, debe tener una licencia para NVE y habilitarse en el clúster. Si NAE está habilitado en el back-end, cualquier volumen aprovisionado en Astra Trident estará habilitado para NAE. Para obtener más información, consulte: "Cómo funciona Astra Trident con NVE y NAE" .	"falso"
luksEncryption	Active el cifrado LUKS. Consulte "Usar la configuración de clave unificada de Linux (LUKS)" . El cifrado LUKS no es compatible con NVMe/TCP.	""
securityStyle	Estilo de seguridad para nuevos volúmenes	unix
tieringPolicy	Política de organización en niveles para utilizar ninguna	«Solo Snapshot» para la configuración SVM-DR anterior a ONTAP 9,5
nameTemplate	Plantilla para crear nombres de volúmenes personalizados.	""
limitVolumePoolSize	Tamaño máximo de FlexVol solicitable al usar LUN en back-end económico de ONTAP-san.	"" (no se aplica de forma predeterminada)

Ejemplos de aprovisionamiento de volúmenes

Aquí hay un ejemplo con los 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 con `ontap-san` el controlador, Astra Trident añade un 10 % de capacidad adicional al FlexVol para acomodar los metadatos del LUN. La LUN se aprovisionará con el tamaño exacto que el usuario solicite en la RVP. Astra Trident añade el 10 % a FlexVol (se muestra como tamaño disponible en ONTAP). Los usuarios obtienen ahora la cantidad de capacidad utilizable que soliciten. Este cambio también impide que las LUN se conviertan en de solo lectura a menos que se utilice completamente el espacio disponible. Esto no se aplica a `ontap-san-economy`.

Para los back-ends que definen `snapshotReserve`, Astra 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 % adicional que Astra Trident añade a FlexVol para acomodar los metadatos de las LUN. Para `snapshotReserve` = 5%, y solicitud de PVC = 5GiB, el tamaño total del volumen es 5,79GiB y el tamaño disponible es 5,5GiB. El `volume show` 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_e42ec6fe_3baa_4af6_996d_134adbbb8e6d	online	RW	5.79GB	5.50GB	0%
		_pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba	online	RW	1GB	511.8MB	0%

3 entries were displayed.

En la actualidad, el cambio de tamaño es la única manera de utilizar el nuevo cálculo para un volumen existente.

Ejemplos de configuración mínima

Los ejemplos siguientes muestran configuraciones básicas que dejan la mayoría de los parámetros en los valores predeterminados. Esta es la forma más sencilla de definir un back-end.



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

Ejemplo de SAN ONTAP

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

```
---
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 economía de SAN ONTAP

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


1. ejemplo

Puede configurar el backend para evitar tener que actualizar manualmente la definición de backend después de la conmutación y la conmutación durante ["Replicación y recuperación de SVM"](#).

Para lograr una conmutación de sitios y una conmutación de estado sin problemas, especifique la SVM con managementLIF y omita dataLIF los parámetros y. svm Por ejemplo:

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

Ejemplo de autenticación basada en certificados

En este ejemplo de configuración básica clientCertificate, clientPrivateKey y trustedCACertificate (opcional, si se utiliza CA de confianza) se rellenan backend.json y toman los valores codificados en base64 del certificado de 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 bidireccional

Estos ejemplos crean un backend con useCHAP el valor definido en true.

Ejemplo de CHAP de SAN de ONTAP

```
---
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 CHAP de economía de SAN ONTAP

```
---
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 el back-end de ONTAP. Esta es una configuración de back-end básica 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 configuración de backend con nameTemplate

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

Ejemplos de back-ends con pools virtuales

En estos archivos de definición de backend de ejemplo, se establecen valores predeterminados específicos para todos los pools de almacenamiento, como `spaceReserve` at `none`, `spaceAllocation` at `false` y `encryption` at `false`. Los pools virtuales se definen en la sección de almacenamiento.

Astra Trident establece etiquetas de aprovisionamiento en el campo «Comentarios». Los comentarios se establecen en la FlexVol. Astra Trident copia todas las etiquetas presentes en un pool virtual al volumen de almacenamiento al aprovisionar. Para mayor comodidad, los administradores de almacenamiento pueden definir etiquetas por pool virtual y agrupar volúmenes por etiqueta.

En estos ejemplos, algunos de los pools de almacenamiento establecen sus propios `spaceReserve` valores , `spaceAllocation` y `encryption`, y algunos pools sustituyen a los valores predeterminados.



```

---
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 de economía de SAN ONTAP

```
---
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'
```

Asigne los back-ends a StorageClass

Las siguientes definiciones de StorageClass hacen referencia a la [Ejemplos de back-ends con pools virtuales](#). En este `parameters.selector` campo, cada StorageClass llama la atención sobre los pools virtuales que se pueden usar para alojar un volumen. El volumen tendrá los aspectos definidos en el pool virtual elegido.

- `protection-gold`StorageClass` se asignará al primer pool virtual del ``ontap-san backend`. Este es el único pool que ofrece protección de nivel Gold.

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


- `protection-not-gold`StorageClass` se asignará al segundo y tercer pool virtual en ``ontap-san` el backend. Estos son los únicos pools que ofrecen un nivel de protección distinto del oro.

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

- `app-mysqldb`StorageClass` se asignará al tercer pool virtual en ``ontap-san-economy` backend. Este es el único pool que ofrece configuración de pool de almacenamiento para la aplicación 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"
```

- `protection-silver-creditpoints-20k`StorageClass` se asignará al segundo pool virtual en ``ontap-san` backend. Este es el único pool 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"
```

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

```

- my-test-app-sc`StorageClass se asignará al `testAPP pool virtual del ontap-san controlador 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"

```

Astra Trident decidirá qué pool virtual se selecciona y garantizará que se cumplan los requisitos de almacenamiento.

Controladores para NAS de ONTAP

Información general del controlador de NAS de ONTAP

Obtenga más información sobre la configuración de un entorno de administración de ONTAP con controladores NAS de ONTAP y Cloud Volumes ONTAP.

Información sobre el controlador de NAS de ONTAP

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



Si utiliza Astra Control para protección, recuperación y movilidad, lea [Compatibilidad de controladores Astra Control](#).

Controlador	Protocolo	VolumeMo de	Modos de acceso compatibles	Sistemas de archivos compatibles
ontap-nas	BLOQUE DE MENSAJES DEL SERVIDOR NFS	Sistema de archivos	RWO, ROX, RWX, RWOP	« », nfs, smb
ontap-nas-economy	BLOQUE DE MENSAJES DEL SERVIDOR NFS	Sistema de archivos	RWO, ROX, RWX, RWOP	« », nfs, smb
ontap-nas-flexgroup	BLOQUE DE MENSAJES DEL SERVIDOR NFS	Sistema de archivos	RWO, ROX, RWX, RWOP	« », nfs, smb

Compatibilidad de controladores Astra Control

Astra Control proporciona protección fluida, recuperación ante desastres y movilidad (mover volúmenes entre clústeres de Kubernetes) para volúmenes creados con los `ontap-nas` controladores, `ontap-nas-flexgroup` y `ontap-san`. Consulte ["Requisitos previos de replicación de Astra Control"](#) para obtener más información.



- Utilice `ontap-san-economy` solo si se espera que el recuento de uso de volúmenes persistentes sea superior a ["Límites de volumen ONTAP compatibles"](#).
- Utilice `ontap-nas-economy` solo si se espera que el recuento de uso de volúmenes persistentes sea superior a ["Límites de volumen ONTAP compatibles"](#) y `ontap-san-economy` no se puede utilizar el controlador.
- No utilice `ontap-nas-economy` si anticipa la necesidad de protección de datos, recuperación ante desastres o movilidad.

Permisos de usuario

Astra Trident espera ejecutarse como administrador de ONTAP o SVM, normalmente usando el usuario del clúster o `vsadmin` un usuario de SVM, `admin` o bien como un usuario con un nombre distinto que tenga el mismo rol.

Para puestas en marcha de Amazon FSx para NetApp ONTAP, Astra Trident espera ejecutarse como administrador de ONTAP o SVM, usando el usuario del clúster `fsxadmin` o `vsadmin` un usuario de SVM, o como un usuario con un nombre distinto que tenga el mismo rol. ``fsxadmin`` El usuario es un sustituto limitado para el usuario administrador del clúster.



Si se usa `limitAggregateUsage` el parámetro, se requieren permisos de administrador del clúster. Cuando se utiliza Amazon FSx para NetApp ONTAP con Astra Trident, `limitAggregateUsage` el parámetro no funcionará con `vsadmin` las cuentas de usuario y `fsxadmin`. La operación de configuración generará un error si se especifica este parámetro.

Si bien es posible crear un rol más restrictivo dentro de ONTAP que puede utilizar un controlador Trident, no lo recomendamos. La mayoría de las nuevas versiones de Trident denominan API adicionales que se tendrían que tener en cuenta, por lo que las actualizaciones son complejas y propensas a errores.

Prepárese para configurar un back-end con controladores NAS de ONTAP

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

Requisitos

- Para todos los back-ends de ONTAP, Astra Trident requiere al menos un agregado asignado a la SVM.
- Puede ejecutar más de un controlador y crear clases de almacenamiento que apunten a uno u otro. Por ejemplo, puede configurar una clase Gold que utilice el `ontap-nas` controlador y una clase Bronze que utilice la clase `ontap-nas-economy` One.
- Todos sus nodos de trabajo de Kubernetes deben tener instaladas las herramientas NFS adecuadas. Consulte ["aquí"](#) si desea obtener más información.
- Astra Trident admite volúmenes de SMB montados en pods que se ejecutan solo en nodos de Windows. Consulte [Prepárese para aprovisionar los volúmenes de SMB](#) para obtener más información.

Autentique el backend de ONTAP

Astra Trident ofrece dos modos de autenticación de un back-end de 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, `admin` como o `vsadmin` para garantizar la máxima compatibilidad con las versiones de ONTAP.
- Basado en certificado: Este modo requiere un certificado instalado en el back-end para que Astra Trident se comuniquen con un clúster de ONTAP. Aquí, la definición de backend debe contener valores codificados en Base64 del certificado de cliente, la clave y el certificado de CA de confianza si se utiliza (recomendado).

Puede actualizar los back-ends existentes para moverse entre métodos basados en credenciales y 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 back-end.



Si intenta proporcionar **tanto credenciales como certificados**, la creación de backend fallará y se producirá un error en el que se haya proporcionado más de un método de autenticación en el archivo de configuración.

Habilite la autenticación basada en credenciales

Astra Trident requiere las credenciales a un administrador con ámbito de SVM o clúster para comunicarse con el back-end de ONTAP. Se recomienda hacer uso de roles estándar, predefinidos como `admin` o `vsadmin`. De este modo se garantiza la compatibilidad con futuras versiones de ONTAP que puedan dar a conocer API de funciones que podrán utilizarse en futuras versiones de Astra Trident. Se puede crear y utilizar una función

de inicio de sesión de seguridad personalizada con Astra Trident, pero no es recomendable.

Una definición de backend de ejemplo tendrá este aspecto:

YAML

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

JSON

```
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password"
}
```

Tenga en cuenta que la definición de backend es el único lugar en el que las credenciales se almacenan en texto sin formato. Una vez creado el back-end, los nombres de usuario y las contraseñas se codifican con Base64 y se almacenan como secretos de Kubernetes. La creación/mejora de un backend es el único paso que requiere conocimiento de las credenciales. Por tanto, es una operación de solo administración que deberá realizar el administrador de Kubernetes o almacenamiento.

Habilite la autenticación basada en certificados

Los back-ends nuevos y existentes pueden utilizar un certificado y comunicarse con el back-end de ONTAP. Se necesitan tres parámetros en la definición de backend.

- **ClientCertificate:** Valor codificado en base64 del certificado de 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, se debe proporcionar este parámetro. Esto se puede ignorar si no se utiliza ninguna CA de confianza.

Un flujo de trabajo típico implica los pasos siguientes.

Pasos

1. Genere una clave y un certificado de cliente. Al generar, establezca el nombre común (CN) en el usuario de ONTAP para autenticarse como.

```
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. Añada un certificado de CA de confianza al clúster ONTAP. Es posible que ya sea gestionado por el administrador de almacenamiento. Ignore 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 de cliente (desde el 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 admite cert el 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. Probar la autenticación mediante un certificado generado. Reemplace <LIF de gestión de ONTAP> y <vserver name> por la IP de LIF de gestión y el nombre de SVM. Debe asegurarse de que el LIF tenga su política de servicio establecida en 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 certificados, claves y certificados 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. Cree un backend utilizando los valores obtenidos del 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...Vaaalllluuueeeee",
  "clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
  "storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
```

NAME	STORAGE DRIVER	UUID
NasBackend	ontap-nas	98e19b74-aec7-4a3d-8dcf-128e5033b214

Actualice los métodos de autenticación o gire las credenciales

Puede actualizar un back-end existente para utilizar un método de autenticación diferente o para rotar sus credenciales. Esto funciona de las dos maneras: Los back-ends que utilizan nombre de usuario/contraseña se pueden actualizar para usar certificados. Los back-ends que utilizan certificados pueden actualizarse a 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 |
+-----+-----+-----+-----+
+-----+-----+
```



Cuando gira contraseñas, el administrador de almacenamiento debe actualizar primero la contraseña del usuario en ONTAP. A esto le sigue una actualización de back-end. Al rotar certificados, se pueden agregar varios certificados al usuario. A continuación, el back-end se actualiza para usar el nuevo certificado, siguiendo el cual se puede eliminar el certificado antiguo del clúster de ONTAP.

La actualización de un back-end no interrumpe el acceso a los volúmenes que se han creado ni afecta a las conexiones de volúmenes realizadas después. Una actualización de back-end correcta indica que Astra Trident puede comunicarse con el back-end de ONTAP y gestionar futuras operaciones de volúmenes.

Gestione las políticas de exportación de NFS

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

Astra Trident ofrece dos opciones al trabajar con directivas de exportación:

- Astra 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. Astra Trident agrega automáticamente las IP de nodo que se incluyen en estos rangos a la directiva de exportación. Como alternativa, cuando no se especifican CIDR,

toda IP de unidifusión de ámbito global encontrada en los nodos se agregará a la política de exportación.

- Los administradores de almacenamiento pueden crear una normativa de exportación y añadir reglas manualmente. Astra Trident utiliza la directiva de exportación predeterminada a menos que se especifique un nombre de directiva de exportación diferente en la configuración.

Gestione de forma dinámica políticas de exportación

Astra Trident proporciona la capacidad de gestionar dinámicamente las políticas de exportación para los back-ends de ONTAP. De este modo, el administrador de almacenamiento puede especificar un espacio de direcciones permitido para las IP de nodos de trabajo, en lugar de definir reglas explícitas de forma manual. Simplifica en gran medida la gestión de políticas de exportación; las modificaciones de 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 solo a nodos de trabajo con IP en el rango especificado, lo que permite una gestión automatizada y de gran granularidad.



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 frontend y no la dirección de host IP real, por lo que el acceso se denegará cuando no se encuentre ninguna coincidencia en las reglas de exportación.

Ejemplo

Hay dos opciones de configuración que deben utilizarse. He aquí un ejemplo de definición de backend:

```
---
version: 1
storageDriverName: ontap-nas
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 usar esta función, debe asegurarse de que la unión raíz de la SVM tenga una política de exportación creada previamente con una regla de exportación que permite el bloque CIDR de nodo (como la política de exportación predeterminada). Siga siempre las prácticas recomendadas de NetApp para dedicar una SVM para Astra Trident.

A continuación se ofrece una explicación del funcionamiento de esta función utilizando el ejemplo anterior:

- `autoExportPolicy` se establece en `true`. Esto indica que Astra Trident creará una política de exportación para `svm1` la SVM y gestionará la adición y eliminación de reglas mediante `autoExportCIDRs` bloques de direcciones. Por ejemplo, un back-end con UUID `403b5326-8482-40db-96d0-d83fb3f4daec` y `autoExportPolicy` establecido en `true` crea una política de exportación llamada `trident-403b5326-8482-40db-96d0-d83fb3f4daec` en la SVM.

- `autoExportCIDRs` contiene una lista de bloques de direcciones. Este campo es opcional y se establece de forma predeterminada en `["0.0.0.0/0", "::/0"]`. Si no se define, Astra Trident agrega todas las direcciones de unidifusión de ámbito global que se encuentran en los nodos de trabajo.

En este ejemplo, `192.168.0.0/24` se proporciona el espacio de la dirección. Esto indica que las IP de nodo de Kubernetes que entran dentro de este rango de direcciones se añadirán a la política de exportación que crea Astra Trident. Cuando Astra Trident registra un nodo en el que se ejecuta, recupera las direcciones IP del nodo y las comprueba con respecto a los bloques de direcciones proporcionados en `autoExportCIDRs`. Después de filtrar las IP, Astra Trident crea reglas de política de exportación para las IP de cliente que detecta, con una regla para cada nodo que identifica.

Puede actualizar `autoExportPolicy` y `autoExportCIDRs` para los back-ends después de crearlos. Puede añadir CIDR nuevos para un back-end que se gestiona o elimina automáticamente CIDR existentes. Tenga cuidado al eliminar CIDR para asegurarse de que las conexiones existentes no se hayan caído. También puede optar por desactivar `autoExportPolicy` un backend y recurrir a una política de exportación creada manualmente. Esto requerirá definir el `exportPolicy` parámetro en la configuración de backend.

Después de que Astra Trident cree o actualice un backend, puedes comprobar el backend mediante `tridentctl` o el CRD correspondiente `tridentbackend`:

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

A medida que los nodos se añaden a un clúster de Kubernetes y se registran en la controladora Astra Trident, se actualizan las políticas de exportación de los back-ends existentes (siempre y cuando estén en el rango de direcciones especificado en el `autoExportCIDRs` back-end).

Cuando se quita un nodo, Astra Trident comprueba todos los back-ends que están en línea para quitar la regla de acceso del nodo. Al eliminar esta IP de nodo de las políticas de exportación de los back-ends gestionados, Astra Trident evita los montajes no autorizados, a menos que se vuelva a utilizar esta IP con un nodo nuevo del clúster.

Para los back-ends existentes anteriormente, la actualización del back-end con `tridentctl update backend` garantizará que Astra Trident gestione las políticas de exportación automáticamente. Esto creará una nueva política de exportación llamada después del UUID del back-end y los volúmenes que están presentes en el back-end utilizarán la política de exportación recién creada cuando se vuelvan a montar.



Si se elimina un back-end con políticas de exportación gestionadas automáticamente, se eliminará la política de exportación creada de forma dinámica. Si se vuelve a crear el back-end, se trata como un nuevo back-end 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 Astra Trident en el nodo. A continuación, Astra Trident actualizará la política de exportación para los back-ends que gestiona para reflejar este cambio de IP.

Prepárese para aprovisionar los volúmenes de SMB

Con un poco de preparación adicional, puede aprovisionar volúmenes SMB por medio `ontap-nas` de controladores.



Debe configurar tanto los protocolos NFS como SMB/CIFS en la SVM para crear `ontap-nas-economy` un volumen SMB para ONTAP en las instalaciones. Si no se configura ninguno de estos protocolos, se producirá un error en la creación del volumen de SMB.

Antes de empezar

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

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

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

- Proxy CSI configurado como servicio de Windows. Para configurar un `csi-proxy`, consulte ["GitHub: Proxy CSI"](#) o ["GitHub: Proxy CSI para Windows"](#) para los nodos de Kubernetes que se ejecutan en Windows.

Pasos

1. Para la ONTAP en las instalaciones, puede crear opcionalmente un recurso compartido de SMB, o bien Astra Trident puede crearlo para usted.



Los recursos compartidos de SMB se requieren para Amazon FSx para ONTAP.

Puede crear los recursos compartidos de administrador de SMB de dos maneras mediante el ["Consola de administración de Microsoft"](#) complemento Carpetas compartidas o mediante la CLI de ONTAP. Para crear los recursos compartidos de SMB mediante la CLI de ONTAP:

- a. Si es necesario, cree la estructura de ruta de acceso de directorio para el recurso compartido.

El `vserver cifs share create` comando comprueba 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 de 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. Compruebe que se ha creado el recurso compartido:

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



Consulte "[Cree un recurso compartido de SMB](#)" para obtener información detallada.

2. Al crear el back-end, debe configurar lo siguiente para especificar volúmenes de SMB. Para ver todas las opciones de configuración del backend de FSx para ONTAP, consulte "[Opciones y ejemplos de configuración de FSX para ONTAP](#)".

Parámetro	Descripción	Ejemplo
smbShare	Puede especificar una de las siguientes opciones: El nombre de un recurso compartido de SMB creado mediante la consola de administración de Microsoft o la interfaz de línea de comandos de ONTAP; un nombre para permitir que Astra Trident cree el recurso compartido de SMB; o bien puede dejar el parámetro en blanco para evitar el acceso de recurso compartido común a los volúmenes. Este parámetro es opcional para ONTAP en las instalaciones. Este parámetro es necesario para los back-ends de Amazon FSx para ONTAP y no puede estar en blanco.	smb-share
nasType	Debe establecerse en smb. Si es nulo, el valor por defecto es <code>nfs</code> .	smb
securityStyle	Estilo de seguridad para nuevos volúmenes. Debe establecerse en ntfs o mixed para volúmenes SMB.	ntfs O mixed para volúmenes de SMB
unixPermissions	Modo para volúmenes nuevos. Se debe dejar vacío para volúmenes SMB.	""

Opciones y ejemplos de configuración NAS de ONTAP

Descubre cómo crear y utilizar controladores NAS de ONTAP con tu instalación de Astra Trident. Esta sección proporciona ejemplos de configuración de backend y detalles para la asignación de back-ends a StorageClasses.

