Documentación de Astra Trident 23,04

Astra Trident

NetApp
April 04, 2024

Tabla de contenidos

Documentación de Astra Trident 23,04

Notas de la versión

Lo nuevo

Las notas de la versión ofrecen información sobre nuevas funciones, mejoras y correcciones de errores en la última versión de Astra Trident.

> La `tridentctl` Binario para Linux que se proporciona en el archivo zip del instalador es la versión probada y compatible. Tenga en cuenta que `macos` binario proporcionado en la `/extras` parte del archivo zip no se ha probado ni es compatible.

¿Cuáles son las novedades de 23,04

> La fuerza de desconexión de volúmenes para volúmenes ONTAP-SAN-* solo es compatible con las versiones de Kubernetes con la puerta de la función de apagado de nodos no agraciados habilitada. La desconexión forzada debe estar habilitada en el momento de la instalación mediante `--enable-force-detach` Indicador del instalador de Trident.

Soluciones

- Se ha corregido el operador Trident para usar IPv6 localhost para la instalación cuando se especifica en SPEC.
- Se corrigieron los permisos de rol de clúster de operador de Trident que estaban sincronizados con los permisos del paquete ("Problema n.o 799").
- Se ha solucionado el problema al conectar un volumen de bloques sin configurar en varios nodos en el modo RWX.
- Compatibilidad con clonado de FlexGroup fijo e importación de volúmenes para volúmenes de SMB.
- Se corrigió el problema por el que la controladora Trident no podía apagarse inmediatamente ("Problema n.o 811").
- Se agregó una corrección para mostrar todos los nombres de igroup asociados con un LUN especificado aprovisionado con controladores ontap-san-*.
- Se ha agregado una corrección para permitir que los procesos externos se ejecuten hasta su finalización.
- Corregido error de compilación para la arquitectura s390 ("Problema n.o 537").
- Se solucionó un nivel de registro incorrecto durante las operaciones de montaje de volúmenes ("Problema n.o 781").
- Se ha corregido el error de afirmación de tipo potencial ("Problema n.o 802").

Mejoras

- Kubernetes:
  - Añadido soporte para Kubernetes 1,27.
  - Se ha añadido soporte para importar volúmenes LUKS.
  - Se ha añadido soporte para el modo de acceso de PVC ReadWriteOncePod.
  - Se añadió compatibilidad con la desconexión forzada para volúmenes ONTAP-SAN-* durante los escenarios de apagado de nodos sin gracia.

    - Todos los volúmenes de ONTAP-SAN-* ahora utilizarán iGroups por nodo. Las LUN solo se asignarán a iGroups, mientras que se publicarán de forma activa en esos nodos para mejorar nuestra política de seguridad. Los volúmenes existentes se cambiarán de forma oportunista al nuevo esquema de igroup cuando Trident determina que es seguro hacerlo sin afectar a las cargas de trabajo activas ("Problema n.o 758").
    - Mejora en la seguridad de Trident mediante la limpieza de los iGroups gestionados por Trident sin utilizar de los back-ends ONTAP-SAN-*.
- Se ha añadido soporte para volúmenes SMB con Amazon FSx para la economía de ontap-nas y los controladores de almacenamiento de ontap-nas-flexgroup.
- Se añadió compatibilidad con recursos compartidos SMB con los controladores de almacenamiento ONTAP-nas, ontap-nas y ontap-nas-flexgroup.
- Se ha añadido compatibilidad con los nodos arm64 ("Problema n.o 732").
- Ha mejorado el procedimiento de apagado de Trident desactivando los servidores API en primer lugar ("Problema n.o 811").
- Agregado soporte de compilación multiplataforma para hosts Windows y arm64 a Makefile; consulte BUILD.md.

Amortización

Kubernetes: Ya no se crearán iGroups en el ámbito del back-end al configurar controladores ontap-san y ontap-san-economy ("Problema n.o 758").

Cambios en 23.01.1

Soluciones

- Se ha corregido el operador Trident para usar IPv6 localhost para la instalación cuando se especifica en SPEC.
- Se han corregido los permisos de rol de clúster del operador de Trident para que estén sincronizados con los permisos del paquete "Problema n.o 799".
- Se ha agregado una corrección para permitir que los procesos externos se ejecuten hasta su finalización.
- Se ha solucionado el problema al conectar un volumen de bloques sin configurar en varios nodos en el modo RWX.
- Compatibilidad con clonado de FlexGroup fijo e importación de volúmenes para volúmenes de SMB.

Cambios en 23,01

> Kubernetes 1,27 ahora es compatible con Trident. Actualice Astra Trident antes de actualizar Kubernetes.

Soluciones

- Kubernetes: Se han añadido opciones para excluir la creación de políticas de seguridad de Pod para corregir las instalaciones de Trident mediante Helm ("Cuestiones #783, #794").

Mejoras

Kubernetes

- Añadido soporte para Kubernetes 1,26.
- Mejora de la utilización general de recursos de RBAC de Trident ("Problema n.o 757").
- Se agregó la automatización para detectar y corregir sesiones iSCSI rotas o obsoletas en los nodos de host.
- Compatibilidad añadida para ampliar volúmenes cifrados de LUKS.
- Kubernetes: Compatibilidad con rotación de credenciales añadida para volúmenes cifrados de LUKS.

Astra Trident

- Se ha agregado compatibilidad para volúmenes SMB con Amazon FSX para ONTAP al controlador de almacenamiento ontap-nas.
- Se añadió soporte para permisos NTFS cuando se utilizan volúmenes SMB.
- Se ha agregado soporte para pools de almacenamiento para volúmenes de GCP con el nivel de servicio CVS.
- Se ha añadido compatibilidad para el uso opcional de flexgroupagregarList al crear FlexGroups con el controlador de almacenamiento ontap-nas-flexgroup.
- Rendimiento mejorado para el controlador de almacenamiento ONTAP-nas-Economy al gestionar múltiples FlexVols.
- Actualizaciones de datLIF activadas para todas las controladoras de almacenamiento NAS de ONTAP.
- Se han actualizado la convención de nomenclatura Trident Deployment y DemonSet para reflejar el sistema operativo del nodo del host.

Amortización

- Kubernetes: Se ha actualizado el mínimo admitido de Kubernetes a 1.21.
- Ya no se deben especificar LIF de datos al realizar la configuración `ontap-san` o. `ontap-san-economy` de windows

Cambios en 22,10

Debe leer la siguiente información crítica antes de actualizar a Astra Trident 22.10.

** información de las Ocampo sobre la Astra Trident 22.10 **

- Kubernetes 1,25 ahora es compatible con Trident. Debe actualizar Astra Trident a 22.10 antes de actualizar a Kubernetes 1.25.
- Astra Trident ahora cumple estrictamente el uso de la configuración de varias rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en el archivo multipath.conf.

  Uso de la configuración sin multivía o el uso de `find_multipaths: yes` o. `find_multipaths: smart` el valor del archivo multipath.conf provocará fallos de montaje. Trident ha recomendado el uso de `find_multipaths: no` desde la versión 21.07.

Soluciones

- Se ha solucionado un problema específico del back-end de ONTAP creado mediante `credentials` el campo no se puede conectar durante la actualización 22.07.0 ("Número 759").

- **Docker:** se ha solucionado un problema que provocaba que el complemento para volúmenes de Docker no empezara en algunos entornos ("Problema n.o 548" y.. "Problema n.o 760").
- Se ha solucionado el problema de SLM específico de los back-ends DE SAN de ONTAP para garantizar que solo se publicara un subconjunto de LIF de datos que pertenecen a nodos de generación de informes.
- Se ha solucionado un problema de rendimiento por el que se realizaron análisis innecesarios de LUN iSCSI al conectar un volumen.
- Se han eliminado reintentos granulares en el flujo de trabajo iSCSI de Astra Trident para fallar rápidamente y reducir los intervalos de reintentos externos.
- Se solucionó un problema cuando se devolvió un error al vaciar un dispositivo iSCSI cuando ya se había vaciado el dispositivo multivía correspondiente.

Mejoras

- Kubernetes:
  - Añadido soporte para Kubernetes 1,25. Debe actualizar Astra Trident a 22.10 antes de actualizar a Kubernetes 1.25.
  - Se ha agregado una cuenta de servicio, ClusterRole y ClusterRoleBinding aparte para la implementación de Trident y DemonSet para permitir futuras mejoras de permisos.
  - Se ha agregado compatibilidad con "uso compartido de volúmenes entre espacios de nombres".
- Todos los Trident `ontap-*` Los controladores de almacenamiento ahora funcionan con la API DE REST de ONTAP.
- Se ha añadido un nuevo operador yaml (`bundle_post_1_25.yaml`) sin a. `PodSecurityPolicy` Para admitir Kubernetes 1.25.
- Añadido "Compatibilidad con volúmenes cifrados LUKS" para `ontap-san` y.. `ontap-san-economy` impulsores del almacenamiento.
- Se ha agregado compatibilidad con nodos de Windows Server 2019.
- Añadido "Compatibilidad con volúmenes SMB en nodos de Windows" a través de la `azure-netapp-files` controlador de almacenamiento.
- La detección de conmutación automática de MetroCluster para controladores ONTAP está disponible por lo general.

Amortización

- **Kubernetes:** Se actualizó el mínimo admitido de Kubernetes a 1,20.
- Se ha eliminado el controlador Astra Data Store (ADS).
- Se ha quitado el soporte de `yes` y.. `smart` opciones para `find_multipaths` Al configurar accesos múltiples de nodos de trabajo para iSCSI.

Cambios en 22,07

Soluciones

Kubernetes

- Se ha solucionado el problema para manejar los valores booleanos and Number para el selector de nodos cuando se configura Trident con Helm o el operador de Trident. ("GitHub Número 700")

- Se ha solucionado el problema al gestionar errores de ruta no CHAP, de modo que kubelet lo volverá a intentar si falla. "GitHub Número 736")

Mejoras

- Pasar de k8s.gcr.io a registry.k8s.io como registro predeterminado para las imágenes CSI
- Los volúmenes de ONTAP-SAN ahora utilizan iGroups por nodo y solo asignan LUN a iGroups, mientras se publican de forma activa en esos nodos para mejorar nuestra política de seguridad. Los volúmenes existentes se cambiarán de forma oportunista al nuevo esquema de igroup cuando Astra Trident determine que es seguro hacerlo sin afectar a las cargas de trabajo activas.
- Se incluye un ResourceQuota con las instalaciones de Trident para garantizar que Trident DemonSet se programe cuando el consumo de PriorityClass esté limitado de forma predeterminada.
- Se ha añadido compatibilidad con funciones de red al controlador ANF. ("GitHub Número 717")
- Se ha añadido una vista previa tecnológica con detección automática de conmutación de MetroCluster a los controladores de ONTAP. ("GitHub Número 228")

Amortización

- **Kubernetes:** Actualizado el mínimo admitido de Kubernetes a 1.19.
- La configuración de back-end ya no permite múltiples tipos de autenticación en una única configuración.

Absorciones

- Se ha eliminado el controlador CVS de AWS (obsoleto desde 22.04).
- Kubernetes
  - Se eliminó la capacidad SYS_ADMIN innecesaria de los POD de nodos.
  - Reduce el ruido a simple información de host y detección de servicios activos para hacer el mejor esfuerzo
    Confirmación de que los servicios NFS/iSCSI están disponibles en los nodos de trabajo.

Documentación

Un nuevo "Estándares de seguridad de POD" Se ha agregado la sección (PSS) detallando los permisos habilitados por Astra Trident en la instalación.

Cambios en 22.04

NetApp mejora y mejora continuamente sus productos y servicios. Estas son algunas de las últimas funciones de Astra Trident. Para las versiones anteriores, consulte "Versiones anteriores de la documentación".

> Si actualiza desde cualquier versión de Trident anterior y utiliza Azure NetApp Files, el `location` el parámetro config es ahora un campo obligatorio singleton.

Soluciones

- Análisis mejorado de nombres de iniciadores iSCSI. ("GitHub Número 681")
- Se ha solucionado un problema en el que no se permitían los parámetros de clase de almacenamiento CSI. ("GitHub Número 598")

- Se ha corregido la declaración de clave duplicada en Trident CRD. ("GitHub Número 671")
- Se han corregido registros de instantánea CSI imprecisos. ("GitHub Número 629"))
- Se ha solucionado el problema con la anulación de la publicación de volúmenes en nodos eliminados. ("GitHub Número 691")
- Se ha añadido el tratamiento de incoherencias del sistema de archivos en dispositivos de bloque. ("GitHub Número 656")
- Se ha solucionado el problema al extraer imágenes de soporte automático al configurar el `imageRegistry` indicador durante la instalación. ("GitHub Número 715")
- Se ha solucionado el problema en el que el controlador ANF no pudo clonar un volumen con varias reglas de exportación.

Mejoras

- Las conexiones entrantes con los extremos seguros de Trident ahora requieren un mínimo de TLS 1.3. ("GitHub Número 698")
- Trident ahora añade encabezados HSTS a las respuestas desde sus extremos seguros.
- Trident ahora intenta habilitar automáticamente la función de permisos de unix de Azure NetApp Files.
- **Kubernetes**: El demonset de Trident ahora se ejecuta en la clase prioritaria del nodo-sistema. ("GitHub Número 694")

Absorciones

Se ha quitado el controlador E-Series (desactivado desde 20.07).

Cambios en 22.01.1

Soluciones

- Se ha solucionado el problema con la anulación de la publicación de volúmenes en nodos eliminados. ("GitHub Número 691")
- Alerta fija al acceder a campos nulos para añadir espacio en respuestas de la API de ONTAP.

Cambios en 22.01.0

Soluciones

- **Kubernetes:** aumente el tiempo de reintento de retroceso de registro de nodos para clústeres grandes.
- Problema fijo donde el controlador Azure-netapp-files podría confundirse con varios recursos con el mismo nombre.
- Los LIF de datos IPv6 DE SAN de ONTAP ahora funcionan si se especifican con paréntesis.
- Un problema fijo en el que intentar importar un volumen ya importado devuelve EOF dejando PVC en estado pendiente. ("GitHub Número 489")
- Problema corregido cuando el rendimiento de Astra Trident se ralentiza cuando se crean más de 32 instantáneas en un volumen SolidFire.
- Se reemplazó SHA-1 por SHA-256 en la creación de certificados SSL.
- Controlador ANF fijo para permitir nombres de recursos duplicados y limitar las operaciones a una sola ubicación.

- Controlador ANF fijo para permitir nombres de recursos duplicados y limitar las operaciones a una sola ubicación.

Mejoras

- Mejoras de Kubernetes:
  - Añadido soporte para Kubernetes 1,23.
  - Añada opciones de programación para los pods de Trident cuando se instalen mediante Trident Operator o Helm. ("GitHub número 651")
- Permitir volúmenes entre regiones en el controlador GCP. ("GitHub Número 633")
- Se ha agregado compatibilidad con la opción 'unixPermissions' a los volúmenes ANF. ("GitHub Número 666")

Amortización

La interfaz DE REST de Trident solo puede escuchar y servir en 127.0.0.1 o direcciones [::1]

Cambios en 21.10.1

> La versión v21.10.0 tiene un problema que puede poner a la controladora Trident en estado CrashLoopBackOff cuando se elimina un nodo y, a continuación, volver a añadirse al clúster de Kubernetes. Este problema se soluciona en v21.10.1 (GitHub número 669).

Soluciones

- Se ha corregido una condición de carrera potencial al importar un volumen en un back-end CVS de GCP, lo que provoca un error al importar.
- Se ha solucionado un problema que puede poner la controladora Trident en estado CrashLoopBackOff cuando se quita un nodo y, a continuación, se vuelve a añadir al clúster de Kubernetes (GitHub número 669).
- Problema fijo donde ya no se detectaron SVM si no se especificó ningún nombre de SVM (GitHub, número 612).

Cambios en 21.10.0

Soluciones

- Se ha solucionado el problema por el que no se podían montar clones de volúmenes XFS en el mismo nodo que el volumen de origen (problema 514 de GitHub).
- Se ha solucionado un problema en el que Astra Trident registraba un error grave al apagar (GitHub, número 597).
- Correcciones relacionadas con Kubernetes:
  - Devuelva el espacio usado de un volumen como el tamaño mínimo de restoreSize a la hora de crear snapshots con `ontap-nas` y.. `ontap-nas-flexgroup` Controladores (GitHub, número 645).
  - Se ha solucionado el problema `Failed to expand filesystem` Se registró el error después de cambiar el tamaño del volumen (problema 560 de GitHub).
  - Se ha solucionado un problema por el que se podría atascar un pod `Terminating` estado (GitHub número 572).

- Se ha fijado la caja donde un `ontap-san-economy` Es posible que FlexVol esté lleno de LUN de snapshot (GitHub, número 533).
- Se ha solucionado el problema del instalador de YAML personalizado con una imagen diferente (GitHub, número 613).
- Se ha corregido el cálculo del tamaño de la instantánea (GitHub, número 611).
- Se ha solucionado un problema por el que todos los instaladores de Astra Trident podían identificar Kubernetes sin formato como OpenShift (GitHub, número 639).
- Se ha solucionado el operador Trident para detener la reconciliación si no se puede acceder al servidor API de Kubernetes (GitHub, número 599).

Mejoras

- Se ha agregado compatibilidad con `unixPermissions` Opción para los volúmenes de rendimiento GCP-CVS.
- Se ha agregado compatibilidad con volúmenes CVS optimizados para el escalado en GCP en el intervalo de 600 GIB a 1 TIB.
- Mejoras relacionadas con Kubernetes:
  - Se ha añadido la compatibilidad con Kubernetes 1.22.
  - Se ha habilitado el operador de Trident y el gráfico Helm para que funcionen con Kubernetes 1.22 (GitHub, número 628).
  - Se ha añadido la imagen del operador a. `tridentctl` Comando images (GitHub, número 570).

Mejoras experimentales

- Se añadió la compatibilidad con la replicación de volúmenes en `ontap-san` controlador.
- Se ha añadido el soporte DE DESCANSO **vista previa tecnológica** para el `ontap-nas-flexgroup`, `ontap-san`, y. `ontap-nas-economy` de windows

Problemas conocidos

Los problemas conocidos identifican problemas por los que el uso correcto del producto puede resultar imposible.

- Cuando actualice un clúster de Kubernetes de 1.24 a 1.25 o posterior que tenga instalado Astra Trident, debe actualizar Values.yaml para establecer `excludePodSecurityPolicy` para `true` o agregar `--set excludePodSecurityPolicy=true` para la `helm upgrade` comando antes de poder actualizar el clúster.
- Astra Trident ahora pone en práctica un espacio en blanco `fsType` (`fsType=""`) para los volúmenes que no tienen `fsType` Especificado en su clase de almacenamiento. Cuando trabaje con Kubernetes 1.17 o posterior, Trident admite proporcionar un espacio en blanco `fsType` Para volúmenes NFS. En los volúmenes iSCSI, se requiere que configure el `fsType` En el clase de almacenamiento al aplicar un `fsGroup` Uso de un contexto de seguridad.
- Si se utiliza un back-end en varias instancias de Astra Trident, cada archivo de configuración de back-end debería tener una diferencia `storagePrefix` Los valores para los back-ends de ONTAP o utilizan una diferencia `TenantName` Para back-ends de SolidFire. Astra Trident no puede detectar los volúmenes que han creado otras instancias de Astra Trident. El intento de crear un volumen existente en los back-ends de ONTAP o SolidFire se realiza correctamente, porque Astra Trident trata la creación de volúmenes como

una operación idempotente. Si `storagePrefix` o. `TenantName` no difieren, es posible que haya colisiones de nombres para los volúmenes creados en el mismo back-end.

- Al instalar Astra Trident (mediante `tridentctl` O el operador de Trident) y uso `tridentctl` Para gestionar Astra Trident, debe garantizar que `KUBECONFIG` la variable de entorno está configurada. Esto es necesario para indicar el clúster de Kubernetes que `tridentctl` debe trabajar en contra. Cuando trabaje con varios entornos de Kubernetes, debe asegurarse de que lo haga `KUBECONFIG` el archivo se ha originado con precisión.
- Para realizar una reclamación de espacio en línea para VP iSCSI, el sistema operativo subyacente del nodo de trabajo puede requerir que se pasen las opciones de montaje al volumen. Esto es así para las instancias de RHEL/RedHat CoreOS, que requieren el `discard` "opción de montaje"; Asegúrese de que la opción de montaje de descarte esté incluida en su[`StorageClass`para admitir descarte de bloques en línea.
- Si dispone de más de una instancia de Astra Trident por clúster de Kubernetes, Astra Trident no puede comunicarse con otras instancias y no puede detectar otros volúmenes que han creado, lo que conduce a un comportamiento inesperado e incorrecto si más de una instancia se ejecuta en un clúster. Solo debe haber una instancia de Astra Trident por clúster de Kubernetes.
- Si se basa en Astra Trident `StorageClass` Los objetos se eliminan de Kubernetes mientras Astra Trident está offline, Astra Trident no elimina las clases de almacenamiento correspondientes de su base de datos cuando vuelve a estar online. Debe eliminar estas clases de almacenamiento mediante `tridentctl` O la API DE REST.
- Si un usuario elimina un VP aprovisionado por Astra Trident antes de eliminar la RVP correspondiente, Astra Trident no elimina automáticamente el volumen del respaldo. Debe eliminar el volumen a través de `tridentctl` O la API DE REST.
- ONTAP no puede aprovisionar simultáneamente más de un FlexGroup a menos que el conjunto de agregados sea único para cada solicitud de aprovisionamiento.
- Cuando utilice Astra Trident sobre IPv6, debe especificar `managementLIF` y.. `dataLIF` en la definición de backend entre corchetes. Por ejemplo: `[fd20:8b1e:b258:2000:f816:3eff:feec:0]`.

> No puede especificar `dataLIF` En un entorno de administración SAN de ONTAP. Astra Trident descubre todos los LIF iSCSI disponibles y los utiliza para establecer la sesión multivía.

- Si utiliza `solidfire-san` Controlador con OpenShift 4.5, asegúrese de que los nodos de trabajo subyacentes utilizan MD5 como algoritmo de autenticación CHAP. Los algoritmos CHAP SHA1, SHA-256 y SHA3-256 compatibles con FIPS están disponibles con Element 12.7.

Obtenga más información

- "Astra Trident GitHub"
- "Blogs de Astra Trident"

Versiones anteriores de la documentación

Si no utiliza Astra Trident 23,04, la documentación de versiones anteriores está disponible según la "Lanzamiento de Astra Trident y ciclo de vida de soporte".

- "Astra Trident 23,01"

- "Astra Trident 22,10"
- "Astra Trident 22,07"
- "Astra Trident 22,04"
- "Astra Trident 22,01"
- "Astra Trident 21,10"
- "Astra Trident 21,07"

Conceptos

Obtenga más información sobre Astra Trident

Astra Trident es un proyecto de código abierto con soporte completo que mantiene NetApp como parte de "Familia de productos Astra". Se ha diseñado para ayudarle a satisfacer las demandas de persistencia de sus aplicaciones contenerizadas mediante interfaces estándar del sector, como Container Storage Interface (CSI).

Descripción general

Astra Trident se pone en marcha en clústeres de Kubernetes como pods y proporciona servicios de orquestación de almacenamiento dinámico para sus cargas de trabajo de Kubernetes. Permite que sus aplicaciones en contenedores consuman de forma rápida y sencilla el almacenamiento persistente de la amplia cartera de NetApp que incluye ONTAP (AFF/FAS/Select/Cloud/Amazon FSX para ONTAP de NetApp), el software Element (HCI/SolidFire de NetApp) y el servicio Azure NetApp Files y Cloud Volumes Service en Google Cloud.

Astra Trident también es una tecnología fundamental de la plataforma Astra de NetApp, que aborda casos de uso en materia de protección de datos, recuperación ante desastres, portabilidad y migración para cargas de trabajo de Kubernetes con la mejor tecnología de gestión de datos del sector de NetApp para copias Snapshot, backup, replicación y clonado.

Arquitecturas de clúster de Kubernetes compatibles

Astra Trident es compatible con las siguientes arquitecturas de Kubernetes:

Arquitecturas de clústeres de Kubernetes	Compatible	Instalación predeterminada
Un único maestro, informática	Sí	Sí
Varios maestros, informáticos	Sí	Sí
Maestro, `etcd`, cálculo	Sí	Sí
Maestro, infraestructura y computación	Sí	Sí

¿Qué es Astra?

Astra facilita a las empresas la gestión, la protección y el movimiento de sus cargas de trabajo en contenedores con una gran cantidad de datos que se ejecutan en Kubernetes en los clouds públicos y en las instalaciones. Astra aprovisiona y proporciona almacenamiento de contenedores persistente mediante Astra Trident de la cartera de almacenamiento probada y amplia de NetApp en el cloud público y en las instalaciones. También ofrece un conjunto amplio de funcionalidades avanzadas de gestión de datos para aplicaciones, como snapshots, backups y restauración, registros de actividades y clonado activo para la protección de datos, recuperación ante desastres/datos, auditoría de datos y casos de uso de migración para cargas de trabajo de Kubernetes.

Puedes inscribirte para una prueba gratuita en la página de Astra.

Si quiere más información

- "Familia de productos Astra de NetApp"
- "Documentación de Astra Control Service"
- "Documentación de Astra Control Center"
- "Documentación de API de Astra"

Controladores ONTAP

Astra Trident proporciona cinco controladores de almacenamiento exclusivos de ONTAP para comunicarse con clústeres de ONTAP.

Controladores compatibles con Astra Control

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

- Debe usar `ontap-nas` para cargas de trabajo de producción que requieren protección de datos, recuperación ante desastres y movilidad.
- Uso `ontap-san-economy` Cuando se espera que el uso previsto de volumen sea mucho superior al soporte de ONTAP.
- Uso `ontap-nas-economy` Únicamente en los casos en los que se espera que el uso previsto del volumen sea mucho superior al soporte de ONTAP y la `ontap-san-economy` no se puede utilizar el conductor.
- No utilizar `ontap-nas-economy` si prevé la necesidad de protección de datos, recuperación ante desastres o movilidad.

Controladores de almacenamiento Astra Trident para ONTAP

Astra Trident proporciona los controladores de almacenamiento siguientes para comunicarse con el clúster de ONTAP. 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
`ontap-nas`	NFS SMB	Sistema de archivos	RWO, ROX, RWX, RWOP	« », nfs, smb
`ontap-nas-economy`	NFS SMB	Sistema de archivos	RWO, ROX, RWX, RWOP	« », nfs, smb

Controlador	Protocolo	VolumeMode	Modos de acceso compatibles	Sistemas de archivos compatibles
`ontap-nas-flexgroup`	NFS SMB	Sistema de archivos	RWO, ROX, RWX, RWOP	« », nfs, smb
`ontap-san`	ISCSI	Bloque	RWO, ROX, RWX, RWOP	Sin sistema de archivos; dispositivo de bloque sin procesar
`ontap-san`	ISCSI	Sistema de archivos	RWO, ROX, RWOP RWX no está disponible en el modo de volumen del sistema de archivos.	`xfs`, `ext3`, `ext4`
`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, ROX, RWOP RWX no está disponible en el modo de volumen del sistema de archivos.	`xfs`, `ext3`, `ext4`

Los back-ends de ONTAP se pueden autenticar utilizando credenciales de inicio de sesión para un rol de seguridad (nombre de usuario/contraseña) o la clave privada y el certificado que se instala en el clúster de ONTAP. Es posible actualizar los back-ends existentes para pasar de un modo de autenticación a otro con `tridentctl update backend`.

El provisionamiento

El aprovisionamiento en Astra Trident tiene dos fases principales. La primera fase asocia una clase de almacenamiento con el conjunto de agrupaciones de almacenamiento back-end adecuadas y tiene lugar como preparación necesaria antes del aprovisionamiento. La segunda fase incluye la creación del volumen y requiere la selección de un pool de almacenamiento de los asociados con la clase de almacenamiento del volumen pendiente.

Asociación de clase de almacenamiento

La asociación de pools de almacenamiento de entorno de administración con una clase de almacenamiento depende tanto de los atributos solicitados de la clase de almacenamiento como de sus `storagePools`, `additionalStoragePools`, y. `excludeStoragePools` listas. Al crear una clase de almacenamiento, Trident compara los atributos y pools que ofrecen cada uno de sus back-ends con los solicitados por la clase de almacenamiento. Si los atributos y el nombre de un pool de almacenamiento coinciden con todos los atributos y los nombres de pool solicitados, Astra Trident añade ese pool de almacenamiento al conjunto de pools de almacenamiento adecuados para esa clase de almacenamiento. Además, Astra Trident añade todos

los pools de almacenamiento que aparecen en la `additionalStoragePools` enumerar este conjunto, incluso si sus atributos no cumplen todos o ninguno de los atributos solicitados de la clase de almacenamiento. Debe utilizar el `excludeStoragePools` enumerar para anular y quitar pools de almacenamiento de usar en una clase de almacenamiento. Astra Trident realiza un proceso similar cada vez que agrega un nuevo back-end, comprueba si sus pools de almacenamiento satisfacen las clases de almacenamiento existentes y eliminan cualquiera que se haya marcado como excluido.

Creación del volumen

A continuación, Astra Trident utiliza las asociaciones entre clases de almacenamiento y pools de almacenamiento para determinar dónde se deben aprovisionar los volúmenes. Cuando se crea un volumen, Astra Trident obtiene primero el conjunto de pools de almacenamiento para la clase de almacenamiento de ese volumen, Asimismo, si especifica un protocolo para el volumen, Astra Trident elimina los pools de almacenamiento que no pueden proporcionar el protocolo solicitado (por ejemplo, un back-end de HCI/SolidFire de NetApp no puede proporcionar un volumen basado en archivos mientras que un back-end NAS de ONTAP no puede proporcionar un volumen basado en bloques). Astra Trident aleatoriza el orden de este conjunto resultante, para facilitar una distribución uniforme de volúmenes y, a continuación, repite el proceso, intentando aprovisionar el volumen en cada pool de almacenamiento a su vez. Si se produce correctamente en una, vuelve con éxito y registra los fallos encontrados en el proceso. Astra Trident devuelve un fallo **sólo si** no consigue aprovisionar en **todos** los pools de almacenamiento disponibles para la clase de almacenamiento y el protocolo solicitados.

Copias de Snapshot de volumen

Más información sobre cómo Astra Trident gestiona la creación de snapshots de volumen para sus controladores.

Obtenga información acerca de la creación de snapshots de volúmenes

- Para la `ontap-nas`, `ontap-san`, `gcp-cvs`, y. `azure-netapp-files` Controladores, cada volumen persistente (PV) se asigna a una FlexVol. Como resultado, las copias de Snapshot de volumen se crean como copias de Snapshot de NetApp. La tecnología Snapshot de NetApp ofrece una mayor estabilidad, escalabilidad, capacidad de recuperación y rendimiento que las tecnologías snapshot de la competencia. Estas copias Snapshot son extremadamente eficientes, tanto en el tiempo necesario para crearlas como en el espacio de almacenamiento.
- Para la `ontap-nas-flexgroup` Cada controlador, cada volumen persistente (PV) se asigna a una FlexGroup. Como resultado, las copias de Snapshot de volumen se crean como copias de Snapshot de FlexGroup de NetApp. La tecnología Snapshot de NetApp ofrece una mayor estabilidad, escalabilidad, capacidad de recuperación y rendimiento que las tecnologías snapshot de la competencia. Estas copias Snapshot son extremadamente eficientes, tanto en el tiempo necesario para crearlas como en el espacio de almacenamiento.
- Para la `ontap-san-economy` Controlador, VP se asigna a las LUN creadas en los FlexVols compartidos. Las copias Snapshot Volumede VP realizan FlexClones del LUN asociado. La tecnología FlexClone de ONTAP posibilita la creación de copias incluso de los conjuntos de datos más grandes de forma casi instantánea. Las copias comparten bloques de datos con sus padres, sin consumir almacenamiento, excepto lo que se necesita para los metadatos.
- Para la `solidfire-san` Cada controlador, cada VP asigna una LUN creada en el software NetApp Element/clúster HCI de NetApp. Las copias Snapshot de volumen están representadas por copias Snapshot de Element de la LUN subyacente. Estas copias Snapshot son copias puntuales y solo ocupan una pequeña cantidad de recursos y espacio del sistema.

- Al trabajar con `ontap-nas` y.. `ontap-san` Controladores, las copias snapshot ONTAP son copias puntuales del FlexVol y consumen espacio en la propia FlexVol. Esto puede dar como resultado la cantidad de espacio editable en el volumen para reducirlo con el tiempo a medida que se crean y se programan las copias Snapshot. Una forma sencilla de abordar esto es aumentar el volumen mediante el cambio de tamaño a través de Kubernetes. Otra opción es eliminar las snapshots que ya no son necesarias. Cuando se elimina una copia Snapshot de volumen creada mediante Kubernetes, Astra Trident elimina la copia Snapshot de ONTAP asociada. También se pueden eliminar las copias de Snapshot de ONTAP que no se crearon con Kubernetes.

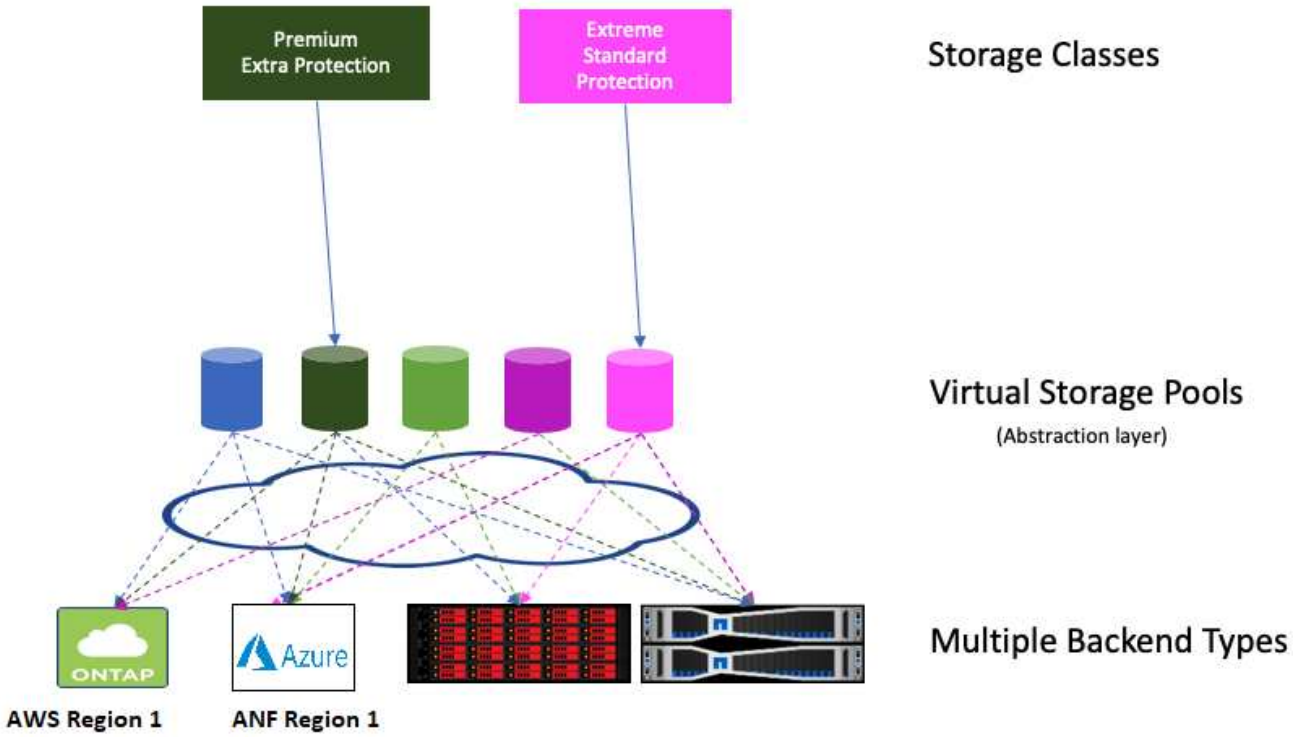
Con Astra Trident, puede usar VolumeSnapshots para crear nuevos VP a partir de ellos. La creación de VP a partir de estas snapshots se realiza usando la tecnología FlexClone para los back-ends de ONTAP y CVS compatibles. Al crear un VP a partir de una instantánea, el volumen de respaldo es un FlexClone del volumen principal de la instantánea. La `solidfire-san` La unidad utiliza clones de volúmenes del software Element para crear VP a partir de copias de Snapshot. Aquí se crea un clon a partir de la copia de Snapshot de Element.

Pools virtuales

Los pools virtuales proporcionan una capa de abstracción entre los back-ends de almacenamiento de Astra Trident y Kubernetes `StorageClasses`. Permiten a un administrador definir aspectos, como la ubicación, el rendimiento y la protección de cada back-end de una manera común y independiente del back-end sin hacer un `StorageClass` especifique qué tipo de backend físico, pool back-end o backend desea utilizar para cumplir los criterios deseados.

Más información sobre los pools virtuales

El administrador de almacenamiento puede definir pools virtuales en cualquiera de los back-ends de Astra Trident en un archivo de definición JSON o YLMA.



Cualquier aspecto especificado fuera de la lista de pools virtuales es global para el back-end y se aplicará a todos los pools virtuales, mientras que cada pool virtual puede especificar uno o más aspectos individualmente (reemplazar cualquier aspecto back-end-global).

- Al definir los pools virtuales, no intente reorganizar el orden de los pools virtuales existentes en una definición de back-end.
- Se aconseja modificar los atributos de un pool virtual existente. Debe definir un nuevo pool virtual para realizar cambios.

La mayoría de los aspectos se especifican en términos específicos del back-end. Lo más importante es que los valores de aspecto no se exponen fuera del controlador del backend y no están disponibles para coincidir en `StorageClasses`. En su lugar, el administrador define una o varias etiquetas para cada pool virtual. Cada etiqueta es una pareja clave:valor y las etiquetas pueden ser comunes en los back-ends únicos. Al igual que en los aspectos, las etiquetas se pueden especificar por grupo o globalmente en el backend. A diferencia de los aspectos, que tienen nombres y valores predefinidos, el administrador tiene la total discreción de definir claves y valores de etiqueta según sea necesario. Para mayor comodidad, los administradores de almacenamiento pueden definir etiquetas por pool virtual y agrupar volúmenes por etiqueta.

1. `StorageClass` identifica el pool virtual que se debe utilizar haciendo referencia a las etiquetas dentro de un parámetro de selector. Los selectores de pools virtuales admiten los siguientes operadores:

Operador	Ejemplo	El valor de etiqueta de un pool debe:
`=`	rendimiento=premium	Coincidencia
`!=`	rendimiento!=extremo	No coincide
`in`	ubicación en (este, oeste)	Esté en el conjunto de valores

Operador	Ejemplo	El valor de etiqueta de un pool debe:
`notin`	rendimiento de la muesca (plata, bronce)	No esté en el conjunto de valores
`<key>`	protección	Existe con cualquier valor
`!<key>`	!protección	No existe

Los grupos de acceso de volúmenes

Obtenga más información sobre el uso de Astra Trident "los grupos de acceso de volúmenes".

