Referencia

Trident

NetApp
September 26, 2025

This PDF was generated from https://docs.netapp.com/es-es/trident-2410/trident-reference/ports.html on September 26, 2025. Always check docs.netapp.com for the latest.

Tabla de contenidos

Referencia . . . 1
- Puertos Trident . . . 1
  - Puertos Trident . . . 1
- API de REST de Trident . . . 1
  - Cuándo utilizar la API DE REST . . . 1
  - Uso de la API DE REST . . . 1
- Opciones de línea de comandos . . . 2
  - Registro . . . 2
  - Kubernetes . . . 2
  - Docker . . . 3
  - DESCANSO . . . 3
- Objetos de Kubernetes y Trident . . . 3
  - ¿Cómo interactúan los objetos entre sí? . . . 3
  - `PersistentVolumeClaim`Objetos de Kubernetes . . . 4
  - `PersistentVolume`Objetos de Kubernetes . . . 6
  - `StorageClass`Objetos de Kubernetes . . . 6
  - `VolumeSnapshotClass`Objetos de Kubernetes . . . 10
  - `VolumeSnapshot`Objetos de Kubernetes . . . 11
  - `VolumeSnapshotContent`Objetos de Kubernetes . . . 11
  - `CustomResourceDefinition`Objetos de Kubernetes . . . 11
  - Objetos deTrident `StorageClass` . . . 12
  - Objetos de back-end de Trident . . . 12
  - Objetos deTrident `StoragePool` . . . 12
  - Objetos deTrident `Volume` . . . 13
  - Objetos deTrident `Snapshot` . . . 14
  - ObjetoTrident `ResourceQuota` . . . 15
- Pod Security Standards (PSS) y las restricciones de contexto de seguridad (SCC) . . . 16
  - Contexto de Kubernetes Security y campos relacionados necesarios . . . 16
  - Estándares de seguridad para POD (PSS) . . . 17
  - Directivas de seguridad de POD (PSP) . . . 17
  - Restricciones de contexto de seguridad (SCC) . . . 19

Referencia

Puertos Trident

Obtenga más información sobre los puertos que Trident utiliza para la comunicación.

Puertos Trident

Trident se comunica a través de los siguientes puertos:

Puerto	Específico
8443	HTTPS de canal posterior
8001	Extremo de métricas de Prometheus
8000	Servidor REST de Trident
17546	Puerto de sonda de presencia/preparación utilizado por los pods demonset de Trident

> El puerto de la sonda de vida/disponibilidad se puede cambiar durante la instalación con el `--probe-port` indicador. Es importante asegurarse de que este puerto no esté siendo utilizado por otro proceso en los nodos de trabajo.

API de REST de Trident

Si bien "comandos y opciones de trimentctl" es la forma más sencilla de interactuar con la API DE REST DE Trident, puede usar el extremo de REST directamente, si lo prefiere.

Cuándo utilizar la API DE REST

La API de REST es útil para instalaciones avanzadas que usan Trident como binario independiente en puestas en marcha que no son de Kubernetes.

Para una mayor seguridad, Trident `REST API` está restringido a localhost de forma predeterminada cuando se ejecuta dentro de un pod. Para cambiar este comportamiento, debe establecer el argumento de Trident `-address` en su configuración de pod.

Uso de la API DE REST

Para ver ejemplos de cómo se llaman a estas API, pase (`-d`el indicador DEBUG). Para obtener más información, consulte "Gestione Trident con tridentctl".

La API funciona de la siguiente manera:

OBTENGA

`GET <trident-address>/trident/v1/<object-type>`

Muestra todos los objetos de ese tipo.

GET <trident-address>/trident/v1/<object-type>/<object-name>

Obtiene los detalles del objeto con nombre.

PUBLICAR

POST <trident-address>/trident/v1/<object-type>

Crea un objeto del tipo especificado.

- Requiere la configuración de JSON para el objeto que se cree. Para conocer la especificación de cada tipo de objeto, consulte "Gestione Trident con tridentctl".
- Si el objeto ya existe, el comportamiento varía: Los back-ends actualizan el objeto existente, mientras que todos los demás tipos de objeto fallarán la operación.

ELIMINAR

DELETE <trident-address>/trident/v1/<object-type>/<object-name>

Suprime el recurso con nombre.

> Seguirán existiendo volúmenes asociados con back-ends o clases de almacenamiento, que deben eliminarse por separado. Para obtener más información, consulte "Gestione Trident con tridentctl".

Opciones de línea de comandos

Trident expone varias opciones de línea de comandos para Trident orchestrator. Puede usar estas opciones para modificar la implementación.

Registro

-debug

Activa la salida de depuración.

-loglevel <level>

Establece el nivel de registro (debug, info, warn, error, fatal). Por defecto es info.

Kubernetes

-k8s_pod

Utilice esta opción o `-k8s_api_server` para habilitar el soporte de Kubernetes. Al configurar esto, Trident usa las credenciales de cuenta del servicio de Kubernetes del pod para contactar con el servidor de API. Esto solo funciona cuando Trident se ejecuta como un pod en un clúster de Kubernetes con cuentas de servicio habilitadas.

-k8s_api_server <insecure-address:insecure-port>

Utilice esta opción o `-k8s_pod` para habilitar el soporte de Kubernetes. Cuando se especifica, Trident se conecta al servidor API de Kubernetes mediante el puerto y la dirección no seguras que se proporcionan. Esto permite que Trident se implemente fuera de un pod; sin embargo, solo admite conexiones no seguras con el servidor API. Para conectarse de forma segura, implemente Trident en un pod con la `-k8s_pod` opción.