Opciones de configuración del back-end

Consulte la siguiente tabla para ver las opciones de configuración del back-end:

Parámetro	Descripción	Predeterminado
version		Siempre 1
storageDriveName	Nombre del controlador de almacenamiento	«ontap-nas», «ontap-nas-economy», «ontap-nas-flexgroup», «ontap-san», «ontap-san-economy»
backendName	Nombre personalizado o el back-end de almacenamiento	Nombre de controlador + «_» + LIF de datos
managementLIF	Dirección IP de un clúster o una LIF de gestión de SVM Se puede especificar un nombre de dominio completo (FQDN). Puede configurarse para que utilice direcciones IPv6 si Astra Trident se instaló mediante la marca IPv6. Las direcciones IPv6 deben definirse entre corchetes, [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555] como . Para una conmutación de sitios MetroCluster fluida, consulte [mcc-best] .	“10.0.0.1”, “[2001:1234:abcd::feffe]”
dataLIF	Dirección IP de LIF de protocolo. Recomendamos especificar dataLIF. En caso de no proporcionar esta información, Astra Trident busca las LIF de datos desde la SVM. Puede especificar un nombre de dominio completo (FQDN) para las operaciones de montaje de NFS, lo que permite crear un DNS round-robin para lograr el equilibrio de carga entre varios LIF de datos. Se puede cambiar después del ajuste inicial. Consulte . Puede configurarse para que utilice direcciones IPv6 si Astra Trident se instaló mediante la marca IPv6. Las direcciones IPv6 deben definirse entre corchetes, [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555] como . Omitir para MetroCluster. Consulte la [mcc-best] .	Dirección especificada o derivada de la SVM, si no se especifica (no recomendada)
svm	Máquina virtual de almacenamiento para usar Omitir para MetroCluster. Consulte la [mcc-best] .	Derivada si se especifica una SVM managementLIF
autoExportPolicy	Habilite la creación y actualización automática de la política de exportación [Boolean]. Mediante las autoExportPolicy opciones y autoExportCIDs, Astra Trident puede gestionar automáticamente las políticas de exportación.	falso
autoExportCIDs	Lista de CIDs para filtrar las IP del nodo de Kubernetes contra cuando autoExportPolicy se habilita. Mediante las autoExportPolicy opciones y autoExportCIDs, Astra Trident puede gestionar automáticamente las políticas de exportación.	[«0,0,0,0/0», «:/0»]

Parámetro	Descripción	Predeterminado
labels	Conjunto de etiquetas con formato JSON arbitrario que se aplica en los volúmenes	""
clientCertificate	Valor codificado en base64 del certificado de cliente. Se utiliza para autenticación basada en certificados	""
clientPrivateKey	Valor codificado en base64 de la clave privada de cliente. Se utiliza para autenticación basada en certificados	""
trustedCACertificate	Valor codificado en base64 del certificado de CA de confianza. Opcional. Se utiliza para autenticación basada en certificados	""
username	Nombre de usuario para conectarse al clúster/SVM. Se utiliza para autenticación basada en credenciales	
password	Contraseña para conectarse al clúster/SVM. Se utiliza para autenticación basada en credenciales	
storagePrefix	El prefijo que se utiliza cuando se aprovisionan volúmenes nuevos en la SVM. No se puede actualizar después de configurarlo	«trident»
limitAggregateUsage	Error al aprovisionar si el uso supera este porcentaje. No se aplica a Amazon FSX para ONTAP	"" (no se aplica de forma predeterminada)
limitVolumeSize	Error en el aprovisionamiento si el tamaño del volumen solicitado es superior a este valor. También restringe el tamaño máximo de los volúmenes que gestiona para qtrees y LUN, y la qtreesPerFlexvol opción permite personalizar el número máximo de qtrees por FlexVol.	"" (no se aplica de forma predeterminada)
lunsPerFlexvol	El número máximo de LUN por FlexVol debe estar comprendido entre [50 y 200]	«100»
debugTraceFlags	Indicadores de depuración que se deben usar para la solución de problemas. Ejemplo, {"api":false, "method":true} no lo utilice debugTraceFlags a menos que esté solucionando problemas y requiera un volcado de log detallado.	nulo
nasType	Configure la creación de volúmenes NFS o SMB. Las opciones son nfs smb o nulas. El valor predeterminado es nulo en volúmenes de NFS.	nfs

Parámetro	Descripción	Predeterminado
nfsMountOptions	Lista de opciones de montaje NFS separadas por comas. Las opciones de montaje para los volúmenes persistentes de Kubernetes se especifican normalmente en tipos de almacenamiento, pero si no se especifican opciones de montaje en una clase de almacenamiento, Astra Trident se pondrá en contacto con las opciones de montaje especificadas en el archivo de configuración del back-end de almacenamiento. Si no se especifican opciones de montaje en la clase de almacenamiento o el archivo de configuración, Astra Trident no configurará ninguna opción de montaje en un volumen persistente asociado.	""
qtreesPerFlexvol	El número máximo de qtrees por FlexVol debe estar comprendido entre [50, 300]	«200»
smbShare	Puede especificar una de las siguientes opciones: El nombre de un recurso compartido de SMB creado mediante la consola de administración de Microsoft o la interfaz de línea de comandos de ONTAP; un nombre para permitir que Astra Trident cree el recurso compartido de SMB; o bien puede dejar el parámetro en blanco para evitar el acceso de recurso compartido común a los volúmenes. Este parámetro es opcional para ONTAP en las instalaciones. Este parámetro es necesario para los back-ends de Amazon FSx para ONTAP y no puede estar en blanco.	smb-share
useREST	Parámetro booleano para usar las API DE REST de ONTAP. useREST Cuando se establece en true, Astra Trident utilizará las API REST DE ONTAP para comunicarse con el back-end. Cuando se establezca en false, Astra Trident utilizará las llamadas ZAPI de ONTAP para comunicarse con el back-end. Esta función requiere ONTAP 9.11.1 o posterior. Además, el rol de inicio de sesión de ONTAP utilizado debe tener acceso a ontap la aplicación. Esto se cumple con los roles predefinidos vsadmin y cluster-admin . A partir de la versión de Astra Trident 24,06 y ONTAP 9.15.1 o posterior, useREST se establece en true de forma predeterminada; cambie useREST a false para utilizar llamadas de ONTAP ZAPI.	true Para ONTAP 9.15.1 o posterior, de lo contrario false.
limitVolumePoolSize	Tamaño máximo de FlexVol solicitable cuando se utilizan qtrees en el back-end económico de ONTAP-nas.	"" (no se aplica de forma predeterminada)

Opciones de configuración de back-end para el aprovisionamiento de volúmenes

Puede controlar el aprovisionamiento predeterminado mediante estas opciones en la `defaults` sección de la configuración. Para ver un ejemplo, vea los ejemplos de configuración siguientes.

Parámetro	Descripción	Predeterminado
spaceAllocation	Asignación de espacio para las LUN	verdadero
spaceReserve	Modo de reserva de espacio; «ninguno» (fino) o «volumen» (grueso)	ninguno
snapshotPolicy	Política de Snapshot que se debe usar	ninguno
qosPolicy	Grupo de políticas de calidad de servicio que se asignará a los volúmenes creados. Elija uno de qosPolicy o adaptiveQosPolicy por pool/back-end de almacenamiento	""
adaptiveQosPolicy	Grupo de políticas de calidad de servicio adaptativo que permite asignar los volúmenes creados. Elija uno de qosPolicy o adaptiveQosPolicy por pool/back-end de almacenamiento. no admitido por ontap-nas-Economy.	""
snapshotReserve	Porcentaje de volumen reservado para las Snapshot	«0» si snapshotPolicy no es «ninguno», de lo contrario «
splitOnClone	Divida un clon de su elemento principal al crearlo	"falso"
encryption	Habilite el cifrado de volúmenes de NetApp (NVE) en el nuevo volumen; los valores predeterminados son false. Para usar esta opción, debe tener una licencia para NVE y habilitarse en el clúster. Si NAE está habilitado en el back-end, cualquier volumen aprovisionado en Astra Trident estará habilitado para NAE. Para obtener más información, consulte: "Cómo funciona Astra Trident con NVE y NAE" .	"falso"
tieringPolicy	Política de organización en niveles para utilizar ninguna	«Solo Snapshot» para la configuración SVM-DR anterior a ONTAP 9,5
unixPermissions	Modo para volúmenes nuevos	«777» para volúmenes NFS; vacío (no aplicable) para volúmenes SMB
snapshotDir	Controla el acceso al .snapshot directorio	"falso"
exportPolicy	Política de exportación que se va a utilizar	"predeterminado"
securityStyle	Estilo de seguridad para nuevos volúmenes. Compatibilidad y unix estilos de seguridad de NFS mixed. Compatibilidad y ntfs estilos de seguridad de SMB mixed.	El valor por defecto de NFS es unix. El valor por defecto de SMB es ntfs.
nameTemplate	Plantilla para crear nombres de volúmenes personalizados.	""



El uso de grupos de políticas de calidad de servicio con Astra Trident requiere ONTAP 9.8 o posterior. Se recomienda utilizar un grupo de políticas de calidad de servicio no compartido y asegurarse de que el grupo de políticas se aplique a cada componente individualmente. Un grupo de políticas de calidad de servicio compartido hará que se aplique el techo para el rendimiento total de todas las cargas de trabajo.

Ejemplos de aprovisionamiento de volúmenes

Aquí hay un ejemplo con los 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'
```

For `ontap-nas` and `ontap-nas-flexgroups`, Astra Trident utiliza ahora un nuevo cálculo para garantizar que el tamaño del FlexVol se ajuste correctamente con el porcentaje de reserva de SnapVault y la RVP. Cuando el usuario solicita una RVP, Astra Trident crea el FlexVol original con más espacio mediante el nuevo cálculo. Este cálculo garantiza que el usuario recibe el espacio de escritura que solicitó en el PVC y no menos espacio que el que solicitó. Antes de v21.07, cuando el usuario solicita una RVP (por ejemplo, 5GiB) con el 50 por ciento de `snapshotReserve`, solo obtiene 2,5 GiB de espacio editable. Esto se debe a que lo que el usuario solicitó es todo el volumen y `snapshotReserve` es un porcentaje de ello. Con Trident 21.07, lo que el usuario solicita es el espacio de escritura y Astra Trident define `snapshotReserve` la cantidad como el porcentaje de todo el volumen. 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 RVP = 5 GiB, el tamaño total del volumen es $2/5 = 10$ GiB y el tamaño disponible es de 5 GiB, lo que es lo que solicitó el usuario en la solicitud de RVP. El `volume show` 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 back-ends existentes de instalaciones anteriores aprovisionan volúmenes como se explicó anteriormente al actualizar Astra Trident. En el caso de los volúmenes que creó antes de actualizar, debe cambiar el tamaño de sus volúmenes para que se observe el cambio. Por ejemplo, una RVP de 2GiB GB con snapshotReserve=50 versiones anteriores dio como resultado un volumen que proporciona 1GiB GB de espacio editable. Cambiar el tamaño del volumen a 3 GiB, por ejemplo, proporciona a la aplicación 3 GiB de espacio editable en un volumen de 6 GiB.

Ejemplos de configuración mínima

Los ejemplos siguientes muestran configuraciones básicas que dejan la mayoría de los parámetros en los valores predeterminados. Esta es la forma más sencilla de definir un back-end.



Si utiliza Amazon FSX en ONTAP de NetApp con Trident, la recomendación es especificar nombres DNS para las LIF en lugar de direcciones IP.

Ejemplo de economía de NAS ONTAP

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

Ejemplo de FlexGroup NAS de ONTAP

```
---
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 de backend después de la conmutación y la conmutación durante ["Replicación y recuperación de SVM"](#).

Para lograr una conmutación de sitios y una conmutación de estado sin problemas, especifique la SVM con managementLIF y omita dataLIF los parámetros y. svm Por ejemplo:

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

Ejemplo de volúmenes de 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 de configuración de backend mínimo. `clientCertificate` `clientPrivateKey`, y `trustedCACertificate` (opcional, si utiliza CA de confianza) se rellenan `backend.json` y toman los valores codificados en base64 del certificado de 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

En este ejemplo se muestra cómo puede indicar a Astra Trident que utilice políticas de exportación dinámicas para crear y gestionar automáticamente la directiva de exportación. Esto funciona igual para `ontap-nas-economy` los controladores y `ontap-nas-flexgroup`

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

Ejemplo de direcciones IPv6

Este ejemplo se muestra managementLIF usando 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 mediante volúmenes de bloque de mensajes del servidor

`smbShare` El parámetro es necesario para FSx para ONTAP mediante volúmenes de 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 nameTemplate

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

Ejemplos de back-ends 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 pools de almacenamiento, como `spaceReserve` at `none`, `spaceAllocation` at `false` y `encryption` at `false`. Los pools virtuales se definen en la sección de almacenamiento.

Astra Trident establece etiquetas de aprovisionamiento en el campo «Comentarios». Los comentarios se establecen en FlexVol for `ontap-nas` o FlexGroup para `ontap-nas-flexgroup`. Astra Trident copia todas las etiquetas presentes en un pool virtual al volumen de almacenamiento al aprovisionar. Para mayor comodidad, los administradores de almacenamiento pueden definir etiquetas por pool virtual y agrupar volúmenes por etiqueta.

En estos ejemplos, algunos de los pools de almacenamiento establecen sus propios `spaceReserve` valores , `spaceAllocation` y `encryption`, y algunos pools sustituyen a los valores predeterminados.

Ejemplo de NAS ONTAP

```
---
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 FlexGroup NAS de ONTAP

```
---
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 de NAS ONTAP

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

```
encryption: 'false'
unixPermissions: '0775'
```

Asigne los back-ends a StorageClass

Las siguientes definiciones de StorageClass se refieren a [Ejemplos de back-ends con pools virtuales](#). En este `parameters.selector` campo, cada StorageClass llama la atención sobre los pools virtuales que se pueden usar para alojar un volumen. El volumen tendrá los aspectos definidos en el pool virtual elegido.

- `protection-gold`StorageClass` se asignará al primer y segundo pool virtual del ``ontap-nas-flexgroup` backend. Estos son los únicos pools que ofrecen protección de nivel Gold.

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

- `protection-not-gold`StorageClass` se asignará al tercer y cuarto pool virtual del ``ontap-nas-flexgroup` backend. Estos son los únicos pools que ofrecen un nivel de protección distinto al Gold.

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

- `app-mysqldb`StorageClass` se asignará al cuarto pool virtual del ``ontap-nas` backend. Este es el único pool que ofrece configuración de pool de almacenamiento para la aplicación 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"

```

- `protection-silver-creditpoints-20k` StorageClass se asignará al tercer pool virtual del `ontap-nas-flexgroup` backend. Este es el único pool 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"

```

- `creditpoints-5k` StorageClass se asignará al tercer pool virtual del `ontap-nas` backend y al segundo pool virtual del backend `ontap-nas-economy`. Estas son las únicas ofertas de grupo 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"

```

Astra Trident decidirá qué pool virtual se selecciona y garantizará que se cumplan los requisitos de almacenamiento.

``dataLIF`` Actualice tras la configuración inicial

Puede cambiar la LIF de datos tras la configuración inicial ejecutando el siguiente comando para proporcionar el nuevo archivo JSON back-end con LIF de datos actualizadas.

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



Si los RVP están conectados a uno o varios pods, deben recuperar todos los pods correspondientes y, a continuación, traerlos para que surta efecto el nuevo LIF de datos.

Amazon FSX para ONTAP de NetApp

Utilice Astra Trident con Amazon FSX para ONTAP de NetApp

"[Amazon FSX para ONTAP de NetApp](#)" Es un servicio de AWS totalmente gestionado que permite a los clientes iniciar y ejecutar sistemas de archivos con tecnología del sistema operativo de almacenamiento NetApp ONTAP. FSX para ONTAP le permite aprovechar las funciones, el rendimiento y las funcionalidades administrativas de NetApp con las que ya está familiarizado, a la vez que aprovecha la simplicidad, la agilidad, la seguridad y la escalabilidad de almacenar datos en AWS. FSX para ONTAP es compatible con las funciones del sistema de archivos ONTAP y las API de administración.

Puede integrar su sistema de archivos Amazon FSX para ONTAP de NetApp con Astra 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.

Un sistema de archivos es el recurso principal de Amazon FSX, similar a un clúster de ONTAP en las instalaciones. En cada SVM, se pueden crear uno o varios volúmenes, que son contenedores de datos que almacenan los archivos y las carpetas en el sistema de archivos. Con Amazon FSX para ONTAP de NetApp, Data ONTAP se proporcionará como un sistema de archivos gestionado en el cloud. El nuevo tipo de sistema de archivos se llama **ONTAP** de NetApp.

Al utilizar Astra Trident con Amazon FSX para ONTAP de NetApp, puede garantizar que los clústeres de Kubernetes que se ejecutan en Amazon Elastic Kubernetes Service (EKS) pueden aprovisionar volúmenes persistentes de bloques y archivos respaldados por ONTAP.

Requisitos

Además "[Requisitos de Astra Trident](#)" de , para integrar FSx para ONTAP con Astra Trident, necesita:

- Un clúster de Amazon EKS existente o un clúster de Kubernetes autogestionado `kubect1` con instalado.
- Un sistema de archivos Amazon FSx para NetApp ONTAP y una máquina virtual de almacenamiento (SVM) a la que se puede acceder desde los nodos de trabajo del clúster.
- Nodos de trabajador preparados para "[NFS o iSCSI](#)".



Asegúrese de seguir los pasos de preparación de nodos necesarios para Amazon Linux y Ubuntu "[Imágenes de máquina de Amazon](#)" (AMI) según el tipo de AMI de EKS.

Consideraciones

- Volúmenes SMB:
 - Los volúmenes SMB solo se admiten mediante `ontap-nas` el controlador.
 - Los volúmenes SMB no son compatibles con el complemento Astra Trident EKS.
 - Astra Trident admite volúmenes de SMB montados en pods que se ejecutan solo en nodos de Windows. Consulte ["Prepárese para aprovisionar los volúmenes de SMB"](#) para obtener más información.
- Antes de Astra Trident 24,02, Trident no podía eliminar los volúmenes creados en el sistema de archivos Amazon FSx que tienen habilitados backups automáticos. Para evitar este problema en Astra Trident 24,02 o posterior, especifique `fsxFilesystemID`, `aws`, `apiKey` `aws` `apiRegion` y `aws` `secretKey` en el archivo de configuración del back-end de AWS FSx para ONTAP.



Si especifica un rol de IAM en Astra Trident, puede omitir la especificación explícita de los `apiRegion` campos, `apiKey` y `secretKey` en Astra Trident. Para obtener más información, consulte ["Opciones y ejemplos de configuración de FSX para ONTAP"](#).

Autenticación

Astra Trident ofrece dos modos de autenticación.

- Basado en credenciales (recomendado): Almacena las credenciales de forma segura en AWS Secrets Manager. Puede usar el `fsxadmin` usuario del sistema de archivos o del `vsadmin` usuario configurado para la SVM.



Astra Trident espera ejecutarse como `vsadmin` usuario de SVM o como usuario con un nombre distinto que tenga el mismo rol. Amazon FSx para NetApp ONTAP tiene un `fsxadmin` usuario que sustituye de forma limitada al usuario del clúster de ONTAP `admin`. Recomendamos encarecidamente utilizar `vsadmin` con Astra Trident.

- Basado en certificados: Astra Trident se comunicará con la SVM en su sistema de archivos FSX mediante un certificado instalado en la SVM.

Para obtener más información sobre cómo habilitar la autenticación, consulte la autenticación del tipo de controlador:

- ["Autenticación NAS ONTAP"](#)
- ["Autenticación SAN ONTAP"](#)

Obtenga más información

- ["Documentación de Amazon FSX para ONTAP de NetApp"](#)
- ["Publicación del blog en Amazon FSX para ONTAP de NetApp"](#)

Cree un rol de IAM y AWS Secret

Puede configurar los pods de Kubernetes para acceder a los recursos de AWS mediante la autenticación como un rol de AWS IAM en lugar de proporcionar credenciales de AWS explícitas.



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

Crear secreto de AWS Secret Manager

En este ejemplo, se crea un secreto de AWS Secret Manager para almacenar las credenciales de CSI de Astra Trident:

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

Crear política de IAM

Los siguientes ejemplos crean una política de IAM mediante 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 Secret manager"
```

Policy JSON archivo:


```

policy.json:
{
  "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"
}

```

Rol Crear e IAM para la cuenta de servicio

En el siguiente ejemplo, se crea 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 arn:aws:iam::aws:policy/service-
role/AmazonFSxNCSIDriverPolicy --approve

```

Instale Astra Trident

Astra Trident optimiza la gestión del almacenamiento de Amazon FSx para NetApp ONTAP en Kubernetes para que sus desarrolladores y administradores se centren en la puesta en marcha de aplicaciones.

Puedes instalar Astra Trident mediante uno de los siguientes métodos:

- Timón

- Complemento EKS

If you want to make use of the snapshot functionality, install the CSI snapshot controller add-on. Refer to <https://docs.aws.amazon.com/eks/latest/userguide/csi-snapshot-controller.html>.

Instala Astra Trident mediante helm

1. Descargue el paquete de instalador de Astra Trident

El paquete de instalación de Astra Trident incluye todo lo necesario para poner en marcha al operador de Trident e instalar Astra Trident. Descargue y extraiga la última versión del instalador de Astra Trident de la sección Activos de GitHub.