> Ignore esta sección si está utilizando CHAP, que se recomienda para simplificar la gestión y evitar el límite de escalado descrito a continuación. Además, si está utilizando Astra Trident en el modo CSI, puede ignorar esta sección. Astra Trident utiliza CHAP cuando se instala como un aprovisionador CSI mejorado.

Obtenga información acerca de los grupos de acceso de volúmenes

Astra Trident puede usar grupos de acceso de volúmenes para controlar el acceso a los volúmenes que aprovisiona. Si CHAP está deshabilitado, se espera encontrar un grupo de acceso llamado `trident` A menos que se especifiquen uno o varios ID del grupo de acceso en la configuración.

Aunque Astra Trident asocia nuevos volúmenes con los grupos de acceso configurados, no crea ni gestiona los propios grupos de acceso. Los grupos de acceso deben existir antes de que el back-end de almacenamiento se añada a Astra Trident y deben contener los IQN de iSCSI desde cada nodo del clúster de Kubernetes que podría montar potencialmente los volúmenes aprovisionados mediante ese back-end. En la mayoría de las instalaciones, esto incluye todos los nodos de trabajo del clúster.

Para los clústeres de Kubernetes con más de 64 nodos, se deben usar varios grupos de acceso. Cada grupo de acceso puede contener hasta 64 IQN, y cada volumen puede pertenecer a cuatro grupos de acceso. Con un máximo de cuatro grupos de acceso configurados, cualquier nodo de un clúster con un tamaño de hasta 256 nodos podrá acceder a cualquier volumen. Para obtener los límites más recientes de los grupos de acceso de volúmenes, consulte "aquí".

Si va a modificar la configuración a partir de una que está utilizando la configuración predeterminada `trident` Incluya el ID de para uno que utilice también otros `trident` grupo de acceso de la lista.

Manos a la obra

Pruébelo

NetApp proporciona una imagen de laboratorio lista para usar a la que puede solicitar "Versión de prueba de NetApp".

Obtenga más información sobre la versión de prueba

La unidad de prueba ofrece un entorno de filtrado que incluye un clúster de Kubernetes de tres nodos y una configuración de Astra Trident instalada y configurada. Es una excelente forma de familiarizarse con Astra Trident y explorar sus funciones.

Otra opción es ver la "Guía de instalación de Kubeadm" Proporcionado por Kubernetes.

> No debe usar el clúster de Kubernetes que cree con las siguientes instrucciones en producción. Use las guías de puesta en marcha de producción que ofrece su distribución para crear clústeres listos para la producción.

Si es la primera vez que utiliza Kubernetes, familiarícese con los conceptos y las herramientas "aquí".

Requisitos

Antes de instalar Astra Trident, debería revisar estos requisitos generales del sistema. Es posible que los back-ends específicos tengan requisitos adicionales.

Información vital sobre Astra Trident 23,01

- Debe leer la siguiente información crítica sobre Astra Trident.*

** información bï¿½tico sobre Astra Tridbï¿½tico **

- Kubernetes 1,27 ahora es compatible con Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident cumple estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en el archivo multipath.conf.

  Uso de la configuración sin multivía o el uso de `find_multipaths: yes` o. `find_multipaths: smart` el valor del archivo multipath.conf provocará fallos de montaje. Trident ha recomendado el uso de `find_multipaths: no` desde la versión 21.07.

Front-ends compatibles (orquestadores)

Astra Trident admite varios orquestadores y motores de contenedor, incluidos los siguientes:

- Anthos on-premises (VMware) y Anthos en 1,12 básico
- Kubernetes 1,21 - 1,27
- Mirantis Kubernetes Engine 3,5

- OpenShift 4.9 - 4.12

El operador Trident es compatible con las siguientes versiones:

- Anthos on-premises (VMware) y Anthos en 1,12 básico
- Kubernetes 1,21 - 1,27
- OpenShift 4.9 - 4.12

Astra Trident también funciona con una gran cantidad de otras ofertas de Kubernetes completamente gestionadas y gestionadas, como Google Kubernetes Engine (GKE), Amazon Elastic Kubernetes Services (EKS), Azure Kubernetes Service (AKS), Rancher y VMware Tanzu Portfolio.

Antes de actualizar un clúster de Kubernetes de 1.24 a 1.25 o posterior que tenga instalado Astra Trident, consulte "Actualice la instalación de un operador basado en Helm".

Back-ends compatibles (almacenamiento)

Para utilizar Astra Trident, se necesitan uno o varios de los siguientes back-ends compatibles:

- Amazon FSX para ONTAP de NetApp
- Azure NetApp Files
- Cloud Volumes ONTAP
- Cloud Volumes Service para GCP
- FAS/AFF/Select 9,5 o posterior
- Cabina All SAN de NetApp (ASA)
- Software HCI/Element de NetApp 11 o posterior

Requisitos de funciones

La siguiente tabla resume las funciones disponibles con esta versión de Astra Trident y las versiones de Kubernetes compatible.

Función	La versión de Kubernetes	¿Se requieren puertas de funciones?
CSI Trident	1,21 - 1,27	No
Snapshots de volumen	1,21 - 1,27	No
RVP desde snapshots de volumen	1,21 - 1,27	No
Cambio de tamaño del VP de iSCSI	1,21 - 1,27	No
CHAP bidireccional de ONTAP	1,21 - 1,27	No
Políticas de exportación dinámicas	1,21 - 1,27	No

Función	La versión de Kubernetes	¿Se requieren puertas de funciones?
Operador de Trident	1,21 - 1,27	No
Topología CSI	1,21 - 1,27	No

Se probaron sistemas operativos host

Aunque Astra Trident no admite oficialmente sistemas operativos específicos, se sabe que lo siguiente es lo siguiente:

- Versiones de RedHat CoreOS (RHCOS) compatibles con OpenShift Container Platform (AMD64 y ARM64)
- RHEL 8+ (AMD64 Y ARM64)
- Ubuntu 22,04 o posterior (AMD64 y ARM64)
- Windows Server 2019 (AMD64)

De forma predeterminada, Astra Trident se ejecuta en un contenedor y, por lo tanto, se ejecutará en cualquier trabajador de Linux. Sin embargo, estos trabajadores deben poder montar los volúmenes que ofrece Astra Trident con el cliente NFS o iniciador iSCSI estándar, en función de los back-ends que utilice.

La `tridentctl` Utility también se ejecuta en cualquiera de estas distribuciones de Linux.

Configuración de hosts

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 o iSCSI según su selección de controlador.

"Prepare el nodo de trabajo"

Configuración del sistema de almacenamiento

Es posible que Astra Trident requiera cambios en un sistema de almacenamiento antes de que una configuración de back-end pueda usarla.

"Configurar los back-ends"

Puertos Astra Trident

Astra Trident requiere acceso a puertos específicos para la comunicación.

"Puertos Astra Trident"

Imágenes de contenedor y las versiones de Kubernetes correspondientes

Para instalaciones con problemas de conexión aérea, la siguiente lista es una referencia de las imágenes de contenedor necesarias para instalar Astra Trident. Utilice la `tridentctl images` comando para verificar la lista de imágenes de contenedor necesarias.

La versión de Kubernetes	Imagen de contenedor
1.21.0	• docker.io/netapp/trident:23.04.0 • docker.io/netapp/trident-autosupport:23,04 • registry.k8s.io/sig-storage/csi-provisioner:v3,4.1 • registry.k8s.io/sig-storage/csi-attacher:v4,2.0 • registry.k8s.io/sig-storage/csi-resizer:v1,7.0 • registry.k8s.io/sig-storage/csi-snapshotter:v6,2.1 • registry.k8s.io/sig-storage/csi-node-driver-registrador:v2,7.0 • docker.io/netapp/trident-operator:23.04.0 (opcional)
v1.22.0	• docker.io/netapp/trident:23.04.0 • docker.io/netapp/trident-autosupport:23,04 • registry.k8s.io/sig-storage/csi-provisioner:v3,4.1 • registry.k8s.io/sig-storage/csi-attacher:v4,2.0 • registry.k8s.io/sig-storage/csi-resizer:v1,7.0 • registry.k8s.io/sig-storage/csi-snapshotter:v6,2.1 • registry.k8s.io/sig-storage/csi-node-driver-registrador:v2,7.0 • docker.io/netapp/trident-operator:23.04.0 (opcional)
v1.23.0	• docker.io/netapp/trident:23.04.0 • docker.io/netapp/trident-autosupport:23,04 • registry.k8s.io/sig-storage/csi-provisioner:v3,4.1 • registry.k8s.io/sig-storage/csi-attacher:v4,2.0 • registry.k8s.io/sig-storage/csi-resizer:v1,7.0 • registry.k8s.io/sig-storage/csi-snapshotter:v6,2.1 • registry.k8s.io/sig-storage/csi-node-driver-registrador:v2,7.0 • docker.io/netapp/trident-operator:23.04.0 (opcional)

La versión de Kubernetes	Imagen de contenedor
v1.24.0	• docker.io/netapp/trident:23.04.0 • docker.io/netapp/trident-autosupport:23,04 • registry.k8s.io/sig-storage/csi-provisioner:v3,4.1 • registry.k8s.io/sig-storage/csi-attacher:v4,2.0 • registry.k8s.io/sig-storage/csi-resizer:v1,7.0 • registry.k8s.io/sig-storage/csi-snapshotter:v6,2.1 • registry.k8s.io/sig-storage/csi-node-driver-registrador:v2,7.0 • docker.io/netapp/trident-operator:23.04.0 (opcional)
v1.25.0	• docker.io/netapp/trident:23.04.0 • docker.io/netapp/trident-autosupport:23,04 • registry.k8s.io/sig-storage/csi-provisioner:v3,4.1 • registry.k8s.io/sig-storage/csi-attacher:v4,2.0 • registry.k8s.io/sig-storage/csi-resizer:v1,7.0 • registry.k8s.io/sig-storage/csi-snapshotter:v6,2.1 • registry.k8s.io/sig-storage/csi-node-driver-registrador:v2,7.0 • docker.io/netapp/trident-operator:23.04.0 (opcional)
v1.26.0	• docker.io/netapp/trident:23.04.0 • docker.io/netapp/trident-autosupport:23,04 • registry.k8s.io/sig-storage/csi-provisioner:v3,4.1 • registry.k8s.io/sig-storage/csi-attacher:v4,2.0 • registry.k8s.io/sig-storage/csi-resizer:v1,7.0 • registry.k8s.io/sig-storage/csi-snapshotter:v6,2.1 • registry.k8s.io/sig-storage/csi-node-driver-registrador:v2,7.0 • docker.io/netapp/trident-operator:23.04.0 (opcional)

La versión de Kubernetes	Imagen de contenedor
v.1.27.0	• docker.io/netapp/trident:23.04.0 • docker.io/netapp/trident-autosupport:23,04 • registry.k8s.io/sig-storage/csi-provisioner:v3,4.1 • registry.k8s.io/sig-storage/csi-attacher:v4,2.0 • registry.k8s.io/sig-storage/csi-resizer:v1,7.0 • registry.k8s.io/sig-storage/csi-snapshotter:v6,2.1 • registry.k8s.io/sig-storage/csi-node-driver-registrador:v2,7.0 • docker.io/netapp/trident-operator:23.04.0 (opcional)

En Kubernetes versión 1,21 y posteriores, utilice el validado `registry.k8s.gcr.io/sig-storage/csi-snapshotter:v6.x` la imagen sólo si la `v1` la versión sirve `volumesnapshots.snapshot.storage.k8s.gcr.io` CRD. Si la `v1beta1` La versión sirve al CRD con/sin el `v1` versión, utilice la validada `registry.k8s.gcr.io/sig-storage/csi-snapshotter:v3.x` imagen.

Instale Astra Trident

Obtenga más información sobre la instalación de Astra Trident

Para garantizar que Astra Trident se puede instalar en una amplia variedad de entornos y organizaciones, NetApp ofrece múltiples opciones de instalación. Puede instalar Astra Trident con el operador Trident (manualmente o mediante Helm) o con `tridentctl`. En este tema se proporciona información importante para seleccionar el proceso de instalación adecuado.

Información vital sobre Astra Trident 23,04

- Debe leer la siguiente información crítica sobre Astra Trident.*

información bï¿½tico sobre Astra Tridbï¿½tico

- Kubernetes 1,27 ahora es compatible con Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident cumple estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en el archivo multipath.conf.

  Uso de la configuración sin multivía o el uso de `find_multipaths: yes` o. `find_multipaths: smart` el valor del archivo multipath.conf provocará fallos de montaje. Trident ha recomendado el uso de `find_multipaths: no` desde la versión 21.07.

Antes de empezar

Independientemente de la ruta de instalación, debe tener:

- Privilegios completos en un clúster de Kubernetes compatible que ejecuta una versión compatible de Kubernetes y requisitos de funciones habilitados. Revise la "requisitos" para obtener más detalles.
- Acceso a un sistema de almacenamiento de NetApp compatible.
- Capacidad para montar volúmenes de todos los nodos de trabajo de Kubernetes.
- Un host Linux con `kubectl` (o. `oc`, Si está utilizando OpenShift) instalado y configurado para administrar el clúster de Kubernetes que desea utilizar.
- La `KUBECONFIG` Variable de entorno establecida en el clúster de Kubernetes.
- Si utiliza Kubernetes con Docker Enterprise, "Siga sus pasos para habilitar el acceso a la CLI".

Si usted no se ha familiarizado con el "conceptos básicos", ahora es un gran momento para hacerlo.

Elija el método de instalación

Seleccione el método de instalación adecuado. También debe revisar las consideraciones de "moverse entre métodos" antes de tomar su decisión.

Utilice el operador Trident

Tanto si se pone en marcha manualmente como si se utiliza Helm, el operador Trident es una forma excelente de simplificar la instalación y gestionar de forma dinámica los recursos de Astra Trident. Incluso puede "Personalice la implementación del operador de Trident" uso de los atributos de la `TridentOrchestrator` Recurso personalizado (CR).

Algunas de las ventajas de usar el operador Trident son:

** Astra Trident de objetos cremunidad **

El operador Trident crea automáticamente los siguientes objetos para la versión de Kubernetes.

- ServiceAccount para el operador
- ClusterRole y ClusterRoleBinding a la cuenta de servicio
- Dedicated PodSecurityPolicy (para Kubernetes 1.25 y versiones anteriores)
- El propio operador

** de capeel de curación de las Ouna**

El operador supervisa la instalación de Astra Trident y toma activamente medidas para resolver problemas, como cuándo se elimina la implementación o si se modifica accidentalmente. A. `trident-operator-<generated-id>` se crea un pod que asocia un `TridentOrchestrator` CR con una instalación de Astra Trident. Esto garantiza que solo haya una instancia de Astra Trident en el clúster y controle su configuración, asegurándose de que la instalación es idempotente. Cuando se realizan cambios en la instalación (como eliminar el despliegue o el conjunto de nodos), el operador los identifica y los corrige individualmente.

® actualizaciones en la existente

Puede actualizar fácilmente una implementación existente con el operador. Sólo tiene que editar el `TridentOrchestrator` CR para realizar actualizaciones de una instalación.

Por ejemplo, piense en una situación en la que necesita habilitar Astra Trident para generar registros de depuración. Para hacer esto, parche su `TridentOrchestrator` para ajustar `spec.debug` para `true`:

```
kubectl patch torc <trident-orchestrator-name> -n trident --type=merge
-p '{"spec":{"debug":true}}'
```

Después `TridentOrchestrator` se actualiza, el operador procesa las actualizaciones y parches de la instalación existente. Esto puede activar la creación de nuevos POD para modificar la instalación según corresponda.

Mejora a de Kubernetes a mano

Cuando la versión de Kubernetes del clúster se actualiza a una versión compatible, el operador actualiza una instalación existente de Astra Trident automáticamente y la cambia para garantizar que cumple los requisitos de la versión de Kubernetes.

> Si se actualiza el clúster a una versión no compatible, el operador evita la instalación de Astra Trident. Si ya se ha instalado Astra Trident con el operador, se muestra una advertencia para indicar que Astra Trident está instalada en una versión de Kubernetes no compatible.

gestiona clústeres a través de BlueXP (anteriormente Cloud Manager)

Con "Astra Trident con BlueXP", Puede realizar una actualización a la versión más reciente de Astra Trident, agregar y gestionar clases de almacenamiento, conectarlos a entornos de trabajo y realizar backups de volúmenes persistentes mediante Cloud Backup Service. BlueXP admite la puesta en marcha de Astra Trident con el operador de Trident, ya sea manualmente o mediante Helm.

Uso `tridentctl`

Si tiene una puesta en marcha existente que debe actualizarse o si desea personalizar altamente su puesta en marcha, debe tener en cuenta . Este es el método convencional de puesta en marcha de Astra Trident.

Puede hacerlo Para generar los manifiestos de los recursos de Trident. Esto incluye la implementación, el conjunto demoníaco, la cuenta de servicio y el rol de clúster que crea Astra Trident como parte de su instalación.

> A partir de la versión 22.04, las claves AES ya no se regenerarán cada vez que se instale Astra Trident. Con este lanzamiento, Astra Trident instalará un nuevo objeto secreto que persiste en todas las instalaciones. Esto significa que, `tridentctl` En la versión 22.04 puede desinstalar versiones anteriores de Trident, pero las versiones anteriores no pueden desinstalar instalaciones de 22.04.
> Seleccione la instalación *method* adecuada.

Elija el modo de instalación

Determine el proceso de implementación según el *installation mode* (Standard, Offline o Remote) requerido por su organización.

Instalación estándar

Esta es la forma más sencilla de instalar Astra Trident y funciona para la mayoría de los entornos que no imponen restricciones de red. El modo de instalación estándar usa registros predeterminados para almacenar Trident necesario (`docker.io`) Y CSI (`registry.k8s.io`) imágenes.

Cuando se utiliza el modo estándar, el instalador de Astra Trident:

- Obtiene las imágenes del contenedor por Internet
- Crea una implementación o un conjunto de nodos demonset, que hace girar las pods de Astra Trident en todos los nodos elegibles del clúster de Kubernetes

Instalación sin conexión

Es posible que se requiera el modo de instalación sin conexión en una ubicación segura o con un sistema de activación por aire. En este escenario, puede crear un único registro privado duplicado o dos registros reflejados para almacenar las imágenes Trident y CSI necesarias.

> Independientemente de la configuración del registro, las imágenes CSI deben residir en un registro.

Instalación remota

A continuación se ofrece una descripción general de alto nivel del proceso de instalación remota:

- Despliegue la versión adecuada de `kubectl` En la máquina remota desde la que desea poner en marcha Astra Trident.
- Copie los archivos de configuración del clúster de Kubernetes y establezca el `KUBECONFIG` variable de entorno en el equipo remoto.
- Inicie un `kubectl get nodes` Comando para verificar que puede conectarse al clúster de Kubernetes necesario.
- Complete la implementación desde la máquina remota mediante los pasos de instalación estándar.

Seleccione el proceso según el método y el modo

Después de tomar sus decisiones, seleccione el proceso apropiado.

Método	Modo de instalación
Operador de Trident (manualmente)	"Instalación estándar" "Instalación sin conexión"
Operador Trident (Helm)	"Instalación estándar" "Instalación sin conexión"

Método	Modo de instalación
`tridentctl`	"Instalación estándar o sin conexión"

Moverse entre los métodos de instalación

Puede decidir cambiar el método de instalación. Antes de hacerlo, tenga en cuenta lo siguiente:

- Utilice siempre el mismo método para instalar y desinstalar Astra Trident. Si ha implementado con `tridentctl`, debe utilizar la versión adecuada de `tridentctl` Binario para desinstalar Astra Trident. Del mismo modo, si está desplegando con el operador, debe editar el `TridentOrchestrator` CR y SET `spec.uninstall=true` Para desinstalar Astra Trident.
- Si tiene una implementación basada en operador que desea quitar y utilizar en su lugar `tridentctl` Para poner en marcha Astra Trident, primero debe editar `TridentOrchestrator` y ajustar `spec.uninstall=true` Para desinstalar Astra Trident. A continuación, elimínelo `TridentOrchestrator` y la puesta en marcha del operador. A continuación, puede realizar la instalación mediante `tridentctl`.
- Si tiene una puesta en marcha manual basada en el operador y desea utilizar la puesta en marcha del operador de Trident basado en Helm, primero debe desinstalar manualmente al operador y, a continuación, llevar a cabo la instalación de Helm. De este modo, Helm puede poner en marcha el operador Trident con las etiquetas y anotaciones necesarias. Si no lo hace, la puesta en marcha del operador de Trident basado en Helm generará un error de validación de la etiqueta y un error de validación de la anotación. Si usted tiene un `tridentctl`La implementación basada en , puede utilizar la puesta en marcha basada en Helm sin que se produzcan problemas.

Otras opciones de configuración conocidas

Al instalar Astra Trident en productos de la cartera tanzu de VMware:

- El clúster debe admitir cargas de trabajo con privilegios.
- La `--kubelet-dir` el indicador se debe establecer en la ubicación del directorio kubelet. De forma predeterminada, esta es `/var/vcap/data/kubelet`.

  Especificación de la ubicación del kubelet mediante `--kubelet-dir` Sabe que funciona para el operador, Helm y. `tridentctl` implementaciones.

Realice la instalación mediante el operador Trident

Implemente manualmente el operador de Trident (modo estándar)

Es posible poner en marcha manualmente el operador de Trident para instalar Astra Trident. Este proceso se aplica a instalaciones en las que las imágenes de contenedor requeridas por Astra Trident no se almacenan en un registro privado. Si dispone de un registro de imágenes privado, utilice "proceso de puesta en marcha sin conexión".

Información vital sobre Astra Trident 23,04

- Debe leer la siguiente información crítica sobre Astra Trident.*

** información bï¿½tico sobre Astra Tridbï¿½tico **

- Kubernetes 1,27 ahora es compatible con Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident cumple estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en el archivo multipath.conf.

  Uso de la configuración sin multivía o el uso de `find_multipaths: yes` o. `find_multipaths: smart` el valor del archivo multipath.conf provocará fallos de montaje. Trident ha recomendado el uso de `find_multipaths: no` desde la versión 21.07.

Implemente manualmente el operador de Trident e instale Trident

Revisar "descripción general de la instalación" para asegurarse de cumplir con los requisitos previos de instalación y seleccionar la opción de instalación correcta para el entorno.

Antes de empezar

Antes de iniciar la instalación, inicie sesión en el host Linux y compruebe que esté gestionando un trabajo, "Clúster de Kubernetes compatible" y que tenga los privilegios necesarios.

Con OpenShift, utilícelo `oc` en lugar de `kubectl` en todos los ejemplos que siguen, e inicie sesión como **system:admin** primero ejecutando `oc login -u system:admin` o. `oc login -u kube-admin`.

1. Compruebe su versión de Kubernetes:

```
kubectl version
```

2. Comprobar los privilegios de administrador de clúster:

```
kubectl auth can-i '*' '*' --all-namespaces
```

3. Compruebe que puede iniciar un pod que utilice una imagen de Docker Hub para llegar al sistema de almacenamiento a través de la red de pod:

```
kubectl run -i --tty ping --image=busybox --restart=Never --rm -- \
  ping <management IP>
```

Paso 1: Descargue el paquete de instalación de 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 versión más reciente del instalador de Trident "La sección *Assets* de GitHub".

```
wget https://github.com/NetApp/trident/releases/download/v23.04.0/trident-
installer-23.04.0.tar.gz
tar -xf trident-installer-23.04.0.tar.gz
cd trident-installer
```

Paso 2: Cree la `TridentOrchestrator` CRD

Cree el `TridentOrchestrator` Definición de recurso personalizado (CRD). Creará una `TridentOrchestrator` Recursos personalizados más adelante. Use la versión adecuada de CRD YAML en `deploy/crds` para crear la `TridentOrchestrator` CRD.

```
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
```

Paso 3: Ponga en marcha el operador de Trident

El instalador de Astra Trident proporciona un archivo de paquete que se puede utilizar para instalar el operador y crear objetos asociados. El archivo de paquete es una forma sencilla de poner en marcha el operador e instalar Astra Trident con una configuración predeterminada.

- Para los clústeres que ejecutan Kubernetes 1,24 o una versión anterior, utilice `bundle_pre_1_25.yaml`.
- Para los clústeres que ejecutan Kubernetes 1,25 o posterior, utilice `bundle_post_1_25.yaml`.

Antes de empezar

- De forma predeterminada, el instalador de Trident implementa el operador en la `trident` espacio de nombres. Si la `trident` el espacio de nombres no existe, créelo con:

  ```
  kubectl apply -f deploy/namespace.yaml
  ```

- Para implementar el operador en un espacio de nombres distinto del `trident` espacio de nombres, actualización `serviceaccount.yaml`, `clusterrolebinding.yaml` y.. `operator.yaml` y genere el archivo del paquete con el `kustomization.yaml`.

  a. Cree el `kustomization.yaml` con el siguiente comando donde está *<bundle>* `bundle_pre_1_25` o. `bundle_post_1_25` Según su versión de Kubernetes.

  ```
  cp kustomization_<bundle>.yaml kustomization.yaml
  ```

  b. Compile el paquete con el siguiente comando donde está *<bundle>* `bundle_pre_1_25` o. `bundle_post_1_25` Según su versión de Kubernetes.

  ```
  kubectl kustomize deploy/ > deploy/<bundle>.yaml
  ```

Pasos

1. Crear los recursos e implementar el operador:

```
kubectl create -f deploy/<bundle>.yaml
```

2. Compruebe que se han creado el operador, el despliegue y los replicasets.

```
kubectl get all -n <operator-namespace>
```

> Solo debe haber **una instancia** del operador en un clúster de Kubernetes. No cree varias implementaciones del operador Trident.

Paso 4: Cree el `TridentOrchestrator` E instale Trident

Ahora puede crear el `TridentOrchestrator` E instale Astra Trident. Opcionalmente, puede hacerlo "Personalice su instalación de Trident" uso de los atributos de la `TridentOrchestrator` espec.

```
kubectl create -f deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created

kubectl describe torc trident

Name:        trident
Namespace:
Labels:      <none>
Annotations: <none>
API Version: trident.netapp.io/v1
Kind:        TridentOrchestrator
...
Spec:
  Debug:     true
  Namespace: trident
Status:
  Current Installation Params:
    IPv6:                      false
    Autosupport Hostname:
    Autosupport Image:         netapp/trident-autosupport:23.04
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                     true
    Image Pull Secrets:
    Image Registry:
    k8sTimeout:           30
    Kubelet Dir:          /var/lib/kubelet
    Log Format:           text
    Silence Autosupport:  false
    Trident Image:        netapp/trident:23.04.0
  Message:                  Trident installed  Namespace:
trident
  Status:                   Installed
  Version:                  v23.04.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

Compruebe la instalación

Existen varias formas de verificar su instalación.

Uso `TridentOrchestrator` estado

El estado de `TridentOrchestrator` Indica si la instalación se realizó correctamente y muestra la versión de

Trident instalada. Durante la instalación, el estado de `TridentOrchestrator` cambios de `Installing` para `Installed`. Si observa la `Failed` y el operador no puede recuperar por sí solo, "compruebe los registros".

Estado	Descripción
Instalación	El operador está instalando Astra Trident con este método `TridentOrchestrator` CR.
Instalado	Astra Trident se ha instalado correctamente.
Desinstalando	El operador está desinstalando Astra Trident, porque `spec.uninstall=true`.
Desinstalado	Astra Trident se desinstala.
Error	El operador no ha podido instalar, aplicar parches, actualizar o desinstalar Astra Trident; el operador intentará automáticamente recuperarse de este estado. Si este estado continúa, necesitará solucionar problemas.
Actualizando	El operador está actualizando una instalación existente.
Error	La `TridentOrchestrator` no se utiliza. Otro ya existe.

Uso del estado de creación de pod

Para confirmar si la instalación de Astra Trident ha finalizado, revise el estado de los pods creados:

```
kubectl get pods -n trident

NAME                                READY   STATUS    RESTARTS
AGE
trident-controller-7d466bf5c7-v4cpw        6/6     Running   0
1m
trident-node-linux-mr6zc                   2/2     Running   0
1m
trident-node-linux-xrp7w                   2/2     Running   0
1m
trident-node-linux-zh2jt                   2/2     Running   0
1m
trident-operator-766f7b8658-ldzsv          1/1     Running   0
3m
```

Uso `tridentctl`

Puede utilizar `tridentctl` Para comprobar la versión de Astra Trident instalada.

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 23.04.0        | 23.04.0        |
+----------------+----------------+
```

El futuro

Ahora es posible "cree una clase de back-end y almacenamiento, aprovisione un volumen y monte el volumen en un pod".

Implemente manualmente el operador Trident (modo sin conexión).

Es posible poner en marcha manualmente el operador de Trident para instalar Astra Trident. Este proceso se aplica a instalaciones en las que las imágenes de contenedor requeridas por Astra Trident se almacenan en un registro privado. Si no dispone de un registro de imágenes privado, utilice "proceso de implementación estándar".

Información vital sobre Astra Trident 23,04

- Debe leer la siguiente información crítica sobre Astra Trident.*

información bï¿½tico sobre Astra Tridbï¿½tico

- Kubernetes 1,27 ahora es compatible con Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident cumple estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en el archivo multipath.conf.

  Uso de la configuración sin multivía o el uso de `find_multipaths: yes` o. `find_multipaths: smart` el valor del archivo multipath.conf provocará fallos de montaje. Trident ha recomendado el uso de `find_multipaths: no` desde la versión 21.07.

Implemente manualmente el operador de Trident e instale Trident

Revisar "descripción general de la instalación" para asegurarse de cumplir con los requisitos previos de instalación y seleccionar la opción de instalación correcta para el entorno.

Antes de empezar

Inicie sesión en el host Linux y compruebe que está gestionando un funcionamiento y. "Clúster de Kubernetes compatible" y que tenga los privilegios necesarios.

Con OpenShift, utilícelo `oc` en lugar de `kubectl` en todos los ejemplos que siguen, e inicie sesión como **system:admin** primero ejecutando `oc login -u system:admin` o. `oc login -u kube-admin`.

1. Compruebe su versión de Kubernetes:

```
kubectl version
```

2. Comprobar los privilegios de administrador de clúster:

```
kubectl auth can-i '*' '*' --all-namespaces
```

3. Compruebe que puede iniciar un pod que utilice una imagen de Docker Hub para llegar al sistema de almacenamiento a través de la red de pod:

```
kubectl run -i --tty ping --image=busybox --restart=Never --rm -- \
  ping <management IP>
```

Paso 1: Descargue el paquete de instalación de 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 versión más reciente del instalador de Trident "La sección *Assets* de GitHub".

```
wget https://github.com/NetApp/trident/releases/download/v23.04.0/trident-
installer-23.04.0.tar.gz
tar -xf trident-installer-23.04.0.tar.gz
cd trident-installer
```

Paso 2: Cree la `TridentOrchestrator` CRD

Cree el `TridentOrchestrator` Definición de recurso personalizado (CRD). Creará una `TridentOrchestrator` Recursos personalizados más adelante. Use la versión adecuada de CRD YAML en `deploy/crds` para crear la `TridentOrchestrator` CRD:

```
kubectl create -f deploy/crds/<VERSION>.yaml
```

Paso 3: Actualice la ubicación del registro en el operador

Pulg `/deploy/operator.yaml`, actualizar `image: docker.io/netapp/trident-operator:23.04.0` para reflejar la ubicación del registro de imágenes. Su "Imágenes Trident y CSI" Se pueden ubicar en un registro o en diferentes registros, pero todas las imágenes CSI deben estar ubicadas en el mismo registro. Por ejemplo:

- `image: <your-registry>/trident-operator:23.04.0` si todas las imágenes están ubicadas en un registro.

- `image: <your-registry>/netapp/trident-operator:23.04.0` Si su imagen Trident se encuentra en un registro diferente de sus imágenes CSI.

Paso 4: Despliegue el operador Trident

El instalador de Astra Trident proporciona un archivo de paquete que se puede utilizar para instalar el operador y crear objetos asociados. El archivo de paquete es una forma sencilla de poner en marcha el operador e instalar Astra Trident con una configuración predeterminada.

- Para los clústeres que ejecutan Kubernetes 1,24 o una versión anterior, utilice `bundle_pre_1_25.yaml`.
- Para los clústeres que ejecutan Kubernetes 1,25 o posterior, utilice `bundle_post_1_25.yaml`.

Antes de empezar

- De forma predeterminada, el instalador de Trident implementa el operador en la `trident` espacio de nombres. Si la `trident` el espacio de nombres no existe, créelo con:

```
kubectl apply -f deploy/namespace.yaml
```

- Para implementar el operador en un espacio de nombres distinto del `trident` espacio de nombres, actualización `serviceaccount.yaml`, `clusterrolebinding.yaml` y.. `operator.yaml` y genere el archivo del paquete con el `kustomization.yaml`.

  a. Cree el `kustomization.yaml` con el siguiente comando donde está *<bundle>* `bundle_pre_1_25` o. `bundle_post_1_25` Según su versión de Kubernetes.

  ```
  cp kustomization_<bundle>.yaml kustomization.yaml
  ```

  b. Compile el paquete con el siguiente comando donde está *<bundle>* `bundle_pre_1_25` o. `bundle_post_1_25` Según su versión de Kubernetes.

  ```
  kubectl kustomize deploy/ > deploy/<bundle>.yaml
  ```

Pasos

1. Crear los recursos e implementar el operador:

```
kubectl kustomize deploy/ > deploy/<bundle>.yaml
```

2. Compruebe que se han creado el operador, el despliegue y los replicasets.

```
kubectl get all -n <operator-namespace>
```

> Solo debe haber **una instancia** del operador en un clúster de Kubernetes. No cree varias implementaciones del operador Trident.

Paso 5: Actualice la ubicación del registro de imágenes en el `TridentOrchestrator`

Su "Imágenes Trident y CSI" Se pueden ubicar en un registro o en diferentes registros, pero todas las imágenes CSI deben estar ubicadas en el mismo registro. Actualizar `deploy/crds/tridentorchestrator_cr.yaml` para agregar las especificaciones de ubicación adicionales basadas en la configuración de su registro.

Imágenes en un registro

```
imageRegistry: "<your-registry>"
autosupportImage: "<your-registry>/trident-autosupport:23.04"
tridentImage: "<your-registry>/trident:23.04.0"
```

Imágenes en diferentes registros

Debe añadir `sig-storage` para la `imageRegistry` para usar diferentes ubicaciones de registro.

```
imageRegistry: "<your-registry>/sig-storage"
autosupportImage: "<your-registry>/netapp/trident-autosupport:23.04"
tridentImage: "<your-registry>/netapp/trident:23.04.0"
```

Paso 6: Cree el `TridentOrchestrator` E instale Trident

Ahora puede crear el `TridentOrchestrator` E instale Astra Trident. Si lo desea, puede ir más allá "Personalice su instalación de Trident" uso de los atributos de la `TridentOrchestrator` espec. En el siguiente ejemplo se muestra una instalación donde las imágenes Trident y CSI se encuentran en diferentes registros.

```
kubectl create -f deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created

kubectl describe torc trident

Name:        trident
Namespace:
Labels:      <none>
Annotations: <none>
API Version: trident.netapp.io/v1
Kind:        TridentOrchestrator
...
Spec:
  Autosupport Image:  <your-registry>/netapp/trident-autosupport:23.04
  Debug:              true
  Image Registry:     <your-registry>/sig-storage
  Namespace:          trident
  Trident Image:      <your-registry>/netapp/trident:23.04.0
Status:
  Current Installation Params:
    IPv6:                       false
    Autosupport Hostname:
    Autosupport Image:          <your-registry>/netapp/trident-
autosupport:23.04
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                      true
    Http Request Timeout:       90s
    Image Pull Secrets:
    Image Registry:       <your-registry>/sig-storage
    k8sTimeout:           30
    Kubelet Dir:          /var/lib/kubelet
    Log Format:           text
    Probe Port:           17546
    Silence Autosupport:  false
    Trident Image:        <your-registry>/netapp/trident:23.04.0
  Message:                Trident installed
  Namespace:              trident
  Status:                 Installed
  Version:                v23.04.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

Compruebe la instalación

Existen varias formas de verificar su instalación.

Uso `TridentOrchestrator` estado

El estado de `TridentOrchestrator` Indica si la instalación se realizó correctamente y muestra la versión de Trident instalada. Durante la instalación, el estado de `TridentOrchestrator` cambios de `Installing` para `Installed`. Si observa la `Failed` y el operador no puede recuperar por sí solo, "compruebe los registros".

Estado	Descripción
Instalación	El operador está instalando Astra Trident con este método `TridentOrchestrator` CR.
Instalado	Astra Trident se ha instalado correctamente.
Desinstalando	El operador está desinstalando Astra Trident, porque `spec.uninstall=true`.
Desinstalado	Astra Trident se desinstala.
Error	El operador no ha podido instalar, aplicar parches, actualizar o desinstalar Astra Trident; el operador intentará automáticamente recuperarse de este estado. Si este estado continúa, necesitará solucionar problemas.
Actualizando	El operador está actualizando una instalación existente.
Error	La `TridentOrchestrator` no se utiliza. Otro ya existe.

Uso del estado de creación de pod

Para confirmar si la instalación de Astra Trident ha finalizado, revise el estado de los pods creados:

```
kubectl get pods -n trident

NAME                                   READY   STATUS    RESTARTS
AGE
trident-controller-7d466bf5c7-v4cpw    6/6     Running   0
1m
trident-node-linux-mr6zc               2/2     Running   0
1m
trident-node-linux-xrp7w               2/2     Running   0
1m
trident-node-linux-zh2jt               2/2     Running   0
1m
trident-operator-766f7b8658-ldzsv      1/1     Running   0
3m
```

Uso `tridentctl`

Puede utilizar `tridentctl` Para comprobar la versión de Astra Trident instalada.

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 23.04.0        | 23.04.0        |
+----------------+----------------+
```

El futuro

Ahora es posible "cree una clase de back-end y almacenamiento, aprovisione un volumen y monte el volumen en un pod".

Puesta en marcha del operador de Trident con Helm (modo estándar)

Puede poner en marcha el operador de Trident e instalar Astra Trident con Helm. Este proceso se aplica a instalaciones en las que las imágenes de contenedor requeridas por Astra Trident no se almacenan en un registro privado. Si dispone de un registro de imágenes privado, utilice "proceso de puesta en marcha sin conexión".