Docker

`-volume_driver <name>`

Nombre del controlador utilizado al registrar el plugin de Docker. El valor por defecto es `netapp`.

`-driver_port <port-number>`

Reciba en este puerto en lugar de en un socket de dominio UNIX.

`-config <file>`

Necesario; debe especificar esta ruta de acceso a un archivo de configuración de backend.

DESCANSO

`-address <ip-or-host>`

Especifica la dirección en la que debe escuchar el servidor REST DE Trident. El valor predeterminado es localhost. Cuando se escucha en localhost y se ejecuta dentro de un pod Kubernetes, la interfaz REST no es accesible desde fuera del pod. Se utiliza `-address ""` para que la interfaz REST sea accesible desde la dirección IP del pod.

> La interfaz DE REST de Trident se puede configurar para escuchar y servir únicamente en 127.0.0.1 (para IPv4) o [::1] (para IPv6).

`-port <port-number>`

Especifica el puerto en el que debe recibir el servidor REST DE Trident. El valor predeterminado es 8000.

`-rest`

Habilita la interfaz DE REST. El valor predeterminado es TRUE.

Objetos de Kubernetes y Trident

Puede interactuar con Kubernetes y Trident mediante las API DE REST a través de la lectura y la escritura de objetos de recursos. Existen varios objetos de recursos que dictan la relación entre Kubernetes y Trident, Trident y el almacenamiento, y Kubernetes y el almacenamiento. Algunos de estos objetos se gestionan mediante Kubernetes y los demás se gestionan mediante Trident.

¿Cómo interactúan los objetos entre sí?

Quizás la forma más sencilla de comprender los objetos, qué hacen y cómo interactúan sea, es seguir una única solicitud de almacenamiento a un usuario de Kubernetes:

1. Un usuario crea un `PersistentVolumeClaim` pedido nuevo de `PersistentVolume` un tamaño concreto a partir de un Kubernetes `StorageClass` que había configurado previamente el administrador.
2. El Kubernetes `StorageClass` identifica a Trident como su aprovisionador e incluye parámetros que indican a Trident cómo aprovisionar un volumen para la clase solicitada.
3. Trident mira por sí mismo `StorageClass` con el mismo nombre que identifica la coincidencia `Backends` y `StoragePools` que puede utilizar para aprovisionar volúmenes para la clase.

4. Trident aprovisiona almacenamiento en un back-end coincidente y crea dos objetos: Un `PersistentVolume` en Kubernetes que indica a Kubernetes cómo encontrar, montar y tratar el volumen, así como un volumen en Trident que conserva la relación entre `PersistentVolume` el y el almacenamiento real.
5. Kubernetes enlaza los `PersistentVolumeClaim` a la nueva `PersistentVolume`. Pods que incluyen el `PersistentVolumeClaim` montaje de Volume persistente en cualquier host en el que se ejecute.
6. Un usuario crea `VolumeSnapshot` un de un RVP existente, utilizando un `VolumeSnapshotClass` que apunta a Trident.
7. Trident identifica el volumen asociado con la RVP y crea una copia Snapshot del volumen en su back-end. También crea un `VolumeSnapshotContent` que le indica a Kubernetes cómo identificar la snapshot.
8. Un usuario puede crear un `PersistentVolumeClaim` uso `VolumeSnapshot` como origen.
9. Trident identifica la instantánea necesaria y realiza el mismo juego de pasos involucrados en la creación de un `PersistentVolume` y `Volume` un .

Para obtener más información sobre los objetos de Kubernetes, le recomendamos que lea "Volúmenes persistentes" la sección de la documentación de Kubernetes.

`PersistentVolumeClaim`Objetos de Kubernetes

Un objeto de Kubernetes `PersistentVolumeClaim` es una solicitud de almacenamiento que realiza un usuario del clúster de Kubernetes.

Además de la especificación estándar, Trident permite a los usuarios especificar las siguientes anotaciones específicas del volumen si desean anular los valores predeterminados que se establecen en la configuración de back-end:

Anotación	Opción de volumen	Controladores compatibles
trident.netapp.io/fileSystem	Sistema de archivos	ontap-san, solidfire-san, ontap-san-economy
trident.netapp.io/cloneFromPVC	ClonSourceVolume	ontap-nas, ontap-san, solidfire-san, azure-netapp-files, gcp-cvs, ontap-san-economía
trident.netapp.io/splitOnClone	SplitOnClone	ontap-nas y ontap-san
trident.netapp.io/protocol	protocolo	cualquiera
trident.netapp.io/exportPolicy	Política de exportoPolicy	ontap-nas, ontap-nas-economy y ontap-nas-flexgroup
trident.netapp.io/snapshotPolicy	Política de copias Snapshot	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup y ontap-san
trident.netapp.io/snapshotReserve	Reserva de copias Snapshot	ontap-nas, ontap-nas-flexgroup, ontap-san, gcp-cvs
trident.netapp.io/snapshotDirectory	Snapshot shotDirectory	ontap-nas, ontap-nas-economy y ontap-nas-flexgroup
trident.netapp.io/unixPermissions	Permisos univalados	ontap-nas, ontap-nas-economy y ontap-nas-flexgroup

Anotación	Opción de volumen	Controladores compatibles
trident.netapp.io/blockSize	Tamaño del bloque	solidfire-san

Si el VP creado tiene la `Delete` política de reclamaciones, Trident elimina el VP y el volumen de respaldo cuando se libera el VP (es decir, cuando el usuario elimina la RVP). Si la acción de eliminación falla, Trident Marca el VP como tal y reintenta periódicamente la operación hasta que esta se complete o se elimine manualmente el VP. Si el VP usa la `Retain` política, Trident la ignora y asume que el administrador la limpiará de Kubernetes y del back-end para permitir que se haga un backup del volumen o se inspeccione antes de su eliminación. Tenga en cuenta que al eliminar el VP, Trident no eliminará el volumen de backup. Se debe quitar mediante la API de REST (`tridentctl`).