```
wget https://github.com/NetApp/trident/releases/download/v24.06.0/trident-installer-24.06.0.tar.gz
tar -xf trident-installer-24.06.0.tar.gz
cd trident-installer
```

2. Establezca los valores para los indicadores **cloud provider** y **cloud identity** utilizando las siguientes variables de entorno:

```
export CP="AWS"
export CI="'eks.amazonaws.com/role-arn:
arn:aws:iam::<accountID>:role/<AmazonEKS_FSxN_CSI_DriverRole>'"
```

En el siguiente ejemplo, se instala Astra Trident y se establece `cloud-provider` la marca en `$CP`, y `cloud-identity` en `$CI`:

```
helm install trident trident-operator-100.2406.0.tgz --set
cloudProvider=$CP --set cloudIdentity=$CI --namespace trident
```

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

```
helm list -n trident
```

NAME	STATUS	CHART	NAMESPACE	REVISION	UPDATED	APP VERSION
trident-operator			trident	1	2024-10-14 14:31:22.463122	
+0300 IDT	deployed		trident-operator-100.2406.1			24.06.1

Instala Astra Trident a través del complemento EKS

El complemento Astra Trident EKS incluye las últimas revisiones de seguridad, correcciones de errores y AWS lo valida para que funcione con Amazon EKS. El complemento EKS le permite garantizar de forma constante que sus clústeres de Amazon EKS sean seguros y estables y reducir la cantidad de trabajo que necesita para instalar, configurar y actualizar complementos.

Requisitos previos

Asegúrate de disponer de lo siguiente antes de configurar el complemento Astra Trident para AWS EKS:

- Una cuenta de clúster de Amazon EKS con suscripción complementaria
- Permisos de AWS para AWS Marketplace:
"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 para NetApp ONTAP existente

Habilita el complemento Astra Trident para AWS

Clúster EKS

Los siguientes comandos de ejemplo instalan el complemento Astra Trident EKS:

```
eksctl create addon --cluster clusterName --name netapp_trident-operator  
--version v24.6.1-eksbuild  
eksctl create addon --cluster clusterName --name netapp_trident-operator  
--version v24.6.1-eksbuild.1 (con una versión dedicada)
```



Al configurar el parámetro opcional `cloudIdentity`, asegúrese de especificar `cloudProvider` al instalar Trident mediante el complemento EKS.

Consola de gestión

1. Abra la consola de Amazon EKS en <https://console.aws.amazon.com/eks/home#/clusters>.
2. En el panel de navegación izquierdo, haga clic en **Clusters**.
3. Haga clic en el nombre del cluster para el que desea configurar el complemento CSI de NetApp Trident.
4. Haga clic en **Complementos** y luego haga clic en **Obtener más complementos**.
5. En la página **Select add-ons**, haz lo siguiente:
 - a. En la sección eks-addons de AWS Marketplace, selecciona la casilla de verificación **Astra Trident by NetApp**.
 - b. Haga clic en **Siguiente**.
6. En la página de configuración **Configure Selected add-ons**, haga lo siguiente:
 - a. Seleccione la **Versión** que desea usar.
 - b. Para **Seleccione el rol de IAM**, déjelo en **No establecido**.
 - c. Expanda la configuración **Opcional**, siga el esquema de configuración **Add-on** y establezca el parámetro `configurationValues` en la sección **Valores de configuración** en el rol-arn que creó en el paso anterior (el valor debe tener el siguiente formato `eks.amazonaws.com/role-arn:arn:aws:iam::464262061435:role/AmazonEKS_FSXN_CSI_DriverRole:`). Si selecciona **Sustituir** para el método de resolución de conflictos, una o más de las configuraciones del complemento existente se pueden sobrescribir con la configuración del complemento Amazon EKS. Si no habilita esta opción y existe un conflicto con la configuración existente, se producirá un error en la operación. Puede utilizar el mensaje de error resultante para solucionar el conflicto. Antes de seleccionar esta opción, asegúrese de que el complemento de Amazon EKS no gestiona la configuración que necesita para autogestionar.



Al configurar el parámetro opcional `cloudIdentity`, asegúrese de especificar `cloudProvider` al instalar Trident mediante el complemento EKS.

7. Elija **Siguiente**.
8. En la página **Revisar y agregar**, selecciona **Crear**.

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

CLI DE AWS

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

```
add-on.json
{
    "clusterName": "<eks-cluster>",
    "addonName": "netapp_trident-operator",
    "addonVersion": "v24.6.1-eksbuild.1",
    "serviceAccountRoleArn": "arn:aws:iam::123456:role/astratrident-
role",
    "configurationValues": "{\"cloudIdentity\":
    \"'eks.amazonaws.com/role-arn: arn:aws:iam::123456:role/astratrident-
role'\",
    \"cloudProvider\": \"AWS\"}"
}
```



Al configurar el parámetro opcional `cloudIdentity`, asegúrese de especificar `AWS` como `cloudProvider` al instalar Trident mediante el complemento EKS.

2. Instala el complemento de Astra Trident EKS»

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

Actualiza el complemento EKS de Astra Trident

Clúster EKS

- Compruebe la versión actual de su complemento FSxN Trident CSI. Sustituya `my-cluster` por el nombre del 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	v24.6.1-eksbuild.1	ACTIVE	0
{ "cloudIdentity": "'eks.amazonaws.com/role-arn:arn:aws:iam::139763910815:role/AmazonEKS_FSXN_CSI_DriverRole'" }			

- Actualice el complemento a la versión devuelta bajo ACTUALIZACIÓN DISPONIBLE en la salida del paso anterior.

```
eksctl update addon --name netapp_trident-operator --version v24.6.1-eksbuild.1 --cluster my-cluster --force
```

Si elimina la `--force` opción y cualquiera de las configuraciones del complemento de Amazon EKS entra en conflicto con la configuración existente, la actualización del complemento de Amazon EKS falla; recibirá un mensaje de error que le ayudará a resolver el conflicto. Antes de especificar esta opción, asegúrese de que el complemento de Amazon EKS no gestiona la configuración que debe administrar, ya que dicha configuración se sobrescribe con esta opción. Para obtener más información acerca de otras opciones para esta configuración, consulte ["Complementos"](#). Para obtener más información sobre la gestión de campos de Amazon EKS Kubernetes, consulte ["Gestión del campo de Kubernetes"](#).

Consola de gestión

1. Abra la consola de Amazon EKS <https://console.aws.amazon.com/eks/home#/clusters>.
2. En el panel de navegación izquierdo, haga clic en **Clusters**.
3. Haga clic en el nombre del cluster para el que desea actualizar el complemento CSI de NetApp Trident.
4. Haga clic en la pestaña **Add-ons**.
5. Haz clic en **Astra Trident by NetApp** y luego haz clic en **Editar**.
6. En la página **Configure Astra Trident by NetApp**, haga lo siguiente:
 - a. Seleccione la **Versión** que desea usar.
 - b. (Opcional) Puede ampliar la **Configuración opcional** y modificarla según sea necesario.
 - c. Haga clic en **Guardar cambios**.

CLI DE AWS

El siguiente ejemplo actualiza el complemento EKS:

```
aws eks update-addon --cluster-name my-cluster netapp_trident-operator vpc-cni  
--addon-version v24.6.1-eksbuild.1 \  
--service-account-role-arn arn:aws:iam::111122223333:role/role-name
```

```
--configuration-values '{}' --resolve-conflicts --preserve
```

Desinstale/quite el complemento de Astra Trident EKS

Tienes dos opciones para eliminar un complemento de Amazon EKS:

- **Preserve el software complementario en su clúster** – Esta opción elimina la administración de Amazon EKS de cualquier configuración. También elimina la posibilidad de que Amazon EKS le notifique las actualizaciones y actualice automáticamente el complemento de Amazon EKS después de iniciar una actualización. Sin embargo, conserva el software complementario en el clúster. Esta opción convierte el complemento en una instalación autogestionada, en lugar de un complemento de Amazon EKS. Con esta opción, no se produce tiempo de inactividad en el complemento. Conserve `--preserve` la opción en el comando para conservar el complemento.
- *** Elimine el software complementario completamente de su clúster ***: Le recomendamos que elimine el complemento Amazon EKS de su clúster solo si no hay recursos en su clúster que dependan de él. Elimine `--preserve` la opción del `delete` comando para eliminar el complemento.



Si el complemento tiene una cuenta de IAM asociada, la cuenta de IAM no se elimina.

Clúster EKS

El siguiente comando desinstala el complemento EKS de Astra Trident:

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

Consola de gestión

1. Abra la consola de Amazon EKS en <https://console.aws.amazon.com/eks/home#/clusters>.
2. En el panel de navegación izquierdo, haga clic en **Clusters**.
3. Haga clic en el nombre del cluster del que desea quitar el complemento CSI de NetApp Trident.
4. Haz clic en la pestaña **Complementos** y luego haz clic en **Astra Trident by NetApp**.*
5. Haga clic en **Quitar**.
6. En el cuadro de diálogo **Remove netapp_trident-operator confirmation**, haga lo siguiente:
 - a. Si desea que Amazon EKS deje de administrar la configuración del complemento, seleccione **Conservar en clúster**. Haga esto si desea conservar el software complementario en su clúster para que pueda gestionar todos los ajustes del complemento por su cuenta.
 - b. Introduzca **netapp_trident-operator**.
 - c. Haga clic en **Quitar**.

CLI DE AWS

Reemplace `my-cluster` por el nombre del clúster y, a continuación, ejecute el siguiente comando.

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

Configure el backend de almacenamiento

Integración de controladores ONTAP SAN y NAS

Puede crear un archivo backend con las credenciales de SVM (nombre de usuario y contraseña) almacenadas en AWS Secret Manager, como se muestra en este ejemplo:

YAML

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
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:xxxxxxx:secret:secret-
name"
    type: awsarn
```

JSON

```
{
  "apiVersion": "trident.netapp.io/v1",
  "kind": "TridentBackendConfig",
  "metadata": {
    "name": "backend-tbc-ontap-nas"
  },
  "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:xxxxxxx:secret:secret-
name",
      "type": "awsarn"
    }
  }
}
```

Para obtener más información sobre la creación de back-ends, consulte estas páginas:

- ["Configurar un back-end con controladores NAS de ONTAP"](#)
- ["Configurar un back-end con controladores SAN de ONTAP"](#)

FSX para ONTAP detalles del controlador

Puede integrar Astra Trident con Amazon FSx para ONTAP de NetApp mediante los siguientes controladores:

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

Para obtener información detallada sobre el conductor, consulte ["Controladores de NAS"](#) y ["Controladores de SAN"](#).

Configuraciones de ejemplo

Configuración para AWS FSx para ONTAP con administrador secreto

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

Configuración de la clase de almacenamiento para volúmenes SMB

Con `nasType`, `node-stage-secret-name` y `node-stage-secret-namespace`, puede especificar un volumen SMB y proporcionar las credenciales de Active Directory necesarias. Los volúmenes SMB solo se admiten mediante `ontap-nas` el controlador.

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

Configuración avanzada de backend y ejemplos

Consulte la siguiente tabla para ver las opciones de configuración del back-end:

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 back-end de almacenamiento	Nombre del conductor + “_” + dataLIF

Parámetro	Descripción	Ejemplo
managementLIF	Dirección IP de un clúster o una LIF de gestión de SVM Se puede especificar un nombre de dominio completo (FQDN). Puede configurarse para que utilice direcciones IPv6 si Astra Trident se instaló mediante la marca IPv6. Las direcciones IPv6 deben definirse entre corchetes, como [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]. Si proporciona el fsxFilesystemID en aws el campo, no necesita proporcionar el managementLIF porque Astra Trident recupera la información de la SVM managementLIF de AWS. Por lo tanto, debe proporcionar credenciales para un usuario en la SVM (por ejemplo: Vsadmin) y el usuario debe tener vsadmin el rol.	"10.0.0.1", "[2001:1234:abcd::fefe]"
dataLIF	Dirección IP de LIF de protocolo. Controladores NAS de ONTAP: Recomendamos especificar dataLIF. En caso de no proporcionar esta información, Astra Trident busca las LIF de datos desde la SVM. Puede especificar un nombre de dominio completo (FQDN) para las operaciones de montaje de NFS, lo que permite crear un DNS round-robin para lograr el equilibrio de carga entre varios LIF de datos. Se puede cambiar después del ajuste inicial. Consulte . Controladores SAN ONTAP: No se especifica para iSCSI. Astra Trident utiliza la asignación selectiva de LUN de ONTAP para descubrir los LIF iSCSI necesarios para establecer una sesión de varias rutas. Se genera una advertencia si dataLIF se define explícitamente. Puede configurarse para que utilice direcciones IPv6 si Astra Trident se instaló mediante la marca IPv6. Las direcciones IPv6 deben definirse entre corchetes, como [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	

Parámetro	Descripción	Ejemplo
autoExportPolicy	Habilite la creación y actualización automática de la política de exportación [Boolean]. Mediante las autoExportPolicy opciones y autoExportCIDRs, Astra Trident puede gestionar automáticamente las políticas de exportación.	false
autoExportCIDRs	Lista de CIDRs para filtrar las IP del nodo de Kubernetes contra cuando autoExportPolicy se habilita. Mediante las autoExportPolicy opciones y autoExportCIDRs, Astra Trident puede gestionar automáticamente las políticas de exportación.	"["0.0.0.0/0", "":"/0"]"
labels	Conjunto de etiquetas con formato JSON arbitrario que se aplica en los volúmenes	""
clientCertificate	Valor codificado en base64 del certificado de cliente. Se utiliza para autenticación basada en certificados	""
clientPrivateKey	Valor codificado en base64 de la clave privada de cliente. Se utiliza para autenticación basada en certificados	""
trustedCACertificate	Valor codificado en base64 del certificado de CA de confianza. Opcional. Se utiliza para autenticación basada en certificados.	""
username	El nombre de usuario para conectarse al clúster o SVM. Se utiliza para autenticación basada en credenciales. Por ejemplo, vsadmin.	
password	La contraseña para conectarse al clúster o SVM. Se utiliza para autenticación basada en credenciales.	
svm	Máquina virtual de almacenamiento que usar	Derivado si se especifica una LIF de gestión de SVM.

Parámetro	Descripción	Ejemplo
storagePrefix	El prefijo que se utiliza cuando se aprovisionan volúmenes nuevos en la SVM. No se puede modificar una vez creada. Para actualizar este parámetro, deberá crear un nuevo backend.	trident
limitAggregateUsage	No especifique para Amazon FSx para NetApp ONTAP. El proporcionado fsxadmin y vsadmin no contiene los permisos necesarios para recuperar el uso de agregados y limitarlo mediante Astra Trident.	No utilizar.
limitVolumeSize	Error en el aprovisionamiento si el tamaño del volumen solicitado es superior a este valor. También restringe el tamaño máximo de los volúmenes que gestiona para qtrees y LUN, y la qtreesPerFlexvol opción permite personalizar el número máximo de qtrees por FlexVol.	"" (no se aplica de forma predeterminada)
lunsPerFlexvol	El número máximo de LUN por FlexVol debe estar comprendido entre [50 y 200]. Solo SAN.	«100»
debugTraceFlags	Indicadores de depuración que se deben usar para la solución de problemas. Por ejemplo, {"api":false, "method":true} no se utiliza debugTraceFlags a menos que esté solucionando problemas y requiera un volcado de log detallado.	nulo

Parámetro	Descripción	Ejemplo
nfsMountOptions	Lista de opciones de montaje NFS separadas por comas. Las opciones de montaje para los volúmenes persistentes de Kubernetes se especifican normalmente en tipos de almacenamiento, pero si no se especifican opciones de montaje en una clase de almacenamiento, Astra Trident se pondrá en contacto con las opciones de montaje especificadas en el archivo de configuración del back-end de almacenamiento. Si no se especifican opciones de montaje en la clase de almacenamiento o el archivo de configuración, Astra Trident no configurará ninguna opción de montaje en un volumen persistente asociado.	""
nasType	Configure la creación de volúmenes NFS o SMB. Las opciones son <code>nfs smb</code> , o nulas. Debe establecerse en <code>smb</code> para volúmenes SMB. El valor predeterminado es nulo en volúmenes de NFS.	nfs
qtreesPerFlexvol	El número máximo de qtrees por FlexVol debe estar comprendido entre [50, 300]	"200"
smbShare	Puede especificar una de las siguientes opciones: El nombre de un recurso compartido de SMB creado con la consola de administración de Microsoft o la interfaz de línea de comandos de ONTAP, o bien un nombre para permitir que Astra Trident cree el recurso compartido de SMB. Este parámetro es obligatorio para los back-ends de Amazon FSx para ONTAP.	smb-share

Parámetro	Descripción	Ejemplo
useREST	<p>Parámetro booleano para usar las API DE REST de ONTAP. Vista previa tecnológica</p> <p>useREST se proporciona como vista previa tecnológica que se recomienda para entornos de prueba y no para cargas de trabajo de producción. Cuando se configura en <code>true</code>, Astra Trident utilizará las API REST DE ONTAP para comunicarse con el back-end. Esta función requiere ONTAP 9.11.1 o posterior. Además, el rol de inicio de sesión de ONTAP utilizado debe tener acceso a <code>ontap</code> la aplicación. Esto se cumple con los roles predefinidos <code>vsadmin</code> y <code>cluster-admin</code>.</p>	false
aws	<p>Puede especificar lo siguiente en el archivo de configuración de AWS FSx para ONTAP: -</p> <p><code>fsxFileSystemID</code>: Especifique el ID del sistema de archivos AWS FSx. <code>apiRegion</code>: AWS API nombre de región. <code>apiKey</code>: AWS API key. - <code>secretKey</code>: AWS clave secreta.</p>	<pre>"" "" ""</pre>
credentials	<p>Especifique las credenciales de FSx SVM que se van a almacenar en AWS Secret Manager. <code>name</code>: Nombre de recurso de Amazon (ARN) del secreto, que contiene las credenciales de SVM. <code>type</code>: Establecido en <code>awsarn</code>. Consulte "Cree un secreto de AWS Secrets Manager" si desea obtener más información.</p>	

Opciones de configuración de back-end para el aprovisionamiento de volúmenes

Puede controlar el aprovisionamiento predeterminado mediante estas opciones en la `defaults` sección de la configuración. Para ver un ejemplo, vea los ejemplos de configuración siguientes.

Parámetro	Descripción	Predeterminado
spaceAllocation	Asignación de espacio para las LUN	true
spaceReserve	Modo de reserva de espacio; "none" (thin) o "VOLUME" (grueso)	none

Parámetro	Descripción	Predeterminado
snapshotPolicy	Política de Snapshot que se debe usar	none
qosPolicy	Grupo de políticas de calidad de servicio que se asignará a los volúmenes creados. Elija uno de qosPolicy o adaptiveQosPolicy por pool de almacenamiento o back-end. El uso de grupos de políticas de calidad de servicio con Astra Trident requiere ONTAP 9.8 o posterior. Recomendamos utilizar un grupo de políticas QoS no compartido y garantizar que el grupo de políticas se aplique a cada componente por separado. Un grupo de políticas de calidad de servicio compartido hará que se aplique el techo para el rendimiento total de todas las cargas de trabajo.	""
adaptiveQosPolicy	Grupo de políticas de calidad de servicio adaptativo que permite asignar los volúmenes creados. Elija uno de qosPolicy o adaptiveQosPolicy por pool de almacenamiento o back-end. no admitido por ontap-nas-Economy.	""
snapshotReserve	Porcentaje del volumen reservado para instantáneas "0"	snapshotPolicy`Si es `none, else ""
splitOnClone	Divida un clon de su elemento principal al crearlo	false
encryption	Habilite el cifrado de volúmenes de NetApp (NVE) en el nuevo volumen; los valores predeterminados son false. Para usar esta opción, debe tener una licencia para NVE y habilitarse en el clúster. Si NAE está habilitado en el back-end, cualquier volumen provisionado en Astra Trident estará habilitado para NAE. Para obtener más información, consulte: "Cómo funciona Astra Trident con NVE y NAE" .	false
luksEncryption	Active el cifrado LUKS. Consulte "Usar la configuración de clave unificada de Linux (LUKS)" . Solo SAN.	""

Parámetro	Descripción	Predeterminado
tieringPolicy	Política de organización en niveles para utilizar <code>none</code>	<code>snapshot-only</code> Para la configuración previa a ONTAP 9.5 SVM-DR
unixPermissions	Modo para volúmenes nuevos. Dejar vacío para volúmenes SMB.	""
securityStyle	Estilo de seguridad para nuevos volúmenes. Compatibilidad y <code>unix</code> estilos de seguridad de NFS <code>mixed</code> . Compatibilidad y <code>ntfs</code> estilos de seguridad de SMB <code>mixed</code> .	El valor por defecto de NFS es <code>unix</code> . El valor por defecto de SMB es <code>ntfs</code> .

Prepárese para aprovisionar los volúmenes de SMB

Puede aprovisionar volúmenes SMB con `ontap-nas` el controlador. Antes de completar [Integración de controladores ONTAP SAN y NAS](#) los siguientes pasos.

Antes de empezar

Para poder aprovisionar volúmenes de SMB con `ontap-nas` el controlador, debe tener lo siguiente.

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

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

- Proxy CSI configurado como servicio de Windows. Para configurar un `csi-proxy`, consulte ["GitHub: Proxy CSI"](#) o ["GitHub: Proxy CSI para Windows"](#) para los nodos de Kubernetes que se ejecutan en Windows.