Información vital sobre Astra Trident 23,04

- Debe leer la siguiente información crítica sobre Astra Trident.*

** información bï¿½tico sobre Astra Tridbï¿½tico **

- Kubernetes 1,27 ahora es compatible con Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident cumple estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en el archivo multipath.conf.

  Uso de la configuración sin multivía o el uso de `find_multipaths: yes` o. `find_multipaths: smart` el valor del archivo multipath.conf provocará fallos de montaje. Trident ha recomendado el uso de `find_multipaths: no` desde la versión 21.07.

Ponga en marcha el operador de Trident e instale Astra Trident con Helm

Usar Trident "Carta del timón" Es posible poner en marcha el operador de Trident e instalar Trident en un paso.

Revisar "descripción general de la instalación" para asegurarse de cumplir con los requisitos previos de instalación y seleccionar la opción de instalación correcta para el entorno.

Antes de empezar

Además de la "requisitos previos a la implementación" que necesita "Versión timón 3".

Pasos

1. Añada el repositorio de Astra Trident Helm:

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

2. Uso `helm install` y especifique un nombre para la implementación como en el ejemplo siguiente donde `23.04.0` Es la versión de Astra Trident que está instalando.

```
helm install <name> netapp-trident/trident-operator --version 23.04.0
--create-namespace --namespace <trident-namespace>
```

> Si ya creó un espacio de nombres para Trident, el `--create-namespace` el parámetro no creará un espacio de nombres adicional.

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

Pasar los datos de configuración durante la instalación

Existen dos formas de pasar los datos de configuración durante la instalación:

Opción	Descripción
`--values` (o. `-f`)	Especifique un archivo YAML con anulaciones. Esto se puede especificar varias veces y el archivo de la derecha tendrá prioridad.
`--set`	Especifique anulaciones en la línea de comandos.

Por ejemplo, para cambiar el valor predeterminado de `debug`, ejecute lo siguiente `--set` comando donde `23.04.0` Es la versión de Astra Trident que está instalando:

```
helm install <name> netapp-trident/trident-operator --version 23.04.0
--create-namespace --namespace --set tridentDebug=true
```

Opciones de configuración

Esta tabla y la `values.yaml` File, que forma parte del gráfico Helm, proporciona la lista de claves y sus valores predeterminados.

Opción	Descripción	Predeterminado
`nodeSelector`	Etiquetas de nodo para la asignación de pod	
`podAnnotations`	Anotaciones del pod	
`deploymentAnnotations`	Anotaciones de despliegue	
`tolerations`	Toleraciones para la asignación de POD	
`affinity`	Afinidad para la asignación de pod	
`tridentControllerPluginNodeSelector`	Selectores de nodos adicionales para POD. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`tridentControllerPluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`tridentNodePluginNodeSelector`	Selectores de nodos adicionales para POD. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	

Opción	Descripción	Predeterminado
`tridentNodePluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`imageRegistry`	Identifica el registro del `trident-operator`, `trident`, y otras imágenes. Déjelo vacío para aceptar el valor predeterminado.	""
`imagePullPolicy`	Establece la política de extracción de imágenes para el `trident-operator`.	`IfNotPresent`
`imagePullSecrets`	Establece los secretos de extracción de imágenes para el `trident-operator`, `trident`, y otras imágenes.	
`kubeletDir`	Permite anular la ubicación del host del estado interno de kubelet.	"/var/lib/kubelet"
`operatorLogLevel`	Permite establecer el nivel de registro del operador Trident en: `trace`, `debug`, `info`, `warn`, `error`, o. `fatal`.	`"info"`
`operatorDebug`	Permite configurar en debug el nivel de registro del operador Trident.	`true`
`operatorImage`	Permite la sustitución completa de la imagen durante `trident-operator`.	""
`operatorImageTag`	Permite sobrescribir la etiqueta del `trident-operator` imagen.	""
`tridentIPv6`	Permite permitir que Astra Trident funcione en clústeres de IPv6.	`false`
`tridentK8sTimeout`	Anula el tiempo de espera predeterminado de 30 segundos para la mayoría de las operaciones de la API de Kubernetes (si no es cero, en segundos).	`0`
`tridentHttpRequestTimeout`	Sustituye el timeout por defecto de 90 segundos para las solicitudes HTTP, con `0s` ser una duración infinita para el timeout. No se permiten valores negativos.	`"90s"`
`tridentSilenceAutosupport`	Permite deshabilitar la generación de informes periódicos de AutoSupport de Astra Trident.	`false`

Opción	Descripción	Predeterminado
`tridentAutosupportImageTag`	Permite sobrescribir la etiqueta de la imagen del contenedor AutoSupport de Astra Trident.	`<version>`
`tridentAutosupportProxy`	Permite que el contenedor Astra Trident AutoSupport telefonee a casa a través de un proxy HTTP.	""
`tridentLogFormat`	Establece el formato de registro de Astra Trident (`text` o. `json`).	`"text"`
`tridentDisableAuditLog`	Deshabilita el registro de auditorías de Astra Trident.	`true`
`tridentLogLevel`	Permite establecer el nivel de registro de Astra Trident en: `trace`, `debug`, `info`, `warn`, `error`, o. `fatal`.	`"info"`
`tridentDebug`	Permite establecer el nivel de registro de Astra Trident `debug`.	`false`
`tridentLogWorkflows`	Permite habilitar flujos de trabajo específicos de Astra Trident para el registro de seguimiento o la supresión de registros.	""
`tridentLogLayers`	Permite habilitar capas específicas de Astra Trident para el registro de seguimiento o la supresión de registros.	""
`tridentImage`	Permite anular por completo la imagen de Astra Trident.	""
`tridentImageTag`	Permite sobrescribir la etiqueta de la imagen para Astra Trident.	""
`tridentProbePort`	Permite sobrescribir el puerto predeterminado utilizado para las sondas de vida/preparación de Kubernetes.	""
`windows`	Permite instalar Astra Trident en el nodo de trabajo de Windows.	`false`
`enableForceDetach`	Permite habilitar la función Forzar separación.	`false`
`excludePodSecurityPolicy`	Excluye la política de seguridad del pod del operador de la creación.	`false`

Descripción de los pods de la controladora y los pods de nodo

Astra Trident se ejecuta como un único pod de la controladora, más un pod de nodos en cada nodo de trabajo del clúster. El pod del nodo debe ejecutarse en cualquier host en el que desee montar potencialmente un volumen Astra Trident.

Kubernetes "selectores de nodos" y.. "toleraciones y tintes" se utilizan para restringir un pod para ejecutarse en un nodo concreto o preferido. Uso del "ControllerPlugin" y. `NodePlugin`, puede especificar restricciones y anulaciones.

- El complemento de la controladora se ocupa del aprovisionamiento y la gestión de volúmenes, como snapshots y redimensionamiento.
- El complemento de nodo se encarga de conectar el almacenamiento al nodo.

El futuro

Ahora es posible "cree una clase de back-end y almacenamiento, aprovisione un volumen y monte el volumen en un pod".

Implementar el operador de Trident con Helm (modo sin conexión)

Puede poner en marcha el operador de Trident e instalar Astra Trident con Helm. Este proceso se aplica a instalaciones en las que las imágenes de contenedor requeridas por Astra Trident se almacenan en un registro privado. Si no dispone de un registro de imágenes privado, utilice "proceso de implementación estándar".

Información vital sobre Astra Trident 23,04

- Debe leer la siguiente información crítica sobre Astra Trident.*

** información bï¿½tico sobre Astra Tridbï¿½tico **

- Kubernetes 1,27 ahora es compatible con Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident cumple estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en el archivo multipath.conf.

  Uso de la configuración sin multivía o el uso de `find_multipaths: yes` o. `find_multipaths: smart` el valor del archivo multipath.conf provocará fallos de montaje. Trident ha recomendado el uso de `find_multipaths: no` desde la versión 21.07.

Ponga en marcha el operador de Trident e instale Astra Trident con Helm

Usar Trident "Carta del timón" Es posible poner en marcha el operador de Trident e instalar Trident en un paso.

Revisar "descripción general de la instalación" para asegurarse de cumplir con los requisitos previos de instalación y seleccionar la opción de instalación correcta para el entorno.

Antes de empezar

Además de la "requisitos previos a la implementación" que necesita "Versión timón 3".

Pasos

1. Añada el repositorio de Astra Trident Helm:

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

2. Uso `helm install` y especifique un nombre para su implementación y ubicación del registro de imágenes. Su "Imágenes Trident y CSI" Se pueden ubicar en un registro o en diferentes registros, pero todas las imágenes CSI deben estar ubicadas en el mismo registro. En los ejemplos: `23.04.0` Es la versión de Astra Trident que está instalando.

Imágenes en un registro

```
helm install <name> netapp-trident/trident-operator --version
23.04.0 --set imageRegistry=<your-registry> --create-namespace
--namespace <trident-namespace>
```

Imágenes en diferentes registros

Debe añadir `sig-storage` para la `imageRegistry` para usar diferentes ubicaciones de registro.

```
helm install <name> netapp-trident/trident-operator --version
23.04.0 --set imageRegistry=<your-registry>/sig-storage --set
operatorImage=<your-registry>/netapp/trident-operator:23.04.0 --set
tridentAutosupportImage=<your-registry>/netapp/trident-
autosupport:23.04 --set tridentImage=<your-
registry>/netapp/trident:23.04.0 --create-namespace --namespace
<trident-namespace>
```

Si ya creó un espacio de nombres para Trident, el `--create-namespace` el parámetro no creará un espacio de nombres adicional.

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

Pasar los datos de configuración durante la instalación

Existen dos formas de pasar los datos de configuración durante la instalación:

Opción	Descripción
`--values` (o. `-f`)	Especifique un archivo YAML con anulaciones. Esto se puede especificar varias veces y el archivo de la derecha tendrá prioridad.
`--set`	Especifique anulaciones en la línea de comandos.

Por ejemplo, para cambiar el valor predeterminado de `debug`, ejecute lo siguiente `--set` comando donde `23.04.0` Es la versión de Astra Trident que está instalando:

```
helm install <name> netapp-trident/trident-operator --version 23.04.0
--create-namespace --namespace --set tridentDebug=true
```

Opciones de configuración

Esta tabla y la `values.yaml` File, que forma parte del gráfico Helm, proporciona la lista de claves y sus valores predeterminados.

Opción	Descripción	Predeterminado
`nodeSelector`	Etiquetas de nodo para la asignación de pod	
`podAnnotations`	Anotaciones del pod	
`deploymentAnnotations`	Anotaciones de despliegue	
`tolerations`	Toleraciones para la asignación de POD	
`affinity`	Afinidad para la asignación de pod	
`tridentControllerPluginNodeSelector`	Selectores de nodos adicionales para POD. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`tridentControllerPluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`tridentNodePluginNodeSelector`	Selectores de nodos adicionales para POD. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`tridentNodePluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`imageRegistry`	Identifica el registro del `trident-operator`, `trident`, y otras imágenes. Déjelo vacío para aceptar el valor predeterminado.	""
`imagePullPolicy`	Establece la política de extracción de imágenes para el `trident-operator`.	`IfNotPresent`

Opción	Descripción	Predeterminado
`imagePullSecrets`	Establece los secretos de extracción de imágenes para el `trident-operator`, `trident`, y otras imágenes.	
`kubeletDir`	Permite anular la ubicación del host del estado interno de kubelet.	"/var/lib/kubelet"
`operatorLogLevel`	Permite establecer el nivel de registro del operador Trident en: `trace`, `debug`, `info`, `warn`, `error`, o. `fatal`.	`"info"`
`operatorDebug`	Permite configurar en debug el nivel de registro del operador Trident.	`true`
`operatorImage`	Permite la sustitución completa de la imagen durante `trident-operator`.	""
`operatorImageTag`	Permite sobrescribir la etiqueta del `trident-operator` imagen.	""
`tridentIPv6`	Permite permitir que Astra Trident funcione en clústeres de IPv6.	`false`
`tridentK8sTimeout`	Anula el tiempo de espera predeterminado de 30 segundos para la mayoría de las operaciones de la API de Kubernetes (si no es cero, en segundos).	`0`
`tridentHttpRequestTimeout`	Sustituye el timeout por defecto de 90 segundos para las solicitudes HTTP, con `0s` ser una duración infinita para el timeout. No se permiten valores negativos.	`"90s"`
`tridentSilenceAutosupport`	Permite deshabilitar la generación de informes periódicos de AutoSupport de Astra Trident.	`false`
`tridentAutosupportImageTag`	Permite sobrescribir la etiqueta de la imagen del contenedor AutoSupport de Astra Trident.	`<version>`
`tridentAutosupportProxy`	Permite que el contenedor Astra Trident AutoSupport telefonee a casa a través de un proxy HTTP.	""
`tridentLogFormat`	Establece el formato de registro de Astra Trident (`text` o. `json`).	`"text"`
`tridentDisableAuditLog`	Deshabilita el registro de auditorías de Astra Trident.	`true`

Opción	Descripción	Predeterminado
`tridentLogLevel`	Permite establecer el nivel de registro de Astra Trident en: `trace`, `debug`, `info`, `warn`, `error`, o. `fatal`.	`"info"`
`tridentDebug`	Permite establecer el nivel de registro de Astra Trident `debug`.	`false`
`tridentLogWorkflows`	Permite habilitar flujos de trabajo específicos de Astra Trident para el registro de seguimiento o la supresión de registros.	""
`tridentLogLayers`	Permite habilitar capas específicas de Astra Trident para el registro de seguimiento o la supresión de registros.	""
`tridentImage`	Permite anular por completo la imagen de Astra Trident.	""
`tridentImageTag`	Permite sobrescribir la etiqueta de la imagen para Astra Trident.	""
`tridentProbePort`	Permite sobrescribir el puerto predeterminado utilizado para las sondas de vida/preparación de Kubernetes.	""
`windows`	Permite instalar Astra Trident en el nodo de trabajo de Windows.	`false`
`enableForceDetach`	Permite habilitar la función Forzar separación.	`false`
`excludePodSecurityPolicy`	Excluye la política de seguridad del pod del operador de la creación.	`false`

Descripción de los pods de la controladora y los pods de nodo

Astra Trident se ejecuta como un único pod de la controladora, más un pod de nodos en cada nodo de trabajo del clúster. El pod del nodo debe ejecutarse en cualquier host en el que desee montar potencialmente un volumen Astra Trident.

Kubernetes "selectores de nodos" y.. "toleraciones y tintes" se utilizan para restringir un pod para ejecutarse en un nodo concreto o preferido. Uso del "ControllerPlugin" y. `NodePlugin`, puede especificar restricciones y anulaciones.

- El complemento de la controladora se ocupa del aprovisionamiento y la gestión de volúmenes, como snapshots y redimensionamiento.
- El complemento de nodo se encarga de conectar el almacenamiento al nodo.

El futuro

Ahora es posible "cree una clase de back-end y almacenamiento, aprovisione un volumen y monte el volumen en un pod".

Personalice la instalación del operador de Trident

El operador Trident le permite personalizar la instalación de Astra Trident con los atributos del `TridentOrchestrator` spec. Si desea personalizar la instalación más allá de qué `TridentOrchestrator` los argumentos lo permiten, considere usar `tridentctl` Para generar manifiestos YAML personalizados y modificarlos según sea necesario.

Descripción de los pods de la controladora y los pods de nodo

Astra Trident se ejecuta como un único pod de la controladora, más un pod de nodos en cada nodo de trabajo del clúster. El pod del nodo debe ejecutarse en cualquier host en el que desee montar potencialmente un volumen Astra Trident.

Kubernetes "selectores de nodos" y.. "toleraciones y tintes" se utilizan para restringir un pod para ejecutarse en un nodo concreto o preferido. Uso del "ControllerPlugin" y. `NodePlugin`, puede especificar restricciones y anulaciones.

- El complemento de la controladora se ocupa del aprovisionamiento y la gestión de volúmenes, como snapshots y redimensionamiento.
- El complemento de nodo se encarga de conectar el almacenamiento al nodo.

Opciones de configuración

`spec.namespace` se especifica en `TridentOrchestrator` Para indicar el espacio de nombres en el que está instalado Astra Trident. Este parámetro **no se puede actualizar después de instalar Astra Trident**. Al intentar hacerlo, se genera el `TridentOrchestrator` estado a cambiar a. `Failed`. Astra Trident no está pensado para la migración entre espacios de nombres.

Esta tabla detalla `TridentOrchestrator` atributos.

Parámetro	Descripción	Predeterminado
`namespace`	Espacio de nombres para instalar Astra Trident en	"predeterminado"
`debug`	Habilite la depuración para Astra Trident	falso

Parámetro	Descripción	Predeterminado
`enableForceDetach`	`ontap-san` y.. `ontap-san-economy` solamente. Funciona con cierre de nodos no controlado (NGN) de Kubernetes para conceder a los administradores de clústeres la capacidad de migrar de forma segura cargas de trabajo con volúmenes montados a nodos nuevos en caso de que un nodo se vuelva en mal estado. **Esta es una característica experimental en 23,04.** Consulte Detalles acerca de forzar separación para obtener detalles importantes.	`false`
`windows`	Ajuste a. `true` Permite la instalación en nodos de trabajo de Windows.	falso
`useIPv6`	Instale Astra Trident sobre IPv6	falso
`k8sTimeout`	Tiempo de espera para las operaciones de Kubernetes	30 seg
`silenceAutosupport`	No envíe paquetes AutoSupport a NetApp automáticamente	falso
`autosupportImage`	La imagen contenedora para telemetría AutoSupport	«netapp/trident-autosupport:23,07»
`autosupportProxy`	La dirección/puerto de un proxy para enviar AutoSupport Telemetría	"http://proxy.example.com:8888"
`uninstall`	Una Marca utilizada para desinstalar Astra Trident	falso
`logFormat`	Formato de registro de Astra Trident para utilizar [text,json]	"texto"
`tridentImage`	Imagen de Astra Trident para instalar	«netapp/trident:23,07»
`imageRegistry`	Ruta de acceso al registro interno, del formato `<registry FQDN>[:port][/subpath]`	«k8s.gcr.io/sig-storage» (k8s 1,19+) o «quay.io/k8scsi»
`kubeletDir`	Ruta al directorio kubelet del host	«/var/lib/kubelet»

Parámetro	Descripción	Predeterminado
`wipeout`	Lista de recursos que se van a suprimir para realizar una eliminación completa de Astra Trident	
`imagePullSecrets`	Secretos para extraer imágenes de un registro interno	
`imagePullPolicy`	Establece la política de extracción de imágenes para el operador Trident. Valores válidos: `Always` para tirar siempre de la imagen. `IfNotPresent` para extraer la imagen solo si aún no existe en el nodo. `Never` para no tirar nunca de la imagen.	`IfNotPresent`
`controllerPluginNodeSelector`	Selectores de nodos adicionales para POD. Sigue el mismo formato que `pod.spec.nodeSelector`.	Sin valores predeterminados; opcional
`controllerPluginTolerations`	Anula la toleración de Kubernetes en pods. Sigue el mismo formato que `pod.spec.Tolerations`.	Sin valores predeterminados; opcional
`nodePluginNodeSelector`	Selectores de nodos adicionales para POD. Sigue el mismo formato que `pod.spec.nodeSelector`.	Sin valores predeterminados; opcional
`nodePluginTolerations`	Anula la toleración de Kubernetes en pods. Sigue el mismo formato que `pod.spec.Tolerations`.	Sin valores predeterminados; opcional

Para obtener más información sobre el formato de los parámetros del POD, consulte "Asignación de pods a nodos".

Detalles acerca de forzar separación

Forzar separación está disponible para `ontap-san` y.. `ontap-san-economy` solamente. Antes de habilitar la desconexión forzada, se debe habilitar el cierre de nodos (NGN) no controlado en el clúster de Kubernetes. Para obtener más información, consulte "Kubernetes: Cierre de nodo sin gracia".

Dado que Astra Trident se basa en NGN de Kubernetes, no lo elimine `out-of-service` mantiene un nodo en mal estado hasta que se reprograman todas las cargas de trabajo no tolerables. La aplicación o eliminación imprudente de la contaminación puede poner en peligro la protección de datos de back-end.

Cuando el administrador del clúster de Kubernetes ha aplicado el `node.kubernetes.io/out-of-`

`service=nodeshutdown:NoExecute` mancha al nodo y. `enableForceDetach` se establece en `true`, Astra Trident determinará el estado del nodo y:

1. Cese el acceso de I/O back-end para los volúmenes montados en ese nodo.
2. Marque el objeto de nodo de Astra Trident como `dirty` (no es seguro para las nuevas publicaciones).

> La controladora Trident rechazará nuevas solicitudes de volumen de publicación hasta que el nodo se vuelva a calificar (después de haberse marcado como `dirty`) Por el pod del nodo de Trident. No se aceptarán todas las cargas de trabajo programadas con una RVP montada (incluso después de que el nodo del clúster esté en buen estado y listo) hasta que Astra Trident pueda verificar el nodo `clean` (seguro para nuevas publicaciones).

Cuando se restaure el estado del nodo y se elimine el tinte, Astra Trident:

1. Identifique y limpie las rutas publicadas obsoletas en el nodo.
2. Si el nodo está en `cleanable` estado (se ha eliminado la contaminación de fuera de servicio y el nodo está en `Ready` estatal) Y todas las rutas obsoletas publicadas están limpias, Astra Trident reenviará el nodo como `clean` y permitir nuevos volúmenes publicados al nodo.

Configuraciones de ejemplo

Puede utilizar los atributos mencionados anteriormente al definir `TridentOrchestrator` para personalizar la instalación.

Ejemplo 1: Configuración personalizada básica

Este es un ejemplo de una configuración personalizada básica.

```
cat deploy/crds/tridentorchestrator_cr_imagepullsecrets.yaml
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullSecrets:
  - thisisasecret
```

Ejemplo 2: Implementar con selectores de nodos

Este ejemplo ilustra cómo se puede implementar Trident con los selectores de nodos:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  controllerPluginNodeSelector:
    nodetype: master
  nodePluginNodeSelector:
    storage: netapp
```

Ejemplo 3: Implementar en nodos de trabajo de Windows

Este ejemplo ilustra la implementación en un nodo de trabajo de Windows.

```
cat deploy/crds/tridentorchestrator_cr.yaml
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  windows: true
```

Instale utilizando trimentctl

Instale utilizando trimentctl

Puede instalar Astra Trident con `tridentctl`. Este proceso se aplica a instalaciones en las que las imágenes de contenedor requeridas por Astra Trident se almacenan o no en un registro privado. Para personalizar su `tridentctl` despliegue, consulte "Personalice la implementación trimentctl".

Información vital sobre Astra Trident 23,04

- Debe leer la siguiente información crítica sobre Astra Trident.*

** información bï¿½tico sobre Astra Tridbï¿½tico **

- Kubernetes 1,27 ahora es compatible con Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident cumple estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en el archivo multipath.conf.

  Uso de la configuración sin multivía o el uso de `find_multipaths: yes` o. `find_multipaths: smart` el valor del archivo multipath.conf provocará fallos de montaje. Trident ha recomendado el uso de `find_multipaths: no` desde la versión 21.07.

Instale Astra Trident con `tridentctl`

Revisar "descripción general de la instalación" para asegurarse de cumplir con los requisitos previos de instalación y seleccionar la opción de instalación correcta para el entorno.

Antes de empezar

Antes de iniciar la instalación, inicie sesión en el host Linux y compruebe que esté gestionando un trabajo, "Clúster de Kubernetes compatible" y que tenga los privilegios necesarios.

> Con OpenShift, utilícelo `oc` en lugar de `kubectl` en todos los ejemplos que siguen, e inicie sesión como **system:admin** primero ejecutando `oc login -u system:admin` o. `oc login -u kube-admin`.

1. Compruebe su versión de Kubernetes:

```
kubectl version
```

2. Comprobar los privilegios de administrador de clúster:

```
kubectl auth can-i '*' '*' --all-namespaces
```

3. Compruebe que puede iniciar un pod que utilice una imagen de Docker Hub para llegar al sistema de almacenamiento a través de la red de pod:

```
kubectl run -i --tty ping --image=busybox --restart=Never --rm -- \
  ping <management IP>
```

Paso 1: Descargue el paquete de instalación de Trident

El paquete de instalación de Astra Trident crea un pod Trident, configura los objetos CRD que se utilizan para mantener su estado e inicializa las sidecs CSI para realizar acciones como aprovisionar y adjuntar volúmenes a los hosts del clúster. Descargue y extraiga la versión más reciente del instalador de Trident "La sección *Assets* de GitHub". Actualice *<trident-installer-XX.XX.X.tar.gz>* en el ejemplo con la versión Astra Trident

seleccionada.

```
wget https://github.com/NetApp/trident/releases/download/v23.04.0/trident-
installer-23.04.0.tar.gz
tar -xf trident-installer-23.04.0.tar.gz
cd trident-installer
```

Paso 2: Instale Astra Trident

Instale Astra Trident en el espacio de nombres deseado ejecutando `tridentctl install` comando. Puede agregar argumentos adicionales para especificar la ubicación del registro de imágenes.

Modo estándar

```
./tridentctl install -n trident
```

Imágenes en un registro

```
./tridentctl install -n trident --image-registry <your-registry>
--autosupport-image <your-registry>/trident-autosupport:23.04 --trident
-image <your-registry>/trident:23.04.0
```

Imágenes en diferentes registros

Debe añadir `sig-storage` para la `imageRegistry` para usar diferentes ubicaciones de registro.

```
./tridentctl install -n trident --image-registry <your-registry>/sig-
storage --autosupport-image <your-registry>/netapp/trident-
autosupport:23.04 --trident-image <your-
registry>/netapp/trident:23.04.0
```

El estado de su instalación debería tener un aspecto parecido a este.

```
....
INFO Starting Trident installation.                namespace=trident
INFO Created service account.
INFO Created cluster role.
INFO Created cluster role binding.
INFO Added finalizers to custom resource definitions.
INFO Created Trident service.
INFO Created Trident secret.
INFO Created Trident deployment.
INFO Created Trident daemonset.
INFO Waiting for Trident pod to start.
INFO Trident pod started.                          namespace=trident
pod=trident-controller-679648bd45-cv2mx
INFO Waiting for Trident REST interface.
INFO Trident REST interface is up.                 version=23.04.0
INFO Trident installation succeeded.
....
```

Compruebe la instalación

Puede verificar la instalación con el estado de creación de un pod o. `tridentctl`.

Uso del estado de creación de pod

Para confirmar si la instalación de Astra Trident ha finalizado, revise el estado de los pods creados:

```
kubectl get pods -n trident

NAME                                READY   STATUS    RESTARTS   AGE
trident-controller-679648bd45-cv2mx   6/6     Running   0          5m29s
trident-node-linux-vgc8n              2/2     Running   0          5m29s
```

> Si el instalador no se completa correctamente o. `trident-controller-<generated id>` (`trident-csi-<generated id>` En versiones anteriores a 23.01) no tiene un estado **en ejecución**, la plataforma no estaba instalada. Uso `-d` para "activa el modo de depuración" y solucionar el problema.

Uso `tridentctl`

Puede utilizar `tridentctl` Para comprobar la versión de Astra Trident instalada.

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 23.04.0        | 23.04.0        |
+----------------+----------------+
```

Configuraciones de ejemplo

Ejemplo 1: Habilitar Astra Trident para que se ejecute en los nodos de Windows

Para permitir que Astra Trident se ejecute en los nodos de Windows:

```
tridentctl install --windows -n trident
```

Ejemplo 2: Activar la desconexión forzada

Para obtener más información acerca de forzar separación, consulte "Personalice la instalación del operador de Trident".

```
tridentctl install --enable-force-detach=true -n trident
```

El futuro

Ahora es posible "cree una clase de back-end y almacenamiento, aprovisione un volumen y monte el volumen en un pod".

Personalice la instalación trimentctl

Puede utilizar el instalador de Astra Trident para personalizar la instalación.

Obtenga más información sobre el instalador

El instalador de Astra Trident le permite personalizar atributos. Por ejemplo, si ha copiado la imagen de Trident en un repositorio privado, puede especificar el nombre de la imagen mediante `--trident-image`. Si ha copiado la imagen Trident así como las imágenes sidecar CSI necesarias en un repositorio privado, puede que sea preferible especificar la ubicación de ese repositorio mediante el `--image-registry` switch, que toma la forma `<registry FQDN>[:port]`.

Si utiliza una distribución de Kubernetes, donde `kubelet` mantiene los datos en una ruta distinta de la habitual `/var/lib/kubelet`, puede especificar la ruta alternativa mediante `--kubelet-dir`.

Si necesita personalizar la instalación más allá de lo que permiten los argumentos del instalador, también puede personalizar los archivos de implementación. Con el `--generate-custom-yaml` El parámetro crea

los siguientes archivos YAML en el instalador `setup` directorio:

- `trident-clusterrolebinding.yaml`
- `trident-deployment.yaml`
- `trident-crds.yaml`
- `trident-clusterrole.yaml`
- `trident-daemonset.yaml`
- `trident-service.yaml`
- `trident-namespace.yaml`
- `trident-serviceaccount.yaml`
- `trident-resourcequota.yaml`

Después de haber generado estos archivos, puede modificarlos según sus necesidades y luego usarlos `--use-custom-yaml` para instalar su implementación personalizada.

```
./tridentctl install -n trident --use-custom-yaml
```

¿Cuál es el siguiente?

Después de instalar Astra Trident, puede continuar con la creación de un entorno de administración, la creación de una clase de almacenamiento, el aprovisionamiento de un volumen y el montaje del volumen en un pod.

Paso 1: Crear un back-end

Ahora puede Adelante y crear un back-end que utilizará Astra Trident para aprovisionar volúmenes. Para ello, cree un `backend.json` archivo que contiene los parámetros necesarios. Se pueden encontrar archivos de configuración de ejemplo para diferentes tipos de backend en la `sample-input` directorio.

Consulte "aquí" para obtener más información acerca de cómo configurar el archivo para el tipo de backend.

```
cp sample-input/<backend template>.json backend.json
vi backend.json
```

```
./tridentctl -n trident create backend -f backend.json
+-------------+---------------+--------------------------------------
+--------+---------+
|    NAME     | STORAGE DRIVER |                 UUID                 |
STATE   | VOLUMES |
+-------------+---------------+--------------------------------------
+--------+---------+
| nas-backend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       0 |
+-------------+---------------+--------------------------------------
+--------+---------+
```

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

```
./tridentctl -n trident logs
```

Después de solucionar el problema, simplemente vuelva al principio de este paso e inténtelo de nuevo. Para obtener más consejos sobre la solución de problemas, consulte "la solución de problemas" sección.

Paso 2: Crear una clase de almacenamiento

Los usuarios de Kubernetes aprovisionan volúmenes mediante reclamaciones de volumen persistente (RVP) que especifican un "clase de almacenamiento" por nombre. Los detalles están ocultos de los usuarios, pero una clase de almacenamiento identifica el aprovisionador que se utiliza para esa clase (en este caso, Trident) y lo que significa esa clase para el aprovisionador.

Cree una clase de almacenamiento que los usuarios de Kubernetes especifiquen cuando quieran un volumen. La configuración de la clase debe modelar el back-end que ha creado en el paso anterior, de modo que Astra Trident lo utilice para aprovisionar nuevos volúmenes.

La clase de almacenamiento más sencilla que se debe empezar por está basada en la `sample-input/storage-class-csi.yaml.templ` archivo que viene con el instalador, reemplazar *`BACKEND_TYPE`* con el nombre del controlador de almacenamiento.

```
./tridentctl -n trident get backend
+-------------+---------------+--------------------------------------
+--------+---------+
|    NAME     | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+-------------+---------------+--------------------------------------
+--------+---------+
| nas-backend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       0 |
+-------------+---------------+--------------------------------------
+--------+---------+

cp sample-input/storage-class-csi.yaml.templ sample-input/storage-class-
basic-csi.yaml

# Modify __BACKEND_TYPE__ with the storage driver field above (e.g.,
ontap-nas)
vi sample-input/storage-class-basic-csi.yaml
```

Este es un objeto de Kubernetes, por lo que se usa `kubectl` Para crear en Kubernetes.

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

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"
        ]
      }
    }
  ]
}
```

Paso 3: Aprovisionar el primer volumen

Ahora está listo para aprovisionar de forma dinámica el primer volumen. Esto se realiza mediante la creación de un Kubernetes "reclamación de volumen persistente" Objeto (PVC).

Cree una RVP para un volumen que utiliza la clase de almacenamiento que acaba de crear.

Consulte `sample-input/pvc-basic-csi.yaml` por ejemplo. Asegúrese de que el nombre de la clase de almacenamiento coincida con el que ha creado.

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

kubectl get pvc --watch
NAME      STATUS    VOLUME                                     CAPACITY
ACCESS MODES  STORAGECLASS   AGE
basic     Pending
basic          1s
basic     Pending   pvc-3acb0d1c-b1ae-11e9-8d9f-5254004dfdb7   0
basic          5s
basic     Bound     pvc-3acb0d1c-b1ae-11e9-8d9f-5254004dfdb7   1Gi
RWO           basic          7s
```

Paso 4: Monte los volúmenes en un pod

Ahora vamos a montar el volumen. Lanzaremos una vaina nginx que monta el PV debajo `/usr/share/nginx/html`.

```
cat << EOF > task-pv-pod.yaml
kind: Pod
apiVersion: v1
metadata:
  name: task-pv-pod
spec:
  volumes:
    - name: task-pv-storage
      persistentVolumeClaim:
       claimName: basic
  containers:
    - name: task-pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/usr/share/nginx/html"
          name: task-pv-storage
EOF
kubectl create -f task-pv-pod.yaml
```

```
# Wait for the pod to start
kubectl get pod --watch

# Verify that the volume is mounted on /usr/share/nginx/html
kubectl exec -it task-pv-pod -- df -h /usr/share/nginx/html

# Delete the pod
kubectl delete pod task-pv-pod
```

En este momento, el pod (la aplicación) ya no existe pero el volumen sigue ahí. Puede utilizarlo desde otro pod si lo desea.

Para eliminar el volumen, elimine la reclamación:

```
kubectl delete pvc basic
```

Ahora puede realizar tareas adicionales, como las siguientes:

- "Configurar back-ends adicionales."
- "Cree clases de almacenamiento adicionales."

Gestione Astra Trident

Actualice Astra Trident

Actualice Astra Trident

Astra Trident sigue una cadencia de lanzamientos trimestrales y ofrece cuatro lanzamientos principales cada año. Cada versión nueva se basa en las versiones anteriores, y ofrece nuevas funciones y mejoras de rendimiento, así como correcciones de errores y mejoras. Le animamos a que realice una actualización al menos una vez al año para aprovechar las nuevas funciones de Astra Trident.

Consideraciones antes de la actualización

Cuando actualice a la versión más reciente de Astra Trident, tenga en cuenta lo siguiente:

- Solo debe haber una instancia de Astra Trident instalada en todos los espacios de nombres en un clúster de Kubernetes determinado.
- A partir de Trident 20.01, solo la versión en beta de "copias de snapshot de volumen" compatible. Los administradores de Kubernetes deben tener cuidado de realizar un backup o convertir de forma segura los objetos Snapshot alfa en beta para conservar las snapshots alfa heredadas.
  - Las instantáneas de volumen CSI son ahora una función de GA que comienza con Kubernetes 1.20. Antes de actualizar, debe eliminar los CRD de instantánea alfa utilizando `tridentctl obliviate alpha-snapshot-crd` Para eliminar los CRD de la especificación de instantánea alfa.
  - La versión beta de instantáneas de volumen introduce un conjunto modificado de CRD y un controlador de instantáneas, que deben configurarse antes de actualizar Astra Trident.
  - Para obtener más información, consulte "Qué necesita saber antes de actualizar su clúster de Kubernetes".
- Todas las actualizaciones de las versiones 19,04 y anteriores requieren la migración de metadatos de Astra Trident desde él mismo `etcd` Para crear objetos. Asegúrese de comprobar el "Documentación específica para la versión de Astra Trident" para entender cómo funciona la actualización.
- Al actualizar, es importante que proporcione `parameter.fsType` pulg `StorageClasses` Utilizado por Astra Trident. Puede eliminar y volver a crear `StorageClasses` sin interrumpir los volúmenes preexistentes.
  - Este es un **requisito** para hacer cumplir "contextos de seguridad" Para volúmenes SAN.
  - El directorio sample input contiene ejemplos, como `storage-class-basic.yaml.templ` y `storage-class-bronze-default.yaml`#. Para obtener más información, consulte "Problemas conocidos".

Paso 1: Seleccione una versión

Las versiones de Astra Trident siguen una fecha basada `YY.MM` Convención de nomenclatura, donde "YY" es los dos últimos dígitos del año y "MM" es el mes. Las versiones de puntos siguen a `YY.MM.X` convención, donde "X" es el nivel de parche. Deberá seleccionar la versión a la que se actualizará en función de la versión desde la que se actualice.

- Puede realizar una actualización directa a cualquier versión de destino que esté dentro de una ventana de

cuatro versiones de la versión instalada. Por ejemplo, puede actualizar a 23,04 desde 22,04 (incluidas las versiones de puntos, como 22.04.1) directamente.

- Si dispone de una versión anterior, debe realizar una actualización en varios pasos con la documentación de la versión correspondiente para obtener instrucciones específicas. Esto requiere que primero actualice a la versión más reciente que se ajuste a su ventana de cuatro lanzamientos. Por ejemplo, si ejecuta 18.07 y desea actualizar a la versión 20.07, siga el proceso de actualización de varios pasos como se indica a continuación:
  a. Primera actualización de 18.07 a 19.07.
  b. A continuación, actualice de 19.07 a 20.07.

> Cuando se actualice con el operador Trident en OpenShift Container Platform, debe actualizar a Trident 21.01.1 o una versión posterior. El operador Trident publicado con 21.01.0 contiene un problema conocido que se ha solucionado en 21.01.1. Si quiere más detalles, consulte "Detalles del problema en GitHub".

Paso 2: Determine el método de instalación original

Por lo general, debe actualizar utilizando el mismo método que utilizó para la instalación inicial, sin embargo, puede hacerlo "desplazarse entre los métodos de instalación".

Para determinar qué versión solías instalar originalmente Astra Trident:

1. Uso `kubectl get pods - trident` para examinar los pods.
   - Si no existe ningún pod de operador, Astra Trident se instaló mediante `tridentctl`.
   - Si hay un pod de operador, se instaló Astra Trident mediante el operador Trident manualmente o mediante Helm.
2. Si hay un pod del operador, utilice `kubectl describe tproc trident` Para determinar si Astra Trident se instaló mediante Helm.
   - Si hay una etiqueta Helm, Astra Trident se instaló usando Helm.
   - Si no hay ninguna etiqueta Helm, Astra Trident se instaló manualmente mediante el operador Trident.

Paso 3: Seleccione un método de actualización

Existen dos métodos para actualizar Astra Trident.