Trident admite la creación de instantáneas de volumen utilizando la especificación CSI: Puede crear una instantánea de volumen y utilizarla como origen de datos para clonar las RVP existentes. De este modo, las copias puntuales de VP pueden exponerse a Kubernetes en forma de snapshots. Las instantáneas pueden utilizarse para crear nuevos VP. Echa un vistazo `On-Demand Volume Snapshots` para ver cómo funcionaría esto.

Trident también proporciona `cloneFromPVC` las anotaciones y `splitOnClone` para crear clones. Puede utilizar estas anotaciones para clonar una RVP sin tener que utilizar la implementación de CSI.

Aquí hay un ejemplo: Si un usuario ya tiene un PVC llamado `mysql`, el usuario puede crear un nuevo PVC llamado `mysqlclone` mediante la anotación, como `trident.netapp.io/cloneFromPVC: mysql` . Con este conjunto de anotaciones, Trident clona el volumen correspondiente a la RVP de mysql, en lugar de aprovisionar un volumen desde cero.

Considere los siguientes puntos:

- Se recomienda clonar un volumen inactivo.
- Una RVP y su clon deben estar en el mismo espacio de nombres de Kubernetes y tener el mismo tipo de almacenamiento.
- Con los `ontap-nas` controladores y `ontap-san`, podría ser deseable establecer la anotación de PVC `trident.netapp.io/splitOnClone` junto con `trident.netapp.io/cloneFromPVC`. Con `trident.netapp.io/splitOnClone` Set to `true`, Trident divide el volumen clonado del volumen principal y, por lo tanto, desvincula por completo el ciclo de vida del volumen clonado de su principal a costa de perder cierta eficiencia del almacenamiento. Si no lo establece ni lo establece `trident.netapp.io/splitOnClone` en, `false` se reduce el consumo de espacio en el back-end a expensas de la creación de dependencias entre los volúmenes principal y el volumen clonado, de tal modo que el volumen principal no se pueda eliminar a menos que el clon se elimine primero. Una situación en la que dividir el clon tiene sentido es clonar un volumen de base de datos vacío donde se espera que tanto el volumen como su clon desvíen enormemente y no se beneficien de las eficiencias del almacenamiento ofrecidas por ONTAP.

  ```
  `sample-input`El directorio contiene ejemplos de definiciones RVP que se
  deben utilizar con Trident. Consulte para obtener una descripción
  completa de los parámetros y la configuración asociados con los
  volúmenes de Trident.
  ```

`PersistentVolume`Objetos de Kubernetes

Un objeto de Kubernetes `PersistentVolume` representa una pieza de almacenamiento que se pone a disposición del clúster de Kubernetes. Tiene un ciclo de vida independiente del pod que lo utiliza.

> Trident crea `PersistentVolume` objetos y los registra en el clúster de Kubernetes automáticamente en función de los volúmenes que aprovisiona. No se espera que usted los gestione usted mismo.

Cuando crea una RVP que hace referencia a una Trident `StorageClass` , Trident aprovisiona un nuevo volumen con la clase de almacenamiento correspondiente y registra un nuevo VP para ese volumen. Al configurar el volumen aprovisionado y el VP correspondiente, Trident sigue las siguientes reglas:

- Trident genera un nombre PV para Kubernetes y un nombre interno que utiliza para aprovisionar el almacenamiento. En ambos casos, se asegura de que los nombres son únicos en su alcance.
- El tamaño del volumen coincide con el tamaño solicitado en el PVC lo más cerca posible, aunque podría redondearse a la cantidad más cercana asignable, dependiendo de la plataforma.

`StorageClass`Objetos de Kubernetes

Los objetos de Kubernetes `StorageClass` se especifican por nombre en `PersistentVolumeClaims` para aprovisionar el almacenamiento con un conjunto de propiedades. La clase de almacenamiento identifica el aprovisionador que se usará y define ese conjunto de propiedades en términos que entiende el aprovisionador.

Es uno de los dos objetos básicos que el administrador debe crear y gestionar. El otro es el objeto back-end de Trident.

Un objeto de Kubernetes `StorageClass` que usa Trident tiene el siguiente aspecto:

```
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
```

Estos parámetros son específicos de Trident y dicen a Trident cómo aprovisionar volúmenes para la clase.

Los parámetros de la clase de almacenamiento son:

Atributo	Tipo	Obligatorio	Descripción
atributos	map[string]string	no	Consulte la sección atributos a continuación

Atributo	Tipo	Obligatorio	Descripción
Pools de almacenamiento	Map[string]StringList	no	Asignación de nombres de back-end a listas de pools de almacenamiento dentro
AdicionalStoragePools	Map[string]StringList	no	Asignación de nombres de back-end a listas de pools de almacenamiento dentro
ExcludeStoragePools	Map[string]StringList	no	Asignación de nombres de back-end a listas de pools de almacenamiento dentro

Los atributos de almacenamiento y sus posibles valores se pueden clasificar en atributos de selección de pools de almacenamiento y atributos de Kubernetes.