Pasos

1. Cree recursos compartidos de SMB. Puede crear los recursos compartidos de administrador de SMB de dos maneras mediante el ["Consola de administración de Microsoft"](#) complemento Carpetas compartidas o mediante la CLI de ONTAP. Para crear los recursos compartidos de SMB mediante la CLI de ONTAP:

- a. Si es necesario, cree la estructura de ruta de acceso de directorio para el recurso compartido.

El `vserver cifs share create` comando comprueba 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 de 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. Compruebe que se ha creado el recurso compartido:

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



Consulte "[Cree un recurso compartido de SMB](#)" para obtener información detallada.

- Al crear el back-end, debe configurar lo siguiente para especificar volúmenes de SMB. Para ver todas las opciones de configuración del backend de FSx para ONTAP, consulte "[Opciones y ejemplos de configuración de FSx para ONTAP](#)".

Parámetro	Descripción	Ejemplo
smbShare	Puede especificar una de las siguientes opciones: El nombre de un recurso compartido de SMB creado con la consola de administración de Microsoft o la interfaz de línea de comandos de ONTAP, o bien un nombre para permitir que Astra Trident cree el recurso compartido de SMB. Este parámetro es obligatorio para los back-ends de Amazon FSx para ONTAP.	smb-share
nasType	Debe establecerse en smb. Si es nulo, el valor por defecto es <code>nfs</code> .	smb
securityStyle	Estilo de seguridad para nuevos volúmenes. Debe establecerse en ntfs o mixed para volúmenes SMB.	ntfs O mixed para volúmenes de SMB
unixPermissions	Modo para volúmenes nuevos. Se debe dejar vacío para volúmenes SMB.	""

Configure una clase de almacenamiento y la RVP

Configure un objeto StorageClass de Kubernetes y cree la clase de almacenamiento para indicar a Astra Trident cómo se aprovisionan los volúmenes. Cree un volumen persistente (VP) y una reclamación de volumen persistente (RVP) que utilice el tipo de almacenamiento de Kubernetes configurado para solicitar acceso al VP. A continuación, puede montar el VP en un pod.

Cree una clase de almacenamiento

Configurar un objeto de Kubernetes StorageClass

El "[Objeto de Kubernetes StorageClass](#)" identifica Astra Trident como el proveedor que se usa para esa clase indica a Astra Trident cómo aprovisionar un volumen. Por ejemplo:

```
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"
```

Consulte el "[Objetos de Kubernetes y Trident](#)" para obtener más detalles sobre cómo interactúan las clases de almacenamiento con PersistentVolumeClaim los parámetros y para controlar cómo Astra Trident aprovisiona los volúmenes.

Cree una clase de almacenamiento

Pasos

1. Se trata de un objeto de Kubernetes, así que utilícelo `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 Astra Trident, y Astra Trident debería haber descubierto las piscinas en el back-end.

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

Cree el VP y la RVP

Un "[Volumen persistente](#)" (VP) es un recurso de almacenamiento físico proporcionado por el administrador del clúster en un clúster de Kubernetes. "[Claim de volumen persistente](#)"La (RVP) es una solicitud para acceder al volumen persistente en el clúster.

La RVP se puede configurar para solicitar almacenamiento de un determinado tamaño o modo de acceso. Mediante el StorageClass asociado, el administrador del clúster puede controlar mucho más que el tamaño de los volúmenes persistentes y el modo de acceso, como el rendimiento o el nivel de servicio.

Después de crear el VP y la RVP, puede montar el volumen en un pod.

Manifiestos de muestra

Manifiesto de muestra de volumen persistente

Este manifiesto de ejemplo muestra un PV básico de 10Gi que está asociado con StorageClass `basic-csi`.

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-storage
  labels:
    type: local
spec:
  storageClassName: basic-csi
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteMany
  hostPath:
    path: "/my/host/path"
```

Manifiestos de muestra de PersistentVolumeClaim

Estos ejemplos muestran opciones básicas de configuración de PVC.

PVC con acceso RWO

Este ejemplo muestra una PVC básica con acceso RWX que está asociada con una clase de almacenamiento llamada `basic-csi`.

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

PVC con NVMe/TCP

En este ejemplo se muestra una PVC básica para NVMe/TCP con acceso RWO asociada con una clase de almacenamiento llamada `protection-gold`.

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

Cree el VP y la RVP

Pasos

1. Cree el VP.

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

2. Compruebe el estado de PV.

```
kubectl get pv
NAME          CAPACITY  ACCESS MODES  RECLAIM POLICY  STATUS  CLAIM
STORAGECLASS  REASON    AGE
pv-storage    4Gi       RWO           Retain          Available
7s
```

3. Cree la RVP.

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

4. Verifique el estado de la RVP.

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

Consulte el ["Objetos de Kubernetes y Trident"](#) para obtener más detalles sobre cómo interactúan las clases de almacenamiento con PersistentVolumeClaim los parámetros y para controlar cómo Astra Trident aprovisiona los volúmenes.

Atributos de Astra Trident

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

Atributo	Tipo	Valores	Oferta	Solicitud	Admitido por
media 1	cadena	hdd, híbrido, ssd	Pool contiene medios de este tipo; híbrido significa ambos	Tipo de medios especificado	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san y solidfire-san
AprovisionaciónTipo	cadena	delgado, grueso	El pool admite este método de aprovisionamiento o	Método de aprovisionamiento o especificado	grueso: all ONTAP; thin: all ONTAP y solidfire-san

Atributo	Tipo	Valores	Oferta	Solicitud	Admitido por
Tipo de backendType	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
snapshot	bool	verdadero, falso	El pool admite volúmenes con Snapshot	Volumen con snapshots habilitadas	ontap-nas, ontap-san, solidfire-san y gcp-cvs
clones	bool	verdadero, falso	Pool admite el clonado de volúmenes	Volumen con clones habilitados	ontap-nas, ontap-san, solidfire-san y gcp-cvs
cifrado	bool	verdadero, falso	El pool admite volúmenes cifrados	Volumen con cifrado habilitado	ontap-nas, ontap-nas-economy, ontap-nas-flexgroups, ontap-san
IOPS	int	entero positivo	El pool es capaz de garantizar IOPS en este rango	El volumen garantizado de estas IOPS	solidfire-san

Esta versión 1: No es compatible con sistemas ONTAP Select

Despliegue la aplicación de muestra

Despliegue la aplicación de muestra.

Pasos

1. Monte el volumen en un pod.

```
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

```



Puede supervisar el progreso utilizando `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
```

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	

1. Ahora puede eliminar el Pod. La aplicación Pod ya no existirá, pero el volumen permanecerá.

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

Configura el complemento EKS de Astra Trident en un clúster de EKS

Astra Trident optimiza la gestión del almacenamiento de Amazon FSx para NetApp ONTAP en Kubernetes para que sus desarrolladores y administradores se centren en la puesta en marcha de aplicaciones. El complemento Astra Trident EKS incluye las últimas revisiones de seguridad, correcciones de errores y AWS lo valida para que funcione con Amazon EKS. El complemento EKS le permite garantizar de forma constante que sus

clústeres de Amazon EKS sean seguros y estables y reducir la cantidad de trabajo que necesita para instalar, configurar y actualizar complementos.

Requisitos previos

Asegúrate de disponer de lo siguiente antes de configurar el complemento Astra Trident para AWS EKS:

- Una cuenta de clúster de Amazon EKS con suscripción complementaria
- Permisos de AWS para AWS Marketplace:
"aws-marketplace:ViewSubscriptions",
"aws-marketplace:Subscribe",
"aws-marketplace:Unsubscribe"
- Tipo de AML: 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 para NetApp ONTAP existente

Pasos

1. En tu clúster de EKS Kubernetes, navega a la pestaña **Add-ons**.

The screenshot shows the AWS EKS console interface for a cluster named 'tri-env-eks'. At the top, there are buttons for 'Delete cluster' and 'Upgrade version'. Below this is a notification bar about the end of standard support for Kubernetes version 1.30 on July 28, 2025, with an 'Upgrade now' button. The main section is titled 'Cluster info' and displays the following details:

Status	Kubernetes version	Support period	Provider
Active	1.30	Standard support until July 28, 2025	EKS

Below the cluster info, there is a horizontal navigation bar with tabs: Overview, Resources, Compute, Networking, **Add-ons** (with a notification icon), Access, Observability, Upgrade insights, Update history, and Tags. The 'Add-ons' tab is selected, showing a notification: 'New versions are available for 3 add-ons.' Below this, the 'Add-ons (3)' section includes a search bar, filters for 'Any category' and 'Any status', and indicates '3 matches'. There are buttons for 'View details', 'Edit', 'Remove', and 'Get more add-ons'.

2. Vaya a **AWS Marketplace add-ons** y elija la categoría *storage*.

AWS Marketplace add-ons (1)

Discover, subscribe to and configure EKS add-ons to enhance your EKS clusters.

Filtering options

Any category ▼
NetApp, Inc. ▼
Any pricing model ▼
Clear filters

NetApp, Inc. ✕
1

NetApp Trident
☐

NetApp Trident streamlines Amazon FSx for NetApp ONTAP storage management in Kubernetes to let your developers and administrators focus on application deployment. FSx for ONTAP flexibility, scalability, and integration capabilities make it the ideal choice for organizations seeking efficient containerized storage workflows. [Product details](#)

Standard Contract

Category	Listed by	Supported versions	Pricing starting at
storage	NetApp, Inc.	1.30, 1.29, 1.28, 1.27, 1.26, 1.25, 1.24, 1.23	View pricing details

Cancel
Next

- Localiza **NetApp Trident** y selecciona la casilla de verificación para el complemento Astra Trident.
- Elija la versión deseada del complemento.

NetApp Trident

Remove add-on

Listed by NetApp	Category storage	Status ✓ Ready to install
---------------------	---------------------	------------------------------

You're subscribed to this software

You can view the terms and pricing details for this product or choose another offer if one is available.

View subscription

×

Version

Select the version for this add-on.

v24.6.1-eksbuild.1

▼

Select IAM role

Select an IAM role to use with this add-on. To create a new custom role, follow the instructions in the [Amazon EKS User Guide](#).

Not set

▼

↺

► Optional configuration settings

Cancel

Previous

Next

5. Seleccione la opción Rol IAM que desea heredar del nodo.

Review and add

Step 1: Select add-ons

[Edit](#)

Selected add-ons (1)

< 1 >

Add-on name



Type



Status

netapp_trident-operator

storage

✓ Ready to install

Step 2: Configure selected add-ons settings

[Edit](#)

Selected add-ons version (1)

< 1 >

Add-on name



Version



IAM role for service account (IRSA)

netapp_trident-operator

v24.6.1-eksbuild.1

Not set

[Cancel](#)[Previous](#)[Create](#)

6. (Opcional) Configure cualquier configuración opcional según sea necesario y seleccione **Siguiente**.

Siga el esquema de configuración **Add-On** y establezca el parámetro `configurationValues` en la sección **Valores de configuración** en el Role-arn que creó en el paso anterior (el valor debe tener el siguiente formato `eks.amazonaws.com/role-arn:`

`arn:aws:iam::464262061435:role/AmazonEKS_FSXN_CSI_DriverRole:`). Si selecciona Sustituir para el método de resolución de conflictos, una o más de las configuraciones del complemento existente se pueden sobrescribir con la configuración del complemento Amazon EKS. Si no habilita esta opción y existe un conflicto con la configuración existente, se producirá un error en la operación. Puede utilizar el mensaje de error resultante para solucionar el conflicto. Antes de seleccionar esta opción, asegúrese de que el complemento de Amazon EKS no gestiona la configuración que necesita para autogestionar.



Al configurar el parámetro opcional `cloudIdentity`, asegúrese de especificar `AWS` como `cloudProvider` al instalar Trident mediante el complemento EKS.

Select IAM role
Select an IAM role to use with this add-on. To create a new custom role, follow the instructions in the [Amazon EKS User Guide](#).

Not set

▼ Optional configuration settings

Add-on configuration schema
Refer to the JSON schema below. The configuration values entered in the code editor will be validated against this schema.

```
{
  "$id": "http://example.com/example.json",
  "$schema": "https://json-schema.org/draft/2019-09/schema",
  "default": {},
  "examples": [
    {
      "cloudIdentity": ""
    }
  ],
  "properties": {
    "cloudIdentity": {
      "default": "",
      "examples": [

```

Configuration values Info
Specify any additional JSON or YAML configurations that should be applied to the add-on.

```
1 {
2   "cloudIdentity": "'eks.amazonaws.com/role-arn: arn:aws
3   :iam::139763910815:role
4   /AmazonEKS_FSXN_CSI_DriverRole'",
  "cloudProvider": "AWS"
}
```

7. Seleccione **Crear**.
8. Compruebe que el estado del complemento es *Active*.

Add-ons (1) Info

View details Edit Remove **Get more add-ons**

netapp Any category Any status 1 match < 1 >

NetApp Astra Trident by NetApp

Astra Trident streamlines Amazon FSx for NetApp ONTAP storage management in Kubernetes to let your developers and administrators focus on application deployment. FSx for ONTAP flexibility, scalability, and integration capabilities make it the ideal choice for organizations seeking efficient containerized storage workflows. [Product details](#)

Category	Status	Version	IAM role for service account (IRSA)	Listed by
storage	Active	v24.6.1-eksbuild.1	Not set	NetApp, Inc.

[View subscription](#)

Instalar/desinstalar el complemento Astra Trident EKS mediante la CLI

Instale el complemento Astra Trident EKS mediante la CLI:

El siguiente comando de ejemplo instala el complemento EKS de Astra Trident:

```
eksctl create addon --cluster K8s-arm --name netapp_trident-operator --version v24.6.1-eksbuild
```

```
eksctl create addon --cluster clusterName --name netapp_trident-operator  
--version v24.6.1-eksbuild.1 (Con una versión dedicada)
```



Al configurar el parámetro opcional `cloudIdentity`, asegúrese de especificar `cloudProvider` al instalar Trident mediante el complemento EKS.

Desinstale el complemento Astra Trident EKS mediante la CLI:

El siguiente comando desinstala el complemento EKS de Astra Trident:

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

Cree back-ends con kubectl

Un back-end define la relación entre Astra Trident y un sistema de almacenamiento. Le indica a Astra Trident cómo se comunica con ese sistema de almacenamiento y cómo debe aprovisionar volúmenes a partir de él. Una vez instalado Astra Trident, el siguiente paso es crear un back-end. La `TridentBackendConfig` definición de recursos personalizados (CRD) le permite crear y administrar backends de Trident directamente a través de la interfaz de Kubernetes. Para ello, puede utilizar `kubectl` o la herramienta CLI equivalente para su distribución de Kubernetes.

`TridentBackendConfig`

`TridentBackendConfig (tbc, , tbconfig tbackendconfig)` Es una interfaz CRD con nombre que le permite administrar los back-ends de Astra Trident utilizando `kubectl`. Los administradores de Kubernetes y de almacenamiento ahora pueden crear y gestionar back-ends directamente mediante la CLI de Kubernetes sin necesidad de una utilidad de línea de comandos dedicada (`tridentctl`).

Al crear `TridentBackendConfig` un objeto, sucede lo siguiente:

- Astra Trident crea automáticamente un back-end en función de la configuración que proporcione. Esto se representa internamente como un `TridentBackend (tbe, tridentbackend)` CR.
- El `TridentBackendConfig` está vinculado de forma exclusiva a una `TridentBackend` instancia creada por Astra Trident.

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



`TridentBackend` Astra Trident crea CRS automáticamente. Usted **no debe** modificarlos. Si desea realizar actualizaciones en los back-ends, haga esto modificando el `TridentBackendConfig` objeto.

Consulte el siguiente ejemplo para conocer el formato de `TridentBackendConfig` la 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 puede echar un vistazo a los ejemplos ["instalador de trident"](#) del directorio para ver ejemplos de configuraciones para el servicio/plataforma de almacenamiento que desee.

``spec`` Toma parámetros de configuración específicos de backend. En este ejemplo, el backend utiliza ``ontap-san`` el controlador de almacenamiento y utiliza los parámetros de configuración que se tabulan aquí. Para obtener la lista de opciones de configuración para el controlador de almacenamiento deseado, consulte la `xref:{relative_path}backends.html["información de configuración del backend para el controlador de almacenamiento"]`.

La `spec` sección también incluye `credentials` campos y `deletionPolicy`, que se han introducido 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. Este juego debe ser un secreto de Kubernetes creado por el usuario. Las credenciales no se pueden pasar en texto sin formato y se producirá un error.
- `deletionPolicy`: Este campo define lo que debe suceder cuando se elimina el `TridentBackendConfig`. Puede ser necesario uno de los dos valores posibles:
 - `delete`: Esto resulta en la eliminación de `TridentBackendConfig` CR y el backend asociado. Este es el valor predeterminado.
 - `retain`: Cuando se elimina un `TridentBackendConfig` CR, la definición de backend seguirá estando presente y se puede gestionar con `tridentctl`. La configuración de la política de eliminación `retain` permite a los usuarios degradar a una versión anterior (anterior a 21,04) y conservar los back-ends creados. El valor de este campo se puede actualizar después de crear un `TridentBackendConfig`.



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



Los back-ends que se crearon con `tridentctl` no tienen un objeto asociado `TridentBackendConfig`. Puede optar por gestionar dichos back-ends con `kubectl` creando una `TridentBackendConfig` CR. Se debe tener cuidado de especificar parámetros de configuración idénticos (`spec.backendName` como `,`, `,`, ``spec.storagePrefix` `spec.storageDriverName` etc.). Astra Trident enlazará automáticamente los recién creados `TridentBackendConfig` con el back-end preexistente.

Descripción general de los pasos

Para crear un nuevo backend mediante `kubectl`, debe hacer lo siguiente:

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

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

Paso 1: Cree un secreto de Kubernetes

Cree un secreto que contenga las credenciales de acceso para el back-end. Esto es único para cada servicio/plataforma de almacenamiento. Veamos 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 campos secretos de la plataforma de almacenamiento	Secreto	Descripción de los campos
Azure NetApp Files	ID del Cliente	El ID de cliente de un registro de aplicación
Cloud Volumes Service para GCP	id_clave_privada	ID de la clave privada. Parte de la clave API de la cuenta de servicio de GCP con el rol de administrador CVS

Descripción de campos secretos de la plataforma de almacenamiento	Secreto	Descripción de los campos
Cloud Volumes Service para GCP	clave_privada	Clave privada. Parte de la clave API de la cuenta de servicio de GCP con el rol de administrador CVS
Element (HCI/SolidFire de NetApp)	Extremo	MVIP para el clúster de SolidFire con credenciales de inquilino
ONTAP	nombre de usuario	Nombre de usuario para conectarse al clúster/SVM. Se utiliza para autenticación basada en credenciales
ONTAP	contraseña	Contraseña para conectarse al clúster/SVM. Se utiliza para autenticación basada en credenciales
ONTAP	ClientPrivateKey	Valor codificado en base64 de la clave privada de cliente. Se utiliza para autenticación basada en certificados
ONTAP	ChapUsername	Nombre de usuario entrante. Necesario si useCHAP=true. Para ontap-san y. ontap-san-economy
ONTAP	InitichapatorSecret	Secreto CHAP del iniciador. Necesario si useCHAP=true. Para ontap-san y. ontap-san-economy
ONTAP	ChapTargetUsername	Nombre de usuario de destino. Necesario si useCHAP=true. Para ontap-san y. ontap-san-economy
ONTAP	ChapTargetInitiatorSecret	Secreto CHAP del iniciador de destino. Necesario si useCHAP=true. Para ontap-san y. ontap-san-economy

El secreto creado en este paso será referenciado en `spec.credentials` el campo del `TridentBackendConfig` objeto que se crea en el siguiente paso.

Paso 2: Crear el TridentBackendConfig CR

Ya está listo para crear su TridentBackendConfig CR. En este ejemplo, se crea un backend que utiliza ontap-san el controlador mediante el TridentBackendConfig objeto mostrado 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: Verifique el estado de la TridentBackendConfig CR

Ahora que ha creado TridentBackendConfig el CR, puede verificar el estado. Consulte el siguiente ejemplo:

```
kubectl -n trident get tbc backend-tbc-ontap-san
```

NAME	BACKEND NAME	BACKEND UUID
backend-tbc-ontap-san	ontap-san-backend	8d24fce7-6f60-4d4a-8ef6-bab2699e6ab8
Bound	Success	

Se ha creado correctamente un backend y se ha enlazado al TridentBackendConfig CR.

La fase puede tomar uno de los siguientes valores:

- **Bound:** El TridentBackendConfig CR está asociado con un backend, y ese backend contiene configRef definido en el uid del TridentBackendConfig CR.
- **Unbound:** Representado usando "". El TridentBackendConfig objeto no está enlazado a un backend. Todos los CRS recién creados TridentBackendConfig se encuentran en esta fase de forma predeterminada. Tras cambiar la fase, no puede volver a «sin límites».
- **Deleting:** TridentBackendConfig Se ha establecido que se supriman las CR deletionPolicy. Cuando TridentBackendConfig se elimina la CR, pasa al estado Supresión.
 - Si no existen reclamaciones de volumen persistentes (RVP) en el back-end, al eliminar el,

TridentBackendConfig Astra Trident eliminará tanto el back-end como la TridentBackendConfig CR.

- Si uno o más EVs están presentes en el backend, pasa a un estado de supresión. TridentBackendConfig`Posteriormente, la CR también entra en la fase de supresión. El backend y `TridentBackendConfig sólo se eliminan después de eliminar todas las EVs.