Cuándo actualizar con el operador

Puede hacerlo "Actualice con el operador Trident" si:

- Originalmente instaló Astra Trident con el operador o con el uso `tridentctl`.
- Desinstaló CSI Trident y los metadatos de la instalación persisten.
- Tiene una instalación de Astra Trident basada en CSI. Todas las versiones de 19.07 en adelante se basan en CSI. Puede examinar los pods en el espacio de nombres de Trident para verificar su versión.
  - Nombres de POD en versiones anteriores a 23,01 Usos: `trident-csi-*`
  - La nomenclatura de POD en 23.01 y versiones posteriores utiliza:
    - `trident-controller-<generated id>` para el pod del controlador
    - `trident-node-<operating system>-<generated id>` para los pods de nodo

- `trident-operator-<generated id>` para el pod del operador

> Si utiliza una, no utilice el operador para actualizar Trident `etcd`Versión de Trident basada en (19.04 o anterior).

Cuándo actualizar mediante `tridentctl`

Puede hacerlo Si originalmente instaló Astra Trident usando `tridenctl'.

`tridentctl` Es el método convencional de instalación de Astra Trident y proporciona la mayor cantidad de opciones para aquellos que requieren complejas personalizaciones. Para obtener información detallada, consulte "Elija el método de instalación".

Cambios en el operador

La versión 21,01 de Astra Trident introdujo cambios de arquitectura para el operador:

- El operador ahora está **ámbito de clúster**. Las instancias anteriores del operador Trident (versiones 20.04 a 20.10) eran **espacio de nombres**. Un operador con ámbito de clúster puede ser ventajoso por los siguientes motivos:
  - Responsabilidad de recursos: Ahora el operador gestiona los recursos asociados con una instalación de Astra Trident a nivel de clúster. Como parte de la instalación de Astra Trident, el operador crea y mantiene varios recursos mediante el uso `ownerReferences`. Mantenimiento `ownerReferences` En los recursos de ámbito de clúster pueden generar errores en determinados distribuidores de Kubernetes como OpenShift. Esto se mitiga con un operador con ámbito de clúster. Para la reparación automática y parches de recursos de Trident, este es un requisito esencial.
  - Limpieza durante la desinstalación: Una eliminación completa de Astra Trident requeriría que se eliminen todos los recursos asociados. Un operador con ámbito de espacio de nombres puede experimentar problemas con la eliminación de recursos con ámbito de clúster (como clusterRole, ClusterRoleBinding y PodSecurityPolicy) y dar lugar a una limpieza incompleta. Un operador con ámbito de clúster elimina este problema. Los usuarios pueden desinstalar por completo Astra Trident e instalar de nuevo si es necesario.
- `TridentProvisioner` se sustituye ahora por `TridentOrchestrator` Como recurso personalizado utilizado para instalar y gestionar Astra Trident. Además, se introduce un nuevo campo en el `TridentOrchestrator` espec. Los usuarios pueden especificar que el espacio de nombres Trident debe instalarse o actualizarse desde mediante el `spec.namespace` campo. Puede echar un vistazo a un ejemplo "aquí".

Actualizar con el operador

Puede actualizar fácilmente una instalación existente de Astra Trident con el operador, ya sea manualmente o mediante Helm.

Actualice con el operador Trident

Normalmente, debe actualizar Astra Trident mediante el mismo método que se utilizaba para instalarlo originalmente. Revisar "Seleccione un método de actualización" Antes de intentar actualizar con el operador Trident.

Al actualizar desde una instancia de Astra Trident instalada con el operador que cuenta con el ámbito de espacio de nombres (de las versiones 20,07 a la 20,10), el operador Trident realiza automáticamente lo siguiente:

- Migra `tridentProvisioner` a `tridentOrchestrator` objeto con el mismo nombre,
- Eliminaciones `TridentProvisioner` los objetos y la `tridentprovisioner` CRD
- Actualiza Astra Trident a la versión del operador en el ámbito del clúster que se usa
- Instale Astra Trident en el mismo espacio de nombres donde estaba instalado originalmente

Actualice una instalación de operador de Trident de ámbito de clúster

Puede actualizar una instalación del operador Trident en el ámbito del clúster. Todas las versiones 21.01 y posteriores de Astra Trident utilizan un operador con ámbito de clúster.

Antes de empezar

Asegúrese de que está utilizando un clúster de Kubernetes en ejecución "Una versión de Kubernetes compatible".

Pasos

1. Compruebe su versión de Astra Trident:

```
./tridentctl -n trident version
```

2. Elimine el operador Trident que se ha utilizado para instalar la instancia actual de Astra Trident. Por ejemplo, si va a actualizar desde 22,01, ejecute el siguiente comando:

```
kubectl delete -f 22.01/trident-installer/deploy/bundle.yaml -n trident
```

3. Si ha personalizado la instalación inicial mediante `TridentOrchestrator` atributos, puede editar `TridentOrchestrator` objeto para modificar los parámetros de instalación. Esto podría incluir cambios realizados para especificar registros de imágenes de Trident y CSI reflejados para el modo sin conexión, habilitar registros de depuración o especificar secretos de extracción de imágenes.
4. Instale Astra Trident con el archivo YAML de paquete correcto para su entorno y la versión Astra Trident. Por ejemplo, si va a instalar Astra Trident 23,04 para Kubernetes 1,27, ejecute el siguiente comando:

```
kubectl create -f 23.04.0/trident-installer/deploy/bundle_post_1_25.yaml
-n trident
```

Trident proporciona un archivo de paquete que se puede usar para instalar el operador y crear objetos asociados para la versión de Kubernetes.

- Para los clústeres que ejecutan Kubernetes 1,24 o una versión anterior, utilice "bundle_pre_1_25.yaml".
- Para los clústeres que ejecutan Kubernetes 1,25 o posterior, utilice "bundle_post_1_25.yaml".

Resultados

El operador de Trident identifica una instalación existente de Astra Trident y la actualiza a la misma versión que el operador.

Actualice la instalación de un operador de ámbito de espacio de nombres

Puede actualizar desde una instancia de Astra Trident instalada mediante el operador con ámbito de espacio de nombres (versiones 20,07 a 20,10) a una instalación de operadores en el ámbito del clúster.

Antes de empezar

Necesita el archivo YAML del paquete utilizado para desplegar el operador de ámbito de espacio de nombres desde `https://github.com/NetApp/trident/tree/stable/vXX.XX/deploy/BUNDLE.YAML` donde `vXX.XX` es el número de versión y. `BUNDLE.YAML` Es el nombre del archivo YAML del grupo.

Pasos

1. Compruebe el `TridentProvisioner` El estado de la instalación existente de Trident es `Installed`.

```
kubectl describe tprov trident -n trident | grep Message: -A 3

Message:  Trident installed
Status:   Installed
Version:  v20.10.1
```

> Si el estado muestra `Updating`, asegúrese de resolverlo antes de continuar. Para obtener una lista de los posibles valores de estado, consulte "aquí".

2. Cree el `TridentOrchestrator` CRD mediante el manifiesto proporcionado con el instalador de Trident.

```
# Download the release required [23.04.0]
mkdir 23.04.0
cd 23.04.0
wget
https://github.com/NetApp/trident/releases/download/v23.04.0/trident-
installer-23.04.0.tar.gz
tar -xf trident-installer-23.04.0.tar.gz
cd trident-installer
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
```

3. Elimine el operador de ámbito del espacio de nombres mediante su manifiesto.
   a. Asegúrese de que está en el directorio correcto.

```
pwd
/root/20.10.1/trident-installer
```

b. Suprima el operador de ámbito de espacio de nombres.

```
kubectl delete -f deploy/<BUNDLE.YAML> -n trident

serviceaccount "trident-operator" deleted
clusterrole.rbac.authorization.k8s.io "trident-operator" deleted
clusterrolebinding.rbac.authorization.k8s.io "trident-operator"
deleted
deployment.apps "trident-operator" deleted
podsecuritypolicy.policy "tridentoperatorpods" deleted
```

c. Confirme que se ha eliminado el operador Trident.

```
kubectl get all -n trident

NAME                                READY   STATUS    RESTARTS   AGE
pod/trident-csi-68d979fb85-dsrmn    6/6     Running   12         99d
pod/trident-csi-8jfhf               2/2     Running   6          105d
pod/trident-csi-jtnjz               2/2     Running   6          105d
pod/trident-csi-lcxvh               2/2     Running   8          105d

NAME                  TYPE        CLUSTER-IP       EXTERNAL-IP
PORT(S)              AGE
service/trident-csi   ClusterIP   10.108.174.125   <none>
34571/TCP,9220/TCP   105d

NAME                           DESIRED   CURRENT   READY   UP-TO-DATE
AVAILABLE   NODE SELECTOR                                        AGE
daemonset.apps/trident-csi   3         3         3       3
3           kubernetes.io/arch=amd64,kubernetes.io/os=linux   105d

NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/trident-csi    1/1     1            1           105d

NAME                                      DESIRED   CURRENT   READY
AGE
replicaset.apps/trident-csi-68d979fb85    1         1         1
105d
```

4. (Opcional) Si es necesario modificar los parámetros de instalación, actualice `TridentProvisioner` espec. Esto puede incluir cambios como cambiar: Los valores de `tridentImage`, `autosupportImage`, repositorio privado de imágenes y provisión `imagePullSecrets`) después de eliminar el operador de ámbito de espacio de nombres y antes de instalar el operador de ámbito de clúster. Para obtener una lista completa de los parámetros que se pueden actualizar, consulte la "opciones de configuración".

```
kubectl patch tprov <trident-provisioner-name> -n <trident-namespace>
--type=merge -p '{"spec":{"debug":true}}'
```

5. Instale el operador del ámbito del clúster Trident.
   a. Asegúrese de que está en el directorio correcto.

```
pwd
/root/23.04.0/trident-installer
```

   b. Instale el operador de ámbito de cluster en el mismo espacio de nombres.

> Trident proporciona un archivo de paquete que se puede usar para instalar el operador y crear objetos asociados para la versión de Kubernetes.
>
> - Para los clústeres que ejecutan Kubernetes 1,24 o una versión anterior, utilice "bundle_pre_1_25.yaml".
> - Para los clústeres que ejecutan Kubernetes 1,25 o posterior, utilice "bundle_post_1_25.yaml".

```
kubectl create -f deploy/<BUNDLE.YAML>

serviceaccount/trident-operator created
clusterrole.rbac.authorization.k8s.io/trident-operator created
clusterrolebinding.rbac.authorization.k8s.io/trident-operator created
deployment.apps/trident-operator created
podsecuritypolicy.policy/tridentoperatorpods created

#All tridentProvisioners will be removed, including the CRD itself
kubectl get tprov -n trident
Error from server (NotFound): Unable to list "trident.netapp.io/v1,
Resource=tridentprovisioners": the server could not find the
requested resource (get tridentprovisioners.trident.netapp.io)

#tridentProvisioners are replaced by tridentOrchestrator
kubectl get torc
NAME      AGE
trident   13s
```

   c. Examine los pods de Trident en el espacio de nombres. La `trident-controller` y los nombres de pod reflejan la convención de nomenclatura introducida en 23.01.

```
kubectl get pods -n trident

NAME                                  READY   STATUS    RESTARTS
AGE
trident-controller-79df798bdc-m79dc   6/6     Running   0
1m41s
trident-node-linux-xrst8              2/2     Running   0
1m41s
trident-operator-5574dbbc68-nthjv     1/1     Running   0
1m52s
```

d. Confirme que Trident se ha actualizado a la versión prevista.

```
kubectl describe torc trident | grep Message -A 3
Message:                Trident installed
Namespace:              trident
Status:                 Installed
Version:                v23.04.0
```

Actualice la instalación de un operador basado en Helm

Realice los pasos siguientes para actualizar la instalación de un operador basado en Helm.

> Cuando actualice un clúster de Kubernetes de 1.24 a 1.25 o posterior que tenga instalado Astra Trident, debe actualizar Values.yaml para establecer `excludePodSecurityPolicy` para `true` o agregar `--set excludePodSecurityPolicy=true` para la `helm upgrade` comando antes de poder actualizar el clúster.

Pasos

1. Descargue la última versión de Astra Trident.
2. Utilice la `helm upgrade` comando donde `trident-operator-23.04.0.tgz` refleja la versión a la que desea actualizar.

```
helm upgrade <name> trident-operator-23.04.0.tgz
```

Si establece cualquier opción no predeterminada durante la instalación inicial (como especificar registros privados reflejados para imágenes Trident y CSI), utilice `--set` para asegurarse de que estas opciones están incluidas en el comando upgrade, de lo contrario, los valores se restablecerán a los valores predeterminados.

Por ejemplo, para cambiar el valor predeterminado de `tridentDebug`, ejecute el siguiente comando:

```
helm upgrade <name> trident-operator-23.04.0-custom.tgz --set
tridentDebug=true
```

3. Ejecución `helm list` para comprobar que la versión de la gráfica y de la aplicación se han actualizado. Ejecución `tridentctl logs` para revisar cualquier mensaje de depuración.

Resultados

El operador de Trident identifica una instalación existente de Astra Trident y la actualiza a la misma versión que el operador.

Actualizar desde una instalación que no sea del operador

Puede actualizarlo a la versión más reciente del operador de Trident desde un `tridentctl` instalación.

Pasos

1. Descargue la última versión de Astra Trident.

```
# Download the release required [23.04.0]
mkdir 23.04.0
cd 23.04.0
wget
https://github.com/NetApp/trident/releases/download/v22.01.1/trident-
installer-23.04.0.tar.gz
tar -xf trident-installer-23.04.0.tar.gz
cd trident-installer
```

2. Cree el `tridentorchestrator` CRD del manifiesto.

```
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
```

3. Despliegue el operador de ámbito de cluster en el mismo espacio de nombres.

```
kubectl create -f deploy/<BUNDLE.YAML>

serviceaccount/trident-operator created
clusterrole.rbac.authorization.k8s.io/trident-operator created
clusterrolebinding.rbac.authorization.k8s.io/trident-operator created
deployment.apps/trident-operator created
podsecuritypolicy.policy/tridentoperatorpods created

#Examine the pods in the Trident namespace
NAME                                  READY   STATUS    RESTARTS   AGE
trident-controller-79df798bdc-m79dc   6/6     Running   0          150d
trident-node-linux-xrst8              2/2     Running   0          150d
trident-operator-5574dbbc68-nthjv     1/1     Running   0          1m30s
```

4. Cree un `TridentOrchestrator` CR para instalar Astra Trident.

```
cat deploy/crds/tridentorchestrator_cr.yaml
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident

kubectl create -f deploy/crds/tridentorchestrator_cr.yaml

#Examine the pods in the Trident namespace
NAME                                READY   STATUS    RESTARTS   AGE
trident-csi-79df798bdc-m79dc        6/6     Running   0          1m
trident-csi-xrst8                   2/2     Running   0          1m
trident-operator-5574dbbc68-nthjv   1/1     Running   0          5m41s
```

5. Confirmar que Trident se ha actualizado a la versión prevista.

```
kubectl describe torc trident | grep Message -A 3

Message:                Trident installed
Namespace:              trident
Status:                 Installed
Version:                v23.04.0
```

Resultados

Los back-ends y las CVP existentes están disponibles automáticamente.

Actualice con trimentctl

Puede actualizar fácilmente una instalación de Astra Trident existente mediante `tridentctl`.

Actualiza Astra Trident con `tridentctl`

La desinstalación y reinstalación de Astra Trident actúa como una actualización. Cuando desinstala Trident, la reclamación de volumen persistente (PVC) y el volumen persistente (PV) que utiliza la puesta en marcha de Astra Trident no se eliminan. Las RVP que ya se han aprovisionado seguirán disponibles mientras Astra Trident está offline y Astra Trident aprovisiona volúmenes para cualquier RVP que se crean interanualmente una vez que vuelve a estar online.

Antes de empezar

Revisar "Seleccione un método de actualización" antes de actualizar mediante `tridentctl`.

Pasos

1. Ejecute el comando uninstall en `tridentctl` Para quitar todos los recursos asociados con Astra Trident, excepto los CRD y los objetos relacionados.

```
./tridentctl uninstall -n <namespace>
```

2. Vuelva a instalar Astra Trident. Consulte "Instalar Astra Trident mediante tridentctl".

> No interrumpa el proceso de actualización. Asegúrese de que el instalador se ejecuta hasta su finalización.

Actualizar volúmenes mediante `tridentctl`

Después de la actualización, puede utilizar el completo conjunto de funciones que están disponibles en las versiones más recientes de Trident (como, las Snapshots de volumen bajo demanda), puede actualizar los volúmenes con el `tridentctl upgrade` comando.

Si hay volúmenes heredados, debe actualizar desde un tipo NFS o iSCSI al tipo CSI para utilizar el conjunto completo de funciones nuevas en Astra Trident. Un VP heredado que ha sido aprovisionado por Trident admite el conjunto tradicional de funciones.

Antes de empezar

Tenga en cuenta lo siguiente antes de decidir actualizar los volúmenes al tipo CSI:

- Es posible que no sea necesario actualizar todos los volúmenes. Los volúmenes creados previamente seguirán siendo accesibles y funcionarán normalmente.
- Un PV se puede montar como parte de un despliegue/Statilusionados al actualizar. No es necesario que los ilusionados traigan el despliegue/StatSet.
- **No puede** conectar un PV a un pod independiente al realizar la actualización. Debe apagar el pod antes de actualizar el volumen.
- Solo puede actualizar un volumen vinculado a una RVP. Los volúmenes que no están enlazados a PVC

deben eliminarse e importarse antes de actualizar.

Pasos

1. Ejecución `kubectl get pv` Para enumerar los VP.

```
kubectl get pv
NAME                        CAPACITY     ACCESS MODES   RECLAIM POLICY
STATUS   CLAIM                   STORAGECLASS    REASON   AGE
default-pvc-1-a8475         1073741824   RWO            Delete
Bound    default/pvc-1          standard                19h
default-pvc-2-a8486         1073741824   RWO            Delete
Bound    default/pvc-2          standard                19h
default-pvc-3-a849e         1073741824   RWO            Delete
Bound    default/pvc-3          standard                19h
default-pvc-4-a84de         1073741824   RWO            Delete
Bound    default/pvc-4          standard                19h
trident                     2Gi          RWO            Retain
Bound    trident/trident                                19h
```

Actualmente hay cuatro VP creados por Trident 20.07 con la `netapp.io/trident` aprovisionador.

2. Ejecución `kubectl describe pv` Para obtener los detalles del PV.

```
kubectl describe pv default-pvc-2-a8486

Name:            default-pvc-2-a8486
Labels:          <none>
Annotations:     pv.kubernetes.io/provisioned-by: netapp.io/trident
                 volume.beta.kubernetes.io/storage-class: standard
Finalizers:      [kubernetes.io/pv-protection]
StorageClass:    standard
Status:          Bound
Claim:           default/pvc-2
Reclaim Policy:  Delete
Access Modes:    RWO
VolumeMode:      Filesystem
Capacity:        1073741824
Node Affinity:   <none>
Message:
Source:
    Type:      NFS (an NFS mount that lasts the lifetime of a pod)
    Server:    10.xx.xx.xx
    Path:      /trid_1907_alpha_default_pvc_2_a8486
    ReadOnly:  false
```

El VP se creó mediante la `netapp.io/trident` provisioner y es del tipo NFS. Para admitir todas las nuevas funciones proporcionadas por Astra Trident, este PV debe actualizarse al tipo CSI.

3. Ejecute el `tridentctl upgrade volume <name-of-trident-volume>` Comando para actualizar un volumen heredado de Astra Trident a la especificación CSI.

```
./tridentctl get volumes -n trident
+---------------------+---------+---------------+----------
+--------------------------------------+--------+---------+
|            NAME     |  SIZE   | STORAGE CLASS | PROTOCOL |
BACKEND UUID             | STATE  | MANAGED |
+---------------------+---------+---------------+----------
+--------------------------------------+--------+---------+
| default-pvc-2-a8486 | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true    |
| default-pvc-3-a849e | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true    |
| default-pvc-1-a8475 | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true    |
| default-pvc-4-a84de | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true    |
+---------------------+---------+---------------+----------
+--------------------------------------+--------+---------+

./tridentctl upgrade volume default-pvc-2-a8486 -n trident
+---------------------+---------+---------------+----------
+--------------------------------------+--------+---------+
|            NAME     |  SIZE   | STORAGE CLASS | PROTOCOL |
BACKEND UUID             | STATE  | MANAGED |
+---------------------+---------+---------------+----------
+--------------------------------------+--------+---------+
| default-pvc-2-a8486 | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true    |
+---------------------+---------+---------------+----------
+--------------------------------------+--------+---------+
```

4. Ejecute un `kubectl describe pv` Para verificar que el volumen es un volumen CSI.

```
kubectl describe pv default-pvc-2-a8486
Name:            default-pvc-2-a8486
Labels:          <none>
Annotations:     pv.kubernetes.io/provisioned-by: csi.trident.netapp.io
                 volume.beta.kubernetes.io/storage-class: standard
Finalizers:      [kubernetes.io/pv-protection]
StorageClass:    standard
Status:          Bound
Claim:           default/pvc-2
Reclaim Policy:  Delete
Access Modes:    RWO
VolumeMode:      Filesystem
Capacity:        1073741824
Node Affinity:   <none>
Message:
Source:
    Type:              CSI (a Container Storage Interface (CSI) volume
source)
    Driver:            csi.trident.netapp.io
    VolumeHandle:      default-pvc-2-a8486
    ReadOnly:          false
    VolumeAttributes:      backendUUID=c5a6f6a4-b052-423b-80d4-
8fb491a14a22

internalName=trid_1907_alpha_default_pvc_2_a8486
                           name=default-pvc-2-a8486
                           protocol=file
Events:                <none>
```

Desinstale Astra Trident

En función de la instalación de Astra Trident, hay varias opciones para desinstalarla.

Desinstalar utilizando Helm

Si ha instalado Astra Trident mediante Helm, puede desinstalarlo mediante `helm uninstall`.

```
#List the Helm release corresponding to the Astra Trident install.
helm ls -n trident
NAME          NAMESPACE       REVISION        UPDATED
STATUS          CHART                           APP VERSION
trident       trident         1               2021-04-20
00:26:42.417764794 +0000 UTC deployed        trident-operator-21.07.1
21.07.1

#Uninstall Helm release to remove Trident
helm uninstall trident -n trident
release "trident" uninstalled
```

Desinstale mediante el operador Trident

Si ha instalado Astra Trident mediante el operador, puede desinstalarlo realizando una de las siguientes acciones:

- **Edición `TridentOrchestrator` Para establecer el indicador de desinstalación:** puede editar `TridentOrchestrator` y ajustar `spec.uninstall=true`. Edite el `TridentOrchestrator` CR y ajuste la `uninstall` marcar como se muestra a continuación:

  ```
  kubectl patch torc <trident-orchestrator-name> --type=merge -p
  '{"spec":{"uninstall":true}}'
  ```

  Cuando la `uninstall` el indicador se establece en `true`, El operador Trident desinstala Trident, pero no quita el propio TridentOrchestrator. Debe limpiar el TridentOrchestrator y crear uno nuevo si lo desea Vuelva a instalar Trident.

- **Eliminar `TridentOrchestrator`:** extrayendo el `TridentOrchestrator` CR utilizado para implementar Astra Trident, indica al operador que desinstale Trident. El operador procesa la eliminación de `TridentOrchestrator` Y procede a eliminar la implementación y el demonset de Astra Trident, con la eliminación de los pods de Trident que ha creado como parte de la instalación.
  Para eliminar completamente Astra Trident (incluidos los CRD que crea) y borrar la pizarra de forma efectiva, puede editar `TridentOrchestrator` para pasar la `wipeout` opción. Consulte el siguiente ejemplo:

```
kubectl patch torc <trident-orchestrator-name> --type=merge -p
'{"spec":{"wipeout":["crds"],"uninstall":true}}'
```

Esto desinstala Astra Trident por completo y borra todos los metadatos relacionados con los back-ends y los volúmenes que gestiona. Las instalaciones posteriores se tratan como instalaciones frescas.

> Sólo debe considerar borrar los CRD al realizar una desinstalación completa. Esta acción no se puede deshacer. **No limpie los CRD a menos que esté buscando empezar y crear una nueva instalación de Astra Trident**.

Desinstale mediante `tridentctl`

Ejecute el `uninstall` comando en `tridentctl` A continuación, se eliminan todos los recursos asociados con Astra Trident, excepto los CRD y los objetos relacionados, lo que facilita la ejecución del instalador de nuevo para actualizar a una versión más reciente.

```
./tridentctl uninstall -n <namespace>
```

Para realizar una eliminación completa de Astra Trident, debe eliminar los finalizadores de los CRD creados por Astra Trident y eliminar los CRD.

Degradar Astra Trident

Conozca los pasos que se deben seguir para cambiar a una versión anterior de Astra Trident.

Cuándo degradar

Es posible que tenga en cuenta la degradación por varios motivos, como los siguientes:

- Planificación de contingencia
- Solución inmediata de errores observados como resultado de una actualización
- Problemas de dependencia, actualizaciones incorrectas e incompletas

Debe considerar una degradación al cambiar a una versión de Astra Trident que utiliza CRD. Como Astra Trident utiliza CRD para el mantenimiento del estado, todas las entidades de almacenamiento creadas (back-ends, clases de almacenamiento, VP y snapshots de volúmenes) tienen objetos CRD asociados en lugar de los datos escritos en la `trident` PV (utilizado por la versión anterior de Astra Trident). Las clases de almacenamiento, los back-ends y los VP recién creados se mantienen como objetos CRD.

Solo intente realizar una degradación a una versión de Astra Trident que se ejecute con CRD (19.07 y posterior). Esto garantiza que las operaciones realizadas en la versión actual de Astra Trident sean visibles una vez que se produce la degradación.

Cuando no se debe degradar

No debe degradar a una versión de Trident que utilice `etcd` mantener el estado (19.04 y anteriores). Todas las operaciones realizadas con la versión actual de Astra Trident no se reflejan después de la degradación. Los VP recién creados no se pueden utilizar al volver a una versión anterior. Los cambios que se realizan en objetos como los back-ends, VP, las clases de almacenamiento y las snapshots de volúmenes (creadas, actualizadas o eliminadas) no son visibles para Astra Trident al volver a una versión anterior. Volver a una versión anterior no interrumpe el acceso a los VP que ya se habían creado con la versión anterior, a menos que se hayan actualizado.

Proceso de degradación cuando se instala Astra Trident mediante el operador

En el caso de las instalaciones realizadas mediante el operador Trident, el proceso de degradación es diferente y no requiere el uso de `tridentctl`.

En el caso de las instalaciones realizadas mediante el operador Trident, es posible reclasificar Astra Trident a

uno de los siguientes:

- Versión que se instala mediante el operador namespace-scoped (20.07 - 20.10).
- Versión que se instala mediante el operador de ámbito del clúster (21.01 y posteriores).

Degradar al operador de ámbito del clúster

Para degradar Astra Trident a una versión que utilice el operador de ámbito del clúster, siga los pasos que se mencionan a continuación.

Pasos

1. "Desinstale Astra Trident". **No elimine los CRD a menos que desee eliminar completamente una instalación existente.**
2. El operador de Trident puede eliminarse mediante el manifiesto de operador asociado con su versión de Trident. Por ejemplo: `https://github.com/NetApp/trident/tree/stable/vXX.XX /deploy/bundle.yaml` donde `vXX.XX` es el número de versión (por ejemplo `v22.10`) y. `bundle.yaml` Es el nombre del archivo YAML del grupo.
3. Continúe con la degradación instalando la versión deseada de Astra Trident. Siga la documentación para la versión deseada.

Degradar al operador de ámbito de espacio de nombres

En esta sección se resumen los pasos necesarios para la degradación a una versión de Astra Trident que está dentro del intervalo comprendido entre el 20.07 y el 20.10, que se instalará utilizando el operador de ámbito del espacio de nombres.

Pasos

1. "Desinstale Astra Trident". **No extraiga los CRD a menos que desee eliminar completamente una instalación existente.**
   Asegúrese de que el `tridentorchestrator` se ha eliminado.

   ```
   #Check to see if there are any tridentorchestrators present
   kubectl get torc
   NAME        AGE
   trident     20h

   #Looks like there is a tridentorchestrator that needs deleting
   kubectl delete torc trident
   tridentorchestrator.trident.netapp.io "trident" deleted
   ```

2. El operador de Trident puede eliminarse mediante el manifiesto de operador asociado con su versión de Trident. Por ejemplo: `https://github.com/NetApp/trident/tree/stable/vXX.XX /deploy/bundle.yaml` donde `vXX.XX` es el número de versión (por ejemplo `v22.10`) y. `bundle.yaml` Es el nombre del archivo YAML del grupo.
3. Elimine el `tridentorchestrator` CRD.

```
#Check to see if ``tridentorchestrators.trident.netapp.io`` CRD is
present and delete it.

kubectl get crd tridentorchestrators.trident.netapp.io

NAME                                        CREATED AT
tridentorchestrators.trident.netapp.io   2021-01-21T21:11:37Z

kubectl delete crd tridentorchestrators.trident.netapp.io

customresourcedefinition.apiextensions.k8s.io
"tridentorchestrators.trident.netapp.io" deleted
```

Astra Trident se ha desinstalado.

4. Continúe con la degradación instalando la versión deseada. Siga la documentación para la versión deseada.

Bajar utilizando Helm

Para degradar, utilice `helm rollback` comando. Consulte el siguiente ejemplo:

```
helm rollback trident [revision #]
```

Proceso de degradación cuando Astra Trident se instala mediante `tridentctl`

Si instaló Astra Trident mediante `tridentctl`, el proceso de degradación implica los siguientes pasos. Esta secuencia le guiará por el proceso de degradación para pasar de Astra Trident 21.07 a 20.07.

Antes de iniciar la degradación, debe tomar una instantánea del clúster de Kubernetes `etcd`. Esto le permite realizar una copia de seguridad del estado actual de los CRD de Astra Trident.

Pasos

1. Compruebe que Trident se instala mediante `tridentctl`. Si no está seguro de cómo está instalado Astra Trident, ejecute esta sencilla prueba:
   a. Enumere los pods presentes en el espacio de nombres de Trident.
   b. Identifique la versión de Astra Trident que se ejecuta en su clúster. Puede utilizar cualquiera de estos dos usos `tridentctl` También puede ver la imagen utilizada en los pods de Trident.
   c. Si **no ve** a `tridentOrchestrator`, (o) un `tridentprovisioner`, (o) un pod llamado `trident-operator-xxxxxxxxxx-xxxxx`, Astra Trident **está instalado** con `tridentctl`.
2. Desinstale Astra Trident con el existente `tridentctl` binario. En este caso, se desinstalará con el binario 21.07.

```
tridentctl version -n trident
+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 21.07.0        | 21.07.0        |
+----------------+----------------+

tridentctl uninstall -n trident
INFO Deleted Trident deployment.
INFO Deleted Trident daemonset.
INFO Deleted Trident service.
INFO Deleted Trident secret.
INFO Deleted cluster role binding.
INFO Deleted cluster role.
INFO Deleted service account.
INFO Deleted pod security policy.
podSecurityPolicy=tridentpods
INFO The uninstaller did not delete Trident's namespace in case it is
going to be reused.
INFO Trident uninstallation succeeded.
```

3. Una vez finalizado este proceso, obtenga el binario de Trident correspondiente a la versión deseada (en este ejemplo, 20.07) y utilícelo para instalar Astra Trident. Puede generar YAML personalizados para un "instalación personalizada" si es necesario.

```
cd 20.07/trident-installer/
./tridentctl install -n trident-ns
INFO Created installer service account.
serviceaccount=trident-installer
INFO Created installer cluster role.              clusterrole=trident-
installer
INFO Created installer cluster role binding.
clusterrolebinding=trident-installer
INFO Created installer configmap.                 configmap=trident-
installer
...
...
INFO Deleted installer cluster role binding.
INFO Deleted installer cluster role.
INFO Deleted installer service account.
```

Se completó el proceso de degradación.

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 o iSCSI según su selección de controlador.

Seleccionar las herramientas adecuadas

Si utiliza una combinación de controladores, debe instalar herramientas NFS e iSCSI.

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 utiliza: `ontap-san`, `ontap-san-economy`, `solidfire-san`

> Las versiones recientes de RedHat CoreOS tienen instaladas NFS e iSCSI de forma predeterminada.

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:

```
kubectl 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 de 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 reparación automática se encuentran en la `trident-main` Contenedor en el dosis de Demonset correspondiente. Para ver los registros, debe haber configurado `debug` A "verdadero" 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

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 Linux compatible con RHEL, con `solidfire-san` Controlador y Element OS 12.5 o anterior, asegúrese de que el algoritmo de autenticación CHAP esté establecido en MD5 in `/etc/iscsi/iscsid.conf`. Los algoritmos CHAP SHA1, SHA-256 y SHA3-256 compatibles con FIPS están disponibles con Element 12.7.

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

- Cuando utilice nodos de trabajo que ejecutan RHEL/RedHat CoreOS con VP iSCSI, especifique el `discard` MountOption en StorageClass para realizar un reclamación de espacio en línea. Consulte "La 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 `etc/multipath.conf` contiene `find_multipaths no` inferior `defaults`.

5. Asegúrese de que así sea `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 `etc/multipath.conf` contiene `find_multipaths no` inferior `defaults`.

5. Asegúrese de que así sea `open-iscsi` y.. `multipath-tools` están habilitadas 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 descubrir los puertos de destino con `iscsiadm` antes de comenzar `open-iscsi` Para que se inicie el daemon iSCSI. También puede modificar el `iscsi` servicio para empezar `iscsid` automáticamente.

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

Configurar los 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"
- "Configurar 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 (ANF) como back-end de Astra Trident. Puede asociar volúmenes de NFS y SMB con un back-end de ANF.

Consideraciones

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

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" back-end, necesita lo siguiente:

- Un pool de capacidad. Consulte "Microsoft: Cree un pool de capacidad para Azure NetApp Files".
- Una subred delegada en Azure NetApp Files. Consulte "Microsoft: Delegue una subred en Azure NetApp Files".
- `subscriptionID` Desde una suscripción de Azure con Azure NetApp Files habilitado.
- `tenantID`, `clientID`, y. `clientSecret` desde una "Registro de aplicaciones" En Azure Active Directory con permisos suficientes para el servicio Azure NetApp Files. El registro de aplicaciones debe usar:
  - El rol propietario o Colaborador "Predefinidos por Azure".
  - A. "Rol Colaborador personalizado" en el nivel de suscripción (`assignableScopes`) Con los

siguientes permisos que están limitados únicamente a lo que Astra Trident necesita. 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/writ
e",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/dele
te",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/subvolumes/rea
d",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/subvolumes/wri
te",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/subvolumes/del
ete",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/subvolumes/Get
```

```
Metadata/action",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/MountTargets/r
ead",
                    "Microsoft.Network/virtualNetworks/read",
                    "Microsoft.Network/virtualNetworks/subnets/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrations
/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrations
/write",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrations
/delete",
                    "Microsoft.Features/features/read",
                    "Microsoft.Features/operations/read",
                    "Microsoft.Features/providers/features/read",

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

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

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

- Azure `location` que contiene al menos uno "subred delegada". A partir de Trident 22.01, la `location` parámetro es un campo obligatorio en el nivel superior del archivo de configuración del back-end. Los valores de ubicación especificados en los pools virtuales se ignoran.

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 2019. 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. 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 nodos Kubernetes que se ejecutan en Windows.

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

Obtenga más información acerca de las opciones de configuración de back-end de NFS y SMB para ANF y revise 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 ANF en pools de capacidad 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 ANF 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 su suscripción de Azure	
`tenantID`	El ID de inquilino de un registro de aplicación	
`clientID`	El ID de cliente de un registro de aplicación	
`clientSecret`	El secreto de cliente de un registro de aplicaciones	
`serviceLevel`	Uno de `Standard`, `Premium`, o. `Ultra`	"" (aleatorio)
`location`	Nombre de la ubicación de Azure donde se crearán los nuevos volúmenes	
`resourceGroups`	Lista de grupos de recursos para filtrar los recursos detectados	"[]" (sin filtro)

Parámetro	Descripción	Predeterminado
`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 a. `Microsoft.Netapp/volumes`	""
`networkFeatures`	Puede que el conjunto de funciones de vnet para un volumen sea `Basic` o. `Standard`. Las funciones de red no están disponibles en todas las regiones y es posible que tengan que activarse en una suscripción. Especificando `networkFeatures` cuando la funcionalidad no está habilitada, hace que no se pueda realizar 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: `\{"api": false, "method": true, "discovery": true}`. 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 `nfs`, `smb` o nulo. El valor predeterminado es nulo en volúmenes de NFS.	`nfs`

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 recibe un error que indica que no se han encontrado pools de capacidad al crear un PVC, es probable que el registro de aplicaciones no tenga asociados los permisos y recursos necesarios (subred, red virtual o pool de capacidad). 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` puede especificarse utilizando 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.

La `resourceGroups`, `netappAccounts`, y. `capacityPools` los valores son filtros que restringen el conjunto de recursos detectados a los disponibles en este back-end de almacenamiento y pueden especificarse en cualquier combinación de estos. Los nombres completos siguen este formato:

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

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 detalles.

Parámetro	Descripción	Predeterminado
`exportRule`	Reglas de exportación de volúmenes nuevos. `exportRule` Debe ser una lista separada por comas con 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

Ejemplo 1: Configuración mínima

Ésta es la configuración mínima absoluta del back-end. Con esta configuración, Astra Trident descubre todas sus cuentas, pools de capacidad y subredes de NetApp delegadas en ANF en la ubicación configurada, y coloca nuevos volúmenes en uno de estos pools y subredes de forma aleatoria. Porque `nasType` se omite, la `nfs` El valor predeterminado es aplicable, y el back-end aprovisionará para volúmenes NFS.

Esta configuración es ideal cuando simplemente va a empezar con ANF e intentar cosas, pero en la práctica va a querer proporcionar un ámbito adicional para los volúmenes que debe aprovisionar.