Atributos de selección del pool de almacenamiento

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

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	Método de aprovisionamiento especificado	grueso: all ONTAP; thin: all ONTAP y solidfire-san
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

Atributo	Tipo	Valores	Oferta	Solicitud	Admitido por
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

En la mayoría de los casos, los valores solicitados influyen directamente en el aprovisionamiento; por ejemplo, solicitar un aprovisionamiento de alto rendimiento da lugar a un volumen considerablemente aprovisionado. Sin embargo, un pool de almacenamiento de Element utiliza el valor mínimo y máximo de IOPS que ofrece para establecer los valores de calidad de servicio, en lugar del valor solicitado. En este caso, el valor solicitado se utiliza solo para seleccionar el pool de almacenamiento.

Lo ideal sería que pueda utilizar `attributes` solo para modelar las cualidades del almacenamiento que necesita para satisfacer las necesidades de una clase determinada. Trident detecta y selecciona automáticamente los pools de almacenamiento que coinciden con *todos* de los `attributes` especificados.

Si no puede utilizar `attributes` para seleccionar automáticamente los pools adecuados para una clase, puede utilizar los `storagePools` parámetros y `additionalStoragePools` para refinar aún más los pools o incluso para seleccionar un juego específico de pools.

Puede utilizar el `storagePools` parámetro para restringir aún más el juego de pools que coinciden con los especificados `attributes`. En otras palabras, Trident utiliza la intersección de pools identificados por los `attributes` parámetros y `storagePools` para el aprovisionamiento. Es posible usar un parámetro solo o ambos juntos.

Puede utilizar el `additionalStoragePools` parámetro para ampliar el conjunto de pools que Trident utiliza para el aprovisionamiento, independientemente de los pools seleccionados por los `attributes` parámetros y. `storagePools`

Es posible usar el `excludeStoragePools` parámetro para filtrar el conjunto de pools que Trident utiliza para el aprovisionamiento. Cuando se usa este parámetro, se quitan todos los pools que coinciden.

En los `storagePools` parámetros y `additionalStoragePools`, cada entrada toma el formato `<backend>:<storagePoolList>`, donde `<storagePoolList>` es una lista separada por comas de pools de almacenamiento para el backend especificado. Por ejemplo, un valor para `additionalStoragePools` puede ser similar a `ontapnas_192.168.1.100:aggr1,aggr2;solidfire_192.168.1.101:bronze`. Estas listas aceptan valores regex para los valores de backend y list. Puede utilizar `tridentctl get backend` para obtener la lista de back-ends y sus pools.

Atributos de Kubernetes

Trident no afecta a la selección de pools y back-ends de almacenamiento durante el aprovisionamiento dinámico. En su lugar, estos atributos simplemente ofrecen parámetros compatibles con los volúmenes persistentes de Kubernetes. Los nodos de trabajo son responsables de las operaciones de creación del sistema de archivos y pueden requerir utilidades del sistema de archivos, como xfsprogs.

Atributo	Tipo	Valores	Descripción	Controladores relevantes	Versión de Kubernetes
Tipo fstype	cadena	ext4, ext3, xfs	El tipo de sistema de archivos para los volúmenes de bloques	solidfire-san, ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economía	Todo
Expansión de allowVolume	booleano	verdadero, falso	Habilite o deshabilite el soporte para aumentar el tamaño de PVC	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economy, solidfire-san, gcp-cvs, azure-netapp-files	1,11 o posterior
VolumeBindingMode	cadena	Inmediatamente, WaitForFirstConsumer	Elija cuándo se producen el enlace de volumen y el aprovisionamiento dinámico	Todo	1,19 - 1,26

- `fsType`El parámetro se utiliza para controlar el tipo de sistema de archivos deseado para los LUN de SAN. Además, Kubernetes también utiliza la presencia de `fsType en una clase de almacenamiento para indicar que existe un sistema de archivos. La propiedad del volumen se puede controlar mediante `fsGroup` el contexto de seguridad de un pod solo si `fsType` se establece. Consulte "Kubernetes: Configure un contexto de seguridad para un Pod o contenedor"para obtener información general sobre la configuración de la propiedad del volumen mediante el `fsGroup` contexto. Kubernetes aplicará el `fsGroup` valor solo si:

  - `fsType` se define en la clase de almacenamiento.
  - El modo de acceso de PVC es RWO.

  Para los controladores de almacenamiento NFS, ya existe un sistema de archivos como parte de la exportación NFS. Para utilizar `fsGroup` la clase de almacenamiento, todavía necesita especificar un `fsType`. Puede definirlo en `nfs` o cualquier valor que no sea nulo.

- Consulte "Expanda los volúmenes"para obtener más información sobre la expansión de volumen.
- El paquete de instalación de Trident proporciona varias definiciones de clase de almacenamiento de ejemplo para su uso con Trident en `sample-input/storage-class-*.yaml`. Al eliminar una clase de almacenamiento Kubernetes, también se elimina el tipo de almacenamiento Trident correspondiente.

`VolumeSnapshotClass`Objetos de Kubernetes

Los objetos de Kubernetes `VolumeSnapshotClass` son análogos a `StorageClasses`. Ayudan a definir varias clases de almacenamiento y las instantáneas de volumen hacen referencia a ellas para asociar la snapshot a la clase de snapshot necesaria. Cada copia de Snapshot de volumen se asocia con una sola clase de copia de Snapshot de volumen.