- **Lost:** El backend asociado con TridentBackendConfig el CR se eliminó accidental o deliberadamente y el TridentBackendConfig CR todavía tiene una referencia al backend eliminado. La TridentBackendConfig CR se puede eliminar independientemente del deletionPolicy valor.
- **Unknown:** Astra 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 falta el tridentbackends.trident.netapp.io CRD. Esto puede requerir intervención.

En esta fase, se ha creado un backend. Hay varias operaciones que, además, se pueden manejar, ["actualizaciones back-end y eliminaciones backend"](#) como .

(Opcional) Paso 4: Obtener más detalles

Puede ejecutar el siguiente comando para obtener más información acerca de su entorno de administración:

```
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 los backendName y el backendUUID del backend que se creó en respuesta a la TridentBackendConfig CR. lastOperationStatus`El campo representa el estado de la última operación `TridentBackendConfig del CR, que puede ser activada por el usuario (por ejemplo, el usuario cambió algo en spec) o activada por Astra Trident (por ejemplo, durante reinicios de Astra Trident). Puede ser Success o Failed. phase Representa el estado de la relación entre TridentBackendConfig el CR y el backend. En el ejemplo anterior, phase tiene el valor bound, lo que significa que TridentBackendConfig el CR está asociado al backend.

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



No puede actualizar ni suprimir un backend que contenga un objeto asociado TridentBackendConfig mediante `tridentctl`. Comprender los pasos que implica cambiar entre `tridentctl` y TridentBackendConfig, ["ver aquí"](#).

Gestionar back-ends

Realice la gestión del entorno de administración con kubectl

Obtenga información sobre cómo realizar operaciones de gestión de backend mediante `kubectl`.

Eliminar un back-end

Al eliminar un `TridentBackendConfig`, le indica a Astra Trident que elimine/conserve back-ends (según `deletionPolicy`). Para suprimir un backend, asegúrese de que `deletionPolicy` está definido como `DELETE`. Para suprimir sólo el `TridentBackendConfig`, asegúrese de que `deletionPolicy` está definido en `Retener`. Esto asegurará que el backend todavía está presente y se puede gestionar mediante el uso `tridentctl`.

Ejecute el siguiente comando:

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

Astra Trident no elimina los secretos de Kubernetes que estaban utilizando `TridentBackendConfig`. El usuario de Kubernetes es responsable de limpiar los secretos. Hay que tener cuidado a la hora de eliminar secretos. Solo debe eliminar secretos si no los están utilizando los back-ends.

Ver los back-ends existentes

Ejecute el siguiente comando:

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

También puede ejecutar `tridentctl get backend -n trident` u `tridentctl get backend -o yaml -n trident` obtener una lista de todos los back-ends existentes. Esta lista también incluirá back-ends creados con `tridentctl`.

Actualizar un back-end

Puede haber varias razones para actualizar un back-end:

- Las credenciales del sistema de almacenamiento han cambiado. Para actualizar las credenciales, se debe actualizar el secreto de Kubernetes utilizado en el `TridentBackendConfig` objeto. Astra Trident actualizará automáticamente el back-end con las últimas credenciales proporcionadas. Ejecute el siguiente comando para actualizar Kubernetes Secret:

```
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).
 - Puede `TridentBackendConfig` actualizar objetos directamente a través de Kubernetes mediante el siguiente comando:

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

- Como alternativa, puede realizar cambios en el CR existente `TridentBackendConfig` mediante el siguiente comando:

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



- Si falla una actualización de back-end, el back-end continúa en su última configuración conocida. Puede ver los logs 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 `update`.

Realizar la administración de back-end con `tridentctl`

Obtenga información sobre cómo realizar operaciones de gestión de backend mediante `tridentctl`.

Cree un back-end

Después de crear un ["archivo de configuración del back-end"](#), ejecute el siguiente comando:

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

Si se produce un error en la creación del back-end, algo estaba mal con la configuración del back-end. Puede ver los registros para determinar la causa ejecutando el siguiente comando:

```
tridentctl logs -n trident
```

Después de identificar y corregir el problema con el archivo de configuración, simplemente puede ejecutar el `create` comando de nuevo.

Eliminar un back-end

Para eliminar un back-end de Astra Trident, haga lo siguiente:

1. Recupere el nombre del backend:

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

2. Eliminar el back-end:

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



Si Astra Trident ha provisionado volúmenes y snapshots de este back-end que aún existen, al eliminar el back-end se impiden que el departamento de tecnología provisione nuevos volúmenes. El back-end continuará existiendo en un estado de “eliminación” y Trident seguirá gestionando esos volúmenes y instantáneas hasta que se eliminen.

Ver los back-ends existentes

Para ver los back-ends 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 back-end

Después de crear un nuevo archivo de configuración de back-end, ejecute el siguiente comando:

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

Si falla la actualización del back-end, algo estaba mal con la configuración del back-end o intentó una actualización no válida. Puede ver los registros para determinar la causa ejecutando el siguiente comando:

```
tridentctl logs -n trident
```

Después de identificar y corregir el problema con el archivo de configuración, simplemente puede ejecutar el update comando de nuevo.

Identifique las clases de almacenamiento que utilizan un back-end

Este es un ejemplo del tipo de preguntas que puede responder con el JSON que `tridentctl` genera los objetos backend. Esto utiliza la `jq` utilidad, que necesita instalar.

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

Esto también se aplica a los back-ends que se crearon mediante el uso ``TridentBackendConfig`` de .

Pasar entre las opciones de administración del back-end

Conozca las distintas formas de gestionar los back-ends en Astra Trident.

Opciones para gestionar back-ends

Con la introducción de `TridentBackendConfig`, los administradores ahora tienen dos formas únicas de gestionar back-ends. Esto plantea las siguientes preguntas:

- ¿Se pueden crear back-ends mediante `tridentctl` ser gestionados con `TridentBackendConfig`?
- Se pueden crear back-ends mediante la utilización `TridentBackendConfig` de `tridentctl`?

Gestionar `tridentctl` back-ends utilizando `TridentBackendConfig`

Esta sección cubre los pasos necesarios para administrar los back-ends que se crearon `tridentctl` directamente a través de la interfaz de Kubernetes mediante la creación de `TridentBackendConfig` objetos.

Esto se aplica a las siguientes situaciones:

- Back-ends preexistentes, que no tienen un `TridentBackendConfig` porque fueron creados con `tridentctl`.
- Nuevos back-ends creados con `tridentctl`, mientras existen otros `TridentBackendConfig` objetos.

En ambos escenarios, continuarán presentes los back-ends, con los volúmenes de programación de Astra Trident y el funcionamiento de ellos. A continuación, los administradores tienen una de estas dos opciones:

- Siga `tridentctl` utilizando para gestionar los back-ends creados con él.
- Backend de enlace creado mediante `tridentctl` a un nuevo `TridentBackendConfig` objeto. Hacerlo significaría que los back-ends se gestionarán usando `kubectl` y no `tridentctl`.

Para gestionar un backend preexistente mediante `kubectl`, deberá crear un `TridentBackendConfig` que se vincule al backend existente. A continuación se ofrece una descripción general de cómo funciona:

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

Paso 0: Identificar el back-end

Para crear un `TridentBackendConfig` que se vincule a un backend existente, deberá obtener la configuración de backend. En este ejemplo, supongamos que se ha creado un back-end mediante 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: Cree un secreto de Kubernetes

Cree un secreto que contenga las credenciales del back-end, 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 consiste en crear un `TridentBackendConfig` CR que se enlazará automáticamente a la preexistente `ontap-nas-backend` (como en este ejemplo). Asegurarse de que se cumplen los siguientes requisitos:

- El mismo nombre de backend se define en `spec.backendName`.
- Los parámetros de configuración son idénticos al backend original.
- Los pools virtuales (si están presentes) deben conservar el mismo orden que en el back-end original.
- Las credenciales se proporcionan a través de un secreto de Kubernetes, pero no en texto sin formato.

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: Verifique el estado de la TridentBackendConfig **CR**

Una vez creado el TridentBackendConfig, su fase debe ser Bound. También debería reflejar el mismo nombre de fondo y UUID que el del back-end existente.

```
kubectl get tbc tbc-ontap-nas-backend -n trident
```

NAME	BACKEND NAME	BACKEND UUID
tbc-ontap-nas-backend	ontap-nas-backend	52f2eb10-e4c6-4160-99fc-96b3be5ab5d7

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

NAME	STATE	VOLUMES	STORAGE DRIVER	UUID
ontap-nas-backend	online	25	ontap-nas	52f2eb10-e4c6-4160-99fc-96b3be5ab5d7

El backend ahora será completamente administrado usando el tbc-ontap-nas-backend TridentBackendConfig objeto.

Gestionar TridentBackendConfig back-ends utilizando tridentctl

`tridentctl` se puede utilizar para mostrar los back-ends creados con `TridentBackendConfig`. Además, los administradores también pueden optar por administrar completamente dichos back-ends `tridentctl` mediante la eliminación `TridentBackendConfig` y asegurarse de `spec.deletionPolicy` que se establece en `retain`.

Paso 0: Identificar el back-end

Por ejemplo, supongamos que el siguiente backend se creó usando TridentBackendConfig:

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
```

NAME	BACKEND NAME	BACKEND UUID
backend-tbc-ontap-san	ontap-san-backend	81abcb27-ea63-49bb-b606-0a5315ac5f82

```
tridentctl get backend ontap-san-backend -n trident
```

NAME	STORAGE DRIVER	UUID
ontap-san-backend	ontap-san	81abcb27-ea63-49bb-b606-0a5315ac5f82

A partir de la salida, se ve que TridentBackendConfig se ha creado correctamente y está enlazado a un backend [observe el UUID del backend].

Paso 1: Confirme deletionPolicy que está establecido en retain

Echemos un vistazo al valor de deletionPolicy. Se debe establecer en retain. Esto asegurará que cuando se elimine un TridentBackendConfig CR, la definición de backend seguirá presente y se pueda gestionar con tridentctl.

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
```

NAME	BACKEND NAME	BACKEND UUID
backend-tbc-ontap-san	ontap-san-backend	81abcb27-ea63-49bb-b606-0a5315ac5f82

```
# 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
backend-tbc-ontap-san	ontap-san-backend	81abcb27-ea63-49bb-b606-0a5315ac5f82



No continúe con el siguiente paso a menos que `deletionPolicy` esté establecido en `retain`.

Paso 2: Eliminar el `TridentBackendConfig` CR

El paso final es eliminar la `TridentBackendConfig` CR. Después de confirmar que el `deletionPolicy` está definido en `retain`, puede continuar 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 `TridentBackendConfig` del objeto, Astra Trident simplemente lo elimina sin eliminar el propio back-end.

Crear y gestionar clases de almacenamiento

Cree una clase de almacenamiento

Configure un objeto `StorageClass` de Kubernetes y cree la clase de almacenamiento para indicar a Astra Trident cómo se aprovisionan los volúmenes.

Configurar un objeto de Kubernetes `StorageClass`

El "[Objeto de Kubernetes `StorageClass`](#)" identifica Astra Trident como el aprovisionador que se usa para esa clase e indica a Astra Trident cómo aprovisionar un volumen. Por ejemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: <Name>
provisioner: csi.trident.netapp.io
mountOptions: <Mount Options>
parameters:
  <Trident Parameters>
allowVolumeExpansion: true
volumeBindingMode: Immediate
```

Consulte el ["Objetos de Kubernetes y Trident"](#) para obtener más detalles sobre cómo interactúan las clases de almacenamiento con PersistentVolumeClaim los parámetros y para controlar cómo Astra Trident aprovisiona los volúmenes.

Cree una clase de almacenamiento

Después de crear el objeto StorageClass, puede crear la clase de almacenamiento. [Muestras de clase de almacenamiento](#) proporciona algunas muestras básicas que puede utilizar o modificar.

Pasos

1. Se trata de un objeto de Kubernetes, así que utilícelo `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 Astra Trident, y Astra Trident debería haber descubierto las piscinas en el back-end.


```

kubectl get sc basic-csi
NAME          PROVISIONER          AGE
basic-csi     csi.trident.netapp.io 15h

./tridentctl -n trident get storageclass basic-csi -o json
{
  "items": [
    {
      "Config": {
        "version": "1",
        "name": "basic-csi",
        "attributes": {
          "backendType": "ontap-nas"
        },
        "storagePools": null,
        "additionalStoragePools": null
      },
      "storage": {
        "ontapnas_10.0.0.1": [
          "aggr1",
          "aggr2",
          "aggr3",
          "aggr4"
        ]
      }
    }
  ]
}

```

Muestras de clase de almacenamiento

Astra Trident ["definiciones simples de clase de almacenamiento para back-ends específicos"](#) proporciona .

Como alternativa, puede editar `sample-input/storage-class-csi.yaml.tmpl` el archivo que viene con el instalador y reemplazarlo `BACKEND_TYPE` por 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
```

Gestione las clases de almacenamiento

Puede ver las clases de almacenamiento existentes, definir una clase de almacenamiento predeterminada, identificar el back-end de la clase de almacenamiento y eliminar clases de almacenamiento.

Consulte las clases de almacenamiento existentes

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

```
kubectl get storageclass
```

- Para ver la información sobre la clase de almacenamiento Kubernetes, ejecute el siguiente comando:

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

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

```
tridentctl get storageclass
```

- Para ver la información detallada de la clase de almacenamiento sincronizado de Astra 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 un tipo de almacenamiento predeterminado. Esta es la clase de almacenamiento que se usará para aprovisionar un volumen persistente si un usuario no especifica una en una solicitud de volumen persistente (PVC).

- Defina una clase de almacenamiento predeterminada estableciendo la anotación `storageclass.kubernetes.io/is-default-class` en `TRUE` en la definición de la clase de almacenamiento. Según la especificación, cualquier otro valor o ausencia de la anotación se interpreta como falso.
- Puede configurar una clase de almacenamiento existente para que sea la 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 forma similar, puede eliminar la anotación predeterminada de la clase de almacenamiento 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 del instalador de Trident que incluyen esta anotación.



Solo debe haber una clase de almacenamiento predeterminada en el clúster a la vez. Si no dispone de más de una, técnicamente, Kubernetes no le impide ofrecer más de una, pero funcionará como si no hubiera una clase de almacenamiento predeterminada en absoluto.

Identifique el back-end para una clase de almacenamiento

Este es un ejemplo del tipo de preguntas que puedes responder con el JSON que `tridentctl` genera los objetos back-end de Astra Trident. Esto utiliza la `jq` utilidad, que es posible que tenga que instalar primero.

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

Elimine una clase de almacenamiento

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

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

`<storage-class>` se debe sustituir por su clase de almacenamiento.

Cualquier volumen persistente que se cree a través de esta clase de almacenamiento no cambiará y Astra Trident seguirá gestionarlo.



Astra Trident aplica un espacio en blanco `fsType` para los volúmenes que crea. En el caso de los back-ends iSCSI, se recomienda aplicar `parameters.fsType` en la clase de almacenamiento. Debe eliminar `StorageClasses` existentes y volver a crearlos con `parameters.fsType` los especificados.

Aprovisione y gestione volúmenes

Aprovisione un volumen

Cree un volumen persistente (VP) y una reclamación de volumen persistente (RVP) que utilice el tipo de almacenamiento de Kubernetes configurado para solicitar acceso al VP. A continuación, puede montar el VP en un pod.

Descripción general

Un "*Volumen persistente*" (VP) es un recurso de almacenamiento físico provisionado por el administrador del clúster en un clúster de Kubernetes. "*Claim de volumen persistente*" La (RVP) es una solicitud para acceder al volumen persistente en el clúster.

La RVP se puede configurar para solicitar almacenamiento de un determinado tamaño o modo de acceso. Mediante el `StorageClass` asociado, el administrador del clúster puede controlar mucho más que el tamaño de los volúmenes persistentes y el modo de acceso, como el rendimiento o el nivel de servicio.

Después de crear el VP y la RVP, puede montar el volumen en un pod.

Manifiestos de muestra

Manifiesto de muestra de volumen persistente

Este manifiesto de ejemplo muestra un PV básico de 10Gi que está asociado con StorageClass `basic-csi`.

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pv-storage
  labels:
    type: local
spec:
  storageClassName: basic-csi
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/my/host/path"
```

Manifiestos de muestra de PersistentVolumeClaim

Estos ejemplos muestran opciones básicas de configuración de PVC.

PVC con acceso RWO

Este ejemplo muestra una PVC básica con acceso RWO que está asociada con una clase de almacenamiento 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

En este ejemplo se muestra una PVC básica para NVMe/TCP con acceso RWO asociada con una clase de almacenamiento llamada `protection-gold`.

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

Muestras de manifiesto de POD

Estos ejemplos muestran configuraciones básicas para conectar la RVP 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
```

Configuración de NVMe/TCP básica

```
---
apiVersion: v1
kind: Pod
metadata:
  creationTimestamp: null
  labels:
    run: nginx
  name: nginx
spec:
  containers:
    - image: nginx
      name: nginx
      resources: {}
      volumeMounts:
        - mountPath: "/usr/share/nginx/html"
          name: task-pv-storage
  dnsPolicy: ClusterFirst
  restartPolicy: Always
  volumes:
    - name: task-pv-storage
      persistentVolumeClaim:
        claimName: pvc-san-nvme
```

Cree el VP y la RVP

Pasos

1. Cree el VP.

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

2. Compruebe el estado de PV.

```
kubectl get pv
NAME          CAPACITY  ACCESS MODES  RECLAIM POLICY  STATUS  CLAIM
STORAGECLASS  REASON    AGE
pv-storage    4Gi       RWO           Retain          Available
7s
```

3. Cree la RVP.


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

4. Verifique el estado de la RVP.

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

5. Monte el volumen en un pod.

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



Puede supervisar el progreso utilizando `kubectl get pod --watch`.

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

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

7. Ahora puede eliminar el Pod. La aplicación Pod ya no existirá, pero el volumen permanecerá.

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

Consulte el ["Objetos de Kubernetes y Trident"](#) para obtener más detalles sobre cómo interactúan las clases de almacenamiento con `PersistentVolumeClaim` los parámetros y para controlar cómo Astra Trident aprovisiona los volúmenes.

Expanda los volúmenes

Astra Trident ofrece a los usuarios de Kubernetes la capacidad de ampliar sus volúmenes una vez que se han creado. Encuentre información sobre las configuraciones que se necesitan para ampliar los volúmenes iSCSI y NFS.

Expanda un volumen iSCSI

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



Los `solidfire-san` controladores , , `ontap-san-economy` admiten la expansión del volumen iSCSI `ontap-san` y requiere Kubernetes 1,16 y posterior.

Paso 1: Configure el tipo de almacenamiento para que admita la ampliación de volumen

Edita la definición de `StorageClass` para definir `allowVolumeExpansion` el campo en `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 un StorageClass ya existente, edítelo para incluir el `allowVolumeExpansion` parámetro.

Paso 2: Cree una RVP con el tipo de almacenamiento que ha creado

Edite la definición de PVC y actualice el `spec.resources.requests.storage` para reflejar el tamaño recién 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
```

Astra Trident crea un volumen persistente (PV) y lo asocia con esta solicitud de volumen persistente (PVC).

```
kubectl get pvc
```

NAME	STATUS	VOLUME	CAPACITY
san-pvc	Bound	pvc-8a814d62-bd58-4253-b0d1-82f2885db671	1Gi

```
kubectl get pv
```

NAME	RECLAIM POLICY	STATUS	CLAIM	CAPACITY	ACCESS MODES	STORAGECLASS	REASON	AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671	Delete	Bound	default/san-pvc	1Gi	RWO	ontap-san		10s

Paso 3: Defina un pod que fije el PVC

Conecte el VP a un pod para que se cambie su tamaño. Existen dos situaciones a la hora de cambiar el tamaño de un VP iSCSI:

- Si el VP está conectado a un pod, Astra Trident amplía el volumen en el back-end de almacenamiento, vuelve a buscar el dispositivo y cambia el tamaño del sistema de archivos.
- Cuando se intenta cambiar el tamaño de un VP sin conectar, Astra Trident amplía el volumen en el back-end de almacenamiento. Una vez que la RVP está Unido a un pod, Trident vuelve a buscar el dispositivo y cambia el tamaño del sistema de archivos. Kubernetes, después, actualiza el tamaño de RVP después de completar correctamente la operación de ampliación.

En este ejemplo, se crea un pod que utiliza `san-pvc` el .

```

kubect1 get pod
NAME          READY   STATUS    RESTARTS   AGE
ubuntu-pod    1/1     Running   0           65s

kubect1 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: Expande el PV

Para cambiar el tamaño del VP que se ha creado de 1Gi a 2Gi, edite la definición de PVC y actualice el `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

Para validar que la ampliación ha funcionado correctamente, compruebe el tamaño del volumen PVC, PV y Astra 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 |
+-----+-----+-----+-----+-----+-----+
|          BACKEND UUID          | STATE | MANAGED |
+-----+-----+-----+-----+-----+-----+
| pvc-8a814d62-bd58-4253-b0d1-82f2885db671 | 2.0 GiB | ontap-san |
+-----+-----+-----+-----+-----+-----+
| block | a9b7bfff-0505-4e31-b6c5-59f492e02d33 | online | true |
+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+-----+
```

Expanda un volumen NFS

Astra Trident es compatible con la expansión de volumen para VP NFS provisionados en ontap-nas, , ontap-nas-economy ontap-nas-flexgroup gcp-cvs y azure-netapp-files back-ends.