```
---
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
```

Ejemplo 2: Configuración específica de nivel de servicio con filtros de pool de capacidad

Esta configuración de back-end coloca volúmenes en las de Azure `eastus` ubicación en una `Ultra` pool de capacidad. Astra Trident descubre automáticamente todas las subredes delegadas a ANF en esa ubicación y coloca un nuevo volumen en una de ellas de forma aleatoria.

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

Ejemplo 3: 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'
```

Ejemplo 4: 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. Se utilizaron etiquetas de pools virtuales para diferenciar los pools según `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
```

Definiciones de clase de almacenamiento

Lo siguiente `StorageClass` las definiciones hacen referencia a los pools de almacenamiento anteriores.

Definiciones de ejemplo mediante `parameter.selector` campo

Uso `parameter.selector` puede especificar para cada una de ellas `StorageClass` el 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: 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

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

Ejemplo 1: Configuración básica del 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"
```

Ejemplo 2: Uso de distintos 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}
```

Ejemplo 3: Uso de distintos 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` Filtra los 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.

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.

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 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 uno de "tres niveles de servicio":
  - `standard`
  - `premium`
  - `extreme`
- **CVS**: El tipo de servicio CVS proporciona una alta disponibilidad zonal con niveles de rendimiento limitados a moderados. El tipo de servicio CVS es una opción de software que 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 uno de "dos niveles de servicio":
  - `standardsw`
  - `zoneredundantstandardsw`

Lo que necesitará

Para configurar y usar el "Cloud Volumes Service para Google Cloud" back-end, 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 el `netappcloudvolumes.admin` función
- 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. Uso `software` Para seleccionar el tipo de servicio CVS. De lo contrario, Astra Trident asume el tipo de servicio CVS-Performance (`hardware`).	
`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.	
`hostProjectNumber`	Se requiere si se utiliza una red VPC compartida. En este escenario, `projectNumber` es el proyecto de servicio, y. `hostProjectNumber` es el proyecto anfitrión.	

Parámetro	Descripción	Predeterminado
`apiRegion`	Región de Google Cloud en la que Astra Trident crea volúmenes de Cloud Volumes Service. Cuando se crean clústeres de Kubernetes de diversas regiones, se crean volúmenes en un `apiRegion` Se puede utilizar en cargas de trabajo programadas en nodos en varias regiones de Google Cloud. El tráfico entre regiones conlleva un coste adicional.	
`apiKey`	Clave de API para la cuenta de servicio de Google Cloud con el `netappcloudvolumes.admin` función. 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)

Parámetro	Descripción	Predeterminado
`serviceLevel`	El nivel de servicio CVS-Performance o CVS para nuevos volúmenes. Los valores de CVS-Performance son `standard`, `premium`, o. `extreme`. Los valores CVS son `standardsw` o. `zoneredundantstandardsw`.	El valor predeterminado de CVS-Performance es "estándar". El valor predeterminado de CVS es "standardsw".
`network`	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: `\{"api":false, "method":true}`. No lo utilice a menos que esté solucionando problemas y necesite un volcado de registro detallado.	nulo

`allowedTopologies`	Para habilitar el acceso a varias regiones, se debe definir StorageClass para `allowedTopologies` debe incluir todas las regiones.
	
	Por ejemplo:
	`- key: topology.kubernetes.io/region`
	`values:`
	`- us-east1`
	`- europe-west1`	

Opciones de aprovisionamiento de volúmenes

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

Parámetro	Descripción	Predeterminado
`exportRule`	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"

Parámetro	Descripción	Predeterminado
`snapshotDir`	Acceso a la `.snapshot` directorio	"falso"
`snapshotReserve`	Porcentaje de volumen reservado para las Snapshot	"" (Aceptar CVS por defecto de 0)
`size`	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-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----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-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----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 los pools virtuales y el `StorageClasses` eso se refiere a ellos. Consulte Definiciones de clases de almacenamiento para ver cómo se definieron las clases de almacenamiento.

Aquí, se establecen valores predeterminados específicos para todos los pools virtuales, con los que se establece el `snapshotReserve` con el 5% y la `exportRule` a 0.0.0.0/0. Los pools virtuales se definen en la `storage` sección. Cada pool virtual individual define el suyo propio `serviceLevel`, y algunos pools sobrescriben los valores predeterminados. Se utilizaron etiquetas de pools virtuales para diferenciar los pools según `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-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
```

```
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----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. Uso `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: netapp.io/trident
parameters:
  selector: "performance=extreme; protection=extra"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-standard-protection
provisioner: netapp.io/trident
parameters:
  selector: "performance=premium; protection=standard"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-premium-extra-protection
provisioner: netapp.io/trident
parameters:
  selector: "performance=premium; protection=extra"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-premium
provisioner: netapp.io/trident
parameters:
  selector: "performance=premium; protection=standard"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-standard
provisioner: netapp.io/trident
parameters:
  selector: "performance=standard"
allowVolumeExpansion: true
```

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

- El primer tipo de almacenamiento (`cvs-extreme-extra-protection`) se asigna al primer grupo virtual. Se trata del único pool que ofrece un rendimiento extremo con una reserva Snapshot del 10%.
- El último tipo de almacenamiento (`cvs-extra-protection`) llama a cualquier agrupación de almacenamiento que ofrezca 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 mínima del back-end mediante `storageClass` Para especificar el tipo de servicio CVS y el valor predeterminado `standardsw` nivel de servicio.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
storageClass: software
apiRegion: us-east4
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----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 entorno de administración 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-----
    MIIEvAIBADANBgkqhkiG9w0BAQEFAASCBKYwggSiAgEAAoIBAQDaT+Oui9FBAw19
    L1AGEkrYU5xd9K5NlO5jMkIFND5wCD+Nv+jd1GvtFRLaLK5RvXyF5wzvztmODNS+
    qtScpQ+5cFpQkuGtv9U9+N6qtuVYYO3b504Kp5CtqVPJCgMJaK2j8pZTIqUiMum/
    5/Y9oTbZrjAHSMgJm2nHzFq2X0rqVMaHghI6ATm4DOuWx8XGWKTGIPlc0qPqJlqS
    LLaWOH4VIZQZCAyW5IUp9CAmwqHgdG0uhFNfCgMmED6PBUvVLsLvcq86X+QSWR9k
    ETqElj/sGCenPF7ti1DhGBFafd9hPnxg9PZY29ArEZwY9G/ZjZQX7WPgs0VvxiNR
    DxZRC3GXAgMBAAECggEACn5c59bG/qnVEVI1CwMAalM5M2z09JFhlLlljKwntNPj
    Vilw2eTW2+UE7HbJru/S7KQgA5Dnn9kvCraEahPRuddUMrD0vG4kTl/IODV6uFuk
    Y0sZfbqd4jMUQ21smvGsqFzwloYWS5qzO1W83ivXH/HW/iqkmY2eW+EPRS/hwSSu
    SscR+SojI7PB0BWSJhlV4yqYf3vcD/D95el2CVHfRCkL85DKumeZ+yHEnpiXGZAE
    t8xSs4a5OOPm6NHhevCw2a/UQ95/foXNUR450HtbjieJo5o+FF6EYZQGfU2ZHZO8
    37FBKuaJkdGW5xqaI9TL7aqkGkFMF4F2qvOZM+vy8QKBgQD4oVuOkJDlhkTHP86W
    esFlw1kpWyJR9ZA7LI0g/rVpslnX+XdDq0WQf4umdLNau5hYEH9LU6ZSGs1Xk3/B
    NHwR6OXFuqEKNiu83d0zSlHhTy7PZpOZdj5a/vVvQfPDMz7OvsqLRd7YCAbdzuQ0
    +Ahq0Ztwvg0HQ64hdW0ukpYRRwKBgQDgyHj98oqswoYuIa+pP1yS0pPwLmjwKyNm
    /HayzCp+Qjiyy7Tzg8AUqlH1Ou83XbV428jvg7kDhO7PCCKFq+mMmfqHmTpb0Maq
    KpKnZg4ipsqP1yHNNEoRmcailXbwIhCLewMqMrggUiLOmCw4PscL5nK+4GKu2XE1
    jLqjWAZFMQKBgFHkQ9XXRAJ1kR3XpGHoGN890pZOkCVSrqju6aUef/5KYlFCt8ew
    F/+aIxM2iQSvmWQYOvVCnhuY/F2GFaQ7d0om3decuwI0CX/xy7PjHMkLXa2uaZs4
    WR17sLduj62RqXRLX0c0QkwBiNFyHbRcpdkZJQujbYMhBa+7j7SxT4BtAoGAWMWT
    UucocRXZm/pdvz9wteNH3YDWnJLMxm1KC06qMXbBoYrliY4sm3ywJWMC+iCd/H8A
    Gecxd/xVu5mA2L2N3KMq18Zhz8Th0G5DwKyDRJgOQ0Q46yuNXOoYEjlo4Wjyk8Me
    +tlQ8iK98E0UmZnhTgfSpSNElbz2AqnzQ3MN9uECgYAqdvdVPnKGfvdtZ2DjyMoJ
    E89UIC41WjjJGmHsd8W65+3X0RwMzKMT6aZc5tK9J5dHvmWIETnbM+lTImdBBFga
    NWOC6f3r2xbGXHhaWSl+nobpTuvlo56ZRJVvVk7lFMsiddzMuHH8pxfgNJemwA4P
    ThDHCejv035NNV6KyoO0tA==
    -----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

Descubra cómo crear y usar un back-end de Element con su instalación de Astra Trident.

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".

Modos de volumen

La `solidfire-san` el controlador de almacenamiento admite ambos modos de volumen: archivo y bloque. Para la `Filesystem` 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	No hay sistema de archivos. Dispositivo de bloque RAW.
`solidfire-san`	ISCSI	Bloque	RWO, ROX, RWX	No hay sistema de archivos. Dispositivo de bloque RAW.
`solidfire-san`	ISCSI	Sistema de archivos	RWO, ROX	`xfs`, `ext3`, `ext4`
`solidfire-san`	ISCSI	Sistema de archivos	RWO, ROX	`xfs`, `ext3`, `ext4`

Astra Trident utiliza CHAP cuando funciona como un aprovisionador CSI mejorado. Si está utilizando CHAP (que es el valor predeterminado para CSI), no es necesario realizar ninguna otra preparación. Se recomienda establecer explícitamente el `UseCHAP` Opción para utilizar CHAP con Trident que no sea CSI. De lo contrario, consulte "aquí".

Los grupos de acceso de volúmenes solo son compatibles con el marco convencional que no es CSI para Astra Trident. Cuando se configura para funcionar en el modo CSI, Astra Trident utiliza CHAP.

Si ninguno `AccessGroups` o. `UseCHAP` están definidas, se aplica una de las siguientes reglas:

- Si es el valor predeterminado `trident` se ha detectado el grupo de acceso; se utilizan los grupos de acceso.
- Si no se detecta ningún grupo de acceso y la versión de Kubernetes es 1.7 o posterior, se utiliza CHAP.

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)

Parámetro	Descripción	Predeterminado
`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 la autenticación de iSCSI	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 utilizar `debugTraceFlags` a menos que esté solucionando problemas y necesite un volcado de registro detallado.

Ejemplo 1: Configuración de back-end para `solidfire-san` 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 clases de almacenamiento para consumir cada una de ellas mediante el `IOPS` parámetro de clase de almacenamiento.

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

Ejemplo 2: Configuración de clase de almacenamiento y de entorno de administración 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 grupos de almacenamiento, que establecen el `type` En 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. Con el

`parameters.selector` Field, cada clase de almacenamiento llama a qué pools virtuales se pueden utilizar para alojar un volumen. El volumen tendrá los aspectos definidos en el pool virtual elegido.

El primer tipo de almacenamiento (`solidfire-gold-four`) se asignará al primer grupo virtual. Este es el único pool que ofrece rendimiento de oro con un `Volume Type QoS` De oro. El último tipo de almacenamiento (`solidfire-silver`) llama a cualquier pool de almacenamiento que ofrezca un rendimiento elevado. 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 información sobre la configuración de un back-end de ONTAP con controladores SAN de ONTAP y Cloud Volumes ONTAP.

Información importante sobre los controladores de SAN de ONTAP

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

- Debe usar `ontap-nas` para cargas de trabajo de producción que requieren protección de datos, recuperación ante desastres y movilidad.
- Uso `ontap-san-economy` Cuando se espera que el uso previsto de volumen sea mucho superior al soporte de ONTAP.
- Uso `ontap-nas-economy` Únicamente en los casos en los que se espera que el uso previsto del volumen sea mucho superior al soporte de ONTAP y la `ontap-san-economy` no se puede utilizar el conductor.
- No utilizar `ontap-nas-economy` si prevé la necesidad de protección de datos, recuperación ante desastres o movilidad.

Permisos de usuario

Astra Trident espera que se ejecute como administrador de ONTAP o SVM, normalmente mediante el `admin` usuario del clúster o un `vsadmin` Usuario de SVM o un usuario con un nombre diferente que tenga el mismo rol. Para puestas en marcha de Amazon FSX para ONTAP de NetApp, Astra Trident espera que se ejecute como administrador de ONTAP o SVM, mediante el clúster `fsxadmin` usuario o un `vsadmin` Usuario de SVM o un usuario con un nombre diferente que tenga el mismo rol. La `fsxadmin` el usuario es un reemplazo limitado para el usuario administrador del clúster.

> Si utiliza la `limitAggregateUsage` parámetro, se necesitan permisos de administrador de clúster. Cuando se utiliza Amazon FSX para ONTAP de NetApp con Astra Trident, el `limitAggregateUsage` el parámetro no funciona con el `vsadmin` y.. `fsxadmin` cuentas de usuario. La operación de configuración generará un error si se especifica este parámetro.

Si bien es posible crear una función más restrictiva dentro de ONTAP que pueda 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 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 un `san-dev` clase que utiliza `ontap-san` controlador y a `san-default` clase que utiliza `ontap-san-economy` uno.

Todos sus nodos de trabajo de Kubernetes deben tener instaladas las herramientas iSCSI adecuadas. Consulte "Prepare el nodo de trabajo" para obtener más detalles.

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 `admin` o. `vsadmin` Garantizar la máxima compatibilidad con 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 utilizar funciones estándar predefinidas 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 los compatibilidad con el rol de inicio de sesión de seguridad ONTAP `cert` método de autenticación.

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

5. 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...Vaaalllluuuueeee",
"clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
"trustedCACertificate": "QNFinfO...SiqOyN",
"storagePrefix": "myPrefix_"
}

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

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 ejecutarse `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` y.. `ontap-san-economy` de windows Esto requiere habilitar el `useCHAP` opción en su definición de backend. Cuando se establece en `true`, Astra Trident configura la seguridad del iniciador predeterminada de la SVM en CHAP bidireccional y establece el nombre de usuario y los secretos del archivo de entorno de administración. 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
```

La `useCHAP` Parameter es una opción booleana que solo se puede configurar una vez. De forma predeterminada, se establece en FALSE. Después de configurarlo en true, no puede establecerlo en false.

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

Cómo funciona

Mediante ajuste `useCHAP` Para true, el administrador de almacenamiento 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 del iniciador predeterminado de la SVM es none (establecido de forma predeterminada) **y** no hay LUN preexistentes en el volumen, Astra Trident establecerá el tipo de seguridad predeterminado en `CHAP` Y continúe configurando el iniciador de CHAP, el nombre de usuario y los secretos de destino.
  - Si la SVM contiene LUN, Astra Trident no habilitará CHAP en la SVM. De esta forma se garantiza que el acceso a las LUN que ya están presentes en la SVM no esté restringido.
- 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).

Una vez creado el back-end, Astra Trident crea una correspondiente `tridentbackend` CRD y almacena los secretos y nombres de usuario de CHAP 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

Para actualizar las credenciales de CHAP, se deben actualizar los parámetros de CHAP en `backend.json` archivo. Para ello, será necesario actualizar los secretos CHAP y utilizar el `tridentctl update` comando para reflejar estos cambios.

Al actualizar los secretos CHAP para un back-end, debe utilizar `tridentctl` para actualizar el back-end. 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-4378-8a33-1a472ffbeb5c |
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 DE SAN ONTAP

Descubra cómo crear y usar controladores SAN de ONTAP con su instalación de Astra Trident. En esta sección, se ofrecen ejemplos de configuración del back-end y detalles sobre cómo asignar 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
`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
`managementLIF`	La dirección IP de una LIF de gestión de clústeres o SVM Para lograr un cambio de MetroCluster sin problemas, debe especificar una LIF de gestión de SVM. Se puede especificar un nombre de dominio completo (FQDN). Se puede configurar para que utilice direcciones IPv6 si se instaló Astra Trident mediante el `--use -ipv6` bandera. Las direcciones IPv6 deben definirse entre corchetes, como [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	“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 los LIF iSCSI necesarios para establecer una sesión de ruta múltiple. Se genera una advertencia if `dataLIF` se define explícitamente.	Derivado del SVM
`useCHAP`	Use CHAP para autenticar iSCSI para los controladores SAN de ONTAP [Boolean]. Establezca en `true` Para Astra Trident, configure y utilice CHAP bidireccional como 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 detalles.	`false`

Parámetro	Descripción	Predeterminado
`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`	""
`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	Derivado si una SVM `managementLIF` está especificado
`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`

Parámetro	Descripción	Predeterminado
`limitAggregateUsage`	Error al aprovisionar si el uso supera este porcentaje. Si utiliza un entorno de administración de Amazon FSX para ONTAP de NetApp, no especifique `limitAggregateUsage`. El proporcionado `fsxadmin` y.. `vsadmin` No incluya 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 por defecto)
`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 necesite un volcado de log detallado.	`null`

Parámetro	Descripción	Predeterminado
`useREST`	Parámetro booleano para usar las API DE REST de ONTAP. **Vista previa técnica** `useREST` se proporciona como **avance técnico** que se recomienda para entornos de prueba y no para cargas de trabajo de producción. Cuando se establece en `true`, Astra Trident utilizará las API DE 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 `ontap` cliente más. Esto está satisfecho por el predefinido `vsadmin` y.. `cluster-admin` funciones. `useREST` No es compatible con MetroCluster.	`false`

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

Puede controlar el aprovisionamiento predeterminado utilizando 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; “none” (thin) o “VOLUME” (grueso)	“ninguna”
`snapshotPolicy`	Política de Snapshot que se debe usar	“ninguna”

Parámetro	Descripción	Predeterminado
`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. 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/back-end de almacenamiento	""
`snapshotReserve`	Porcentaje de volumen reservado para snapshots «0»	Si `snapshotPolicy` no es “ninguno”, sino “”
`splitOnClone`	Divida un clon de su elemento principal al crearlo	“falso”
`encryption`	Habilite el cifrado de volúmenes de NetApp (NVE) en el volumen nuevo; el valor predeterminado es `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)".	""
`securityStyle`	Estilo de seguridad para nuevos volúmenes	`unix`

Parámetro	Descripción	Predeterminado
`tieringPolicy`	Política de organización en niveles para usar «ninguno»	“Solo Snapshot” para configuración previa a ONTAP 9.5 SVM-DR

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 mediante la `ontap-san` Controlador, Astra Trident añade un 10 % adicional de capacidad a FlexVol para acomodar los metadatos de las 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 la solicitud de PVC = 5GIB, el tamaño total del volumen es de 5.79GIB y el

tamaño disponible es de 5.5GIB. La `volume show` el comando debería mostrar resultados similares a los de este ejemplo:

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 configuración mínima con SAN de ONTAP

Se trata de una configuración básica que utiliza el `ontap-san` 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 configuración mínima de la economía de SAN de ONTAP

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

Ejemplo de autenticación basada en certificados

En este ejemplo de configuración básica `clientCertificate`, `clientPrivateKey`, y. `trustedCACertificate` (Opcional, si se utiliza una CA de confianza) se completan en `backend.json` Y tome los valores codificados 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` establezca 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>
```

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` en ninguno, `spaceAllocation` en falso, y. `encryption` en falso. 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`, `spaceAllocation`, y. `encryption` y algunos pools sustituyen los valores predeterminados.

Ejemplo de SAN ONTAP

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

Ejemplo 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'
```

Asigne los back-ends a StorageClass

Las siguientes definiciones de StorageClass hacen referencia a la Ejemplos de back-ends con pools virtuales. Con el `parameters.selector` Cada StorageClass llama la atención sobre qué pools virtuales pueden usarse para alojar un volumen. El volumen tendrá los aspectos definidos en el pool virtual elegido.

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

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

- La `protection-not-gold` StorageClass se asignará al segundo y tercer pool virtual en `ontap-san` back-end. 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: netapp.io/trident
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
```

- La `app-mysqldb` StorageClass se asignará al tercer pool virtual en `ontap-san-economy` back-end. 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: netapp.io/trident
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
```

- La `protection-silver-creditpoints-20k` StorageClass se asignará al segundo pool virtual de `ontap-san` back-end. 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: netapp.io/trident
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
```

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

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

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

Unidades NAS de ONTAP

Información general del controlador 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 importante acerca de los controladores NAS de ONTAP

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

- Debe usar `ontap-nas` para cargas de trabajo de producción que requieren protección de datos, recuperación ante desastres y movilidad.
- Uso `ontap-san-economy` Cuando se espera que el uso previsto de volumen sea mucho superior al soporte de ONTAP.
- Uso `ontap-nas-economy` Únicamente en los casos en los que se espera que el uso previsto del volumen sea mucho superior al soporte de ONTAP y la `ontap-san-economy` no se puede utilizar el conductor.
- No utilizar `ontap-nas-economy` si prevé la necesidad de protección de datos, recuperación ante desastres o movilidad.

Permisos de usuario

Astra Trident espera que se ejecute como administrador de ONTAP o SVM, normalmente mediante el `admin` usuario del clúster o un `vsadmin` Usuario de SVM o un usuario con un nombre diferente que tenga el mismo rol.

Para puestas en marcha de Amazon FSX para ONTAP de NetApp, Astra Trident espera que se ejecute como administrador de ONTAP o SVM, mediante el clúster `fsxadmin` usuario o un `vsadmin` Usuario de SVM o un usuario con un nombre diferente que tenga el mismo rol. La `fsxadmin` el usuario es un reemplazo limitado para el usuario administrador del clúster.

> Si utiliza la `limitAggregateUsage` parámetro, se necesitan permisos de administrador de clúster. Cuando se utiliza Amazon FSX para ONTAP de NetApp con Astra Trident, el `limitAggregateUsage` el parámetro no funciona con el `vsadmin` y.. `fsxadmin` cuentas de usuario. 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 `ontap-nas` Controlador y clase Bronze que utiliza `ontap-nas-economy` uno.
- Todos sus nodos de trabajo de Kubernetes deben tener instaladas las herramientas NFS adecuadas. Consulte "aquí" para obtener más detalles.

- 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 detalles.

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 `admin` o. `vsadmin` Garantizar la máxima compatibilidad con 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 utilizar funciones estándar predefinidas 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 los compatibilidad con el rol de inicio de sesión de seguridad ONTAP `cert` método de autenticación.

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

5. 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 la LIF tiene 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...Vaaalllluuuueeee",
"clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
"storagePrefix": "myPrefix_"
}

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

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 ejecutarse `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

La versión 20.04 de CSI Trident ofrece la capacidad de gestionar dinámicamente 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.

> La gestión dinámica de las políticas de exportación sólo está disponible para CSI Trident. Es importante asegurarse de que los nodos de trabajo no estén siendo atados.

Ejemplo

Hay dos opciones de configuración que deben utilizarse. A continuación se muestra 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 la mejor práctica recomendada por 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 directiva de exportación para `svm1` SVM y gestionan 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` establezca 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, la `192.168.0.0/24` se proporciona espacio de 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` para un back-end y caer en una política de exportación creada manualmente. Esto requerirá establecer la `exportPolicy` parámetro en la configuración del back-end.

Una vez que Astra Trident crea o actualiza un back-end, puede comprobar el backend mediante `tridentctl` o el correspondiente `tridentbackend` CRD:

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

A medida que se añaden nodos a un clúster de Kubernetes y se registran con la controladora Astra Trident, se actualizan las políticas de exportación de los back-ends existentes (siempre que entren en el rango de direcciones especificado en la `autoExportCIDRs` para el 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 anteriores, actualizando el back-end con `tridentctl update backend` Se asegurará de que Astra Trident gestiona las políticas de exportación de forma automática. Esto creará una nueva política de exportación denominada después de que el UUID del back-end y los volúmenes presentes en el back-end utilicen 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 con `ontap-nas` de windows

> Debe configurar tanto los protocolos NFS como SMB/CIFS en la SVM para crear un `ontap-nas-economy` 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 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. 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 nodos 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 recursos compartidos de administrador de SMB de una de dos formas mediante el "Consola de administración de Microsoft" Complemento carpetas compartidas o uso de 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.

      La `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 todos los detalles.

2. Al crear el back-end, debe configurar lo siguiente para especificar volúmenes de SMB. Para obtener información sobre todas las opciones de configuración del entorno de administración de 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 establecer en `smb`. Si es nulo, el valor predeterminado es `nfs`.	`smb`	`securityStyle`

Parámetro	Descripción	Ejemplo
Estilo de seguridad para nuevos volúmenes. **Debe estar configurado en `ntfs` o. `mixed` Para volúmenes SMB.**	`ntfs` o. `mixed` Para volúmenes de SMB	`unixPermissions`

Opciones y ejemplos de configuración NAS de ONTAP

Descubre cómo crear y utilizar controladores NAS de ONTAP con tu instalación de Astra Trident. En esta sección, se ofrecen ejemplos de configuración del back-end y detalles sobre cómo asignar 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
`storageDriverName`	Nombre del controlador de almacenamiento	“ontap-nas”, “ontap-nas-economy”, “ontap-nas-flexgroup”, “ontap-san” y “ontap-san-economy”
`backendName`	Nombre personalizado o el back-end de almacenamiento	Nombre del conductor + “_” + dataLIF
`managementLIF`	La dirección IP de una LIF de gestión de clústeres o SVM Para lograr un cambio de MetroCluster sin problemas, debe especificar una LIF de gestión de SVM. Se puede especificar un nombre de dominio completo (FQDN). Se puede configurar para que utilice direcciones IPv6 si se instaló Astra Trident mediante el `--use -ipv6` bandera. Las direcciones IPv6 deben definirse entre corchetes, como [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	“10.0.0.1”, “[2001:1234:abcd::fefe]”

Parámetro	Descripción	Predeterminado
`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 . Se puede configurar para que utilice direcciones IPv6 si se instaló Astra Trident mediante el `--use -ipv6` bandera. Las direcciones IPv6 deben definirse entre corchetes, como [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	Dirección especificada o derivada de la SVM, si no se especifica (no recomendada)
`autoExportPolicy`	Habilite la creación y actualización automática de la política de exportación [Boolean]. Con el `autoExportPolicy` y.. `autoExportCIDRs` Astra Trident puede gestionar automáticamente las políticas de exportación.	falso
`autoExportCIDRs`	Lista de CIDR para filtrar las IP de nodo de Kubernetes contra cuándo `autoExportPolicy` está habilitado. Con el `autoExportPolicy` 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	""

Parámetro	Descripción	Predeterminado
`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	
`svm`	Máquina virtual de almacenamiento que usar	Derivado si una SVM `managementLIF` está especificado
`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.	“” (no se aplica por defecto)
`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` Permite personalizar el número máximo de qtrees por FlexVol.	“” (no se aplica por defecto)
`lunsPerFlexvol`	El número máximo de LUN por FlexVol debe estar comprendido entre [50 y 200]	«100»

Parámetro	Descripción	Predeterminado
`debugTraceFlags`	Indicadores de depuración que se deben usar para la solución de problemas. Ejemplo, {“api”:false, “method”:true} No utilizar `debugTraceFlags` 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 `nfs`, `smb` o nulo. El valor predeterminado es nulo en volúmenes de NFS.	`nfs`
`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»

Parámetro	Descripción	Predeterminado
`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. **Vista previa técnica** `useREST` se proporciona como **avance técnico** que se recomienda para entornos de prueba y no para cargas de trabajo de producción. Cuando se establece en `true`, Astra Trident utilizará las API DE 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 `ontap` cliente más. Esto está satisfecho por el predefinido `vsadmin` y.. `cluster-admin` funciones. `useREST` No es compatible con MetroCluster.	falso

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

Puede controlar el aprovisionamiento predeterminado utilizando 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; “none” (thin) o “VOLUME” (grueso)	“ninguna”
`snapshotPolicy`	Política de Snapshot que se debe usar	“ninguna”
`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 snapshots «0»	Si `snapshotPolicy` no es “ninguno”, sino “”
`splitOnClone`	Divida un clon de su elemento principal al crearlo	“falso”
`encryption`	Habilite el cifrado de volúmenes de NetApp (NVE) en el volumen nuevo; el valor predeterminado es `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 usar «ninguno»	“Solo Snapshot” para configuración previa a ONTAP 9.5 SVM-DR
`unixPermissions`	Modo para volúmenes nuevos	“777” para volúmenes NFS; vacío (no aplicable) para volúmenes SMB
`snapshotDir`	Controla la visibilidad de `.snapshot` directorio	“falso”

Parámetro	Descripción	Predeterminado
`exportPolicy`	Política de exportación que se va a utilizar	“predeterminado”
`securityStyle`	Estilo de seguridad para nuevos volúmenes. Compatibilidad con NFS `mixed` y.. `unix` estilos de seguridad. SMB admite `mixed` y.. `ntfs` estilos de seguridad.	El valor predeterminado de NFS es `unix`. La opción predeterminada de SMB es `ntfs`.

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'
```

Para `ontap-nas` y.. `ontap-nas-flexgroups`, Astra Trident utiliza ahora un nuevo cálculo para garantizar que el tamaño de la FlexVol sea correcto con el porcentaje snapshotReserve 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 el usuario solicitó es todo el volumen y. `snapshotReserve` es un porcentaje de esta situación. Con Trident 21.07, lo que el usuario solicita es el espacio editable y Astra Trident define el `snapshotReserve` número como porcentaje del volumen completo. 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. La `volume show` el comando debería mostrar resultados similares a los de este ejemplo:

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 2 GIB con `snapshotReserve=50` Anteriormente, se produjo un volumen que proporciona 1 GIB 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.

Configuración mínima para <code>ontap-nas-economy</code>

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

Configuración mínima para <code>ontap-nas-flexgroup</code>

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

Configuración mínima para 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
```

Autenticación basada en certificados

Este es un ejemplo de configuración de backend mínima. `clientCertificate`, `clientPrivateKey`, y. `trustedCACertificate` (Opcional, si se utiliza una CA de confianza) se completan en `backend.json` Y tome los valores codificados 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_
```

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 el `ontap-nas-economy` y.. `ontap-nas-flexgroup` de windows

```
---
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
```

El uso de direcciones IPv6

Este ejemplo muestra `managementLIF` Uso de 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
```

Amazon FSx para ONTAP mediante volúmenes SMB

La `smbShare` El parámetro es obligatorio para FSx para ONTAP mediante volúmenes de bloque de mensajes del servidor.

```
---
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_
```

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` en ninguno, `spaceAllocation` en falso, y. `encryption` en falso. 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 para `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`, `spaceAllocation`, y. `encryption` y algunos pools sustituyen los valores predeterminados.

Ejemplo de NAS de 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 NAS de ONTAP

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

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

Asigne los back-ends a StorageClass

Las siguientes definiciones de StorageClass se refieren a Ejemplos de back-ends con pools virtuales. Con el `parameters.selector` Cada StorageClass llama la atención sobre qué pools virtuales pueden usarse para alojar un volumen. El volumen tendrá los aspectos definidos en el pool virtual elegido.

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

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

- La `protection-not-gold` StorageClass se asignará al tercer y cuarto pool virtual del `ontap-nas-flexgroup` back-end. 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: netapp.io/trident
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
```

- La `app-mysqldb` StorageClass se asignará al cuarto pool virtual del `ontap-nas` back-end. 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: netapp.io/trident
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
```

- T. `protection-silver-creditpoints-20k` StorageClass se asignará al tercer pool virtual del `ontap-nas-flexgroup` back-end. 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: netapp.io/trident
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
```

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

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

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

Actualizar `dataLIF` 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 AWS totalmente gestionado que permite a los clientes iniciar y ejecutar sistemas de archivos con tecnología del sistema operativo de almacenamiento ONTAP de NetApp. 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.

Descripción general

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.

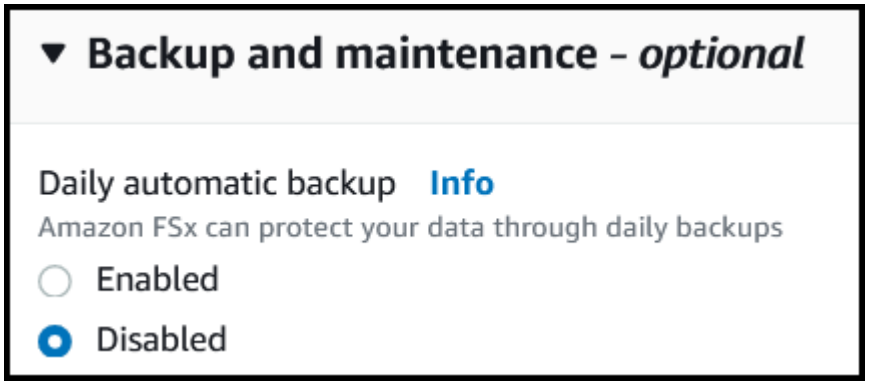
Al utilizar Astra Trident con Amazon FSX para ONTAP de NetApp, puede garantizar que los clústeres de Kubernetes que se ejecutan en Amazon Elastic Kubernetes Service (EKS) pueden aprovisionar volúmenes persistentes de bloques y archivos respaldados por ONTAP.

Usos de Amazon FSX para ONTAP de NetApp "FabricPool" para gestionar los niveles de almacenamiento. Le permite almacenar datos en un nivel, según la frecuencia de acceso a estos.

Consideraciones

- Volúmenes SMB:
  - Se admiten los volúmenes de SMB mediante el `ontap-nas` sólo conductor.
  - Astra Trident admite volúmenes de SMB montados en pods que se ejecutan solo en nodos de Windows.
- Trident no puede eliminar los volúmenes creados en sistemas de archivos Amazon FSX con backups automáticos habilitados. Para eliminar las RVP, es necesario eliminar manualmente el VP y el FSX para el volumen ONTAP. Para evitar este problema:
  - No utilice **creación rápida** para crear el sistema de archivos FSX para ONTAP. El flujo de trabajo de creación rápida permite realizar backups automáticos y no ofrece la opción de anulación de suscripción.
  - Cuando utilice **Standard create**, desactive la copia de seguridad automática. Al deshabilitar los

backups automáticos, Trident puede eliminar correctamente un volumen sin intervención manual adicional.



De Windows

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

- `ontap-san`: Cada VP aprovisionado es una LUN dentro de su propio Amazon FSX para el volumen ONTAP de NetApp.
- `ontap-san-economy`: Cada VP aprovisionado es un LUN con un número configurable de LUN por Amazon FSX para el volumen ONTAP de NetApp.
- `ontap-nas`: Cada VP aprovisionado es un Amazon FSX completo para el volumen ONTAP de NetApp.
- `ontap-nas-economy`: Cada VP aprovisionado es un qtree, con un número configurable de qtrees por Amazon FSX para el volumen ONTAP de NetApp.
- `ontap-nas-flexgroup`: Cada VP aprovisionado es un Amazon FSX completo para el volumen ONTAP FlexGroup de NetApp.

Para obtener más información sobre el controlador, consulte "Controladores ONTAP".

Autenticación

Astra Trident ofrece dos modos de autenticación.

- Basado en certificados: Astra Trident se comunicará con la SVM en su sistema de archivos FSX mediante un certificado instalado en la SVM.
- Basado en credenciales: Puede utilizar el `fsxadmin` usuario del sistema de archivos o del `vsadmin` Usuario configurado para la SVM.

> Astra Trident espera que se ejecute como un `vsadmin` Usuario de SVM o como usuario con un nombre diferente que tenga el mismo rol. Amazon FSX para NetApp ONTAP cuenta con una `fsxadmin` Usuario que es una sustitución limitada de ONTAP `admin` usuario de clúster. Le recomendamos encarecidamente que utilice `vsadmin` Con Astra Trident.

Puede actualizar los back-ends para moverse entre los métodos basados en credenciales y los basados en certificados. Sin embargo, si intenta proporcionar **credenciales y certificados**, la creación de backend fallará. Para cambiar a un método de autenticación diferente, debe eliminar el método existente de la configuración del back-end.

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

- "Autenticación NAS de 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"

Integración de Amazon FSX para ONTAP de NetApp

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.

Requisitos

Además de "Requisitos de Astra Trident", Para integrar FSX para ONTAP con Astra Trident, necesita:

- Un clúster de Amazon EKS existente o un clúster de Kubernetes autogestionado con `kubectl` 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 trabajo 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) en función del tipo de IAM 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 detalles.

Integración de controladores ONTAP SAN y NAS

> Si está configurando para volúmenes SMB, debe leer Prepárese para aprovisionar los volúmenes de SMB antes de crear el back-end.

Pasos

1. Ponga en marcha Astra Trident con una de las "métodos de implementación".
2. Recoja el nombre de DNS del LIF de gestión de SVM. Por ejemplo, si utiliza la CLI de AWS, busque el `DNSName` entrada en `Endpoints` → `Management` tras ejecutar el siguiente comando:

```
aws fsx describe-storage-virtual-machines --region <file system region>
```

3. Cree e instale certificados para "Autenticación de back-end NAS" o. "Autenticación de entorno de administración DE SAN".

Puede iniciar sesión en el sistema de archivos (por ejemplo, para instalar certificados) con SSH desde cualquier lugar que pueda llegar al sistema de archivos. Utilice la `fsxadmin` Usuario, la contraseña que configuró al crear el sistema de archivos y el nombre DNS de gestión desde `aws fsx describe-file-systems`.

4. Cree un archivo de entorno de administración mediante sus certificados y el nombre DNS de la LIF de gestión, como se muestra en el ejemplo siguiente:

YAML

```
---
version: 1
storageDriverName: ontap-san
backendName: customBackendName
managementLIF: svm-XXXXXXXXXXXXXXXXX.fs-XXXXXXXXXXXXXXXXX.fsx.us-
east-2.aws.internal
svm: svm01
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
```

JSON

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "customBackendName",
  "managementLIF": "svm-XXXXXXXXXXXXXXXXX.fs-
XXXXXXXXXXXXXXXXX.fsx.us-east-2.aws.internal",
  "svm": "svm01",
  "clientCertificate": "ZXR0ZXJwYXB...ICMgJ3BhcGVyc2",
  "clientPrivateKey": "vciwKIyAgZG...0cnksIGRlc2NyaX",
  "trustedCACertificate": "zcyBbaG...b3Igb3duIGNsYXNz"
 }
```

Para obtener información sobre la creación de back-ends, consulte estos enlaces:

- "Configurar un back-end con controladores NAS de ONTAP"
- "Configuración de un back-end con controladores SAN de ONTAP"

Resultados

Después de la implementación, puede crear una "clase de almacenamiento, aprovisione un volumen y monte el volumen en un pod".

Prepárese para aprovisionar los volúmenes de SMB

Puede aprovisionar volúmenes SMB mediante el `ontap-nas` controlador. Antes de completar la tarea Integración de controladores ONTAP SAN y NAS complete los siguientes pasos.

Antes de empezar

Para poder aprovisionar volúmenes de SMB con el `ontap-nas` conductor, 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. 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 nodos Kubernetes que se ejecutan en Windows.