Un administrador debe definir a `VolumeSnapshotClass` para crear instantáneas. Una clase de snapshot de volumen se crea con la siguiente definición:

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: csi-snapclass
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

El `driver` especifica a Kubernetes que Trident gestiona las solicitudes de instantáneas de volumen de `csi-snapclass` la clase. El `deletionPolicy` especifica la acción que se debe realizar cuando se debe eliminar una instantánea. `deletionPolicy`Cuando se establece en `Delete`, los objetos Snapshot del volumen, así como la snapshot subyacente en el clúster de almacenamiento, se eliminan cuando se elimina una snapshot. Como alternativa, si se configura en `Retain`, `VolumeSnapshotContent` se conservan la instantánea física y la física.

`VolumeSnapshot`Objetos de Kubernetes

Un objeto de Kubernetes `VolumeSnapshot` es una solicitud para crear una snapshot de un volumen. Del mismo modo que la RVP representa una solicitud al usuario para un volumen, un snapshot de volumen es una solicitud al que hace un usuario para crear una copia Snapshot de una RVP existente.

Cuando se recibe una solicitud de copia de Snapshot de volumen, Trident gestiona automáticamente la creación de la copia de Snapshot para el volumen en el back-end y expone la copia de Snapshot mediante la creación de un objeto único `VolumeSnapshotContent`. Puede crear instantáneas a partir de EVs existentes y utilizar las instantáneas como DataSource al crear nuevas CVP.

> El ciclo de vida de un VolumeSnapshot es independiente del PVC de origen: Una instantánea persiste incluso después de eliminar el PVC de origen. Cuando se elimina un PVC que tiene instantáneas asociadas, Trident Marca el volumen de respaldo de este PVC con el estado **Eliminación**, pero no lo elimina por completo. El volumen se elimina cuando se eliminan todas las Snapshot asociadas.

`VolumeSnapshotContent`Objetos de Kubernetes

Un objeto de Kubernetes `VolumeSnapshotContent` representa una snapshot tomada de un volumen ya aprovisionado. Es análogo a A `PersistentVolume` y significa una snapshot aprovisionada en el clúster de almacenamiento. Al igual que `PersistentVolumeClaim` y `PersistentVolume` objetos, cuando se crea una snapshot, `VolumeSnapshotContent` el objeto mantiene una asignación uno a uno con el `VolumeSnapshot` objeto, que había solicitado la creación de la snapshot.

El `VolumeSnapshotContent` objeto contiene detalles que identifican de forma exclusiva la instantánea, como el `snapshotHandle`. Esta `snapshotHandle` es una combinación única del nombre del VP y el nombre del `VolumeSnapshotContent` objeto.

Cuando llega una solicitud de Snapshot, Trident crea la snapshot en el back-end. Después de crear la copia Snapshot, Trident configura un `VolumeSnapshotContent` objeto y, por lo tanto, la copia Snapshot se expone a la API de Kubernetes.

> Por lo general, no es necesario administrar el `VolumeSnapshotContent` objeto. Una excepción a esto es cuando se desea "importe una copia de snapshot de volumen"crear fuera de Trident.

`CustomResourceDefinition`Objetos de Kubernetes

Los recursos personalizados de Kubernetes son extremos en la API de Kubernetes que define el administrador y que se usan para agrupar objetos similares. Kubernetes admite la creación de recursos personalizados para almacenar un conjunto de objetos. Puede obtener estas definiciones de recursos ejecutando `kubectl get crds`.

Kubernetes almacena en su almacén de metadatos las definiciones de recursos personalizadas (CRD) y los metadatos de objetos asociados. De este modo, no es necesario disponer de un almacén aparte para Trident.

Trident usa `CustomResourceDefinition` objetos para conservar la identidad de objetos de Trident, como los back-ends de Trident, las clases de almacenamiento Trident y los volúmenes de Trident. Trident gestiona estos objetos. Además, el marco de instantáneas de volumen CSI introduce algunos CRD necesarios para definir instantáneas de volumen.

Los multos son una estructura de Kubernetes. Trident crea los objetos de los recursos definidos anteriormente. Como ejemplo sencillo, cuando se crea un backend con `tridentctl`, se crea un objeto CRD correspondiente `tridentbackends` para su consumo por Kubernetes.

A continuación se indican algunos puntos que hay que tener en cuenta sobre los CRD de Trident:

- Cuando se instala Trident, se crea un conjunto de CRD que se puede utilizar como cualquier otro tipo de recurso.
- Al desinstalar Trident mediante el `tridentctl uninstall` comando, los pods de Trident se eliminan pero los CRD creados no se limpian. Consulte "Desinstale Trident"para comprender cómo Trident se puede eliminar por completo y volver a configurar desde cero.

Objetos deTrident `StorageClass`

Trident crea clases de almacenamiento coincidentes para los objetos de Kubernetes `StorageClass` que se especifican `csi.trident.netapp.io` en su campo aprovisionador. El nombre de la clase de almacenamiento coincide con el del objeto de Kubernetes `StorageClass` que representa.

> Con Kubernetes, estos objetos se crean automáticamente cuando se registra un Kubernetes `StorageClass` que utiliza Trident como aprovisionador.

Las clases de almacenamiento comprenden un conjunto de requisitos para los volúmenes. Trident enlaza estos requisitos con los atributos presentes en cada pool de almacenamiento; si coinciden, ese pool de almacenamiento es un objetivo válido para aprovisionar volúmenes que utilizan esa clase de almacenamiento.

Puede crear configuraciones de clase de almacenamiento para definir clases de almacenamiento directamente mediante la API DE REST. Sin embargo, para implementaciones de Kubernetes, esperamos que se creen al registrar nuevos objetos de Kubernetes `StorageClass`.