Paso 1: Configure el tipo de almacenamiento para que admita la ampliación de volumen

Para cambiar el tamaño de un PV de NFS, en primer lugar, el administrador debe configurar la clase de almacenamiento para permitir la expansión del volumen estableciendo `allowVolumeExpansion` el campo en `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 creó un tipo de almacenamiento sin esta opción, puede editar el tipo de almacenamiento existente mediante el uso `kubectl edit storageclass` de para permitir la expansión de volumen.

Paso 2: Cree una RVP con el tipo de almacenamiento que ha creado

```
cat pvc-ontapnas.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: ontapnas20mb
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 20Mi
  storageClassName: ontapnas
```

Astra Trident debe crear un PV NFS de 20 MiB para esta RVP:

```
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: Expande el PV

Para cambiar el tamaño del PV de 20MiB recién creado a 1GiB, edite la RVP y ajústelo `spec.resources.requests.storage` a 1GiB:

```

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 validar que el tamaño de la configuración ha funcionado correctamente comprobando el tamaño del volumen PVC, PV y Astra 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      |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```

Importar volúmenes

Puede importar volúmenes de almacenamiento existentes como VP de Kubernetes mediante `tridentctl import`.

Descripción general y consideraciones

Es posible importar un volumen en Astra Trident para lo siguiente:

- Agrupe en contenedores una aplicación y vuelva a utilizar su conjunto de datos existente
- Utilice el clon de un conjunto de datos para una aplicación efímera
- Reconstruya un clúster de Kubernetes que haya fallado
- Migración de datos de aplicaciones durante la recuperación ante desastres

Consideraciones

Antes de importar un volumen, revise las siguientes consideraciones.

- Astra Trident solo puede importar volúmenes de ONTAP de tipo RW (lectura y escritura). Los volúmenes del tipo DP (protección de datos) son volúmenes de destino de SnapMirror. Debe romper la relación de reflejo antes de importar el volumen a Astra Trident.

- Sugerimos importar volúmenes sin conexiones activas. Para importar un volumen que se usa activamente, clone el volumen y, a continuación, realice la importación.



Esto es especialmente importante en el caso de volúmenes de bloque, ya que Kubernetes no sabía que la conexión anterior y podría conectar fácilmente un volumen activo a un pod. Esto puede provocar daños en los datos.

- Si bien `StorageClass` debe especificarse en una RVP, Astra Trident no utiliza este parámetro durante la importación. Durante la creación de volúmenes, se usan las clases de almacenamiento para seleccionar entre los pools disponibles según las características de almacenamiento. Como el volumen ya existe, no se requiere ninguna selección de pool durante la importación. Por lo tanto, la importación no fallará incluso si el volumen existe en un back-end o pool que no coincide con la clase de almacenamiento especificada en la RVP.
- El tamaño del volumen existente se determina y se establece en la RVP. Una vez que el controlador de almacenamiento importa el volumen, se crea el PV con un `ClaimRef` al PVC.
 - La política de reclamaciones se establece inicialmente en `retain` el VP. Una vez que Kubernetes enlaza correctamente la RVP y el VP, se actualiza la política de reclamaciones para que coincida con la política de reclamaciones de la clase de almacenamiento.
 - Si la política de reclamación de la clase de almacenamiento es `delete`, el volumen de almacenamiento se eliminará al eliminar el VP.
- De forma predeterminada, Astra Trident gestiona la RVP y cambia el nombre de FlexVol y LUN en el back-end. Puede pasar `--no-manage` la marca para importar un volumen no gestionado. Si utiliza `--no-manage`, Astra Trident no realiza ninguna operación adicional en la RVP o el VP durante el ciclo de vida de los objetos. El volumen de almacenamiento no se elimina cuando se elimina el VP, y también se ignoran otras operaciones como el clon de volumen y el cambio de tamaño de volumen.



Esta opción es útil si desea usar Kubernetes para cargas de trabajo en contenedores, pero de lo contrario desea gestionar el ciclo de vida del volumen de almacenamiento fuera de Kubernetes.

- Se agrega una anotación a la RVP y al VP que tiene el doble propósito de indicar que el volumen se importó y si se administran la PVC y la VP. Esta anotación no debe modificarse ni eliminarse.

Importe un volumen

Puede usar `tridentctl import` para importar un volumen.

Pasos

1. Cree el archivo de reclamación de volumen persistente (RVP) (por ejemplo, `pvc.yaml`) que se utilizará para crear la RVP. El archivo PVC debe incluir `name`, `namespace`, `accessModes` y `storageClassName`. Opcionalmente, puede especificar `unixPermissions` en la definición de RVP.

A continuación se muestra 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 VP o el tamaño del volumen. Esto puede provocar un error en el comando de importación.

2. Utilice `tridentctl import volume` el comando para especificar el nombre del back-end de Astra 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 Cloud Volumes Service). El `-f` argumento es necesario 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 volúmenes para los controladores compatibles.

NAS de ONTAP y NAS FlexGroup de ONTAP

Astra Trident admite la importación de volúmenes mediante los `ontap-nas` controladores y `ontap-nas-flexgroup`



- ``ontap-nas-economy`` El controlador no puede importar ni gestionar qtrees.
- `ontap-nas`` Los controladores y ``ontap-nas-flexgroup` no permiten nombres de volúmenes duplicados.

Cada volumen creado con el `ontap-nas` controlador es un FlexVol en el clúster de ONTAP. La importación de FlexVols con `ontap-nas` el controlador funciona igual. Una FlexVol que ya existe en un clúster de ONTAP puede importarse como `ontap-nas` una RVP. Del mismo modo, los volúmenes de FlexGroup se pueden importar como `ontap-nas-flexgroup` RVP.

Ejemplos de NAS de ONTAP

A continuación, se muestra un ejemplo de un volumen gestionado y una importación de volumen no gestionada.

Volumen gestionado

En el ejemplo siguiente se importa un volumen `managed_volume` llamado en un back-end llamado `ontap_nas`:

```
tridentctl import volume ontap_nas managed_volume -f <path-to-pvc-file>
```

NAME	SIZE	STORAGE CLASS
PROTOCOL	BACKEND UUID	STATE
pvc-bf5ad463-afbb-11e9-8d9f-5254004dfdb7	1.0 GiB	standard
file	c5a6f6a4-b052-423b-80d4-8fb491a14a22	online

Volumen no gestionado

Cuando se utiliza `--no-manage` el argumento, Astra 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
pvc-df07d542-afbc-11e9-8d9f-5254004dfdb7	1.0 GiB	standard
file	c5a6f6a4-b052-423b-80d4-8fb491a14a22	online

SAN de ONTAP

Astra Trident admite la importación de volúmenes con `ontap-san` el controlador. La importación de volúmenes no es compatible con `ontap-san-economy` el controlador.

Astra Trident puede importar volúmenes FlexVol de SAN de ONTAP que contengan una única LUN. Esto es coherente con `ontap-san` el controlador, que crea una FlexVol para cada RVP y una LUN dentro de la FlexVol. Astra Trident importa el FlexVol y lo asocia con la definición de RVP.

A continuación, se muestra un ejemplo de un volumen gestionado y una importación de volumen no gestionada.

En el caso de los volúmenes gestionados, Astra Trident cambia el nombre FlexVol al `pvc-<uuid>` formato y la LUN dentro de FlexVol a `lun0`.

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

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

179

Vserver	Igroup	Protocol	OS Type	Initiators
svm0	k8s-nodename.example.com-fe5d36f2-cded-4f38-9eb0-c7719fc2f9f3	iscsi	linux	iqn.1994-05.com.redhat:4c2e1cf35e0
svm0	unmanaged-example-igroup	mixed	linux	iqn.1994-05.com.redhat:4c2e1cf35e0

Elemento

Astra Trident admite el software NetApp Element y la importación de volúmenes de NetApp HCI mediante `solidfire-san` el controlador.



El controlador Element admite los nombres de volúmenes duplicados. Sin embargo, Astra Trident devuelve un error si hay nombres de volúmenes duplicados. Como solución alternativa, clone el volumen, proporcione un nombre de volumen único e importe el volumen clonado.

Ejemplo de elemento

En el siguiente ejemplo se importa un `element-managed` volumen en el back-end `element_default`.

```
tridentctl import volume element_default element-managed -f pvc-basic-import.yaml -n trident -d
```

NAME	SIZE	STORAGE CLASS
pvc-970ce1ca-2096-4ecd-8545-ac7edc24a8fe	10 GiB	basic-element

PROTOCOL	BACKEND UUID	STATE	MANAGED
block	d3ba047a-ea0b-43f9-9c42-e38e58301c49	online	true

Google Cloud Platform

Astra Trident admite la importación de volúmenes con `gcp-cvs` el controlador.



Para importar un volumen respaldado por NetApp Cloud Volumes Service en Google Cloud Platform, identifique el volumen según la ruta de volumen. La ruta del volumen es la parte 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

En el ejemplo siguiente se importa `gcp-cvs` un volumen en el back-end `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

Astra Trident admite la importación de volúmenes con `azure-netapp-files` el controlador.



Para importar un volumen de Azure NetApp Files, identifique el volumen por su ruta de volumen. La ruta del volumen es la parte de la ruta de exportación del volumen después de `:/`. Por ejemplo, si la ruta de montaje es `10.0.0.2:/importvol1`, la ruta de volumen es `importvol1`.

Ejemplo de Azure NetApp Files

En el ejemplo siguiente se importa `azure-netapp-files` un volumen en el back-end `azurenetaappfiles_40517` con la ruta de volumen `importvol1`.

```
tridentctl import volume azurenetaappfiles_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      |
+-----+-----+-----+
+-----+-----+-----+-----+
```

Personalizar nombres y etiquetas de volúmenes

Con Astra Trident, puedes asignar nombres y etiquetas significativos a los volúmenes que creas. Esto le ayuda a identificar y asignar fácilmente volúmenes a sus respectivos recursos de Kubernetes (RVP). También puede definir plantillas en el nivel de back-end para crear nombres de volúmenes y etiquetas personalizadas; los volúmenes que cree, importe o clone respetarán las plantillas.

Antes de empezar

Nombres de volumen y etiquetas personalizables admiten:

1. Operaciones de creación, importación y clonado de volúmenes.
2. En el caso del controlador económico ontap-nas, solo el nombre del volumen Qtree debe cumplir con la plantilla de nombre.
3. En el caso del controlador ontap-san-economy, solo el nombre de la LUN cumple con la plantilla de nombre.

Limitaciones

1. Los nombres de volumen personalizables solo son compatibles con los controladores locales de ONTAP.
2. Los nombres de volúmenes personalizables no se aplican a los volúmenes existentes.

Comportamientos clave de los nombres de volúmenes personalizables

1. Si se produce un fallo debido a una sintaxis no válida en una plantilla de nombre, se produce un error en la creación del backend. Sin embargo, si la aplicación de plantilla falla, el volumen se nombrará de acuerdo con la convención de nomenclatura existente.
2. El prefijo de almacenamiento no es aplicable cuando se asigna el nombre de un volumen mediante una plantilla de nombres en la configuración back-end. Cualquier valor de prefijo deseado se puede agregar directamente a la plantilla.

Ejemplos de configuración de backend con plantilla de nombre y etiquetas

Las plantillas de nombre personalizado se pueden definir en el nivel 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.RequestName}}"
  },
  "labels": {"cluster": "ClusterA", "PVC":
    "{{.volume.Namespace}}_{{.volume.RequestName}}"
  }
}
```

Ejemplo de nivel de pool

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "ontap-nfs-backend",
  "managementLIF": "<ip address>",
  "svm": "svm0",
  "username": "<admin>",
  "password": "<password>",
  "useREST": true,
  "storage": [
    {
      "labels":{"labelname":"label1", "name": "{{ .volume.Name }}"},
      "defaults":
      {
        "nameTemplate": "pool01_{{ .volume.Name }}_{{ .labels.cluster
}}_{{ .volume.Namespace }}_{{ .volume.RequestName }}"
      }
    },
    {
      "labels":{"cluster":"label2", "name": "{{ .volume.Name }}"},
      "defaults":
      {
        "nameTemplate": "pool02_{{ .volume.Name }}_{{ .labels.cluster
}}_{{ .volume.Namespace }}_{{ .volume.RequestName }}"
      }
    }
  ]
}
```

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 que considerar

1. En el caso de las 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 a la plantilla de nombres definida en el nivel raíz en la definición de backend.
3. Astra Trident no admite el uso de un operador de segmentos con el prefijo de almacenamiento.
4. Si las plantillas no dan como resultado nombres de volúmenes únicos, Astra Trident añadirá algunos caracteres aleatorios para crear nombres de volumen únicos.
5. Si el nombre personalizado de un volumen de ahorro de NAS supera los 64 caracteres de longitud, Astra Trident asignará el nombre a los volúmenes según la convención de nomenclatura existente. Para el resto de los controladores ONTAP, si el nombre del volumen supera el límite de nombre, se produce un error en el proceso de creación de volúmenes.

Comparta un volumen NFS en espacios de nombres

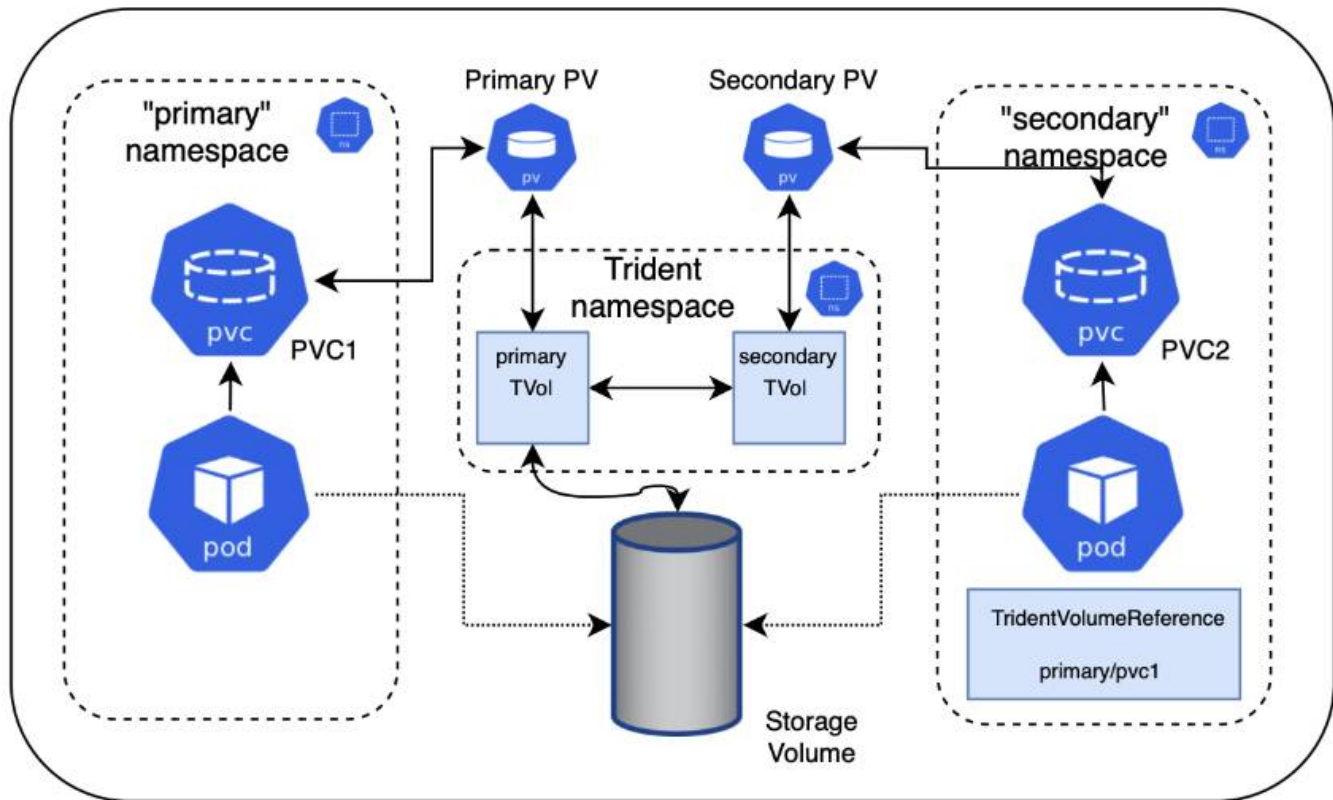
Con Astra Trident, se puede crear un volumen en un espacio de nombres primario y compartirlo en uno o más espacios de nombres secundarios.

Funciones

La Astra TridentVolumeReference CR le permite compartir de forma segura volúmenes NFS ReadWriteMany (RWX) en uno o más espacios de nombres de Kubernetes. Esta solución nativa de Kubernetes tiene las siguientes ventajas:

- Varios niveles de control de acceso para garantizar la seguridad
- Funciona con todos los controladores de volúmenes NFS de Trident
- No depende de tridentctl ni de ninguna otra función de Kubernetes no nativa

Este diagrama ilustra el uso compartido de volúmenes de NFS en dos espacios de nombres de Kubernetes.



Inicio rápido

Puede configurar el uso compartido del volumen NFS en unos pocos pasos.

1

Configure la PVC de origen para compartir el volumen

El propietario del espacio de nombres de origen concede permiso para acceder a los datos de la RVP de origen.

2

Otorgar permiso para crear una CR en el espacio de nombres de destino

El administrador del clúster concede permiso al propietario del espacio de nombres de destino para crear el sistema `TridentVolumeReference` CR.

3

Cree `TridentVolumeReference` en el espacio de nombres de destino

El propietario del espacio de nombres de destino crea el `TridentVolumeReference` CR para hacer referencia al PVC de origen.

4

Cree la RVP subordinada en el espacio de nombres de destino

El propietario del espacio de nombres de destino crea el PVC subordinado para utilizar el origen de datos desde el 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 de clúster y el propietario del espacio de nombres de destino. La función de 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 la `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
```

Astra Trident crea el VP y su volumen de almacenamiento NFS back-end.



- Puede compartir el PVC en varios espacios de nombres utilizando una lista delimitada por comas. Por ejemplo, `trident.netapp.io/shareToNamespace: namespace2, namespace3, namespace4`.
- Puede compartir en todos los espacios de nombres utilizando `*`. Por ejemplo: `trident.netapp.io/shareToNamespace: *`
- Puede actualizar la RVP para incluir la `shareToNamespace` anotación en cualquier momento.

2. **Administrador de clúster:** cree la función personalizada y kubeconfig para conceder permiso al propietario del espacio de nombres de destino para crear el sistema `TridentVolumeReference` CR en el espacio de nombres de destino.
3. **Propietario del espacio de nombres de destino:** Crear un CR de `TridentVolumeReference` en el espacio de nombres de destino que se refiere 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 (namespace2)(pvc2 en el espacio de nombres de destino) Utilizando la `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 PVC de destino debe ser menor o igual que el PVC de origen.

Resultados

Astra Trident lee la `shareFromPVC` anotación en la RVP de destino y crea el VP de destino como un volumen subordinado sin ningún recurso de almacenamiento propio que apunte al VP de origen y comparta el recurso de almacenamiento VP de origen. La RVP y el VP de destino aparecen vinculados como normales.

Elimine un volumen compartido

Es posible eliminar un volumen que se comparte en varios espacios de nombres. Astra Trident eliminará el acceso al volumen en el espacio de nombres de origen y mantendrá el acceso a otros espacios de nombres que comparten el volumen. Cuando se eliminan todos los espacios de nombres que hacen referencia al volumen, Astra Trident elimina el volumen.

Se utiliza `tridentctl get` para consultar los volúmenes subordinados

Con `tridentctl` la utilidad, se puede ejecutar `get` el comando para obtener volúmenes subordinados. Para obtener más información, consulte [LINK:../Trident-reference/tridentctl.html](#) `tridentctl` commands and

options].

```
Usage:
  tridentctl get [option]
```

Indicadores:

- ``-h, --help`: Ayuda para volúmenes.
- `--parentOfSubordinate string`: Limite la consulta al volumen fuente subordinado.
- `--subordinateOf string`: Limite la consulta a los subordinados de volumen.

Limitaciones

- Astra Trident no puede evitar que los espacios de nombres de destino se escriban en el volumen compartido. Se debe usar el bloqueo de archivos u otros procesos para evitar la sobrescritura de datos de volúmenes compartidos.
- No puede revocar el acceso a la PVC de origen eliminando `shareToNamespace` las anotaciones o `shareFromNamespace` eliminando la `TridentVolumeReference` CR. Para revocar el acceso, debe eliminar el PVC subordinado.
- Las snapshots, los clones y el mirroring no son posibles en los volúmenes subordinados.

Si quiere más información

Para obtener más información sobre el acceso de volúmenes entre espacios de nombres:

- Visite ["Uso compartido de volúmenes entre espacios de nombres: Dé la bienvenida al acceso al volumen entre espacios de nombres"](#).
- Vea la demostración en ["NetAppTV"](#).

Replicar volúmenes mediante SnapMirror

Con Astra Control Provisioning, puede crear relaciones de mirroring entre un volumen de origen en un clúster y el volumen de destino en el clúster con relación de paridad para replicar datos para la recuperación de desastres. Puede utilizar una definición de recursos personalizados (CRD) con nombre para realizar las siguientes operaciones:

- Crear relaciones de mirroring entre volúmenes (RVP)
- Elimine las relaciones de reflejo entre volúmenes
- Rompa las relaciones de reflejo
- Promocionar el volumen secundario durante condiciones de desastre (conmutaciones al respaldo).
- Realice una transición de las aplicaciones sin pérdidas de un clúster a otro (durante las migraciones y las conmutaciones al respaldo planificadas).

Requisitos previos de replicación

Asegúrese de que se cumplen los siguientes requisitos previos antes de comenzar:

Clústeres ONTAP

- **Astra Control Provisionador:** Astra Control Provisionador versión 23,10 o posterior debe existir en los clústeres de Kubernetes de origen y destino que utilizan ONTAP como backend.
- **Licencias:** Las licencias asíncronas de SnapMirror de ONTAP que utilizan el paquete de protección de datos deben estar habilitadas en los clústeres de ONTAP de origen y de destino. Consulte ["Información general sobre las licencias de SnapMirror en ONTAP"](#) si desea obtener más información.

Interconexión

- **Cluster y SVM:** Los back-ends de almacenamiento ONTAP deben ser peered. Consulte ["Información general sobre relaciones entre iguales de clústeres y SVM"](#) si desea obtener más información.



Compruebe que los nombres de las SVM utilizados en la relación de replicación entre dos clústeres de ONTAP sean únicos.

- **Astra Control Provisionador y SVM:** Las SVM remotas entre iguales deben estar disponibles para Astra Control Provisionador en el clúster de destino.

Controladores compatibles

- La replicación de volúmenes es compatible con los controladores ontap-nas y ontap-san.

Cree una RVP reflejada

Siga estos pasos y utilice los ejemplos de CRD para crear una relación de reflejo entre los volúmenes primario y secundario.

Pasos

1. Realice los siguientes pasos en el clúster de Kubernetes principal:
 - a. Cree 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 una RVP con el tipo de almacenamiento creado anteriormente.

Ejemplo

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: csi-nas
```

- c. Cree un CR de MirrorRelationship con información local.