Pasos

1. Cree recursos compartidos de SMB. Puede crear recursos compartidos de administrador de SMB de una de dos formas mediante el "Consola de administración de Microsoft" Complemento carpetas compartidas o uso de 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.

      La `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 todos los detalles.

2. Al crear el back-end, debe configurar lo siguiente para especificar volúmenes de SMB. Para obtener información sobre todas las opciones de configuración del entorno de administración de 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 establecer en `smb`.** Si es nulo, el valor predeterminado es `nfs`.	`smb`
`securityStyle`	Estilo de seguridad para nuevos volúmenes. **Debe estar configurado 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 de FSX para ONTAP

Obtenga información acerca de las opciones de configuración de back-end para Amazon FSX para ONTAP. Esta sección proporciona ejemplos de configuración de fondo.

Opciones de configuración del back-end

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`	La dirección IP de una LIF de gestión de clústeres o SVM Para lograr un cambio de MetroCluster sin problemas, debe especificar una LIF de gestión de SVM. Se puede especificar un nombre de dominio completo (FQDN). Se puede configurar para que utilice direcciones IPv6 si se instaló Astra Trident mediante el `--use -ipv6` bandera. Las direcciones IPv6 deben definirse entre corchetes, como [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	“10.0.0.1”, “[2001:1234:abcd::fefe]”

Parámetro	Descripción	Ejemplo
`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. Se puede configurar para que utilice direcciones IPv6 si se instaló Astra Trident mediante el `--use -ipv6` bandera. Las direcciones IPv6 deben definirse entre corchetes, como [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	
`autoExportPolicy`	Habilite la creación y actualización automática de la política de exportación [Boolean]. Con el `autoExportPolicy` y.. `autoExportCIDRs` Astra Trident puede gestionar automáticamente las políticas de exportación.	`false`

Parámetro	Descripción	Ejemplo
`autoExportCIDRs`	Lista de CIDR para filtrar las IP de nodo de Kubernetes contra cuándo `autoExportPolicy` está habilitado. Con el `autoExportPolicy` 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.
`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`

Parámetro	Descripción	Ejemplo
`limitAggregateUsage`	**No especifiques para Amazon FSx para NetApp ONTAP.** El proporcionado `fsxadmin` y.. `vsadmin` No incluya 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` 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. Ejemplo, {“api”:false, “method”:true} No utilizar `debugTraceFlags` a menos que esté solucionando problemas y necesite un volcado de registro 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 `nfs`, `smb`, o nulo. **Debe establecer en `smb` Para volúmenes SMB.** el valor predeterminado es null en volúmenes 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`	Parámetro booleano para usar las API DE REST de ONTAP. **Vista previa técnica** `useREST` se proporciona como **avance técnico** que se recomienda para entornos de prueba y no para cargas de trabajo de producción. Cuando se establece en `true`, Astra Trident utilizará las API DE 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 `ontap` cliente más. Esto está satisfecho por el predefinido `vsadmin` y.. `cluster-admin` funciones.	`false`

Actualizar `dataLIF` 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.

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

Puede controlar el aprovisionamiento predeterminado utilizando 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`
`snapshotPolicy`	Política de Snapshot que se debe usar	`none`

Parámetro	Descripción	Predeterminado
`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 de volumen reservado para snapshots «0»	Si `snapshotPolicy` 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 volumen nuevo; el valor predeterminado es `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".	`false`

Parámetro	Descripción	Predeterminado
`luksEncryption`	Active el cifrado LUKS. Consulte "Usar la configuración de clave unificada de Linux (LUKS)". Solo SAN.	""
`tieringPolicy`	Política de organización en niveles para utilizar `none`	`snapshot-only` Para configuraciones anteriores a ONTAP 9,5 SVM-DR
`unixPermissions`	Modo para volúmenes nuevos. **Dejar vacío para volúmenes SMB.**	""
`securityStyle`	Estilo de seguridad para nuevos volúmenes. Compatibilidad con NFS `mixed` y.. `unix` estilos de seguridad. SMB admite `mixed` y.. `ntfs` estilos de seguridad.	El valor predeterminado de NFS es `unix`. La opción predeterminada de SMB es `ntfs`.

Ejemplo

Uso `nasType`, `node-stage-secret-name`, y. `node-stage-secret-namespace`, Puede especificar un volumen SMB y proporcionar las credenciales necesarias de Active Directory. Se admiten los volúmenes de SMB mediante el `ontap-nas` sólo conductor.

```
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"
```

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` Custom Resource Definition (CRD) permite crear y gestionar back-ends de Trident directamente a través de la

interfaz de Kubernetes. Para ello, utilice `kubectl` O la herramienta CLI equivalente para su distribución de Kubernetes.

`TridentBackendConfig`

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

Sobre la creación de un `TridentBackendConfig` 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.
- La `TridentBackendConfig` está vinculado de manera exclusiva a un `TridentBackend` Eso fue creado por Astra Trident.

Cada uno `TridentBackendConfig` mantiene una asignación de uno a uno con un `TridentBackend`. El primero es la interfaz que se ofrece al usuario para diseñar y configurar los back-ends. El segundo es cómo Trident representa el objeto back-end real.

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

Consulte el siguiente ejemplo para ver el formato del `TridentBackendConfig` CR:

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret
```

También puede echar un vistazo a los ejemplos de la "instalador de trident" directorio para configuraciones de ejemplo para la plataforma o servicio de almacenamiento que desee.

La `spec` toma parámetros de configuración específicos del back-end. En este ejemplo, el back-end utiliza el `ontap-san` controlador de almacenamiento y utiliza los parámetros de configuración que se tabulan aquí. Para obtener una lista de las opciones de configuración del controlador de almacenamiento que desee, consulte "información de configuración del back-end para el controlador de almacenamiento".

La `spec` la sección también incluye `credentials` y.. `deletionPolicy` campos, que se introducen recientemente en `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 `TridentBackendConfig` se ha eliminado. Puede ser necesario uno de los dos valores posibles:
  - `delete`: Esto resulta en la eliminación de ambos `TridentBackendConfig` CR y el back-end asociado. Este es el valor predeterminado.
  - `retain`: Cuando un `TridentBackendConfig` Se elimina la CR, la definición de backend seguirá estando presente y se puede gestionar con `tridentctl`. Establecimiento de la política de eliminación como `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 un `TridentBackendConfig` se ha creado.

El nombre de un back-end 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 establecer explícitamente nombres de backend mediante `spec.backendName`.

Back-ends creados con `tridentctl` no tienen asociado `TridentBackendConfig` objeto. Se pueden optar por gestionar estos back-ends con `kubectl` mediante la creación de un `TridentBackendConfig` CR. Se debe tener cuidado para especificar parámetros de configuración idénticos (como `spec.backendName`, `spec.storagePrefix`, `spec.storageDriverName`, y así sucesivamente). Astra Trident enlazará automáticamente los recién creados `TridentBackendConfig` con el backend preexistente.

Descripción general de los pasos

Para crear un nuevo back-end mediante `kubectl`, debe hacer lo siguiente:

1. Cree un "Secreto Kubernetes". El secreto contiene las credenciales que Astra Trident necesita para comunicarse con el clúster/servicio de almacenamiento.
2. Cree un `TridentBackendConfig` 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 utilizando `kubectl get tbc <tbc-name> -n <trident-namespace>` y recopile 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. A continuación, se muestra 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: t@Ax@7q(>
```

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

Descripción de los campos secretos de la plataforma de almacenamiento	Secreto	Descripción de los campos
Azure NetApp Files	ID 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
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

Descripción de los campos secretos de la plataforma de almacenamiento	Secreto	Descripción de los campos
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 el `spec.credentials` del `TridentBackendConfig` objeto creado en el paso siguiente.

Paso 2: Cree la `TridentBackendConfig` CR

Ya está listo para crear su `TridentBackendConfig` CR. En este ejemplo, un back-end que utiliza `ontap-san` el controlador se crea mediante `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: Compruebe el estado del `TridentBackendConfig` CR

Ahora que creó la `TridentBackendConfig` CR, puede comprobar el estado. Consulte el siguiente ejemplo:

```
kubectl -n trident get tbc backend-tbc-ontap-san
NAME                     BACKEND NAME          BACKEND UUID
PHASE   STATUS
backend-tbc-ontap-san   ontap-san-backend     8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8   Bound   Success
```

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

La fase puede tomar uno de los siguientes valores:

- `Bound`: La `TridentBackendConfig` CR está asociado con un backend, y ese backend contiene `configRef` establezca en la `TridentBackendConfig` El uid de la CR.
- `Unbound`: Representado usando `""`. La `TridentBackendConfig` el objeto no está enlazado a un back-end. Creadas recientemente `TridentBackendConfig` CRS se encuentra en esta fase de forma predeterminada. Tras cambiar la fase, no puede volver a «sin límites».
- `Deleting`: La `TridentBackendConfig` CR `deletionPolicy` se ha configurado para eliminar. Cuando la `TridentBackendConfig` La CR se elimina y pasa al estado de supresión.
  - Si no existen reclamaciones de volumen persistente (RVP) en el back-end, eliminando el `TridentBackendConfig` Como resultado, Astra Trident elimina el back-end, así como el `TridentBackendConfig` CR.
  - Si uno o más EVs están presentes en el backend, pasa a un estado de supresión. La `TridentBackendConfig` Posteriormente, CR también entra en fase de eliminación. El back-end y. `TridentBackendConfig` Se eliminan sólo después de que se hayan eliminado todas las EVs.
- `Lost`: El backend asociado con `TridentBackendConfig` La CR se eliminó accidental o deliberadamente y la `TridentBackendConfig` CR todavía tiene una referencia al backend eliminado. La `TridentBackendConfig` La CR puede ser eliminada independientemente de la `deletionPolicy`

valor.

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

En esta fase, se ha creado un backend. Hay varias operaciones que se pueden realizar además, como "actualizaciones back-end y eliminaciones backend".

(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 YLMA/JSON de `TridentBackendConfig`.

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

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  creationTimestamp: "2021-04-21T20:45:11Z"
  finalizers:
  - trident.netapp.io
  generation: 1
  name: backend-tbc-ontap-san
  namespace: trident
  resourceVersion: "947143"
  uid: 35b9d777-109f-43d5-8077-c74a4559d09c
spec:
  backendName: ontap-san-backend
  credentials:
    name: backend-tbc-ontap-san-secret
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  storageDriverName: ontap-san
  svm: trident_svm
  version: 1
status:
  backendInfo:
    backendName: ontap-san-backend
    backendUUID: 8d24fce7-6f60-4d4a-8ef6-bab2699e6ab8
  deletionPolicy: delete
  lastOperationStatus: Success
  message: Backend 'ontap-san-backend' created
  phase: Bound
```

`backendInfo` contiene el `backendName` y la `backendUUID` del backend que se creó en respuesta a la `TridentBackendConfig` CR. La `lastOperationStatus` el campo representa el estado de la última operación de `TridentBackendConfig` CR, que se puede activar por el usuario (por ejemplo, el usuario ha cambiado algo en `spec`) O activado por Astra Trident (por ejemplo, durante el reinicio de Astra Trident). Puede ser un éxito o un fracaso. `phase` representa el estado de la relación entre el `TridentBackendConfig` CR y el back-end. En el ejemplo anterior, `phase` Tiene el valor enlazado, lo que significa que `TridentBackendConfig` CR está asociado con el backend.

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

> No puede actualizar ni eliminar un backend que contenga un archivo asociado `TridentBackendConfig` objeto con `tridentctl`. Comprender los pasos que implica cambiar entre `tridentctl` y.. `TridentBackendConfig`, "ver aquí".

Realice la gestión del entorno de administración con kubectl

Obtenga información sobre cómo realizar operaciones de administración de back-end mediante `kubectl`.

Eliminar un back-end

Eliminando una `TridentBackendConfig`, Usted instruye a Astra Trident a que elimine/conserve los back-ends (basados en `deletionPolicy`). Para eliminar un back-end, asegúrese de que `deletionPolicy` está configurado para eliminar. Para eliminar sólo la `TridentBackendConfig`, asegúrese de que `deletionPolicy` se establece en retener. De esta forma se asegurará de que el backend esté todavía presente y se pueda gestionar utilizando `tridentctl`.

Ejecute el siguiente comando:

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

Astra Trident no elimina los secretos de Kubernetes que estaban en uso `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` o. `tridentctl get backend -o yaml -n trident` obtener una lista de todos los back-ends que existen. Esta lista también incluirá los back-ends que se crearon 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, el secreto Kubernetes que se utiliza en la `TridentBackendConfig` el objeto debe actualizarse. 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). En este caso, `TridentBackendConfig` Los objetos se pueden actualizar directamente mediante Kubernetes.

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

Como alternativa, realice cambios en el existente `TridentBackendConfig` CR ejecutando 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 registros para determinar la causa ejecutando `kubectl get tbc <tbc-name> -o yaml -n trident` o. `kubectl describe tbc <tbc-name> -n trident`.

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

Realizar la administración de back-end con trimentctl

Obtenga información sobre cómo realizar operaciones de administración de back-end 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 aprovisionado volúmenes y snapshots de este back-end que aún existen, al eliminar el back-end se impiden que el departamento de tecnología aprovisione 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` salidas para objetos backend. Utiliza la `jq` utilidad, que debe 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 con el uso `TridentBackendConfig`.

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 administrar los back-ends. Esto plantea las siguientes preguntas:

- Pueden crearse back-ends con `tridentctl` administrarse con `TridentBackendConfig`?
- Pueden crearse back-ends con `TridentBackendConfig` se gestionan mediante `tridentctl`?

Gestione `tridentctl` con los back-ends `TridentBackendConfig`

En esta sección se describen los pasos necesarios para gestionar los back-ends creados con `tridentctl` Directamente mediante la interfaz de Kubernetes creando `TridentBackendConfig` objetos.

Esto se aplica a las siguientes situaciones:

- Los back-ends preexistentes, que no tienen `TridentBackendConfig` porque fueron creados con `tridentctl`.
- Nuevos back-ends que se crearon con `tridentctl`, mientras que otros `TridentBackendConfig` existen 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 utilizando `tridentctl` para gestionar los back-ends que se crearon con él.
- Enlazar los back-ends creados con `tridentctl` a un nuevo `TridentBackendConfig` objeto. Hacerlo significaría que se gestionarán los back-ends `kubectl` y no `tridentctl`.

Para administrar un back-end preexistente mediante `kubectl`, tendrá que crear un `TridentBackendConfig` que enlaza con el 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. Cree un `TridentBackendConfig` objeto. Este contiene detalles sobre el servicio/clúster de almacenamiento y hace referencia al secreto creado en el paso anterior. Se debe tener cuidado para especificar parámetros de configuración idénticos (como `spec.backendName`, `spec.storagePrefix`, `spec.storageDriverName`, y así sucesivamente). `spec.backendName` se debe establecer el nombre del backend existente.

Paso 0: Identificar el back-end

Para crear un `TridentBackendConfig` que se enlaza a un backend existente, necesitará 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: Cree un `TridentBackendConfig` CR

El paso siguiente es 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 fondo 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` tendrá este aspecto:

```
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: Compruebe el estado del `TridentBackendConfig` CR

Después del `TridentBackendConfig` se ha creado, 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
PHASE   STATUS
tbc-ontap-nas-backend  ontap-nas-backend     52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7   Bound   Success

#confirm that no new backends were created (i.e., TridentBackendConfig did
not end up creating a new backend)
tridentctl get backend -n trident
+---------------------+----------------
+--------------------------------------+--------+---------+
|          NAME       | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+---------------------+----------------
+--------------------------------------+--------+---------+
| ontap-nas-backend   | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |      25 |
+---------------------+----------------
+--------------------------------------+--------+---------+
```

El back-end se gestionará completamente mediante el `tbc-ontap-nas-backend TridentBackendConfig` objeto.

Gestione `TridentBackendConfig` con los back-ends `tridentctl`

```
`tridentctl` se puede utilizar para enumerar los back-ends que se crearon
con `TridentBackendConfig`. Además, los administradores también pueden
optar por gestionar completamente estos back-ends `tridentctl` eliminando
`TridentBackendConfig` y eso seguro `spec.deletionPolicy` se establece en
`retain`.
```

Paso 0: Identificar el back-end

Por ejemplo, supongamos que se ha creado el siguiente back-end mediante `TridentBackendConfig`:

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                    BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        delete

tridentctl get backend ontap-san-backend -n trident
+-------------------+----------------
+--------------------------------------+--------+---------+
|       NAME        | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+-------------------+----------------
+--------------------------------------+--------+---------+
| ontap-san-backend | ontap-san      | 81abcb27-ea63-49bb-b606-
0a5315ac5f82 | online |      33 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

Desde la salida, se ve eso `TridentBackendConfig` Se creó correctamente y está enlazado a un backend [observe el UUID del backend].

Paso 1: Confirmar `deletionPolicy` **se establece en** `retain`

Echemos un vistazo al valor de `deletionPolicy`. Esto debe definirse como `retain`. Esto asegurará que cuando un `TridentBackendConfig` Se elimina la CR, la definición de backend seguirá estando presente y se puede gestionar con `tridentctl`.

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                    BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        delete

# Patch value of deletionPolicy to retain
kubectl patch tbc backend-tbc-ontap-san --type=merge -p
'{"spec":{"deletionPolicy":"retain"}}' -n trident
tridentbackendconfig.trident.netapp.io/backend-tbc-ontap-san patched

#Confirm the value of deletionPolicy
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                    BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   81abcb27-ea63-49bb-b606-
0a5315ac5f82   Bound   Success   ontap-san        retain
```

No continúe con el siguiente paso a menos que `deletionPolicy` se establece en `retain`.

Paso 2: Elimine la `TridentBackendConfig` CR

El paso final es eliminar la `TridentBackendConfig` CR. Tras confirmar la `deletionPolicy` se establece en `retain`, puede utilizar Adelante 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 del `TridentBackendConfig` Astra Trident simplemente la elimina sin eliminar realmente el back-end.

Gestione las clases de almacenamiento

Puede crear una clase de almacenamiento, eliminar una clase de almacenamiento y ver las clases de almacenamiento existentes.

Diseñe una clase de almacenamiento

Consulte "aquí" para obtener más información acerca de las clases de almacenamiento y cómo las configura.

Cree una clase de almacenamiento

Después de tener un archivo de clase de almacenamiento, ejecute el siguiente comando:

```
kubectl create -f <storage-class-file>
```

`<storage-class-file>` debe sustituirse por el nombre de archivo de clase de almacenamiento.

Elimine una clase de almacenamiento

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

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

`<storage-class>` debe sustituirse 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 pone en práctica un espacio en blanco `fsType` para los volúmenes que crea. Para los back-ends de iSCSI, se recomienda aplicar `parameters.fsType` En el tipo de almacenamiento. Debe eliminar las clases de almacenamiento existentes y volver a crearlas con `parameters.fsType` especificado.

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 configurando la anotación `storageclass.kubernetes.io/is-default-class` a 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 tener una clase de almacenamiento predeterminada en el clúster en un momento dado. 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 puede responder con el JSON que `tridentctl` Salidas para objetos de backend de Astra Trident. Utiliza la `jq` utilidad, que puede necesitar instalar primero.

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

Realizar operaciones de volumen

Utilice Topología CSI

Astra Trident puede crear y conectar volúmenes a los nodos presentes en un clúster de Kubernetes de forma selectiva mediante el uso de "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 característica de topología CSI "aquí".

Kubernetes ofrece dos modos de enlace de volúmenes únicos:

- Con `VolumeBindingMode` establezca en `Immediate`, Astra Trident crea el volumen sin conocimiento de la topología. La vinculación de volúmenes y el aprovisionamiento dinámico se manejan cuando se crea la

RVP. Este es el valor predeterminado `VolumeBindingMode` y es adecuado para clústeres que no aplican restricciones de topología. Los volúmenes persistentes se crean sin dependencia alguna de los requisitos de programación del POD solicitante.

- Con `VolumeBindingMode` establezca en `WaitForFirstConsumer`, La creación y enlace de un volumen persistente para una RVP se retrasa hasta que se programa y crea un pod que usa 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.

> La `WaitForFirstConsumer` el modo de encuadernación 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 incluyan 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"
[node1,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node1","kubernetes.io/
os":"linux","node-
role.kubernetes.io/master":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-a"}]
[node2,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node2","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-b"}]
[node3,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node3","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-c"}]
```

Paso 1: 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 back-end puede llevar un opcional `supportedTopologies` bloque que representa una lista de zonas y regiones que se deben admitir. 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.

Puede definir `supportedTopologies` por pool de almacenamiento también. 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
  region: Iowa-DC
  zone: Iowa-DC-A
  supportedTopologies:
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-a
- labels:
    workload: dev
  region: Iowa-DC
  zone: Iowa-DC-B
  supportedTopologies:
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-b
```

En este ejemplo, la `region` y.. `zone` las etiquetas indican la ubicación del pool de almacenamiento. `topology.kubernetes.io/region` y.. `topology.kubernetes.io/zone` dicte desde donde 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 del tipo de almacenamiento que se proporciona 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. La `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:

```
kubectl create -f pvc.yaml
persistentvolumeclaim/pvc-san created
kubectl get pvc
NAME      STATUS    VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS
AGE
pvc-san   Pending                                      netapp-san-us-east1
2s
kubectl describe pvc
Name:          pvc-san
Namespace:     default
StorageClass:  netapp-san-us-east1
Status:        Pending
Volume:
Labels:        <none>
Annotations:   <none>
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:
Access Modes:
VolumeMode:    Filesystem
Mounted By:    <none>
Events:
  Type    Reason                Age   From                         Message
  ----    ------                ----  ----                         -------
  Normal  WaitForFirstConsumer  6s    persistentvolume-controller  waiting
for first consumer to be created before binding
```

Para que Trident cree un volumen y lo 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 de los nodos presentes en el `us-east1` region y elija de cualquier nodo que esté presente en el `us-east1-a` o. `us-east1-b` zonas.

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 incluirán `supportedTopologies`

Se pueden actualizar los back-ends preexistentes 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

Es posible crear snapshots de Kubernetes (snapshot de volumen) de volúmenes persistentes (VP) para mantener copias de un momento específico de los volúmenes Astra Trident. Además, es posible crear un nuevo volumen, también conocido como *clone*, a partir de una snapshot de volumen existente. Admite copias de Snapshot de volumen `ontap-nas`, `ontap-nas-flexgroup`, `ontap-san`, `ontap-san-economy`, `solidfire-san`, `gcp-cvs`, y. `azure-netapp-files` de windows

Antes de empezar

Debe tener un controlador de instantánea externo y definiciones de recursos personalizados (CRD). Esta es la responsabilidad del orquestador de Kubernetes (por ejemplo: Kubeadm, GKE, OpenShift).

Si su distribución de Kubernetes no incluye el controlador de instantáneas ni los CRD, consulte Implementar 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.

Paso 1: Cree un `VolumeSnapshotClass`

En este ejemplo, se crea una clase de snapshot de volumen.

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

La `driver` Señala el controlador CSI de Astra Trident. `deletionPolicy` puede ser `Delete` o. `Retain`. Cuando se establece en `Retain`, la instantánea física subyacente en el clúster de almacenamiento se conserva incluso cuando `VolumeSnapshot` el objeto se ha eliminado.

Para obtener más información, consulte el enlace:./trident-reference/objects.html#kubernetes-volumesnapshotclass-objects[`VolumeSnapshotClass`].

Paso 2: Crear una instantánea de una RVP existente

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, la snapshot se crea para una RVP llamada `pvc1` y el nombre de la copia de snapshot se establece en `pvc1-snap`.

```
kubectl create -f snap.yaml
volumesnapshot.snapshot.storage.k8s.io/pvc1-snap created

kubectl get volumesnapshots
NAME                   AGE
pvc1-snap              50s
```

Esto creó un `VolumeSnapshot` objeto. Un VolumeSnapshot es análogo a un PVC y está asociado a un `VolumeSnapshotContent` objeto que representa la instantánea real.

Es posible identificar la `VolumeSnapshotContent` objeto para `pvc1-snap` VolumeSnapshot, describiéndolo.

```
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
.
.
```

La `Snapshot Content Name` Identifica el objeto VolumeSnapshotContent que sirve esta snapshot. La `Ready To Use` Parámetro indica que la Snapshot se puede usar para crear una RVP nueva.

Paso 3: Creación de EVs a partir de VolumeSnapshots

En este ejemplo, se crea una RVP con una copia Snapshot.

```
cat pvc-from-snap.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: golden
  resources:
    requests:
      storage: 3Gi
  dataSource:
    name: pvc1-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

`dataSource` Muestra que la RVP debe crearse con un VolumeSnapshot llamado `pvc1-snap` como la fuente

de los datos. Esto le indica a Astra Trident que cree una RVP a partir de la snapshot. Una vez creada la RVP, se puede conectar a un pod y utilizarla como cualquier otro PVC.

> La RVP debe crearse en el mismo espacio de nombres que su `dataSource`.

Eliminación de un VP con instantáneas

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.

Implementar 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.yam
l
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshots.yaml
```

2. 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 actualícelo `namespace` en el espacio de nombres.

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 con el `ontap-nas` y.. `ontap-nas-economy` de windows Habilite el `.snapshot` directorio para recuperar 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.

Enlaces relacionados

- "Copias de Snapshot de volumen"
- "VolumeSnapshotClass"

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.

> La ampliación del volumen iSCSI se admite en el `ontap-san`, `ontap-san-economy`, `solidfire-san` Requiere Kubernetes 1.16 o posterior.

Descripción general

Para expandir un VP iSCSI, se deben realizar los siguientes pasos:

- Editar la definición de StorageClass para establecer el `allowVolumeExpansion` campo a. `true`.
- Edición de la definición de PVC y actualización de `spec.resources.requests.storage` para reflejar el nuevo tamaño deseado, que debe ser mayor que el tamaño original.
- Para que se pueda cambiar el tamaño, se debe conectar el PV a un pod. 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.

El ejemplo siguiente muestra cómo funcionan las VP iSCSI.

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

```
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
```

En el caso de un tipo de almacenamiento existente, edítelo para incluir el `allowVolumeExpansion` parámetro.

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

```
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
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi
RWO            ontap-san      8s

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

Paso 3: Defina un pod que fije el PVC

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

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

 kubectl describe pvc san-pvc
Name:          san-pvc
Namespace:     default
StorageClass:  ontap-san
Status:        Bound
Volume:        pvc-8a814d62-bd58-4253-b0d1-82f2885db671
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
               volume.beta.kubernetes.io/storage-provisioner:
csi.trident.netapp.io
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      1Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Mounted By:    ubuntu-pod
```

Paso 4: Expanda 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 |
PROTOCOL |             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 admite la ampliación de volúmenes para los VP de NFS aprovisionados 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 VP de NFS, el administrador primero tiene que configurar la clase de almacenamiento para permitir la expansión del volumen estableciendo el `allowVolumeExpansion` campo a. `true`:

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

Si ya ha creado una clase de almacenamiento sin esta opción, puede simplemente editar la clase de almacenamiento existente mediante `kubectl edit storageclass` 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 VP de 20 MiB recién creado a 1 GIB, edite el RVP y establezca `spec.resources.requests.storage` A 1 GB:

```
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

Es posible importar volúmenes de almacenamiento existentes como un 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.

- Sin embargo `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` En el PV. 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 reclamaciones de la clase de almacenamiento es `delete`, El volumen de almacenamiento se eliminará cuando se elimine el PV.
- De forma predeterminada, Astra Trident gestiona la RVP y cambia el nombre de FlexVol y LUN en el back-end. Puede pasar el `--no-manage` indicador para importar un volumen no gestionado. Si utiliza `--no-manage`, Astra Trident no realiza ninguna operación adicional en el PVC o PV durante el ciclo de vida de los objetos. El volumen de almacenamiento no se elimina cuando se elimina el 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 utilizar `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, se puede especificar `unixPermissions` En su definición de PVC.

   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 la `tridentctl import` 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). La `-f` Se necesita un argumento para especificar la ruta al archivo PVC.

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

Ejemplos

Revise los siguientes ejemplos de importación de volúmenes para los controladores compatibles.

NAS de ONTAP y NAS FlexGroup de ONTAP

Astra Trident admite la importación de volúmenes mediante el `ontap-nas` y.. `ontap-nas-flexgroup` de windows

- La `ontap-nas-economy` el controlador no puede importar y gestionar qtrees.
- La `ontap-nas` y.. `ontap-nas-flexgroup` las controladoras no permiten nombres de volúmenes duplicados.

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

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 llamado `managed_volume` en un backend llamado `ontap_nas`:

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

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

Volumen no gestionado

Cuando utilice la `--no-manage` Argumento, Astra Trident no cambia el nombre del volumen.

El siguiente ejemplo importa `unmanaged_volume` en la `ontap_nas` backend:

```
tridentctl import volume nas_blog unmanaged_volume -f <path-to-pvc-
file> --no-manage

+------------------------------------------+---------+---------------
+----------+------------------------------------------+---------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------+---------------
+----------+------------------------------------------+---------+---------+
| pvc-df07d542-afbc-11e9-8d9f-5254004dfdb7 | 1.0 GiB | standard      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | false   |
+------------------------------------------+---------+---------------
+----------+------------------------------------------+---------+---------+
```

SAN de ONTAP

Astra Trident admite la importación de volúmenes mediante el `ontap-san` controlador.

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

Ejemplos de SAN de ONTAP

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

Volumen gestionado

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

El siguiente ejemplo importa el `ontap-san-managed` FlexVol que está presente en el `ontap_san_default` backend:

```
tridentctl import volume ontapsan_san_default ontap-san-managed -f pvc-
basic-import.yaml -n trident -d

+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
| pvc-d6ee4f54-4e40-4454-92fd-d00fc228d74a | 20 MiB | basic         |
block    | cd394786-ddd5-4470-adc3-10c5ce4ca757 | online | true    |
+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
```

Volumen no gestionado

El siguiente ejemplo importa `unmanaged_example_volume` en la `ontap_san` backend:

```
tridentctl import volume -n trident san_blog unmanaged_example_volume
-f pvc-import.yaml --no-manage
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
| pvc-1fc999c9-ce8c-459c-82e4-ed4380a4b228 | 1.0 GiB | san-blog      |
block    | e3275890-7d80-4af6-90cc-c7a0759f555a | online | false   |
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
```

Si tiene LUN asignadas a iGroups que comparten un IQN con un IQN de nodo de Kubernetes, como se muestra en el ejemplo siguiente, recibirá el error: `LUN already mapped to initiator(s) in this group`. Deberá quitar el iniciador o desasignar la LUN para importar el volumen.

```
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 el `solidfire-san` 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

El siguiente ejemplo 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 |
PROTOCOL |              BACKEND UUID              | STATE  | MANAGED |
+------------------------------------------+--------+--------------
+----------+------------------------------------------+--------+---------+
| pvc-970ce1ca-2096-4ecd-8545-ac7edc24a8fe | 10 GiB | basic-element |
block    | d3ba047a-ea0b-43f9-9c42-e38e58301c49 | online | true    |
+------------------------------------------+--------+--------------
+----------+------------------------------------------+--------+---------+
```

Google Cloud Platform

Astra Trident admite la importación de volúmenes mediante el `gcp-cvs` 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 del `:/`. Por ejemplo, si la ruta de exportación es `10.0.0.1:/adroit-jolly-swift`, la ruta de volumen es `adroit-jolly-swift`.

Ejemplo de Google Cloud Platform

El siguiente ejemplo importa a. `gcp-cvs` volumen en el back-end `gcpcvs_YEppr` con la ruta del 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 mediante el `azure-netapp-files` y.. `azure-netapp-files-subvolume` de windows

> 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 del `:/`. Por ejemplo, si la ruta de montaje es `10.0.0.2:/importvol1`, la ruta de volumen es `importvol1`.

Ejemplo de Azure NetApp Files

El siguiente ejemplo importa un `azure-netapp-files` volumen en el back-end `azurenetappfiles_40517` con la ruta del volumen `importvol1`.

```
tridentctl import volume azurenetappfiles_40517 importvol1 -f <path-to-
pvc-file> -n trident

+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
| pvc-0ee95d60-fd5c-448d-b505-b72901b3a4ab | 100 GiB | anf-storage   |
file     | 1c01274f-d94b-44a3-98a3-04c953c9a51e | online | true    |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
```

Comparta un volumen NFS en espacios de nombres

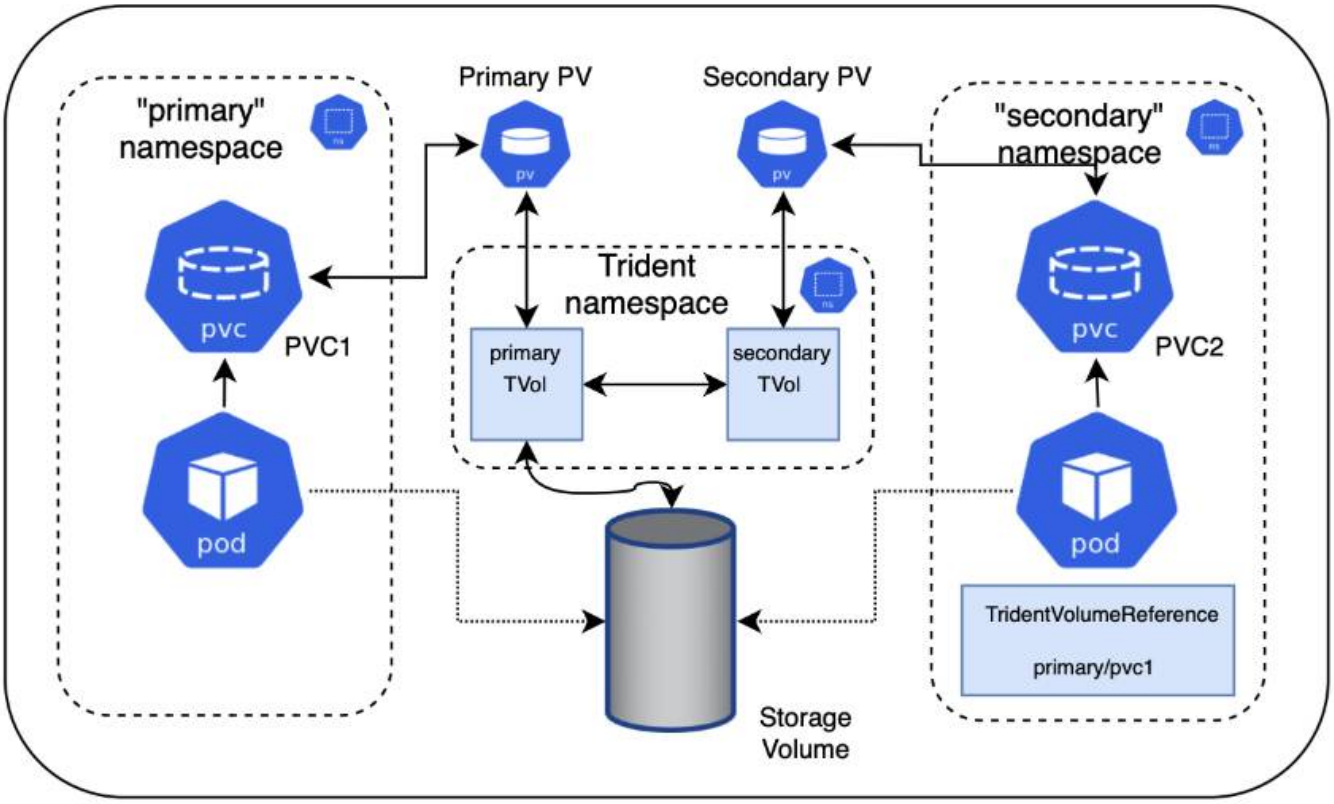
Con Astra Trident, se puede crear un volumen en un espacio de nombres primario y compartirlo en uno o más espacios de nombres secundarios.

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 trimentctl 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 RVP 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 Conceder 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 el PVC subordinado 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:** cree el PVC (`pvc1`) en el espacio de nombres de origen que concede permiso para compartir con el espacio de nombres de destino (`namespace2`) utilizando el `shareToNamespace` anotación.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1
  namespace: namespace1
  annotations:
    trident.netapp.io/shareToNamespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

   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 todos los espacios de nombres mediante `*`. Por ejemplo: `trident.netapp.io/shareToNamespace: *`
- Puede actualizar la RVP para incluir el `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:** cree un sistema TridentVolumeReference CR en el espacio de nombres de destino que haga referencia al espacio de nombres de origen `pvc1`.

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. **Propietario del espacio de nombres de destino:** cree un PVC (`pvc2`) en el espacio de nombres de destino (`namespace2`) utilizando el `shareFromPVC` Anotación para designar el PVC de origen.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  annotations:
    trident.netapp.io/shareFromPVC: namespace1/pvc1
  name: pvc2
  namespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

El tamaño del 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 apunta al VP de origen y comparte 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.

Uso `tridentctl get` para consultar volúmenes subordinados

Con el[`tridentctl` puede ejecutar la `get` comando para obtener volúmenes subordinados. Para obtener más información, consulte el enlace:./trident-reference/tridentctl.html[`tridentctl` comandos y opciones].

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

Indicadores:

- `-h, --help`: Ayuda para volúmenes.
- `--parentOfSubordinate string`: Limite la consulta al volumen de origen subordinado.
- `--subordinateOf string`: Limite la consulta a las subordinadas del 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 al PVC de origen quitando el `shareToNamespace` o. `shareFromNamespace` anotaciones o eliminar `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".

Supervisión de Astra Trident

Astra Trident proporciona un conjunto de extremos de métricas de Prometheus que puedes utilizar para supervisar el rendimiento de Astra Trident.

Descripción general

Las métricas proporcionadas por Astra Trident le permiten hacer lo siguiente:

- Mantenga pestañas sobre el estado y la configuración de Astra Trident. Puede examinar la eficacia de las operaciones y si puede comunicarse con los back-ends como se esperaba.
- Examine la información de uso del back-end, y comprenda cuántos volúmenes se aprovisionan en un entorno de administración y la cantidad de espacio consumido, etc.
- Mantenga una asignación de la cantidad de volúmenes aprovisionados en los back-ends disponibles.
- Seguimiento del rendimiento. Podrá observar el tiempo que tarda Astra Trident en comunicarse con los back-ends y realizar operaciones.

De forma predeterminada, las métricas de Trident se exponen en el puerto de destino `8001` en la `/metrics` extremo. Estas métricas están **activadas de forma predeterminada** cuando se instala Trident.

Lo que necesitará

- Un clúster de Kubernetes con Astra Trident instalado.
- Una instancia Prometheus. Esto puede ser un "Puesta en marcha de Prometeo en contenedores" También puede optar por ejecutar Prometheus como a. "aplicación nativa".

Paso 1: Definir un objetivo Prometheus

Debe definir un destino Prometheus para recopilar las métricas y obtener información sobre los back-ends que administra Astra Trident, los volúmenes que crea, etc. Este "blog" Explica cómo puede usar Prometheus y Grafana con Astra Trident para recuperar métricas. En el blog se explica cómo puede ejecutar Prometheus como operador de su clúster de Kubernetes y la creación de un ServiceMonitor para obtener las métricas de Astra Trident.

Paso 2: Cree un Prometheus ServiceMonitor

Para usar las métricas de Trident, debe crear un Prometheus ServiceMonitor que vaya a ver el `trident-csi` el servicio y escucha el `metrics` puerto. Un ejemplo de ServiceMonitor tiene este aspecto:

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: trident-sm
  namespace: monitoring
  labels:
      release: prom-operator
  spec:
    jobLabel: trident
    selector:
      matchLabels:
        app: controller.csi.trident.netapp.io
    namespaceSelector:
      matchNames:
      - trident
    endpoints:
    - port: metrics
      interval: 15s
```

Esta definición de ServiceMonitor recupera las métricas devueltas por `trident-csi` servicio y busca específicamente la `metrics` extremo del servicio. Como resultado, ahora Prometheus está configurado para comprender el de Astra Trident
métricas.