Objetos de back-end de Trident

Los back-ends representan a los proveedores de almacenamiento, además de los cuales Trident aprovisiona volúmenes; una única instancia de Trident puede gestionar cualquier número de back-ends.

> Éste es uno de los dos tipos de objeto que se crean y administran a sí mismo. El otro es el objeto de Kubernetes `StorageClass`.

Para obtener más información sobre cómo construir estos objetos, consulte "configuración de los back-ends".

Objetos deTrident `StoragePool`

Los pools de almacenamiento representan las distintas ubicaciones disponibles para aprovisionar en cada back-end. Para ONTAP, corresponden a los agregados en las SVM. Para HCI/SolidFire de NetApp, corresponden a las bandas de calidad de servicio especificadas por el administrador. Para Cloud Volumes Service, se corresponden con las regiones de proveedores de cloud. Cada pool de almacenamiento tiene un conjunto de atributos de almacenamiento distintos que definen sus características de rendimiento y sus características de protección de datos.

Al contrario de lo que ocurre con otros objetos aquí, los candidatos de pools de almacenamiento siempre se detectan y gestionan automáticamente.

Objetos deTrident `Volume`

Los volúmenes son la unidad básica de aprovisionamiento y constan de extremos back-end, como recursos compartidos de NFS y LUN iSCSI. En Kubernetes, estos corresponden directamente a `PersistentVolumes`. Cuando crea un volumen, asegúrese de que tiene una clase de almacenamiento, que determina dónde se puede aprovisionar ese volumen junto con un tamaño.

- En Kubernetes, estos objetos se gestionan automáticamente. Es posible verlos para ver qué ha aprovisionado Trident.
- Al eliminar un VP con instantáneas asociadas, el volumen Trident correspondiente se actualiza a un estado **Eliminación**. Para que se elimine el volumen de Trident, es necesario quitar las snapshots del volumen.

Una configuración de volumen define las propiedades que debe tener un volumen aprovisionado.

Atributo	Tipo	Obligatorio	Descripción
versión	cadena	no	Versión de la API de Trident ("1")
nombre	cadena	sí	Nombre del volumen que se va a crear
Clase de almacenamiento	cadena	sí	Clase de almacenamiento que se utilizará al aprovisionar el volumen
tamaño	cadena	sí	El tamaño del volumen que se va a aprovisionar en bytes
protocolo	cadena	no	Tipo de protocolo que se va a utilizar; "archivo" o "bloque"
InternalName	cadena	no	Nombre del objeto en el sistema de almacenamiento, generado por Trident
ClonSourceVolume	cadena	no	ONTAP (nas, san) y SolidFire-*: Nombre del volumen desde el que se va a clonar
SplitOnClone	cadena	no	ONTAP (nas, san): Divida el clon entre su primario
Política de copias Snapshot	cadena	no	ONTAP-*: Política de instantánea a utilizar
Reserva de copias Snapshot	cadena	no	ONTAP-*: Porcentaje del volumen reservado para instantáneas

Atributo	Tipo	Obligatorio	Descripción
Política de exportoPolicy	cadena	no	ontap-nas*: Política de exportación que se va a utilizar
Snapshot shotDirectory	bool	no	ontap-nas*: Si el directorio de instantáneas está visible
Permisos univalados	cadena	no	ontap-nas*: Permisos iniciales de UNIX
Tamaño del bloque	cadena	no	SolidFire-*: Tamaño de bloque/sector
Sistema de archivos	cadena	no	Tipo de sistema de archivos

Trident genera `internalName` al crear el volumen. Esto consta de dos pasos. En primer lugar, antepone el prefijo de almacenamiento (ya sea el predeterminado `trident` o el prefijo en la configuración de backend) al nombre del volumen, lo que da como resultado un nombre del formulario `<prefix>-<volume-name>`. A continuación, procede a desinfectar el nombre y a reemplazar los caracteres no permitidos en el backend. En el caso de los back-ends de ONTAP, reemplaza guiones con guiones bajos (por lo tanto, el nombre interno se convierte `<prefix>_<volume-name>`en). En los back-ends de Element, reemplaza guiones bajos por guiones.

Puedes utilizar las configuraciones de volúmenes para aprovisionar volúmenes directamente mediante la API de REST, pero en las implementaciones de Kubernetes esperamos que la mayoría de los usuarios usen el método Kubernetes estándar `PersistentVolumeClaim`. Trident crea este objeto de volumen automáticamente como parte del proceso de aprovisionamiento.

Objetos deTrident `Snapshot`

Las Snapshot son una copia de un momento específico de los volúmenes, que se pueden usar para aprovisionar nuevos volúmenes o restaurar el estado. En Kubernetes, estos corresponden directamente `VolumeSnapshotContent` a objetos. Cada copia de Snapshot se asocia con un volumen, que es el origen de los datos de la copia de Snapshot.

Cada `Snapshot` objeto incluye las propiedades enumeradas a continuación:

Atributo	Tipo	Obligatorio	Descripción
versión	Cadena	Sí	Versión de la API de Trident ("1")
nombre	Cadena	Sí	Nombre del objeto Snapshot de Trident
InternalName	Cadena	Sí	Nombre del objeto Snapshot de Trident en el sistema de almacenamiento

Atributo	Tipo	Obligatorio	Descripción
Nombre de volumen	Cadena	Sí	Nombre del volumen persistente para el que se crea la snapshot
VolumeInternalName	Cadena	Sí	Nombre del objeto de volumen de Trident asociado en el sistema de almacenamiento

En Kubernetes, estos objetos se gestionan automáticamente. Es posible verlos para ver qué ha aprovisionado Trident.