Ejemplo

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: promoted
  volumeMappings:
    - localPVCName: csi-nas
```

Astra Control Provisioner obtiene la información interna del volumen y el estado actual de protección de datos (DP) del volumen y, a continuación, rellena el campo de estado del MirrorRelationship.

- d. Obtenga el TridentMirrorRelationship CR para obtener el nombre interno y SVM de la PVC.

```
kubect1 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 de Kubernetes secundario:

- 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 de 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"

```

El proveedor de control de Astra creará una relación de SnapMirror con el nombre de la política de relaciones configurada (o predeterminado para ONTAP) e inicializarla.

- c. Crear una RVP con StorageClass creado anteriormente para que actúe como secundario (destino de 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
```

El proveedor de control de Astra comprobará el CRD de TridentMirrorRelationship y no podrá crear el volumen si la relación no existe. Si existe la relación, el proveedor de Astra Control se asegurará de que el nuevo volumen de FlexVol se coloque en una SVM vinculada con la SVM remota definida en MirrorRelationship.

Estados de replicación de volúmenes

Una relación de mirroring de Trident (TMR) es un CRD que representa un extremo de una relación de replicación entre RVP. El TMR de destino tiene un estado, que le dice a Astra Control Provisioner 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 ReadWrite y montable, sin relación de espejo actualmente en vigor.
- **Reestablecido:** El PVC local es el volumen de destino de una relación de espejo y también estaba anteriormente en esa relación de espejo.
 - El estado reestablecido se debe usar si el volumen de destino alguna vez mantuvo una relación con el volumen de origen debido a que sobrescribe el contenido del volumen de destino.
 - El estado reestablecido generará un error si el volumen no mantuvo una relación anteriormente con el origen.

Promocione la RVP secundaria durante una conmutación al respaldo no planificada

Realice el siguiente paso en el clúster de Kubernetes secundario:

- Actualice el campo `spec.state` de TridentMirrorRelationship a `promoted`.

Promocione la RVP secundaria durante una conmutación al respaldo planificada

Durante una conmutación al respaldo planificada (migración), realice los siguientes pasos para promocionar la RVP secundaria:

Pasos

1. En el clúster de Kubernetes principal, cree una snapshot de la RVP y espere hasta que se cree la snapshot.
2. En el clúster de Kubernetes principal, cree SnapshotInfo CR para obtener información interna.

Ejemplo

```
kind: SnapshotInfo
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  snapshot-name: csi-nas-snapshot
```

3. En el clúster de Kubernetes secundario, actualice el campo *spec.state* de *TridentMirrorRelationship* CR a *promoted* y *spec.promotedSnapshotHandle* para que sea InternalName de la snapshot.
4. En un clúster de Kubernetes secundario, confirme el estado (campo *status.state*) de *TridentMirrorRelationship* a *Promoted*.

Restaurar una relación de mirroring después de una conmutación al nodo de respaldo

Antes de restaurar una relación de reflejo, elija el lado que desea realizar como el nuevo primario.

Pasos

1. En el clúster de Kubernetes secundario, compruebe que se actualicen los valores del campo *spec.remoteVolumeHandle* del *TridentMirrorRelationship*.
2. En el clúster de Kubernetes secundario, actualice el campo *spec.mirror* de *TridentMirrorRelationship* a *reestablished*.

Operaciones adicionales

Astra Control Provisioning admite las siguientes operaciones en los volúmenes primarios y secundarios:

Replica la PVC primaria a una nueva PVC secundaria

Asegúrese de que ya tiene un PVC primario y un PVC secundario.

Pasos

1. Elimine los CRD de *PersistentVolumeClaim* y *TridentMirrorRelationship* del clúster secundario (destino) establecido.
2. Elimine el CRD de *TridentMirrorRelationship* del clúster primario (origen).
3. Cree un nuevo CRD de *TridentMirrorRelationship* en el clúster primario (de origen) para la nueva PVC secundaria (de destino) que desea establecer.

Cambie el tamaño de una RVP reflejada, primaria o secundaria

El PVC se puede cambiar de tamaño como normal, ONTAP expandirá automáticamente cualquier flexvol de destino si la cantidad de datos excede el tamaño actual.

Elimine la replicación de una RVP

Para eliminar la replicación, realice una de las siguientes operaciones en el volumen secundario actual:

- Elimine el `MirrorRelationship` en la RVP secundaria. Esto interrumpe la relación de replicación.
- O bien, actualice el campo `spec.state` a *Promoted*.

Eliminar una RVP (que se había duplicado previamente)

Astra Control Provisioning comprueba si existen las RVP replicadas y libera la relación de replicación antes de intentar eliminar el volumen.

Eliminar un TMR

Al eliminar un TMR en un lado de una relación reflejada, el TMR restante pasará al estado *Promoted* antes de que Astra Control Provisioner complete la eliminación. Si el TMR seleccionado para eliminación ya se encuentra en el estado *Promoted*, no existe ninguna relación de reflejo y el TMR se eliminará y el proveedor de Astra Control promoverá la RVP local a *ReadWrite*. Esta eliminación libera los metadatos de SnapMirror del volumen local en ONTAP. Si este volumen se utiliza en una relación de reflejo en el futuro, debe utilizar un nuevo TMR con un estado de replicación de volumen *established* al crear la nueva relación de reflejo.

Actualice las relaciones de reflejo cuando el ONTAP esté en línea

Las relaciones de reflejos se pueden actualizar en cualquier momento una vez establecidas. Puede utilizar los `state: promoted` campos o `state: reestablished` para actualizar las relaciones. Al promocionar un volumen de destino a un volumen de *ReadWrite* normal, se puede usar *promotedSnapshotHandle* para especificar una snapshot específica a la que restaurar el volumen actual.

Actualice las relaciones de reflejo cuando la ONTAP esté sin conexión

Puede utilizar un CRD para realizar una actualización de SnapMirror sin Astra Control para tener conectividad directa con el clúster de ONTAP. Consulte el siguiente formato de ejemplo 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*.

Habilita el proveedor de Astra Control

Las versiones 23,10 y posteriores de Trident incluyen la opción de usar Astra Control Provisioning, que permite a los usuarios de Astra Control con licencia acceder a funcionalidades avanzadas de aprovisionamiento del almacenamiento. El proveedor Astra Control ofrece esta funcionalidad ampliada, además de la funcionalidad estándar basada en CSI de Astra Trident. Puedes usar este procedimiento para habilitar e instalar el proveedor de Astra Control.

Tu suscripción al servicio de Astra Control incluye automáticamente la licencia para el uso del proveedor de Astra Control.

En las próximas actualizaciones de Astra Control, el proveedor de Astra Control reemplazará a Astra Trident como proveedor de almacenamiento y orquestador y será obligatorio para su uso en Astra Control. Por este motivo, se recomienda encarecidamente que los usuarios de Astra Control habiliten el proveedor de Astra Control. Astra Trident seguirá siendo de código abierto y se seguirá lanzando, manteniendo, admitiendo y actualizando con las nuevas funciones CSI y otras de NetApp.

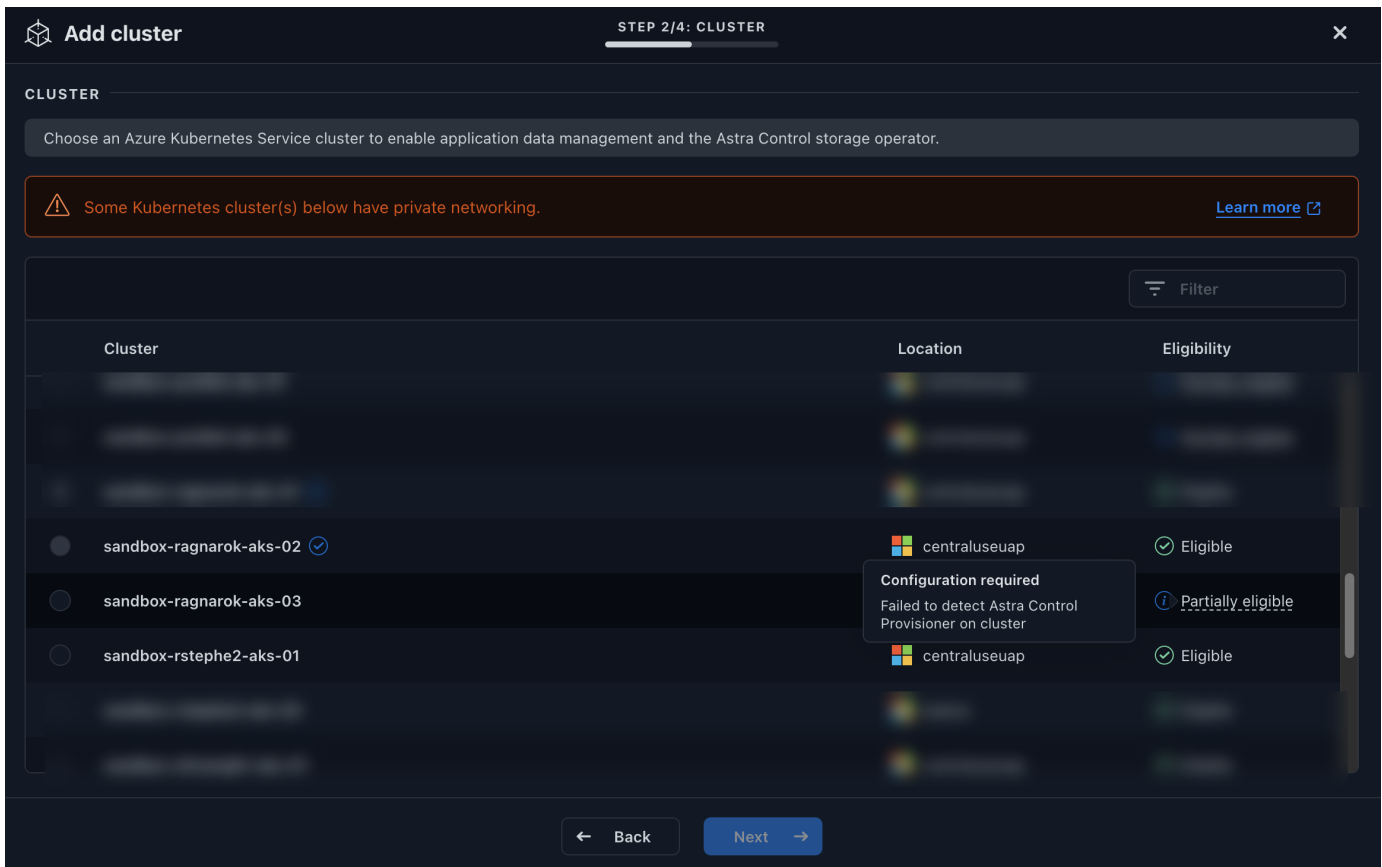
¿Cómo puedo saber si debo habilitar Astra Control Provisioner?

Si agrega un clúster a Astra Control Service que no tiene Astra Trident instalado previamente, el clúster se marcará como `Eligible`. Una vez que usted ["Añada el clúster a Astra Control"](#), Astra Control Provisioner se habilitará automáticamente.

Si su clúster no está marcado `Eligible`, se marcará `Partially eligible` debido a una de las siguientes acciones:

- Está usando una versión anterior de Astra Trident
- Se utiliza un Astra Trident 23,10 que aún no tiene habilitada la opción de proveedor
- Se trata de un tipo de clúster que no permite la habilitación automática

En `Partially eligible` casos, sigue estas instrucciones para habilitar manualmente el proveedor de control Astra para tu clúster.



Antes de habilitar Astra Control Provisioner

Si ya tienes un Astra Trident sin el aprovisionador de Astra Control y quieres habilitar el aprovisionador de Astra Control, haz lo siguiente primero:

- **Si tienes Astra Trident instalado, confirma que su versión está dentro de una ventana de cuatro versiones:** Puedes realizar una actualización directa a Astra Trident 24,02 con el aprovisionador de control de Astra si tu Astra Trident está dentro de una ventana de cuatro versiones de la versión 24,02. Por ejemplo, puedes actualizar directamente de Astra Trident 23,04 a 24,02.
- **Confirme que su clúster tiene una arquitectura de sistema AMD64:** La imagen del aprovisionador de Astra Control se proporciona en las arquitecturas de CPU AMD64 y ARM64, pero solo AMD64 es compatible con Astra Control.

Pasos

1. Acceda al registro de imágenes de Astra Control de NetApp:
 - a. Inicia sesión en la interfaz de usuario de Astra Control Service y registra tu ID de cuenta de Astra Control.
 - i. Seleccione el icono de figura en la parte superior derecha de la página.
 - ii. Seleccione **acceso API**.
 - iii. Escriba su ID de cuenta.
 - b. En la misma página, selecciona **Generar token de API** y copia la cadena de token de API en el portapapeles y guárdalo en tu editor.
 - c. Inicia sesión en el registro de Astra Control usando el método que prefieras:

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

```
crane auth login cr.astra.netapp.io -u <account-id> -p <api-token>
```

2. (Solo registros personalizados) Siga estos pasos para mover la imagen a su registro personalizado. Si no está utilizando un registro, siga los pasos del operador Trident en el [siguiente sección](#).



Puede usar Podman en lugar de Docker para los siguientes comandos. Si se utiliza un entorno de Windows, se recomienda PowerShell.

Docker

- a. Extrae la imagen del proveedor de Astra Control del registro:



La imagen extraída no soportará múltiples plataformas y solo soportará la misma plataforma que el host que sacó la imagen, como Linux AMD64.

```
docker pull cr.astra.netapp.io/astra/trident-acp:24.02.0
--platform <cluster platform>
```

Ejemplo:

```
docker pull cr.astra.netapp.io/astra/trident-acp:24.02.0
--platform linux/amd64
```

- b. Etiqueta la imagen:

```
docker tag cr.astra.netapp.io/astra/trident-acp:24.02.0
<my_custom_registry>/trident-acp:24.02.0
```

- c. Introduzca la imagen en el registro personalizado:

```
docker push <my_custom_registry>/trident-acp:24.02.0
```

Grúa

- a. Copie el manifiesto de Astra Control Provisioner en su registro personalizado:

```
crane copy cr.astra.netapp.io/astra/trident-acp:24.02.0
<my_custom_registry>/trident-acp:24.02.0
```

3. Determine si el método de instalación original de Astra Trident utilizó un.
4. Habilita el aprovisionamiento de Astra Control en Astra Trident con el método de instalación que solías originalmente:

Operador Astra Trident

- a. "Descarga el instalador de Astra Trident y extraígalo".
- b. Complete estos pasos si todavía no ha instalado Astra Trident o si ha quitado el operador de la implementación original de Astra Trident:
 - i. Cree el CRD:

```
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.1
6.yaml
```

- ii. Cree el espacio de nombres trident (`kubectl create namespace trident`) o confirme que el espacio de nombres trident aún existe (`kubectl get all -n trident`). Si el espacio de nombres se ha eliminado, vuelva a crearlo.

- c. Actualice Astra Trident a 24.06.0:



Para los clústeres que ejecutan Kubernetes 1,24 o una versión anterior, `bundle_pre_1_25.yaml` utilice . Para los clústeres que ejecutan Kubernetes 1,25 o posterior, utilice `bundle_post_1_25.yaml`.

```
kubectl -n trident apply -f trident-installer/deploy/<bundle-
name.yaml>
```

- d. Compruebe que Astra Trident está ejecutando:

```
kubectl get torc -n trident
```

Respuesta:

NAME	AGE
trident	21m

- e. Si tienes un registro que usa secretos, crea un secreto para extraer la imagen del aprovisionador de Astra Control:

```
kubectl create secret docker-registry <secret_name> -n trident
--docker-server=<my_custom_registry> --docker-username=<username>
--docker-password=<token>
```

- f. Edite el CR de TridentOrchestrator y realice las siguientes modificaciones:

```
kubectl edit torc trident -n trident
```

- i. Establezca una ubicación de registro personalizada para la imagen de Astra Trident o extráigala del registro de Astra Control (tridentImage:
<my_custom_registry>/trident:24.02.0 o tridentImage:
netapp/trident:24.06.0).
- ii. Habilite Astra Control Provisioner (enableACP: true).
- iii. Establezca la ubicación de registro personalizada para la imagen del aprovisionador de Astra Control o extráigala del registro de Astra Control (acpImage:
<my_custom_registry>/trident-acp:24.02.0 o acpImage:
cr.astra.netapp.io/astra/trident-acp:24.02.0).
- iv. Si estableció [la imagen descubre los secretos](#) anteriormente en este procedimiento, puede establecerlos aquí (imagePullSecrets: - <secret_name>). Utilice el mismo nombre secreto que estableció en los pasos anteriores.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  tridentImage: <registry>/trident:24.06.0
  enableACP: true
  acpImage: <registry>/trident-acp:24.06.0
  imagePullSecrets:
    - <secret_name>
```

- g. Guarde y salga del archivo. El proceso de despliegue comenzará automáticamente.
- h. Compruebe que se han creado el operador, el despliegue y los replicaset.