Además de las métricas disponibles directamente de Astra Trident, kubelet expone muchas `kubelet_volume_*` métricas a través de su propio extremo de métricas. Kubelet puede proporcionar información sobre los volúmenes adjuntos y los pods y otras operaciones internas que realiza. Consulte "aquí".

Paso 3: Consulte las métricas de Trident con PromQL

PromQL es adecuado para crear expresiones que devuelvan datos tabulares o de series temporales.

A continuación se muestran algunas consultas PromQL que se pueden utilizar:

Obtenga información de estado de Trident

- **Porcentaje de respuestas HTTP 2XX de Astra Trident**

```
(sum (trident_rest_ops_seconds_total_count{status_code=~"2.."} OR on()
vector(0)) / sum (trident_rest_ops_seconds_total_count)) * 100
```

- **Porcentaje de respuestas DE DESCANSO de Astra Trident a través del código de estado**

```
(sum (trident_rest_ops_seconds_total_count) by (status_code)  / scalar
(sum (trident_rest_ops_seconds_total_count))) * 100
```

- **Duración media en ms de operaciones realizadas por Astra Trident**

```
sum by (operation)
(trident_operation_duration_milliseconds_sum{success="true"}) / sum by
(operation)
(trident_operation_duration_milliseconds_count{success="true"})
```

Obtenga la información de uso de Astra Trident

- **Tamaño medio del volumen**

```
trident_volume_allocated_bytes/trident_volume_count
```

- **Espacio total por volumen aprovisionado por cada backend**

```
sum (trident_volume_allocated_bytes) by (backend_uuid)
```

Obtenga el uso de cada volumen

Esto solo se habilita si también se recopilan las métricas Kubelet.

- **Porcentaje de espacio usado para cada volumen**

```
kubelet_volume_stats_used_bytes / kubelet_volume_stats_capacity_bytes *
100
```

Obtenga más información sobre la telemetría Astra Trident AutoSupport

De forma predeterminada, Astra Trident envía a NetApp métricas y información básica sobre los back-end a través de una cadencia diaria.

- Para que Astra Trident deje de enviar métricas Prometheus e información básica del back-end a NetApp, pase el `--silence-autosupport` Durante la instalación de Astra Trident.
- Astra Trident también puede enviar registros de contenedores al soporte de NetApp bajo demanda a través `tridentctl send autosupport`. Deberá activar Astra Trident para cargar los registros. Antes de enviar los registros, debe aceptar las "política de privacidad".
- A menos que se especifique lo contrario, Astra Trident recupera los registros de las últimas 24 horas.
- Se puede especificar el lapso de retención del registro con el `--since` bandera. Por ejemplo:

`tridentctl send autosupport --since=1h`. Esta información se recopila y se envía a través de un `trident-autosupport` contenedor
Que se instala junto con Astra Trident. Puede obtener la imagen del contenedor en "AutoSupport de Trident".

- Trident AutoSupport no recopila ni transmite información personal identificable (PII) ni Información personal. Viene con un "CLUF" Esto no se aplica a la imagen del contenedor Trident. Puede obtener más información sobre el compromiso de NetApp con la seguridad y la confianza de los datos "aquí".

Una carga útil de ejemplo enviada por Astra Trident tiene el siguiente aspecto:

```
---
items:
- backendUUID: ff3852e1-18a5-4df4-b2d3-f59f829627ed
  protocol: file
  config:
    version: 1
    storageDriverName: ontap-nas
    debug: false
    debugTraceFlags:
    disableDelete: false
    serialNumbers:
    - nwkvzfanek_SN
    limitVolumeSize: ''
  state: online
  online: true
```

- Los mensajes de AutoSupport se envían al extremo AutoSupport de NetApp. Si está utilizando un Registro privado para almacenar imágenes contenedoras, puede utilizar `--image-registry` bandera.
- También puede configurar direcciones URL proxy generando los archivos YLMA de instalación. Esto se puede hacer usando `tridentctl install --generate-custom-yaml` Para crear los archivos YAML y agregar `--proxy-url` argumento para `trident-autosupport` contenedor en `trident-deployment.yaml`.

Deshabilite las métricas de Astra Trident

Para **desactivar las métricas** de ser reportadas, debe generar YAMLs personalizados (utilizando la `--generate-custom-yaml` y editarlas para eliminar `--metrics` no se invoca el indicador para el `trident-main`
contenedor.

Astra Trident para Docker

Requisitos previos para la implementación

Debe instalar y configurar los requisitos previos del protocolo necesarios en su host antes de poder poner en marcha Astra Trident.

Compruebe los requisitos

- Compruebe que la implementación se adapte a todas las "requisitos".
- Compruebe que tiene instalada una versión compatible de Docker. Si la versión de Docker no está actualizada, "instálelo o actualícelo".

```
docker --version
```

- Compruebe que los requisitos previos de protocolo estén instalados y configurados en el host:

Protocolo	De NetApp	Comandos
NFS	RHEL 8 O POSTERIOR	`sudo yum install -y nfs-utils`
NFS	Ubuntu	`sudo apt-get install -y nfs-common`

Protocolo	De NetApp	Comandos

ISCSI	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 `etc/multipath.conf` contiene `find_multipaths no` inferior `defaults`.
		
		5. Asegúrese de que así sea `iscsid` y.. `multipathd` están en ejecución:
		
		`sudo systemctl enable --now iscsid multipathd`
		
		6. Activar e iniciar `iscsi`:
		
		`sudo systemctl enable --now iscsi`

Protocolo	De NetApp	Comandos

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 `etc/multipath.conf` contiene `find_multipaths no` inferior `defaults`.
		
		5. Asegúrese de que así sea `open-iscsi` y.. `multipath-tools` están habilitadas y en ejecución:
		
		`sudo systemctl status multipath-tools`

Ponga en marcha Astra Trident

Astra Trident para Docker proporciona integración directa con el ecosistema de Docker para las plataformas de almacenamiento de NetApp. Admite el aprovisionamiento y la gestión de recursos de almacenamiento desde la plataforma de almacenamiento hasta hosts Docker, con un marco para añadir plataformas adicionales en el futuro.

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

Pueden ejecutarse múltiples instancias de Astra Trident a la vez en el mismo host. Esto permite conexiones simultáneas a varios sistemas de almacenamiento y tipos de almacenamiento, con la capacidad de personalizar el almacenamiento usado para los volúmenes de Docker.

Lo que necesitará

Consulte "requisitos previos para la implementación". Una vez que se cumplan los requisitos previos, estará listo para poner en marcha Astra Trident.

Método de complemento gestionado por Docker (versión 1.13/17.03 y posteriores)

Antes de empezar

Si ha utilizado Astra Trident pre Docker 1.13/17.03 en el método tradicional del demonio, asegúrese de detener el proceso Astra Trident y reiniciar su daemon Docker antes de utilizar el método de complemento gestionado.

1. Detener todas las instancias en ejecución:

```
pkill /usr/local/bin/netappdvp
pkill /usr/local/bin/trident
```

2. Reinicie Docker.

```
systemctl restart docker
```

3. Asegúrese de tener instalado Docker Engine 17.03 (nuevo 1.13) o una versión posterior.

```
docker --version
```

Si su versión no está actualizada, "instale o actualice la instalación".

Pasos

1. Cree un archivo de configuración y especifique las opciones siguientes:
   - `config`: El nombre de archivo predeterminado es `config.json`, sin embargo, puede utilizar cualquier nombre que elija si especifica `config` opción con el nombre de archivo. El archivo de configuración debe estar ubicado en la `/etc/netappdvp` directorio en el sistema host.
   - `log-level`: Especifique el nivel de registro (`debug`, `info`, `warn`, `error`, `fatal`). El valor predeterminado es `info`.

- `debug`: Especifique si el registro de depuración está activado. El valor predeterminado es false. Reemplaza el nivel de registro si es TRUE.

  i. Cree una ubicación para el archivo de configuración:

  ```
  sudo mkdir -p /etc/netappdvp
  ```

  ii. Cree el archivo de configuración:

  ```
  cat << EOF > /etc/netappdvp/config.json
  {
      "version": 1,
      "storageDriverName": "ontap-nas",
      "managementLIF": "10.0.0.1",
      "dataLIF": "10.0.0.2",
      "svm": "svm_nfs",
      "username": "vsadmin",
      "password": "password",
      "aggregate": "aggr1"
  }
  EOF
  ```

2. Inicie Astra Trident con el sistema de complementos gestionado. Sustituya `<version>` con la versión del complemento (xxx.xx.x) que está utilizando.

```
docker plugin install --grant-all-permissions --alias netapp
netapp/trident-plugin:<version> config=myConfigFile.json
```

3. Comience a usar Astra Trident para consumir almacenamiento desde el sistema configurado.

   a. Cree un volumen denominado "firstVolume":

   ```
   docker volume create -d netapp --name firstVolume
   ```

   b. Cree un volumen predeterminado cuando el contenedor comience:

   ```
   docker run --rm -it --volume-driver netapp --volume
   secondVolume:/my_vol alpine ash
   ```

   c. Quite el volumen "firstVolume":

   ```
   docker volume rm firstVolume
   ```

Método tradicional (versión 1.12 o anterior)

Antes de empezar

1. Asegúrese de que tiene Docker versión 1.10 o posterior.

   ```
   docker --version
   ```

   Si la versión no está actualizada, actualice la instalación.

   ```
   curl -fsSL https://get.docker.com/ | sh
   ```

   O bien, "siga las instrucciones de su distribución".

2. Asegúrese de que esté configurado NFS y/o iSCSI para su sistema.

Pasos

1. Instale y configure el complemento NetApp Docker Volume Plugin:
   a. Descargue y desembale la aplicación:

      ```
      wget
      https://github.com/NetApp/trident/releases/download/v23.04.0/trident-
      installer-23.04.0.tar.gz
      tar zxf trident-installer-23.04.0.tar.gz
      ```

   b. Desplazarse a una ubicación en la ruta de la bandeja:

      ```
      sudo mv trident-installer/extras/bin/trident /usr/local/bin/
      sudo chown root:root /usr/local/bin/trident
      sudo chmod 755 /usr/local/bin/trident
      ```

   c. Cree una ubicación para el archivo de configuración:

      ```
      sudo mkdir -p /etc/netappdvp
      ```

   d. Cree el archivo de configuración:

```
cat << EOF > /etc/netappdvp/ontap-nas.json
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "username": "vsadmin",
    "password": "password",
    "aggregate": "aggr1"
}
EOF
```

2. Después de colocar el archivo binario y crear los archivos de configuración, inicie el daemon Trident con el archivo de configuración deseado.

```
sudo trident --config=/etc/netappdvp/ontap-nas.json
```

> A menos que se especifique, el nombre predeterminado para el controlador de volumen es “netapp”.

Después de iniciar el daemon, puede crear y gestionar volúmenes mediante la interfaz CLI de Docker

3. Cree un volumen:

```
docker volume create -d netapp --name trident_1
```

4. Aprovisione un volumen de Docker al iniciar un contenedor:

```
docker run --rm -it --volume-driver netapp --volume trident_2:/my_vol
alpine ash
```

5. Quite un volumen de Docker:

```
docker volume rm trident_1
docker volume rm trident_2
```

Inicie Astra Trident cuando se inicie el sistema

Puede encontrar un archivo de ejemplo de unidad para sistemas basados en el sistema en `contrib/trident.service.example` En el Git repo. Para utilizar el archivo con RHEL, realice lo siguiente:

1. Copie el archivo en la ubicación correcta.

   Debe utilizar nombres únicos para los archivos de unidad si tiene más de una instancia en ejecución.

   ```
   cp contrib/trident.service.example
   /usr/lib/systemd/system/trident.service
   ```

2. Edite el archivo, cambie la descripción (línea 2) para que coincida con el nombre del controlador y la ruta del archivo de configuración (línea 9) para reflejar su entorno.
3. Vuelva a cargar systemd para que procese los cambios:

   ```
   systemctl daemon-reload
   ```

4. Active el servicio.

   Este nombre varía en función de lo que haya nombrado el archivo en el `/usr/lib/systemd/system` directorio.

   ```
   systemctl enable trident
   ```

5. Inicie el servicio.

   ```
   systemctl start trident
   ```

6. Ver el estado.

   ```
   systemctl status trident
   ```

Siempre que modifique el archivo de unidad, ejecute el `systemctl daemon-reload` comando para que tenga en cuenta los cambios.

Actualice o desinstale Astra Trident

Puede actualizar Astra Trident para Docker sin que ello afecte a los volúmenes que se estén utilizando. Durante el proceso de actualización, habrá un breve periodo en el que `docker volume` los comandos dirigidos al plugin no se llevarán a cabo correctamente y las aplicaciones no podrán montar volúmenes hasta que el plugin se vuelva a ejecutar. En la mayoría de las circunstancias, esto es cuestión de segundos.

Renovar

Realice los siguientes pasos para actualizar Astra Trident for Docker.

Pasos

1. Enumere los volúmenes existentes:

```
docker volume ls
DRIVER              VOLUME NAME
netapp:latest       my_volume
```

2. Desactivar el complemento:

```
docker plugin disable -f netapp:latest
docker plugin ls
ID                  NAME                DESCRIPTION
ENABLED
7067f39a5df5        netapp:latest       nDVP - NetApp Docker Volume
Plugin   false
```

3. Actualizar el complemento:

```
docker plugin upgrade --skip-remote-check --grant-all-permissions
netapp:latest netapp/trident-plugin:21.07
```

> La versión 18.01 de Astra Trident sustituye a nDVP. Debe actualizar directamente desde la `netapp/ndvp-plugin` imagen de `netapp/trident-plugin` imagen.

4. Habilitar el plugin:

```
docker plugin enable netapp:latest
```

5. Compruebe que el plugin está habilitado:

```
docker plugin ls
ID                  NAME                DESCRIPTION
ENABLED
7067f39a5df5        netapp:latest       Trident - NetApp Docker Volume
Plugin   true
```

6. Compruebe que los volúmenes estén visibles:

```
docker volume ls
DRIVER              VOLUME NAME
netapp:latest       my_volume
```

Si actualiza de una versión anterior de Astra Trident (anterior a 20.10) a Astra Trident 20.10 o posterior, es posible que se produzca un error. Para obtener más información, consulte "Problemas conocidos". Si se produce el error, primero debe desactivar el plugin, después eliminar el plugin y, a continuación, instalar la versión Astra Trident necesaria pasando un parámetro de configuración adicional: `docker plugin install netapp/trident-plugin:20.10 --alias netapp --grant-all-permissions config=config.json`

Desinstalar

Siga estos pasos para desinstalar Astra Trident for Docker.

Pasos

1. Quite los volúmenes que haya creado el plugin.
2. Desactivar el complemento:

```
docker plugin disable netapp:latest
docker plugin ls
ID                  NAME                DESCRIPTION
ENABLED
7067f39a5df5        netapp:latest       nDVP - NetApp Docker Volume
Plugin   false
```

3. Quitar el plugin:

```
docker plugin rm netapp:latest
```

Trabaje con volúmenes

Puede crear, clonar y eliminar volúmenes fácilmente siguiendo el estándar `docker volume` Los comandos con el nombre del controlador Astra Trident se especifican cuando es necesario.

Cree un volumen

- Cree un volumen con un controlador con el nombre predeterminado:

```
docker volume create -d netapp --name firstVolume
```

- Cree un volumen con una instancia específica de Astra Trident:

```
docker volume create -d ntap_bronze --name bronzeVolume
```

> Si no especifica ninguna "opciones", se utilizan los valores predeterminados del controlador.

- Anule el tamaño de volumen predeterminado. Consulte el siguiente ejemplo para crear un volumen de 20 GIB con un controlador:

```
docker volume create -d netapp --name my_vol --opt size=20G
```

> Los tamaños de volumen se expresan como cadenas que contienen un valor entero con unidades opcionales (por ejemplo: 10G, 20 GB, 3 TIB). Si no se especifica ninguna unidad, el valor predeterminado es G. Las unidades de tamaño se pueden expresar como potencias de 2 (B, KiB, MIB, GIB, TIB) o de 10 (B, KB, MB, GB, TB). Las unidades abreviadas utilizan potencias de 2 (G = GIB, T = TIB, …).

Quitar un volumen

- Quite el volumen como cualquier otro volumen de Docker:

```
docker volume rm firstVolume
```

> Cuando utilice la `solidfire-san` driver, el ejemplo anterior elimina y purga el volumen.

Realice los siguientes pasos para actualizar Astra Trident for Docker.

Clonar un volumen

Cuando utilice la `ontap-nas`, `ontap-san`, `solidfire-san`, y. `gcp-cvs storage drivers`, Astra Trident puede clonar volúmenes. Cuando utilice la `ontap-nas-flexgroup` o. `ontap-nas-economy` controladores, no se admite la clonación. La creación de un volumen nuevo a partir de un volumen existente dará como resultado la creación de una copia de Snapshot nueva.

- Examine el volumen para enumerar las instantáneas:

```
docker volume inspect <volume_name>
```

- Cree un volumen nuevo a partir de un volumen existente. Esto dará como resultado la creación de una nueva snapshot:

```
docker volume create -d <driver_name> --name <new_name> -o
from=<source_docker_volume>
```

- Cree un volumen nuevo a partir de una snapshot existente en un volumen. Esto no creará una nueva snapshot:

```
docker volume create -d <driver_name> --name <new_name> -o
from=<source_docker_volume> -o fromSnapshot=<source_snap_name>
```

Ejemplo

```
docker volume inspect firstVolume

[
    {
        "Driver": "ontap-nas",
        "Labels": null,
        "Mountpoint": "/var/lib/docker-volumes/ontap-
nas/netappdvp_firstVolume",
        "Name": "firstVolume",
        "Options": {},
        "Scope": "global",
        "Status": {
            "Snapshots": [
                {
                    "Created": "2017-02-10T19:05:00Z",
                    "Name": "hourly.2017-02-10_1505"
                }
            ]
        }
    }
]

docker volume create -d ontap-nas --name clonedVolume -o from=firstVolume
clonedVolume

docker volume rm clonedVolume
docker volume create -d ontap-nas --name volFromSnap -o from=firstVolume
-o fromSnapshot=hourly.2017-02-10_1505
volFromSnap

docker volume rm volFromSnap
```

Acceso a volúmenes creados externamente

Puede acceder a los dispositivos de bloque creados externamente (o a sus clones) mediante contenedores con Trident **only** si no tienen particiones y si su sistema de archivos es compatible con Astra Trident (por ejemplo: An `ext4`-formateado `/dev/sdc1` No se podrá acceder a él a través de Astra Trident).

Opciones de volumen específicas del controlador

Cada controlador de almacenamiento tiene un conjunto diferente de opciones que se pueden especificar al crear un volumen para personalizar el resultado. Consulte a continuación las opciones que se aplican al sistema de almacenamiento configurado.

Usar estas opciones durante la operación de creación de volúmenes es simple. Proporcione la opción y el valor con `-o` Operador durante el funcionamiento de la CLI. Estos sustituyen cualquier valor equivalente al archivo de configuración JSON.

Opciones de volumen de ONTAP

Las opciones de creación de volumen para NFS e iSCSI son las siguientes:

Opción	Descripción
`size`	El tamaño del volumen, de manera predeterminada es 1 GIB.
`spaceReserve`	Aprovisione el volumen de manera thin o thick, de manera predeterminada, es thin. Los valores válidos son `none` (con thin provisioning) y. `volume` (thick-provisioning).
`snapshotPolicy`	Esto establecerá la política de instantáneas en el valor deseado. El valor predeterminado es `none`, es decir, no se creará automáticamente instantáneas para el volumen. A menos que el administrador de almacenamiento lo modifique, existe una política llamada "predeterminada" en todos los sistemas de ONTAP que crea y retiene seis copias Snapshot cada hora, dos días y dos semanas. Los datos conservados en una instantánea se pueden recuperar navegando hacia la `.snapshot` directorio en cualquier directorio del volumen.

Opción	Descripción
`snapshotReserve`	Esto establecerá la reserva de instantáneas en el porcentaje deseado. El valor predeterminado es no, lo que significa que ONTAP seleccionará la reserva de copias Snapshot (generalmente 5 %) si se seleccionó una política de copias Snapshot o 0 % si la política de copias Snapshot no es ninguna. Es posible establecer el valor predeterminado de snapshotReserve en el archivo de configuración para todos los back-ends de ONTAP, y se puede usar como opción de creación de volúmenes para todos los back-ends de ONTAP excepto ontap-nas-Economy.
`splitOnClone`	Al clonar un volumen, ONTAP dividirá inmediatamente el clon de su principal. El valor predeterminado es `false`. Algunos casos de uso para el clonado de volúmenes se sirven mejor dividiendo el clon de su elemento principal inmediatamente después de la creación, ya que es poco probable que haya ninguna oportunidad para la eficiencia del almacenamiento. Por ejemplo, la clonación de una base de datos vacía puede ofrecer un gran ahorro de tiempo pero poco ahorro de espacio de almacenamiento, por lo que es mejor dividir los clones de forma inmediata.
`encryption`	Habilite el cifrado de volúmenes de NetApp (NVE) en el volumen nuevo; el valor predeterminado es `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".
`tieringPolicy`	Establece la política de organización en niveles que se utilizará para el volumen. Esto decide si los datos se mueven al nivel de cloud cuando quedan inactivos (inactivos).

Las siguientes opciones adicionales son para NFS **sólo**:

Opción	Descripción
`unixPermissions`	Esto controla el conjunto de permisos para el propio volumen. De forma predeterminada, los permisos se establecerán en `---rwxr-xr-x, o en notación numérica 0755, y. `root` será el propietario. El texto o el formato numérico funcionará.
`snapshotDir`	Configuración de esta opción en `true` hará la `.snapshot` directorio visible para los clientes que acceden al volumen. El valor predeterminado es `false`, que significa la visibilidad del `.snapshot` el directorio está desactivado de forma predeterminada. Algunas imágenes, por ejemplo la imagen oficial de MySQL, no funcionan como se espera cuando la `.snapshot` el directorio es visible.
`exportPolicy`	Establece la política de exportación que se utilizará para el volumen. El valor predeterminado es `default`.
`securityStyle`	Configura el estilo de seguridad que se usará para acceder al volumen. El valor predeterminado es `unix`. Los valores válidos son `unix` y.. `mixed`.

Las siguientes opciones adicionales son para iSCSI **sólo**:

Opción	Descripción
`fileSystemType`	Configura el sistema de archivos utilizado para formatear volúmenes iSCSI. El valor predeterminado es `ext4`. Los valores válidos son `ext3`, `ext4`, y. `xfs`.
`spaceAllocation`	Configuración de esta opción en `false` Desactivará la función de asignación de espacio de la LUN. El valor predeterminado es `true`, Es decir, ONTAP notifica al host cuando el volumen se ha quedado sin espacio y el LUN del volumen no puede aceptar escrituras. Esta opción también permite que ONTAP reclame espacio automáticamente cuando el host elimina los datos.

Ejemplos

Vea los ejemplos siguientes:

- Cree un volumen de 10 GIB:

```
docker volume create -d netapp --name demo -o size=10G -o
encryption=true
```

- Cree un volumen de 100 GIB con Snapshot:

```
docker volume create -d netapp --name demo -o size=100G -o
snapshotPolicy=default -o snapshotReserve=10
```

- Cree un volumen con el bit setuid activado:

```
docker volume create -d netapp --name demo -o unixPermissions=4755
```

El tamaño de volumen mínimo es 20 MiB.

Si no se especifica la reserva de instantánea y la política de instantánea es `none`, Trident utilizará una reserva de instantáneas del 0%.

- Crear un volumen sin política de Snapshot y sin reserva de Snapshot:

```
docker volume create -d netapp --name my_vol --opt snapshotPolicy=none
```

- Crear un volumen sin política de copias Snapshot y una reserva de copias Snapshot personalizada del 10%:

```
docker volume create -d netapp --name my_vol --opt snapshotPolicy=none
--opt snapshotReserve=10
```

- Crear un volumen con una política de Snapshot y una reserva de Snapshot personalizada del 10%:

```
docker volume create -d netapp --name my_vol --opt
snapshotPolicy=myPolicy --opt snapshotReserve=10
```

- Cree un volumen con una política de snapshots y acepte la reserva de snapshots predeterminada de ONTAP (generalmente 5 %):

```
docker volume create -d netapp --name my_vol --opt
snapshotPolicy=myPolicy
```

Opciones de volumen del software Element

Las opciones del software Element exponen las políticas de tamaño y calidad de servicio asociadas con el volumen. Cuando se crea el volumen, la política de calidad de servicio asociada con él se especifica mediante el `-o type=service_level` terminología

El primer paso para definir un nivel de servicio de calidad de servicio con el controlador de Element es crear al

menos un tipo y especificar las IOPS mínimas, máximas y de ráfaga asociadas con un nombre en el archivo de configuración.

Otras opciones de creación de volúmenes del software Element incluyen las siguientes:

Opción	Descripción
`size`	El tamaño del volumen, de manera predeterminada es 1 GIB o la entrada de configuración… "Valores predeterminados": {"tamaño": "5G"}.
`blocksize`	Utilice 512 o 4096, de forma predeterminada en 512 o en la entrada de configuración DefaultBlockSize.

Ejemplo

Consulte el siguiente archivo de configuración de ejemplo con definiciones de QoS:

```
{
    "...": "..."
    "Types": [
        {
            "Type": "Bronze",
            "Qos": {
                "minIOPS": 1000,
                "maxIOPS": 2000,
                "burstIOPS": 4000
            }
        },
        {
            "Type": "Silver",
            "Qos": {
                "minIOPS": 4000,
                "maxIOPS": 6000,
                "burstIOPS": 8000
            }
        },
        {
            "Type": "Gold",
            "Qos": {
                "minIOPS": 6000,
                "maxIOPS": 8000,
                "burstIOPS": 10000
            }
        }
    ]
}
```

En la configuración anterior, tenemos tres definiciones de normas: Bronce, plata y oro. Estos nombres son arbitrarios.

- Cree un volumen Gold de 10 GIB:

```
docker volume create -d solidfire --name sfGold -o type=Gold -o size=10G
```

- Cree un volumen Bronze de 100 GIB:

```
docker volume create -d solidfire --name sfBronze -o type=Bronze -o
size=100G
```

Recopilar registros

Puede recopilar registros para obtener ayuda con la solución de problemas. El método que se utiliza para recopilar los registros varía en función de cómo se ejecuta el complemento Docker.

Recopile registros para solucionar problemas

Pasos

1. Si ejecuta Astra Trident con el método de complemento gestionado recomendado (por ejemplo, mediante `docker plugin` comandos), visualizarlos de la siguiente manera:

```
docker plugin ls
ID                  NAME                DESCRIPTION
ENABLED
4fb97d2b956b        netapp:latest       nDVP - NetApp Docker Volume
Plugin   false
journalctl -u docker | grep 4fb97d2b956b
```

   El nivel de registro estándar debe permitirle diagnosticar la mayoría de los problemas. Si encuentra que eso no es suficiente, puede habilitar el registro de depuración.

2. Para habilitar el registro de depuración, instale el plugin con el registro de depuración activado:

```
docker plugin install netapp/trident-plugin:<version> --alias <alias>
debug=true
```

   O bien, active el registro de depuración cuando el plugin ya esté instalado:

```
docker plugin disable <plugin>
docker plugin set <plugin> debug=true
docker plugin enable <plugin>
```

3. Si ejecuta el binario en sí mismo en el host, los registros están disponibles en el host `/var/log/netappdvp` directorio. Para habilitar el registro de depuración, especifique `-debug` al ejecutar el plugin.

Sugerencias generales para la solución de problemas

- El problema más común en el que se ejecutan los nuevos usuarios es una configuración errónea que impide que el plugin se inicialice. Cuando esto sucede, es probable que vea un mensaje como este cuando intente instalar o activar el plugin:

  ```
  Error response from daemon: dial unix /run/docker/plugins/<id>/netapp.sock:
  connect: no such file or directory
  ```

  Esto significa que el plugin no se pudo iniciar. Afortunadamente, el complemento se ha creado con una completa capacidad de registro que le ayudará a diagnosticar la mayoría de los problemas que es probable que se encuentren.

- Si hay problemas con el montaje de un PV en un contenedor, asegúrese de que `rpcbind` está instalado y en ejecución. Utilice el administrador de paquetes necesario para el sistema operativo del host y compruebe si `rpcbind` está en ejecución. Puede comprobar el estado del servicio rpcbind ejecutando un `systemctl status rpcbind` o su equivalente.

Gestione varias instancias de Astra Trident

Se necesitan varias instancias de Trident cuando se desean que varias configuraciones de almacenamiento estén disponibles de forma simultánea. La clave para varias instancias es darles nombres diferentes mediante el `--alias` opción con el plugin en contenedor, o. `--volume-driver` Opción al crear una instancia de Trident en el host.

Pasos para el complemento gestionado de Docker (versión 1.13/17.03 o posterior)

1. Inicie la primera instancia que especifique un alias y un archivo de configuración.

```
docker plugin install --grant-all-permissions --alias silver
netapp/trident-plugin:21.07 config=silver.json
```

2. Inicie la segunda instancia, especificando un alias y un archivo de configuración distintos.

```
docker plugin install --grant-all-permissions --alias gold
netapp/trident-plugin:21.07 config=gold.json
```

3. Cree volúmenes que especifiquen el alias como el nombre del controlador.

Por ejemplo, para el volumen Gold:

```
docker volume create -d gold --name ntapGold
```

Por ejemplo, en el caso del volumen Silver:

```
docker volume create -d silver --name ntapSilver
```

Pasos para tradicional (versión 1.12 o anterior)

1. Inicie el plugin con una configuración NFS mediante un ID de controlador personalizado:

```
sudo trident --volume-driver=netapp-nas --config=/path/to/config
-nfs.json
```

2. Inicie el plugin con una configuración iSCSI mediante un ID de controlador personalizado:

```
sudo trident --volume-driver=netapp-san --config=/path/to/config
-iscsi.json
```

3. Aprovisione volúmenes Docker para cada instancia de controlador:

   Por ejemplo, para NFS:

```
docker volume create -d netapp-nas --name my_nfs_vol
```

   Por ejemplo, para iSCSI:

```
docker volume create -d netapp-san --name my_iscsi_vol
```

Opciones de configuración de almacenamiento

Consulte las opciones de configuración disponibles para las configuraciones de Astra Trident.

Opciones de configuración global

Estas opciones de configuración se aplican a todas las configuraciones de Astra Trident, independientemente de la plataforma de almacenamiento que se utilice.

Opción	Descripción	Ejemplo
`version`	Número de versión del archivo de configuración	`1`
`storageDriverName`	Nombre del controlador de almacenamiento	`ontap-nas`, `ontap-san`, `ontap-nas-economy`, `ontap-nas-flexgroup`, `solidfire-san`
`storagePrefix`	Prefijo opcional para los nombres de volúmenes. Valor predeterminado: `netappdvp_`.	`staging_`
`limitVolumeSize`	Restricción opcional de los tamaños de volumen. Por defecto: "" (no se aplica)	`10g`

No utilizar `storagePrefix` (Incluido el valor predeterminado) para los back-ends de Element. De forma predeterminada, la `solidfire-san` el controlador ignorará este ajuste y no utilizará un prefijo. Se recomienda utilizar un tenantID específico para la asignación de volúmenes de Docker o utilizar los datos de atributos que se rellenan con la versión de Docker, la información del controlador y el nombre sin formato de Docker en casos en los que se pueda haber utilizado cualquier comando de asignación de nombres.

Las opciones predeterminadas están disponibles para evitar tener que especificarlas en cada volumen que cree. La `size` la opción está disponible para todos los tipos de controladoras. Consulte la sección ONTAP Configuration para obtener un ejemplo de cómo establecer el tamaño de volumen predeterminado.

Opción	Descripción	Ejemplo
`size`	Tamaño predeterminado opcional para los nuevos volúmenes. Valor predeterminado: `1G`	`10G`

Configuración de ONTAP

Además de los valores de configuración global anteriores, al utilizar ONTAP, están disponibles las siguientes opciones de nivel superior.

Opción	Descripción	Ejemplo
`managementLIF`	Dirección IP de LIF de gestión de ONTAP. Es posible especificar un nombre de dominio completo (FQDN).	`10.0.0.1`

Opción	Descripción	Ejemplo
`dataLIF`	Dirección IP de LIF de protocolo. **Controladores NAS 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. **Controladores SAN ONTAP**: No se especifica para iSCSI. Usos de Astra Trident "Asignación de LUN selectiva de ONTAP" Para descubrir los LIF iSCSI necesarios para establecer una sesión de ruta múltiple. Se genera una advertencia if `dataLIF` se define explícitamente.	`10.0.0.2`
`svm`	Utilizar máquinas virtuales de almacenamiento (necesaria, si LIF de gestión es una LIF de clúster)	`svm_nfs`
`username`	Nombre de usuario para conectarse al dispositivo de almacenamiento	`vsadmin`
`password`	Contraseña para conectarse al dispositivo de almacenamiento	`secret`
`aggregate`	Agregado para el aprovisionamiento (opcional; si se establece, se debe asignar a la SVM). Para la `ontap-nas-flexgroup` controlador, esta opción se ignora. Todos los agregados asignados al SVM se utilizan para aprovisionar un volumen de FlexGroup.	`aggr1`
`limitAggregateUsage`	Opcional, fallo en el aprovisionamiento si el uso supera este porcentaje	`75%`

Opción	Descripción	Ejemplo
`nfsMountOptions`	Control detallado de las opciones de montaje NFS; valor predeterminado es “-o nfsvers=3”. **Disponible sólo para `ontap-nas` y.. `ontap-nas-economy` controladores**. "Consulte la información de configuración del host NFS aquí".	`-o nfsvers=4`
`igroupName`	Astra Trident crea y gestiona por nodo `igroups` como `netappdvp`. Este valor no se puede cambiar ni omitir. **Disponible sólo para `ontap-san` conductor**.	`netappdvp`
`limitVolumeSize`	Tamaño máximo del volumen que se puede volver a realizar la consulta y tamaño del volumen principal de qtree. **Para `ontap-nas-economy` Controlador, esta opción limita además el tamaño de los FlexVols que crea**.	`300g`
`qtreesPerFlexvol`	El número máximo de qtrees por FlexVol debe estar comprendido entre [50, 300], y el valor predeterminado es 200. **Para `ontap-nas-economy` Controlador, esta opción permite personalizar el número máximo de qtrees por FlexVol**.	`300`

Las opciones predeterminadas están disponibles para evitar tener que especificarlas en cada volumen que cree:

Opción	Descripción	Ejemplo
`spaceReserve`	Modo de reserva de espacio; `none` (con thin provisioning) o `volume` (grueso)	`none`
`snapshotPolicy`	La política de Snapshot que se va a utilizar, el valor predeterminado es `none`	`none`

Opción	Descripción	Ejemplo
`snapshotReserve`	Porcentaje de reserva de Snapshot, el valor predeterminado es « » para aceptar el valor predeterminado de ONTAP	`10`
`splitOnClone`	Divida un clon de su elemento principal tras su creación (el valor predeterminado es `false`	`false`
`encryption`	Permite el cifrado de volúmenes de NetApp (NVE) en el volumen nuevo; 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".	verdadero
`unixPermissions`	La opción de NAS para volúmenes NFS aprovisionados, de forma predeterminada a. `777`	`777`
`snapshotDir`	Opción NAS para acceder a `.snapshot` directorio, el valor predeterminado es `false`	`true`
`exportPolicy`	La opción de NAS para la política de exportación de NFS que va a utilizar, de forma predeterminada a. `default`	`default`
`securityStyle`	Opción NAS para acceder al volumen NFS aprovisionado. Compatibilidad con NFS `mixed` y.. `unix` estilos de seguridad. El valor predeterminado es `unix`.	`unix`

Opción	Descripción	Ejemplo
`fileSystemType`	Opción SAN para seleccionar el tipo de sistema de archivos, de forma predeterminada a. `ext4`	`xfs`
`tieringPolicy`	Política de organización en niveles que se va a utilizar, el valor predeterminado es `none`; `snapshot-only` Para configuraciones anteriores a ONTAP 9,5 SVM-DR	`none`

Opciones de escala

La `ontap-nas` y.. `ontap-san` Los controladores crean un ONTAP FlexVol para cada volumen Docker. ONTAP admite un máximo de 1000 FlexVols por nodo del clúster con un máximo de 12,000 FlexVols. Si los requisitos de su volumen de Docker se ajustan a esa limitación, el `ontap-nas` El controlador es la solución NAS preferida debido a las características adicionales que ofrece FlexVols, como las copias Snapshot granulares en Docker-volume y el clonado.

Si necesita más volúmenes de Docker de los que pueden alojar los límites de FlexVol, seleccione la `ontap-nas-economy` o la `ontap-san-economy` controlador.

La `ontap-nas-economy` El controlador crea volúmenes Docker como ONTAP Qtrees dentro de un pool de FlexVols gestionados automáticamente. Qtrees ofrece un escalado mucho mayor, hasta 100,000 por nodo de clúster y 2,400,000 por clúster, a expensas de algunas funciones. La `ontap-nas-economy` El controlador no admite el clonado o copias Snapshot granulares en volúmenes de Docker.

> La `ontap-nas-economy` Actualmente, Docker Swarm no admite el controlador, porque Swarm no orqueste la creación de volúmenes entre varios nodos.

La `ontap-san-economy` El controlador crea volúmenes Docker como LUN de ONTAP en un pool compartido de FlexVols gestionados automáticamente. De este modo, cada FlexVol no está restringido a solo un LUN y ofrece una mejor escalabilidad para cargas DE trabajo SAN. Según la cabina de almacenamiento, ONTAP admite hasta 16384 LUN por clúster. Dado que los volúmenes son LUN en el interior, este controlador admite copias Snapshot granulares en Docker y clonado de volúmenes.