Cuando se crea una solicitud de objetos de Kubernetes `VolumeSnapshot`, Trident funciona creando un objeto Snapshot en el sistema de almacenamiento de respaldo. Para `internalName` este objeto Snapshot se genera combinando el prefijo `snapshot-` con el `UID` `VolumeSnapshot` objeto (por ejemplo, `snapshot-e8d8a0ca-9826-11e9-9807-525400f3f660`). `volumeName` y `volumeInternalName` se completan mediante la obtención de los detalles del volumen de respaldo.

ObjetoTrident `ResourceQuota`

El desamonset de Trident consume una `system-node-critical` clase de prioridad, la clase de prioridad más alta disponible en Kubernetes, para garantizar que Trident pueda identificar y limpiar volúmenes durante el apagado de nodo correcto y permitir que los pods de inicio de datos de Trident se adelanten a las cargas de trabajo con una prioridad más baja en los clústeres donde hay una alta presión de recursos.

Para ello, Trident emplea un `ResourceQuota` objeto para garantizar que se satisfaga una clase de prioridad «crítica para el sistema en el inicio de datos de Trident. Antes de la implementación y la creación de daemonset, Trident busca `ResourceQuota` el objeto y, si no se detecta, lo aplica.

Si necesita más control sobre la cuota de recursos predeterminada y la clase de prioridad, puede generar `custom.yaml` o configurar el `ResourceQuota` objeto mediante el gráfico Helm.

A continuación se muestra un ejemplo de un objeto "ResourceQuota'object que da prioridad al demonset de Trident.

```
apiVersion: <version>
kind: ResourceQuota
metadata:
  name: trident-csi
  labels:
    app: node.csi.trident.netapp.io
spec:
  scopeSelector:
     matchExpressions:
       - operator : In
         scopeName: PriorityClass
         values: ["system-node-critical"]
```

Para obtener más información sobre las cuotas de recursos, consulte "Kubernetes: Cuotas de recursos".

Limpie `ResourceQuota` si la instalación falla

En el caso raro de que la instalación falle después de `ResourceQuota` crear el objeto, primero intente "desinstalando" y luego vuelva a instalarlo.

Si eso no funciona, elimine manualmente el `ResourceQuota` objeto.

Quitar `ResourceQuota`

Si prefiere controlar su propia asignación de recursos, puede eliminar el objeto Trident `ResourceQuota` mediante el comando:

```
kubectl delete quota trident-csi -n trident
```

Pod Security Standards (PSS) y las restricciones de contexto de seguridad (SCC)

Los estándares de seguridad de Kubernetes Pod (PSS) y las políticas de seguridad de Pod (PSP) definen los niveles de permisos y restringen el comportamiento de los POD. OpenShift Security Context restriction (SCC) define de forma similar la restricción de POD específica para OpenShift Kubernetes Engine. Para proporcionar esta personalización, Trident permite ciertos permisos durante la instalación. En las siguientes secciones se detallan los permisos establecidos por Trident.

> PSS reemplaza las políticas de seguridad de Pod (PSP). PSP quedó obsoleto en Kubernetes v1.21 y se eliminará en la versión 1.25. Para obtener más información, consulte "Kubernetes: Seguridad".

Contexto de Kubernetes Security y campos relacionados necesarios

Permiso	Descripción
Privilegiado	CSI requiere que los puntos de montaje sean bidireccionales, lo que significa que el receptáculo del nodo Trident debe ejecutar un contenedor privilegiado. Para obtener más información, consulte "Kubernetes: Propagación de montaje".
Conexión a redes del host	Necesario para el daemon de iSCSI. `iscsiadm` Gestiona montajes iSCSI y utiliza redes de host para comunicarse con el daemon iSCSI.
IPC del host	NFS utiliza la comunicación entre procesos (IPC) para comunicarse con NFSD.

Permiso	Descripción
PID del host	Necesario para iniciar `rpc-statd` para NFS. Trident consulta los procesos de host para determinar `rpc-statd` si se están ejecutando antes de montar los volúmenes de NFS.

Funcionalidades	La `SYS_ADMIN` capacidad se proporciona como parte de las capacidades predeterminadas para contenedores con privilegios. Por ejemplo, Docker establece estas capacidades para contenedores con privilegios:
	`CapPrm: 0000003ffffffffff`
	`CapEff: 0000003ffffffffff`

Seccomp	El perfil de Seccomp siempre está «sin confinar» en contenedores privilegiados; por lo tanto, no se puede activar en Trident.
SELinux	En OpenShift, los contenedores con privilegios se ejecutan en el `spc_t` dominio («Super Privileged Container») y los contenedores sin privilegios se ejecutan en el `container_t` dominio. Activado `containerd`, con `container-selinux` instalado, todos los contenedores se ejecutan en el `spc_t` dominio, lo que desactiva efectivamente SELinux. Por lo tanto, Trident no `seLinuxOptions` agrega a los contenedores.
DAC	Los contenedores con privilegios deben ejecutarse como root. Los contenedores no privilegiados se ejecutan como root para acceder a los sockets unix necesarios para CSI.

Estándares de seguridad para POD (PSS)

Etiqueta	Descripción	Predeterminado
`pod-security.kubernetes.io/enforce` `pod-security.kubernetes.io/enforce-version`	Permite admitir la controladora Trident y los nodos en el espacio de nombres de instalación. No cambie la etiqueta de espacio de nombres.	`enforce: privileged` `enforce-version: <version of the current cluster or highest version of PSS tested.>`

> El cambio de las etiquetas del espacio de nombres puede provocar que los POD no se programen, un "error al crear: …" O bien, "Advertencia: trident-csi-…". Si esto sucede, compruebe si se ha cambiado la etiqueta de espacio de nombres para `privileged`. En ese caso, vuelva a instalar Trident.