```
kubectl get all -n trident
```



Solo debe haber **una instancia** del operador en un clúster de Kubernetes. No cree varias implementaciones del operador Trident de Astra.

- i. Compruebe que trident-acp el contenedor se está ejecutando y que acpVersion tiene 24.02.0 un estado de Installed:

```
kubectl get torc -o yaml
```

Respuesta:

```

status:
  acpVersion: 24.02.0
  currentInstallationParams:
    ...
    acpImage: <registry>/trident-acp:24.02.0
    enableACP: "true"
    ...
  ...
status: Installed

```

tridentctl

- "Descarga el instalador de Astra Trident y extraígalo".
- "Si ya tiene un Astra Trident existente, desinstálelo del clúster que lo aloja".
- Instalar Astra Trident con el aprovisionador de control de Astra habilitado (`--enable-acp=true`):

```

./tridentctl -n trident install --enable-acp=true --acp
-image=mycustomregistry/trident-acp:24.02

```

- Confirme que se ha habilitado el aprovisionador de Astra Control:

```

./tridentctl -n trident version

```

Respuesta:

```

+-----+-----+-----+ | SERVER
VERSION | CLIENT VERSION | ACP VERSION | +-----+
+-----+-----+-----+ | 24.02.0 | 24.02.0 | 24.02.0. |
+-----+-----+-----+

```

Timón

- Si tienes Astra Trident 23.07.1 o anterior instalado, "[desinstalar](#)" el operador y otros componentes.
- Si tu clúster de Kubernetes ejecuta la versión 1,24 o anterior, elimina psp:

```

kubectl delete psp tridentoperatorpod

```

- Añada el repositorio de Astra Trident Helm:

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

d. Actualice el gráfico Helm:

```
helm repo update netapp-trident
```

Respuesta:

```
Hang tight while we grab the latest from your chart
repositories...
...Successfully got an update from the "netapp-trident" chart
repository
Update Complete. ☐Happy Helming!☐
```

e. Enumere las imágenes:

```
./tridentctl images -n trident
```

Respuesta:

```
| v1.28.0          | netapp/trident:24.06.0|
|                  | docker.io/netapp/trident-
autosupport:24.06|
|                  | registry.k8s.io/sig-storage/csi-
provisioner:v4.0.0|
|                  | registry.k8s.io/sig-storage/csi-
attacher:v4.5.0|
|                  | registry.k8s.io/sig-storage/csi-
resizer:v1.9.3|
|                  | registry.k8s.io/sig-storage/csi-
snapshotter:v6.3.3|
|                  | registry.k8s.io/sig-storage/csi-node-
driver-registrar:v2.10.0 |
|                  | netapp/trident-operator:24.06.0 (optional)
```

f. Asegúrese de que el trident-operator 24.06.0 esté disponible:

```
helm search repo netapp-trident/trident-operator --versions
```

Respuesta:

NAME	CHART VERSION	APP VERSION	
DESCRIPTION			
netapp-trident/trident-operator	100.2406.0	24.06.0	A

g. Utilice `helm install` y ejecute una de las siguientes opciones que incluyen estos valores:

- Un nombre para la ubicación de despliegue
- La versión de Trident de Astra
- El nombre de la imagen del aprovisionador de Astra Control
- La marca para habilitar el aprovisionador
- (Opcional) Una ruta de registro local. Si está utilizando un registro local, "[Imágenes de Trident](#)" puede estar ubicado en un registro o registros diferentes, pero todas las imágenes CSI deben estar ubicadas en el mismo registro.
- El espacio de nombres de Trident

Opciones

- Imágenes sin registro

```
helm install trident netapp-trident/trident-operator --version
100.2402.0 --set acpImage=cr.astra.netapp.io/astra/trident-
acp:24.06.0 --set enableACP=true --set operatorImage=netapp/trident-
operator:24.06.0 --set
tridentAutosupportImage=docker.io/netapp/trident-autosupport:24.06
--set tridentImage=netapp/trident:24.06.0 --namespace trident
```

- Imágenes en uno o más registros

```
helm install trident netapp-trident/trident-operator --version
100.2402.0 --set acpImage=<your-registry>:<acp image> --set
enableACP=true --set imageRegistry=<your-registry>/sig-storage --set
operatorImage=netapp/trident-operator:24.06.0 --set
tridentAutosupportImage=docker.io/netapp/trident-autosupport:24.06
--set tridentImage=netapp/trident:24.06.0 --namespace trident
```

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

Si tiene problemas para poner en marcha Trident mediante Helm, ejecute este comando para desinstalar completamente Astra Trident:

```
./tridentctl uninstall -n trident
```

No ["Elimina por completo los CRD de Astra Trident"](#) como parte de tu desinstalación antes de intentar habilitar Astra Control Provisionador de nuevo.

Resultado

Está habilitada la funcionalidad de aprovisionamiento de Astra Control y es posible usar cualquier función disponible para la versión que esté ejecutando.

Después de instalar el aprovisionador de Astra Control, el clúster que aloja el aprovisionador en la interfaz de usuario de Astra Control mostrará un `ACP version` número de versión instalada actual y un campo en lugar de `Trident version`.

CLUSTER STATUS

Available

Version v1.24.9+rke2r2	Managed 2024/03/15 17:32 UTC	Kube-system namespace UID <div></div>	ACP Version <div></div>
Private route identifier <div>...</div>	Cloud instance private	Default bucket astra-bucket1 (inherited)	

Overview

Namespaces

Storage

Activity

Si quiere más información

- ["Documentación sobre actualizaciones de Astra Trident"](#)

Utilice Topología CSI

Astra Trident puede crear y conectar volúmenes a los nodos que están presentes en un clúster de Kubernetes de forma selectiva utilizando el ["Función de topología CSI"](#).

Descripción general

Con la función de topología CSI, el acceso a los volúmenes puede limitarse a un subconjunto de nodos, en función de regiones y zonas de disponibilidad. En la actualidad, los proveedores de cloud permiten a los administradores de Kubernetes generar nodos basados en zonas. Los nodos se pueden ubicar 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 de varias zonas, Astra Trident utiliza la topología CSI.



Obtenga más información sobre la función Topología de CSI ["aquí"](#).

Kubernetes ofrece dos modos de enlace de volúmenes únicos:

- Con `VolumeBindingMode` la opción establecida en `Immediate`, Astra Trident crea el volumen sin tener en cuenta la topología. La vinculación de volúmenes y el aprovisionamiento dinámico se manejan cuando se crea la RVP. Este es el valor por defecto `VolumeBindingMode` y es adecuado para clusters que no aplican restricciones de topología. Los volúmenes persistentes se crean sin depender de los requisitos de programación del pod solicitante.

- Con `VolumeBindingMode` establecido en `WaitForFirstConsumer`, la creación y vinculación de un volumen persistente para una RVP se retrasa hasta que se programe y cree un pod que utilice la RVP. De esta forma, se crean volúmenes con el fin de cumplir las restricciones de programación que se aplican en los requisitos de topología.



El modo de enlace `WaitForFirstConsumer` no requiere etiquetas de topología. Esto se puede utilizar independientemente de la característica de topología CSI.

Lo que necesitará

Para utilizar la topología CSI, necesita lo siguiente:

- Un clúster de Kubernetes que ejecuta un ["Compatible con la versión de Kubernetes"](#)

```
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 introduzcan el reconocimiento de topología (`topology.kubernetes.io/region`y`topology.kubernetes.io/zone`). Estas etiquetas * deben estar presentes en los nodos del clúster* antes de instalar Astra Trident para que Astra Trident tenga en cuenta la topología.


```
kubectl get nodes -o=jsonpath='{range .items[*]}[{.metadata.name},
{.metadata.labels}]{"\n"}{end}' | grep --color "topology.kubernetes.io"
[nodel1,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kubernetes.io/arch":"amd64","kubernetes.io/hostname":"nodel1","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","kubernetes.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","kubernetes.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: Cree un backend con detección de topología

Los back-ends de almacenamiento de Astra Trident se pueden diseñar para aprovisionar de forma selectiva volúmenes en función de las zonas de disponibilidad. Cada backend puede llevar un bloque opcional `supportedTopologies` que representa una lista de zonas y regiones soportadas. En el caso de `StorageClasses` que utilizan dicho back-end, solo se creará un volumen si lo solicita una aplicación programada en una región/zona admitida.

A continuación se muestra un ejemplo de definición de backend:

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

```
{
  "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 un `StorageClass`. En el caso de `StorageClasses` que contienen un subconjunto de las regiones y zonas proporcionadas en un back-end, Astra Trident creará un volumen en el back-end.

También puede definir `supportedTopologies` por pool de almacenamiento. Consulte 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, `region` las etiquetas y `zone` representan la ubicación del pool de almacenamiento. `topology.kubernetes.io/region` `topology.kubernetes.io/zone` y dicte dónde se pueden consumir los pools de almacenamiento.

Paso 2: Defina las clases de almacenamiento que tienen en cuenta la topología

En función de las etiquetas de topología que se proporcionan a los nodos del clúster, se puede definir `StorageClasse` para que contenga información de topología. Esto determinará los pools de almacenamiento que sirven como candidatos para las solicitudes de RVP y el subconjunto de nodos que pueden usar los volúmenes aprovisionados mediante Trident.

Consulte el siguiente ejemplo:

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: netapp-san-us-east1
provisioner: csi.trident.netapp.io
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
- matchLabelExpressions:
- 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` se establece en `WaitForFirstConsumer`. Las RVP solicitadas con este tipo de almacenamiento no se verán en cuestión hasta que se mencionan en un pod. Y, `allowedTopologies` proporciona las zonas y la región que se van a utilizar. `netapp-san-us-east1` StorageClass creará EVs en el `san-backend-us-east1` backend definido anteriormente.

Paso 3: Cree y utilice un PVC

Con el clase de almacenamiento creado y asignado a un back-end, ahora puede crear RVP.

Vea el ejemplo spec a continuación:

```

---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-san
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 300Mi
  storageClassName: netapp-san-us-east1

```

La creación de una RVP con este manifiesto daría como resultado lo siguiente:

```

kubect1 create -f pvc.yaml
persistentvolumeclaim/pvc-san created
kubect1 get pvc
NAME          STATUS      VOLUME      CAPACITY    ACCESS MODES    STORAGECLASS
AGE
pvc-san      Pending                                netapp-san-us-east1
2s
kubect1 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 enlace a la RVP, use la RVP en un pod. Consulte 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

```

Este podSpec indica a Kubernetes que programe el pod en los nodos que están presentes en us-east1 la región y que elija entre cualquier nodo que esté presente en las us-east1-a zonas o. us-east1-b

Consulte la siguiente salida:

```
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
```

Actualice los back-ends que se van a incluir `supportedTopologies`

Los back-ends preexistentes se pueden actualizar para incluir una lista de `supportedTopologies` uso `tridentctl backend update`. Esto no afectará a los volúmenes que ya se han aprovisionado, y sólo se utilizarán en las siguientes CVP.

Obtenga más información

- ["Gestione recursos para contenedores"](#)
- ["Selector de nodos"](#)
- ["Afinidad y anti-afinidad"](#)
- ["Tolerancias y taints"](#)

Trabajar con instantáneas

Las snapshots de volúmenes de Kubernetes de Persistent Volumes (VP) permiten copias puntuales de volúmenes. Es posible crear una copia Snapshot de un volumen creado con Astra Trident, importar una copia de Snapshot creada fuera de Astra Trident, crear un volumen nuevo a partir de una copia de Snapshot existente y recuperar datos de volumen de copias Snapshot.

Descripción general

La instantánea de volumen es compatible con `ontap-nas`, `ontap-nas-flexgroup`, `ontap-san`, `ontap-san-economy`, `solidfire-san`, `gcp-cvs`, y `azure-netapp-files` los conductores.

Antes de empezar

Debe tener un controlador de instantánea externo y definiciones de recursos personalizados (CRD) para trabajar con instantáneas. Esta es la responsabilidad del orquestador de Kubernetes (por ejemplo: Kubeadm, GKE, OpenShift).

Si su distribución de Kubernetes no incluye el controlador de instantáneas y los CRD, consulte [Implemente una controladora Snapshot de volumen](#).



No cree una controladora Snapshot si crea instantáneas de volumen bajo demanda en un entorno de GKE. GKE utiliza un controlador de instantáneas oculto integrado.

Cree una copia de Snapshot de volumen

Pasos

1. Cree un VolumeSnapshotClass. Para obtener más información, consulte "[VolumeSnapshotClass](#)".
 - `driver` Los puntos al controlador CSI de Astra Trident.
 - `deletionPolicy` puede ser `Delete` o `Retain` Cuando se establece en `Retain`, la instantánea física subyacente del clúster de almacenamiento se conserva incluso si se elimina el `VolumeSnapshot` objeto.

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. Crear una instantánea de una RVP existente.

Ejemplos

- En este ejemplo, se crea una copia Snapshot de una RVP existente.

```
cat snap.yaml
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: pvc1-snap
spec:
  volumeSnapshotClassName: csi-snapclass
  source:
    persistentVolumeClaimName: pvc1
```

- En este ejemplo, se crea un objeto Snapshot de volumen para una RVP denominada `pvc1` y el nombre de la Snapshot se establece en `pvc1-snap`. Una Snapshot de volumen es similar a una RVP y se asocia con `VolumeSnapshotContent` un objeto que representa la snapshot real.

```
kubectl create -f snap.yaml
volumesnapshot.snapshot.storage.k8s.io/pvc1-snap created

kubectl get volumesnapshots
NAME                AGE
pvc1-snap           50s
```


- Es posible identificar `VolumeSnapshotContent` el objeto de `pvc1-snap` la Snapshot de volumen describiéndolo. El `Snapshot Content Name` identifica el objeto `VolumeSnapshotContent` que sirve para esta snapshot. `Ready To Use``El parámetro indica que la snapshot se puede usar para crear una nueva RVP.

```
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 una RVP a partir de una snapshot de volumen

Puede usar `dataSource` para crear una RVP con una `VolumeSnapshot` llamada `<pvc-name>` como origen de los datos. Una vez creada la RVP, se puede conectar a un pod y utilizarla como cualquier otro PVC.



La RVP se creará en el mismo back-end que el volumen de origen. Consulte "[KB: La creación de una RVP a partir de una snapshot de RVP de Trident no se puede crear en un back-end alternativo](#)".

En el siguiente ejemplo se crea la RVP utilizando `pvc1-snap` como origen 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: pvcl-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io

```

Importe una copia de Snapshot de volumen

Astra Trident admite "[Proceso de snapshot aprovisionado previamente de Kubernetes](#)" para permitir al administrador de clúster crear VolumeSnapshotContent un objeto e importar copias de Snapshot creadas fuera de Astra Trident.

Antes de empezar

Astra Trident debe haber creado o importado el volumen principal del snapshot.

Pasos

1. **Cluster admin:** Crear un VolumeSnapshotContent objeto que haga referencia a la instantánea backend. Esto inicia el flujo de trabajo de las copias Snapshot en Astra Trident.
 - Especifique el nombre de la instantánea de 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 Astra Trident por el Snapshotter externo en la `ListSnapshots` llamada.



`<volumeSnapshotContentName>` No siempre puede coincidir con el nombre de instantánea de backend debido a restricciones de nomenclatura de CR.

Ejemplo

En el siguiente ejemplo se crea un VolumeSnapshotContent objeto que hace referencia a la instantánea 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>

```

2. **Cluster admin:** Crear el VolumeSnapshot CR que hace referencia al VolumeSnapshotContent objeto. Esto solicita acceso para utilizar VolumeSnapshot en un espacio de nombres determinado.

Ejemplo

En el siguiente ejemplo se crea una VolumeSnapshot CR denominada import-snap que hace referencia a la VolumeSnapshotContent 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 acción):** El Snapshotter externo reconoce el recién creado VolumeSnapshotContent y ejecuta la ListSnapshots llamada. Astra Trident crea el TridentSnapshot.
 - El dispositivo de instantáneas externo establece el VolumeSnapshotContent en readyToUse y el VolumeSnapshot en true.
 - Trident devuelve readyToUse=true.
4. **Cualquier usuario:** Crear un PersistentVolumeClaim para hacer referencia al nuevo VolumeSnapshot, donde el spec.dataSource nombre (o spec.dataSourceRef) es el nombre VolumeSnapshot.

Ejemplo

En el siguiente ejemplo se crea una RVP que hace referencia a la VolumeSnapshot llamada 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 copias Snapshot

El directorio de snapshots está oculto de forma predeterminada para facilitar la máxima compatibilidad de los volúmenes aprovisionados mediante los `ontap-nas` controladores y `ontap-nas-economy`. Permita que `.snapshot` el directorio recupere datos de snapshots directamente.

Use la interfaz de línea de comandos de ONTAP para restaurar un volumen en un estado registrado en una snapshot anterior.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```



Cuando se restaura una copia Snapshot, se sobrescribe la configuración de volúmenes existente. Se pierden los cambios que se hagan en los datos del volumen después de crear la copia Snapshot.

El directorio de snapshots está oculto de forma predeterminada para facilitar la máxima compatibilidad de los volúmenes aprovisionados mediante los `ontap-nas` controladores y `ontap-nas-economy`. Permita que `.snapshot` el directorio recupere datos de snapshots directamente.



Cuando se restaura una copia Snapshot, se sobrescribe la configuración de volúmenes existente. Se pierden los cambios que se hagan en los datos del volumen después de crear la copia Snapshot.

Restauración de volumen sin movimiento a partir de una copia de Snapshot

Astra Control Provisioning ofrece una restauración de volumen rápida y in situ a partir de una copia Snapshot mediante `TridentActionSnapshotRestore` CR (TASR). Esta CR funciona como una acción

imprescindible de Kubernetes y no persiste una vez que finaliza la operación.

Astra Control Provisioner admite la restauración de instantáneas en el `ontap-san`, `ontap-san-economy`, `ontap-nas`, `ontap-nas-flexgroup`, `azure-netapp-files`, `gcp-cvs`, y `solidfire-san` los conductores.

Antes de empezar

Debe tener una snapshot de volumen disponible y la RVP vinculada.

- Compruebe que el estado de la RVP es de enlace.

```
kubectl get pvc
```

- Compruebe que la copia de Snapshot de volumen esté lista para utilizarse.

```
kubectl get vs
```

Pasos

1. Cree el CR de TASR. En este ejemplo, se crea una CR para la RVP `pvc1` y una instantánea de volumen `pvc1-snapshot`.

```
cat tasr-pvc1-snapshot.yaml

apiVersion: v1
kind: TridentActionSnapshotRestore
metadata:
  name: this-doesnt-matter
  namespace: trident
spec:
  pvcName: pvc1
  volumeSnapshotName: pvc1-snapshot
```

2. Aplique el CR para restaurar a partir de la instantánea. Este ejemplo restaura desde la instantánea `pvc1`.

```
kubectl create -f tasr-pvc1-snapshot.yaml

tridentactionsnapshotrestore.trident.netapp.io/this-doesnt-matter
created
```

Resultados

El aprovisionador de Astra Control restaura los datos a partir de la snapshot. Es posible verificar el estado de restauración de la Snapshot.

```
kubectl get tasr -o yaml

apiVersion: v1
items:
- apiVersion: trident.netapp.io/v1
  kind: TridentActionSnapshotRestore
  metadata:
    creationTimestamp: "2023-04-14T00:20:33Z"
    generation: 3
    name: this-doesnt-matter
    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, el proveedor de Astra Control no volverá a intentar automáticamente la operación en caso de fallo. Deberá realizar la operación de nuevo.
- Es posible que el administrador deba conceder permiso al usuario de Kubernetes sin acceso de administrador para crear una CR TASR en su espacio de nombres de la aplicación.

Eliminar un VP con snapshots asociadas

Cuando se elimina un volumen persistente con instantáneas asociadas, el volumen Trident correspondiente se actualiza a un “estado de eliminación”. Quite las snapshots de volumen para eliminar el volumen de Astra Trident.

Implemente una controladora Snapshot de volumen

Si su distribución de Kubernetes no incluye el controlador de snapshots y los CRD, puede implementarlos de la siguiente manera.

Pasos

1. Crear CRD de snapshot 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.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshots.yaml
```

2. Cree la controladora Snapshot.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/deploy/kubernetes/snapshot-controller/setup-snapshot-controller.yaml
```



Si es necesario, abra `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` y actualice `namespace` su espacio de nombres.

Enlaces relacionados

- ["Copias de Snapshot de volumen"](#)
- ["VolumeSnapshotClass"](#)

Información de copyright

Copyright © 2026 NetApp, Inc. Todos los derechos reservados. Imprimido en EE. UU. No se puede reproducir este documento protegido por copyright ni parte del mismo de ninguna forma ni por ningún medio (gráfico, electrónico o mecánico, incluidas fotocopias, grabaciones o almacenamiento en un sistema de recuperación electrónico) sin la autorización previa y por escrito del propietario del copyright.

El software derivado del material de NetApp con copyright está sujeto a la siguiente licencia y exención de responsabilidad:

ESTE SOFTWARE LO PROPORCIONA NETAPP «TAL CUAL» Y SIN NINGUNA GARANTÍA EXPRESA O IMPLÍCITA, INCLUYENDO, SIN LIMITAR, LAS GARANTÍAS IMPLÍCITAS DE COMERCIALIZACIÓN O IDONEIDAD PARA UN FIN CONCRETO, CUYA RESPONSABILIDAD QUEDA EXIMIDA POR EL PRESENTE DOCUMENTO. EN NINGÚN CASO NETAPP SERÁ RESPONSABLE DE NINGÚN DAÑO DIRECTO, INDIRECTO, ESPECIAL, EJEMPLAR O RESULTANTE (INCLUYENDO, ENTRE OTROS, LA OBTENCIÓN DE BIENES O SERVICIOS SUSTITUTIVOS, PÉRDIDA DE USO, DE DATOS O DE BENEFICIOS, O INTERRUPCIÓN DE LA ACTIVIDAD EMPRESARIAL) CUALQUIERA SEA EL MODO EN EL QUE SE PRODUJERON Y LA TEORÍA DE RESPONSABILIDAD QUE SE APLIQUE, YA SEA EN CONTRATO, RESPONSABILIDAD OBJETIVA O AGRAVIO (INCLUIDA LA NEGLIGENCIA U OTRO TIPO), QUE SURJAN DE ALGÚN MODO DEL USO DE ESTE SOFTWARE, INCLUSO SI HUBIEREN SIDO ADVERTIDOS DE LA POSIBILIDAD DE TALES DAÑOS.

NetApp se reserva el derecho de modificar cualquiera de los productos aquí descritos en cualquier momento y sin aviso previo. NetApp no asume ningún tipo de responsabilidad que surja del uso de los productos aquí descritos, excepto aquello expresamente acordado por escrito por parte de NetApp. El uso o adquisición de este producto no lleva implícita ninguna licencia con derechos de patente, de marcas comerciales o cualquier otro derecho de propiedad intelectual de NetApp.

Es posible que el producto que se describe en este manual esté protegido por una o más patentes de EE. UU., patentes extranjeras o solicitudes pendientes.

LEYENDA DE DERECHOS LIMITADOS: el uso, la copia o la divulgación por parte del gobierno están sujetos a las restricciones establecidas en el subpárrafo (b)(3) de los derechos de datos técnicos y productos no comerciales de DFARS 252.227-7013 (FEB de 2014) y FAR 52.227-19 (DIC de 2007).

Los datos aquí contenidos pertenecen a un producto comercial o servicio comercial (como se define en FAR 2.101) y son propiedad de NetApp, Inc. Todos los datos técnicos y el software informático de NetApp que se proporcionan en este Acuerdo tienen una naturaleza comercial y se han desarrollado exclusivamente con fondos privados. El Gobierno de EE. UU. tiene una licencia limitada, irrevocable, no exclusiva, no transferible, no sublicenciable y de alcance mundial para utilizar los Datos en relación con el contrato del Gobierno de los Estados Unidos bajo el cual se proporcionaron los Datos. Excepto que aquí se disponga lo contrario, los Datos no se pueden utilizar, desvelar, reproducir, modificar, interpretar o mostrar sin la previa aprobación por escrito de NetApp, Inc. Los derechos de licencia del Gobierno de los Estados Unidos de América y su Departamento de Defensa se limitan a los derechos identificados en la cláusula 252.227-7015(b) de la sección DFARS (FEB de 2014).

Información de la marca comercial

NETAPP, el logotipo de NETAPP y las marcas que constan en <http://www.netapp.com/TM> son marcas comerciales de NetApp, Inc. El resto de nombres de empresa y de producto pueden ser marcas comerciales de sus respectivos propietarios.