Directivas de seguridad de POD (PSP)

Campo	Descripción	Predeterminado
`allowPrivilegeEscalation`	Los contenedores con privilegios deben permitir la escala de privilegios.	`true`
`allowedCSIDrivers`	Trident no utiliza volúmenes efímeros de CSI en línea.	Vacío
`allowedCapabilities`	Los contenedores Trident no con privilegios no requieren más funcionalidades de las que se establece de forma predeterminada y se conceden todas las funcionalidades posibles a los contenedores con privilegios.	Vacío
`allowedFlexVolumes`	Trident no utiliza un "Controlador FlexVolume", por lo tanto, no se incluyen en la lista de volúmenes permitidos.	Vacío
`allowedHostPaths`	El pod del nodo Trident monta el sistema de archivos raíz del nodo, por lo que no hay ninguna ventaja para configurar esta lista.	Vacío
`allowedProcMountTypes`	Trident no utiliza ninguna `ProcMountTypes`.	Vacío
`allowedUnsafeSysctls`	Trident no requiere ningún tipo inseguro `sysctls`.	Vacío
`defaultAddCapabilities`	No es necesario añadir capacidades a contenedores con privilegios.	Vacío
`defaultAllowPrivilegeEscalation`	En cada POD de Trident, se permite el escalado de privilegios.	`false`
`forbiddenSysctls`	No `sysctls` se permiten.	Vacío
`fsGroup`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`hostIPC`	El montaje de los volúmenes de NFS requiere que el IPC del host se comunique con el `nfsd`	`true`
`hostNetwork`	Iscsiadm requiere que la red del host se comunique con el demonio iSCSI.	`true`
`hostPID`	El PID del host es necesario para comprobar si `rpc-statd` se está ejecutando en el nodo.	`true`
`hostPorts`	Trident no utiliza puertos de host.	Vacío

Campo	Descripción	Predeterminado
`privileged`	Los pods de nodo Trident deben ejecutar un contenedor privilegiado para montar volúmenes.	`true`
`readOnlyRootFilesystem`	Los contenedores de nodos Trident deben escribir en el sistema de archivos del nodo.	`false`
`requiredDropCapabilities`	Los pods de nodo de Trident ejecutan un contenedor privilegiado y no pueden soltar las funcionalidades.	`none`
`runAsGroup`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`runAsUser`	Los contenedores Trident se ejecutan como raíz.	`runAsAny`
`runtimeClass`	Trident no utiliza `RuntimeClasses`.	Vacío
`seLinux`	Trident no se establece `seLinuxOptions` porque actualmente hay diferencias en cómo los tiempos de ejecución de los contenedores y las distribuciones de Kubernetes gestionan SELinux.	Vacío
`supplementalGroups`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`volumes`	Los pods de Trident requieren estos complementos de volumen.	`hostPath, projected, emptyDir`

Restricciones de contexto de seguridad (SCC)

Etiquetas	Descripción	Predeterminado
`allowHostDirVolumePlugin`	Los contenedores de nodos Trident montan el sistema de archivos raíz del nodo.	`true`
`allowHostIPC`	El montaje de volúmenes NFS requiere que el IPC del host se comunique con `nfsd`el .	`true`
`allowHostNetwork`	Iscsiadm requiere que la red del host se comunique con el demonio iSCSI.	`true`
`allowHostPID`	El PID del host es necesario para comprobar si `rpc-statd` se está ejecutando en el nodo.	`true`
`allowHostPorts`	Trident no utiliza puertos de host.	`false`

Etiquetas	Descripción	Predeterminado
`allowPrivilegeEscalation`	Los contenedores con privilegios deben permitir la escala de privilegios.	`true`
`allowPrivilegedContainer`	Los pods de nodo Trident deben ejecutar un contenedor privilegiado para montar volúmenes.	`true`
`allowedUnsafeSysctls`	Trident no requiere ningún tipo inseguro `sysctls`.	`none`
`allowedCapabilities`	Los contenedores Trident no con privilegios no requieren más funcionalidades de las que se establece de forma predeterminada y se conceden todas las funcionalidades posibles a los contenedores con privilegios.	Vacío
`defaultAddCapabilities`	No es necesario añadir capacidades a contenedores con privilegios.	Vacío
`fsGroup`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`groups`	Este SCC es específico de Trident y está vinculado a su usuario.	Vacío
`readOnlyRootFilesystem`	Los contenedores de nodos Trident deben escribir en el sistema de archivos del nodo.	`false`
`requiredDropCapabilities`	Los pods de nodo de Trident ejecutan un contenedor privilegiado y no pueden soltar las funcionalidades.	`none`
`runAsUser`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`seLinuxContext`	Trident no se establece `seLinuxOptions` porque actualmente hay diferencias en cómo los tiempos de ejecución de los contenedores y las distribuciones de Kubernetes gestionan SELinux.	Vacío
`seccompProfiles`	Los contenedores privilegiados siempre funcionan "sin confinar".	Vacío
`supplementalGroups`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`

Etiquetas	Descripción	Predeterminado
`users`	Se proporciona una entrada para vincular este SCC al usuario Trident en el espacio de nombres Trident.	n.a.
`volumes`	Los pods de Trident requieren estos complementos de volumen.	`hostPath, downwardAPI, projected, emptyDir`

Información de copyright

Copyright © 2025 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.