Elija la `ontap-nas-flexgroup` controlador para aumentar el paralelismo con un único volumen que puede crecer hasta llegar a la gama de petabytes con miles de millones de archivos. Algunos casos de uso ideales para FlexGroups incluyen IA/ML/DL, Big Data y análisis, creación de software, streaming, repositorios de archivos, etc. Trident usa todos los agregados asignados a una SVM cuando se aprovisiona un volumen de FlexGroup. La compatibilidad con FlexGroup en Trident también tiene las siguientes consideraciones:

- Requiere ONTAP versión 9.2 o posterior.
- En el momento en el que se ha redactado este documento, FlexGroups solo admite NFS v3.
- Se recomienda habilitar los identificadores de NFSv3 de 64 bits para la SVM.
- El tamaño mínimo recomendado de FlexGroup es de 100 GB.
- No se admite la clonado en volúmenes de FlexGroup.

Para obtener información acerca de FlexGroups y las cargas de trabajo adecuadas para FlexGroups, consulte "Prácticas recomendadas y guía de implementación de los volúmenes FlexGroup de NetApp".

Para obtener funciones avanzadas y obtener un enorme escalado en el mismo entorno, puede ejecutar varias instancias del complemento para volúmenes de Docker, utilizando una `ontap-nas` y otro uso `ontap-nas-economy`.

Archivos de configuración de ONTAP de ejemplo

Ejemplo de NFS para `ontap-nas` controlador

```
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "username": "vsadmin",
    "password": "password",
    "aggregate": "aggr1",
    "defaults": {
      "size": "10G",
      "spaceReserve": "none",
      "exportPolicy": "default"
    }
}
```

Ejemplo de NFS para `ontap-nas-flexgroup` controlador

```
{
    "version": 1,
    "storageDriverName": "ontap-nas-flexgroup",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "username": "vsadmin",
    "password": "password",
    "defaults": {
      "size": "100G",
      "spaceReserve": "none",
      "exportPolicy": "default"
    }
}
```

Ejemplo de NFS para `ontap-nas-economy` controlador

```
{
    "version": 1,
    "storageDriverName": "ontap-nas-economy",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "username": "vsadmin",
    "password": "password",
    "aggregate": "aggr1"
}
```

Ejemplo de iSCSI para `ontap-san` controlador

```
{
    "version": 1,
    "storageDriverName": "ontap-san",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.3",
    "svm": "svm_iscsi",
    "username": "vsadmin",
    "password": "password",
    "aggregate": "aggr1",
    "igroupName": "netappdvp"
}
```

Ejemplo de NFS para `ontap-san-economy` controlador

```
{
    "version": 1,
    "storageDriverName": "ontap-san-economy",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.3",
    "svm": "svm_iscsi_eco",
    "username": "vsadmin",
    "password": "password",
    "aggregate": "aggr1",
    "igroupName": "netappdvp"
}
```

Configuración del software Element

Además de los valores de configuración global, cuando se utiliza el software Element (HCI/SolidFire de NetApp), existen estas opciones disponibles.

Opción	Descripción	Ejemplo
`Endpoint`	https://<login>:<password>@<mvip>/json-rpc/<element-version>	https://admin:admin@192.168.160.3/json-rpc/8.0
`SVIP`	Puerto y dirección IP de iSCSI	10.0.0.7:3260
`TenantName`	Debe utilizar el inquilino SolidFireF (creado si no encontrado)	`docker`
`InitiatorIFace`	Especifique la interfaz cuando restrinja el tráfico de iSCSI a una interfaz no predeterminada	`default`
`Types`	Especificaciones de calidad de servicio	Vea el ejemplo siguiente
`LegacyNamePrefix`	Prefijo para instalaciones actualizadas de Trident. Si utilizó una versión de Trident anterior a la 1.3.2 y realiza una actualización con volúmenes existentes, deberá establecer este valor para acceder a los volúmenes antiguos que se asignaron mediante el método volume-name.	`netappdvp-`

La `solidfire-san` El controlador no es compatible con Docker Swarm.

Ejemplo del archivo de configuración del software Element

```
{
    "version": 1,
    "storageDriverName": "solidfire-san",
    "Endpoint": "https://admin:admin@192.168.160.3/json-rpc/8.0",
    "SVIP": "10.0.0.7:3260",
    "TenantName": "docker",
    "InitiatorIFace": "default",
    "Types": [
        {
            "Type": "Bronze",
            "Qos": {
                "minIOPS": 1000,
                "maxIOPS": 2000,
                "burstIOPS": 4000
            }
        },
        {
            "Type": "Silver",
            "Qos": {
                "minIOPS": 4000,
                "maxIOPS": 6000,
                "burstIOPS": 8000
            }
        },
        {
            "Type": "Gold",
            "Qos": {
                "minIOPS": 6000,
                "maxIOPS": 8000,
                "burstIOPS": 10000
            }
        }
    ]
}
```

Problemas y limitaciones conocidos

Encuentre información sobre problemas y limitaciones conocidos al usar Astra Trident con Docker.

Si se actualiza el complemento Trident Docker Volume Plugin a 20.10 y versiones posteriores, se produce un error de actualización sin dicho archivo o directorio.

Solución alternativa

1. Desactivar el plugin.

```
docker plugin disable -f netapp:latest
```

2. Quitar el plugin.

```
docker plugin rm -f netapp:latest
```

3. Vuelva a instalar el plugin proporcionando el extra `config` parámetro.

```
docker plugin install netapp/trident-plugin:20.10 --alias netapp --grant
-all-permissions config=config.json
```

Los nombres de volumen deben tener una longitud mínima de 2 caracteres.

Esta es una limitación de cliente de Docker. El cliente interpretará un nombre de carácter único como una ruta de Windows. "Consulte el error 25773".

Docker Swarm tiene determinados comportamientos que impiden a Astra Trident admitirla con cada combinación de controladores y almacenamiento.

- Docker Swarm actualmente utiliza el nombre del volumen en lugar del ID de volumen como su identificador de volumen único.
- Las solicitudes de volúmenes se envían simultáneamente a cada nodo de un clúster Swarm.
- Los complementos de volumen (incluida Astra Trident) deben ejecutarse por separado en cada nodo de un clúster Swarm.
  Por la forma en que funciona ONTAP y cómo `ontap-nas` y.. `ontap-san` los conductores funcionan, son los únicos que se pueden operar dentro de estas limitaciones.

El resto de los conductores están sujetos a problemas como las condiciones de la carrera que pueden dar lugar a la creación de un gran número de volúmenes para una sola solicitud sin un claro “ganador”; por ejemplo, Element tiene una función que permite que los volúmenes tengan el mismo nombre pero ID diferentes.

NetApp ha proporcionado comentarios al equipo de Docker, pero no tiene ningún indicio de recurso futuro.

Si se está provisionando un FlexGroup, ONTAP no aprovisiona una segunda FlexGroup si el segundo FlexGroup tiene uno o más agregados en común con el FlexGroup que se está aprovisionando.

Preguntas frecuentes

Encuentre respuestas a las preguntas frecuentes sobre la instalación, configuración, actualización y solución de problemas de Astra Trident.

Preguntas generales

¿Con qué frecuencia se lanza Astra Trident?

Astra Trident se lanza cada tres meses: Enero, abril, julio y octubre. Este es un mes después de un lanzamiento de Kubernetes.

¿Es compatible Astra Trident con todas las funciones que se comercializan en una versión concreta de Kubernetes?

Astra Trident no suele admitir funciones alfa en Kubernetes. Trident puede ser compatible con las funciones beta en las dos versiones de Trident que se indican a continuación de la versión beta de Kubernetes.

¿Astra Trident tiene alguna dependencia de otros productos de NetApp en cuanto a su funcionamiento?

Astra Trident no tiene dependencia de otros productos de software de NetApp y funciona como una aplicación independiente. Sin embargo, debe disponer de un dispositivo de almacenamiento de entorno de administración de NetApp.

¿Cómo puedo obtener detalles completos de la configuración de Astra Trident?

Utilice la `tridentctl get` Comando para obtener más información acerca de la configuración de Astra Trident.

¿Puedo obtener mediciones sobre cómo aprovisiona Astra Trident el almacenamiento?

Sí. Trident 20.01 presenta extremos que se pueden utilizar para recopilar información sobre el funcionamiento de Astra Trident, como el número de back-ends gestionados, el número de volúmenes aprovisionados, bytes consumidos, etc. También puede usar Cloud Insights para la supervisión y el análisis.

¿Cambia la experiencia del usuario al utilizar Astra Trident como aprovisionador CSI?

No No hay cambios en cuanto a la experiencia del usuario y las funcionalidades. El nombre de aprovisionador usado es `csi.trident.netapp.io`. Se recomienda este método de instalación de Astra Trident si desea utilizar todas las funciones nuevas que proporcionan las versiones actuales y futuras.

Instale y use Astra Trident en un clúster de Kubernetes

¿De qué versiones son compatibles `etcd`?

Astra Trident ya no necesita una `etcd`. Utiliza CRD para mantener el estado.

¿Admite Astra Trident una instalación sin conexión desde un registro privado?

Sí, Astra Trident se puede instalar sin conexión. Consulte "aquí".

¿Puedo instalar Astra Trident de forma remota?

Sí. A partir de la versión Trident 18.10 de Astra se admiten la capacidad de instalación remota desde cualquier máquina que tenga `kubectl` acceso al clúster. Después `kubectl` el acceso se verifica (por ejemplo, inicie un `kubectl get nodes` comando desde la máquina remota para verificar), siga las instrucciones de instalación.

¿Puedo configurar la alta disponibilidad con Astra Trident?

Astra Trident se instala como una implementación de Kubernetes (ReplicaSet) con una instancia, por lo que ha incorporado funciones de alta disponibilidad. No debe aumentar el número de réplicas en la implementación. Si se pierde el nodo en el que se ha instalado Astra Trident o no se puede acceder al pod, Kubernetes vuelve a poner en marcha automáticamente el pod en un nodo correcto del clúster. Astra Trident solo es plano de control, por lo que los pods montados actualmente no se ven afectados si se vuelve a poner en marcha Astra Trident.

¿Necesita Astra Trident acceder al espacio de nombres del sistema kube?

Astra Trident lee desde el servidor de API de Kubernetes para determinar cuándo las aplicaciones solicitan nuevos RVP, de modo que necesita acceso al sistema kube.

¿Cuáles son las funciones y los privilegios que utiliza Astra Trident?

El instalador de Trident crea un archivo Kubernetes ClusterRole que tiene acceso específico a los recursos Persistent Volume, Persistent Claim, StorageClass y Secret del clúster de Kubernetes. Consulte "aquí".

¿Puedo generar de forma local los archivos de manifiesto exactos que utiliza Astra Trident para la instalación?

Si es necesario, puede generar y modificar localmente los archivos de manifiesto exactos que Astra Trident utiliza para la instalación. Consulte "aquí".

¿Puedo compartir la misma SVM back-end de ONTAP con dos instancias separadas de Astra Trident para dos clústeres de Kubernetes independientes?

Aunque no se aconseja, puede utilizar la misma SVM back-end para dos instancias de Astra Trident. Especifique un nombre de volumen único para cada instancia durante la instalación o especifique un valor único `StoragePrefix` en la `setup/backend.json` archivo. De este modo, se garantiza que no se utiliza el mismo FlexVol para ambas instancias.

¿Es posible instalar Astra Trident en ContainerLinux (anteriormente CoreOS)?

Astra Trident es simplemente un pod de Kubernetes y se puede instalar dondequiera que se ejecute Kubernetes.

¿Puedo usar Astra Trident con Cloud Volumes ONTAP de NetApp?

Sí, Astra Trident es compatible con AWS, Google Cloud y Azure.

¿Funciona Astra Trident con Cloud Volumes Services?

Sí, Astra Trident es compatible con el servicio Azure NetApp Files en Azure y con Cloud Volumes Service en GCP.

Solución de problemas y soporte técnico

¿Es compatible NetApp con Astra Trident?

Aunque Astra Trident es un código abierto y se proporciona de forma gratuita, NetApp ofrece total compatibilidad con ella, siempre y cuando su entorno de administración de NetApp sea compatible.

¿Cómo levanto un caso de soporte?

Para levantar un caso de soporte, realice una de las siguientes acciones:

1. Póngase en contacto con su responsable técnico de soporte y obtenga ayuda para emitir una incidencia.
2. Levante un caso de soporte con el contacto "Soporte de NetApp".

¿Cómo se genera un bundle del registro de soporte?

Puede crear un paquete de soporte en ejecución `tridentctl logs -a`. Además de los registros capturados en el paquete, capture el registro kubelet para diagnosticar los problemas de montaje en el lado de Kubernetes. Las instrucciones para obtener el registro de Kubelet varían en función de cómo se instale Kubernetes.

¿Qué debo hacer si necesito solicitar una nueva función?

Cree un problema en "Astra Trident Github" Y mencionar **RFE** en el tema y descripción del tema.

¿Dónde puedo elevar un defecto?

Cree un problema en "Astra Trident Github". Asegúrese de incluir toda la información y registros necesarios relacionados con el problema.

¿Qué sucede si tengo una pregunta rápida sobre Astra Trident sobre la que necesito aclaraciones? ¿Hay una comunidad o un foro?

Si tiene alguna pregunta, problema o solicitud, póngase en contacto con nosotros a través de nuestra Astra "Canal de discordia" O GitHub.

La contraseña de mi sistema de almacenamiento ha cambiado y Astra Trident ya no funciona, ¿cómo puedo recuperar?

Actualice la contraseña del back-end con `tridentctl update backend myBackend -f </path/to_new_backend.json> -n trident`. Sustituya `myBackend` en el ejemplo con su nombre de fondo, y. `/path/to_new_backend.json` con la ruta a la correcta `backend.json` archivo.

Astra Trident no encuentra mi nodo Kubernetes. ¿Cómo se soluciona esto?

Hay dos supuestos posibles por los que Astra Trident no puede encontrar un nodo de Kubernetes. Puede

deberse a un problema de red en Kubernetes o a un problema con el DNS. El conjunto de nodos de Trident que se ejecuta en cada nodo de Kubernetes debe poder comunicarse con la controladora Trident para registrar el nodo en Trident. Si se produjeron cambios en la red después de instalar Astra Trident, solo se produce este problema con los nodos de Kubernetes nuevos que se añaden al clúster.

Si el pod de Trident se destruye, ¿perderé los datos?

No se perderán los datos si el pod de Trident se destruye. Los metadatos de Trident se almacenan en objetos CRD. Todos los VP aprovisionados por Trident funcionarán normalmente.

Actualice Astra Trident

¿Puedo actualizar directamente desde una versión anterior a una versión nueva (omitiendo algunas versiones)?

NetApp admite la actualización de Astra Trident de una versión principal a la siguiente inmediata mayor. Puede actualizar de la versión 18.xx a la 19.xx, 19.xx a la 20.xx, etc. Debe realizar pruebas de actualización en un laboratorio antes de la implementación de producción.

¿Es posible degradar Trident a una versión anterior?

Hay una serie de factores que se deben evaluar si se desea cambiar a una versión inferior. Consulte "la sección de la degradación".

Gestione back-ends y volúmenes

¿Debo definir tanto las LIF de gestión como las LIF de datos en un archivo de definición del back-end de ONTAP?

El LIF de gestión es obligatorio. Data LIF varía:

- SAN de ONTAP: No se especifica para iSCSI. Usos de Astra Trident "Asignación de LUN selectiva de ONTAP" Para descubrir los LIF iSCSI necesarios para establecer una sesión de ruta múltiple. Se genera una advertencia if `dataLIF` se define explícitamente. Consulte "Opciones y ejemplos de configuración DE SAN ONTAP" para obtener más detalles.
- 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. Consulte "Opciones y ejemplos de configuración NAS de ONTAP" para obtener más detalles

¿Puede Astra Trident configurar CHAP para los back-ends de ONTAP?

Sí. A partir de 20.04, Astra Trident admite CHAP bidireccional para los back-ends de ONTAP. Esto requiere configuración `useCHAP=true` en la configuración de back-end.

¿Cómo puedo gestionar las políticas de exportación con Astra Trident?

Astra Trident puede crear y gestionar dinámicamente políticas de exportación a partir de la versión 20.04. Esto permite al administrador de almacenamiento proporcionar uno o varios bloques CIDR en la configuración back-end y hacer que Trident añada IP de nodo dentro de estos rangos a una política de exportación que

cree. De esta forma, Astra Trident gestiona automáticamente la adición y eliminación de reglas para nodos con IP en los CIDR dados. Esta función requiere CSI Trident.

¿Podemos especificar un puerto en DataLIF?

Astra Trident 19.01 y versiones posteriores admiten la especificación de un puerto en DataLIF. Configúrelo en la `backend.json` archivo como "`managementLIF": <ip address>:<port>`". Por ejemplo, si la dirección IP del LIF de gestión es 192.0.2.1 y el puerto es 1000, configure "`managementLIF": "192.0.2.1:1000`".

¿Las direcciones IPv6 se pueden utilizar para los LIF de gestión y datos?

Astra Trident admite la definición de direcciones IPv6 para:

- `managementLIF` y.. `dataLIF` Para back-ends NAS de ONTAP.
- `managementLIF` Para back-ends DE SAN de ONTAP. No puede especificar `dataLIF` En un entorno de administración SAN de ONTAP.

Astra Trident debe instalarse mediante `--use-ipv6` Le permite que funcione a través de IPv6.

¿Se puede actualizar la LIF de gestión en el back-end?

Sí, es posible actualizar la LIF de gestión del back-end mediante el `tridentctl update backend` comando.

¿Es posible actualizar la LIF de datos en el back-end?

Puede actualizar el LIF de datos en `ontap-nas` y.. `ontap-nas-economy` solamente.

¿Puedo crear varios back-ends en Astra Trident para Kubernetes?

Astra Trident puede admitir muchos back-ends simultáneamente, ya sea con el mismo controlador o con distintos controladores.

¿Cómo almacena Astra Trident las credenciales de back-end?

Astra Trident almacena las credenciales de back-end como secretos de Kubernetes.

¿Cómo selecciona Astra Trident un back-end específico?

Si los atributos back-end no se pueden utilizar para seleccionar automáticamente los grupos adecuados para una clase, el `storagePools` y.. `additionalStoragePools` los parámetros se usan para seleccionar un conjunto específico de pools.

¿Cómo puedo asegurarme de que Astra Trident no se provisione desde un back-end específico?

La `excludeStoragePools` El parámetro se utiliza para filtrar el conjunto de pools que utilizará Astra Trident para el aprovisionamiento y eliminará cualquier pool que coincida.

Si hay varios back-ends del mismo tipo, ¿cómo selecciona Astra Trident qué back-end utilizar?

Si hay varios back-ends configurados del mismo tipo, Astra Trident selecciona el back-end adecuado en función de los parámetros presentes en `StorageClass` y.. `PersistentVolumeClaim`. Por ejemplo, si hay varios back-ends de unidades ontap-nas, Astra Trident intenta coincidir con los parámetros en `StorageClass` y.. `PersistentVolumeClaim` combine y haga coincidir un back-end que pueda cumplir los requisitos enumerados en `StorageClass` y.. `PersistentVolumeClaim`. Si hay varios back-ends que coincidan con la solicitud, Astra Trident selecciona de uno de ellos al azar.

¿Admite Astra Trident CHAP bidireccional con Element/SolidFire?

Sí.

¿Cómo pone en marcha Astra Trident Qtrees en un volumen de ONTAP? ¿Cuántos qtrees pueden ponerse en marcha en un único volumen?

La `ontap-nas-economy` El controlador crea hasta 200 qtrees en la misma FlexVol (que se puede configurar entre 50 y 300), 100,000 qtrees por nodo del clúster y 2,4 MILLONES por clúster. Al introducir un nuevo `PersistentVolumeClaim` Este servicio es prestado por el conductor económico y busca ver si ya existe una FlexVol que pueda dar servicio al nuevo qtree. Si no existe la FlexVol que pueda dar servicio al qtree, se crea una nueva FlexVol.

¿Cómo puedo establecer los permisos de Unix para los volúmenes aprovisionados en NAS de ONTAP?

Puede establecer permisos Unix en el volumen aprovisionado por Astra Trident mediante la configuración de un parámetro en el archivo de definición del back-end.

¿Cómo puedo configurar un conjunto explícito de opciones de montaje NFS de ONTAP al aprovisionar un volumen?

De forma predeterminada, Astra Trident no establece las opciones de montaje en ningún valor con Kubernetes. Para especificar las opciones de montaje en la clase de almacenamiento Kubernetes, siga el ejemplo dado "aquí".

¿Cómo se configuran los volúmenes aprovisionados en una política de exportación específica?

Para permitir el acceso de hosts adecuados a un volumen, use el `exportPolicy` parámetro configurado en el archivo de definición de backend.

¿Cómo se configura el cifrado de volúmenes mediante Astra Trident con ONTAP?

Puede establecer el cifrado en el volumen aprovisionado por Trident mediante el parámetro Encryption del archivo de definición del back-end. Para obtener más información, consulte: "Cómo funciona Astra Trident con NVE y NAE"

¿Cuál es la mejor forma de implementar la calidad de servicio para ONTAP a través de Astra Trident?

Uso `StorageClasses` Para implementar QoS en ONTAP.

¿Cómo se especifica thin provisioning o thick provisioning a través de Astra Trident?

Los controladores ONTAP admiten thin provisioning o thick. Los controladores ONTAP, de manera predeterminada, son thin provisioning. Si se desea un aprovisionamiento grueso, debe configurar el archivo de definición back-end o el `StorageClass`. Si se configuran ambas, `StorageClass` tiene prioridad. Configure lo siguiente para ONTAP:

1. Encendido `StorageClass`, establezca la `provisioningType` atributo como grueso.
2. En el archivo de definición del back-end, habilite los volúmenes gruesos mediante la configuración `backend spaceReserve parameter` como volumen.

¿Cómo se asegura de que los volúmenes que se están utilizando no se eliminen incluso si se elimina accidentalmente la RVP?

La protección contra RVP se habilita automáticamente en Kubernetes a partir de la versión 1.10.

¿Puedo aumentar las RVP de NFS creadas por Astra Trident?

Sí. Puede ampliar una RVP creada por Astra Trident. Tenga en cuenta que el crecimiento automático del volumen es una función de ONTAP que no se aplica a Trident.

Si tengo un volumen creado fuera de Astra Trident, ¿puedo importarlo a Astra Trident?

A partir de la versión 19.04, se puede usar la función de importación de volúmenes para llevar los volúmenes a Kubernetes.

¿Puedo importar un volumen mientras está en SnapMirror Data Protection (DP) o en modo sin conexión?

Se produce un error en la importación del volumen si el volumen externo está en modo DP o sin conexión. Recibe el siguiente mensaje de error:

```
Error: could not import volume: volume import failed to get size of
volume: volume <name> was not found (400 Bad Request) command terminated
with exit code 1.
Make sure to remove the DP mode or put the volume online before importing
the volume.
```

¿Puedo ampliar las RVP iSCSI creadas por Astra Trident?

Trident 19.10 admite la ampliación de VP iSCSI mediante el aprovisionador CSI.

¿Cómo se traduce la cuota de recursos en un clúster de NetApp?

La cuota de recursos de almacenamiento de Kubernetes debe funcionar siempre que el almacenamiento de NetApp tenga capacidad. Cuando el almacenamiento de NetApp no puede respetar la configuración de cuota de Kubernetes por falta de capacidad, Astra Trident intenta aprovisionar, pero con errores.

¿Puedo crear copias Snapshot de volumen con Astra Trident?

Sí. Astra Trident admite la creación de snapshots de volúmenes bajo demanda y volúmenes persistentes a partir de snapshots. Para crear VP a partir de instantáneas, asegúrese de que `VolumeSnapshotDataSource` se ha habilitado la puerta de operaciones.

¿Cuáles son los controladores compatibles con las instantáneas de volumen de Astra Trident?

A partir de ahora, ofrecemos soporte de copias Snapshot bajo demanda para nuestro `ontap-nas`, `ontap-nas-flexgroup`, `ontap-san`, `ontap-san-economy`, `solidfire-san`, `gcp-cvs`, y. `azure-netapp-files` controladores de back-end.

¿Cómo puedo realizar un backup con Snapshot de un volumen aprovisionado por Astra Trident con ONTAP?

Este está disponible en `ontap-nas`, `ontap-san`, y. `ontap-nas-flexgroup` de windows También puede especificar un `snapshotPolicy` para la `ontap-san-economy` Controlador en el nivel FlexVol.

También está disponible en la `ontap-nas-economy` Pero con la granularidad del nivel de FlexVol, no con la granularidad del qtree. Para permitir la capacidad de realizar copias Snapshot de volúmenes aprovisionados por Astra Trident, establezca la opción de parámetro backend `snapshotPolicy` A la política de Snapshot deseada según se define en el back-end de ONTAP. Astra Trident no conoce las instantáneas que tome la controladora de almacenamiento.

¿Puedo configurar un porcentaje de reserva de Snapshot para un volumen aprovisionado a través de Astra Trident?

Sí, puede reservar un porcentaje específico de espacio en disco para almacenar las copias Snapshot mediante Astra Trident estableciendo el `snapshotReserve` atributo en el archivo de definición de backend. Si se configuró `snapshotPolicy` y.. `snapshotReserve` en el archivo de definición de backend, el porcentaje de reserva de instantánea se establece según la `snapshotReserve` porcentaje mencionado en el archivo back-end. Si la `snapshotReserve` No se menciona ningún número de porcentaje. ONTAP toma el porcentaje de reserva de snapshots de forma predeterminada en 5. Si la `snapshotPolicy` la opción se establece en none, el porcentaje de reserva de snapshot se establece en 0.

¿Puedo acceder directamente al directorio de snapshot del volumen y copiar los archivos?

Sí, es posible acceder al directorio Snapshot en el volumen aprovisionado por Trident estableciendo el `snapshotDir` parámetro en el archivo de definición de backend.

¿Puedo configurar SnapMirror para volúmenes a través de Astra Trident?

Actualmente, SnapMirror debe configurarse externamente mediante la CLI de ONTAP o System Manager de OnCommand.

¿Cómo se restauran los volúmenes persistentes en una snapshot de ONTAP específica?

Para restaurar un volumen a una copia de Snapshot de ONTAP, realice los siguientes pasos:

1. Desactive el pod de la aplicación que utiliza el volumen persistente.
2. Revertir a la snapshot necesaria mediante la interfaz de línea de comandos de ONTAP o System Manager de OnCommand.
3. Reinicie el pod de la aplicación.

¿Trident puede aprovisionar volúmenes en SVM que tengan configurado un reflejo de carga compartida?

Se pueden crear reflejos de uso compartido de carga para volúmenes raíz de los SVM que sirven datos mediante NFS. ONTAP actualiza automáticamente los reflejos de uso compartido de carga para los volúmenes creados por Trident. Esto puede provocar retrasos en el montaje de volúmenes. Cuando se crean varios volúmenes mediante Trident, el aprovisionamiento de un volumen depende de que ONTAP actualice el reflejo de uso compartido de carga.

¿Cómo puedo separar el uso de la clase de almacenamiento para cada cliente/cliente?

Kubernetes no permite las clases de almacenamiento en espacios de nombres. Sin embargo, puede utilizar Kubernetes para limitar el uso de una clase de almacenamiento específica por espacio de nombres mediante las cuotas de recursos de almacenamiento, que se encuentran por espacio de nombres. Para denegar el acceso a un espacio de nombres específico a un almacenamiento específico, establezca la cuota de recursos en 0 para esa clase de almacenamiento.

Soporte técnico

Astra Trident es un proyecto de NetApp con soporte oficial. Puede ponerse en contacto con NetApp mediante cualquiera de los mecanismos estándar y obtener el soporte de nivel empresarial que necesita.

Además, nuestra Astra cuenta con una vibrante comunidad pública de usuarios de contenedores (incluidos los desarrolladores de Astra Trident) "Canal de discordia". Este es un gran lugar para hacer preguntas generales sobre el proyecto y discutir temas relacionados con compañeros de ideas afines.

Resolución de problemas

Utilice los punteros que se proporcionan aquí para solucionar problemas que puedan surgir durante la instalación y el uso de Astra Trident.

Si necesita ayuda con Astra Trident, cree un paquete de soporte con `tridentctl logs -a -n trident` y envíelo a. `NetApp Support <Getting Help>`.

Para obtener una lista completa de los artículos de solución de problemas, consulte "Base de conocimientos de NetApp (se requiere inicio de sesión)". También puede encontrar información sobre la solución de problemas relacionados con Astra "aquí".

Resolución de problemas generales

- Si el pod de Trident no sale correctamente (por ejemplo, cuando el pod de Trident se encuentra atascado en el `ContainerCreating` fase con menos de dos contenedores listos para usar) en ejecución `kubectl -n trident describe deployment trident` y.. `kubectl -n trident describe pod trident-**` puede proporcionar información adicional. Obtención de registros de kubelet (por ejemplo, mediante `journalctl -xeu kubelet`) también puede ser útil.
- Si no hay suficiente información en los registros de Trident, puede intentar habilitar el modo de depuración para Trident aprobando el `-d` marque el parámetro install en función de su opción de instalación.

  A continuación, confirme que la depuración se ha configurado mediante `./tridentctl logs -n trident` y buscando `level=debug msg` en el registro.

  Instalado con el operador

  ```
  kubectl patch torc trident -n <namespace> --type=merge -p
  '{"spec":{"debug":true}}'
  ```

  Así se reiniciarán todos los pods de Trident, que pueden tardar varios segundos. Puede comprobarlo observando la columna "ANTIGÜEDAD" en la salida de `kubectl get pod -n trident`.

  Para uso de Astra Trident 20.07 y 20.10 `tprov` en lugar de `torc`.

  Instalado con Helm

  ```
  helm upgrade <name> trident-operator-21.07.1-custom.tgz --set
  tridentDebug=true`
  ```

  Instalado con trimentctl

  ```
  ./tridentctl uninstall -n trident
  ./tridentctl install -d -n trident
  ```

- También puede obtener registros de depuración para cada back-end incluyendo `debugTraceFlags` en

su definición de backend. Por ejemplo, incluya `debugTraceFlags: {"api":true, "method":true,}` Para obtener llamadas API y recorridos de métodos en los registros de Trident. Los back-ends existentes pueden tener `debugTraceFlags` configurado con una `tridentctl backend update`.

- Cuando utilice RedHat CoreOS, asegúrese de ello `iscsid` está activado en los nodos de trabajo e iniciado de forma predeterminada. Esto se puede hacer usando OpenShift MachineConfigs o modificando las plantillas de ignición.
- Un problema común que se puede encontrar cuando se usa Trident con "Azure NetApp Files" es cuando el inquilino y los secretos de cliente provienen de un registro de aplicación con permisos insuficientes. Si desea ver una lista completa de los requisitos de Trident, consulte "Azure NetApp Files" configuración.
- Si hay problemas con el montaje de un PV en un contenedor, asegúrese de que `rpcbind` está instalado y en ejecución. Utilice el administrador de paquetes necesario para el sistema operativo del host y compruebe si `rpcbind` está en ejecución. Puede comprobar el estado de `rpcbind` ejecutar un `systemctl status rpcbind` o su equivalente.
- Si un back-end de Trident informa que está en la `failed` estado a pesar de haber trabajado antes, es probable que sea por el cambio de las credenciales de SVM/administrador asociadas con el back-end. Actualización de la información del back-end mediante `tridentctl update backend` O rebotando el Pod de Trident se corregirá este problema.
- Si va a actualizar el clúster de Kubernetes y/o Trident para utilizar copias Snapshot de volumen beta, asegúrese de que se hayan eliminado completamente todas las copias de Snapshot alfa del CRS existentes. A continuación, puede utilizar la `tridentctl obliviate alpha-snapshot-crd` Comando para eliminar CRD de instantánea alfa. Consulte "este blog" comprender los pasos que implica la migración de instantáneas alfa.
- Si se producen problemas de permisos al instalar Trident con Docker como el tiempo de ejecución de contenedores, intente instalar Trident con el `--in cluster=false` bandera. Esto no utilizará un módulo de instalación y evitará problemas de permisos vistos debido a la `trident-installer` usuario.
- Utilice la `uninstall parameter <Uninstalling Trident>` para limpiar después de una ejecución fallida. De forma predeterminada, la secuencia de comandos no elimina los CRD creados por Trident, por lo que es seguro desinstalar e instalar de nuevo incluso en una implementación en ejecución.
- Si desea degradar a una versión anterior de Trident, primero debe ejecutar el `tridentctl uninstall` Comando para quitar Trident. Descargue el contenido que desee "Versión de Trident" e instálela utilizando `tridentctl install` comando. Considere una degradación únicamente si no se crean nuevos VP y si no se han realizado cambios en las clases de almacenamiento/VP/back-ends existentes. Como Trident ahora utiliza CRD para mantener el estado, todas las entidades de almacenamiento creadas (back-ends, clases de almacenamiento, VP y copias Snapshot de volumen) `associated CRD objects <Kubernetes CustomResourceDefinition Objects>` En lugar de los datos escritos en el VP que se utilizaban con la versión anterior instalada de Trident. **Los VP recién creados no se podrán utilizar al volver a una versión anterior. los cambios realizados a objetos, como los back-ends, los VP, las clases de almacenamiento y las instantáneas de volumen (creadas/actualizadas/eliminadas) no serán visibles para Trident cuando se degraden**. Trident seguirá visible el VP utilizado en la versión anterior de Trident instalada. Volver a una versión anterior no interrumpirá el acceso a los VP que ya se hayan creado con la versión anterior, a menos que se hayan actualizado.
- Para eliminar completamente Trident, ejecute la `tridentctl obliviate crd` comando. Esto eliminará todos los objetos CRD y deselectará los CRD. Trident ya no gestionará los VP a los que ya había aprovisionado.

Trident deberá volver a configurarse desde cero después de esto.

- Después de una instalación correcta, si un PVC está atascado en el `Pending` fase, en marcha `kubectl describe pvc` Puede proporcionar información adicional acerca de por qué Trident no pudo aprovisionar un VP para esta RVP.

Solución de problemas de una implementación de Trident incorrecta con el operador

Si utiliza el operador, el estado de implementación de Trident `TridentOrchestrator` cambios de `Installing` para `Installed`. Si observa la `Failed` y el operador no puede recuperarse por sí solo, debe comprobar los registros del operador ejecutando el siguiente comando:

```
tridentctl logs -l trident-operator
```

Al dejar atrás los registros del contenedor del operador-trident, puede indicar dónde se encuentra el problema. Por ejemplo, uno de estos problemas podría ser la incapacidad de extraer las imágenes contenedoras necesarias de los registros de entrada en un entorno con conexión aérea.

Para comprender por qué la instalación de Trident no se ha realizado correctamente, usted debería echar un vistazo a la `TridentOrchestrator` estado.

```
kubectl describe torc trident-2
Name:         trident-2
Namespace:
Labels:       <none>
Annotations:  <none>
API Version:  trident.netapp.io/v1
Kind:         TridentOrchestrator
...
Status:
  Current Installation Params:
    IPv6:
    Autosupport Hostname:
    Autosupport Image:
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:
    Image Pull Secrets:         <nil>
    Image Registry:
    k8sTimeout:
    Kubelet Dir:
    Log Format:
    Silence Autosupport:
    Trident Image:
  Message:                      Trident is bound to another CR 'trident'
  Namespace:                    trident-2
  Status:                       Error
  Version:
Events:
  Type     Reason  Age                From                          Message
  ----     ------  ----               ----                          -------
  Warning  Error   16s (x2 over 16s)  trident-operator.netapp.io  Trident
is bound to another CR 'trident'
```

Este error indica que ya existe un `TridentOrchestrator`
Que se utilizó para instalar Trident. Puesto que cada clúster de Kubernetes solo puede
Tener una instancia de Trident, el operador se asegura de que en cualquier momento dado
el tiempo solo existe uno activo `TridentOrchestrator` que puede
cree.

Además, observar el estado de los pods de Trident puede indicar con frecuencia si algo no es correcto.

```
kubectl get pods -n trident

NAME                                READY   STATUS             RESTARTS
AGE
trident-csi-4p5kq                   1/2     ImagePullBackOff   0
5m18s
trident-csi-6f45bfd8b6-vfrkw        4/5     ImagePullBackOff   0
5m19s
trident-csi-9q5xc                   1/2     ImagePullBackOff   0
5m18s
trident-csi-9v95z                   1/2     ImagePullBackOff   0
5m18s
trident-operator-766f7b8658-ldzsv   1/1     Running            0
8m17s
```

Puede ver claramente que los pods no son capaces de inicializarse completamente porque no se han recuperado una o más imágenes de contenedor.

Para solucionar el problema, debe editar el `TridentOrchestrator` CR. Como alternativa, puede eliminar `TridentOrchestrator`, y crear un nuevo uno con la definición modificada y precisa.

Solucione problemas de una implementación de Trident incorrecta mediante `tridentctl`

Para ayudar a averiguar qué fue lo que salió mal, puede ejecutar el instalador de nuevo utilizando el `-d` argumento, que activa el modo de depuración y le ayuda a comprender cuál es el problema:

```
./tridentctl install -n trident -d
```

Después de solucionar el problema, puede limpiar la instalación de la siguiente manera y, a continuación, ejecutar el `tridentctl install` comando:

```
./tridentctl uninstall -n trident
INFO Deleted Trident deployment.
INFO Deleted cluster role binding.
INFO Deleted cluster role.
INFO Deleted service account.
INFO Removed Trident user from security context constraint.
INFO Trident uninstallation succeeded.
```

Prácticas recomendadas y recomendaciones

Puesta en marcha

Utilice las recomendaciones que se enumeran aquí cuando ponga en marcha Astra Trident.

Póngalo en marcha a un espacio de nombres dedicado

"Espacios de nombres" proporcione separación administrativa entre distintas aplicaciones y una barrera para el uso compartido de recursos. Por ejemplo, una RVP de un espacio de nombres no se puede consumir de otro. Astra Trident proporciona recursos PV a todos los espacios de nombres en el clúster de Kubernetes y, por lo tanto, aprovecha una cuenta de servicio con privilegios elevados.

Además, el acceso al pod de Trident puede permitir a un usuario acceder a las credenciales del sistema de almacenamiento y a otra información confidencial. Es importante asegurarse de que los usuarios de aplicaciones y aplicaciones de gestión no tengan la capacidad de acceder a las definiciones de objetos de Trident o a los pods mismos.

Utilice cuotas y límites de rango para controlar el consumo de almacenamiento

Kubernetes cuenta con dos funciones que, al combinarse, ofrecen un potente mecanismo que limita el consumo de recursos que consumen las aplicaciones. La "mecanismo de cuotas de almacenamiento" permite al administrador implementar límites de consumo globales, específicos para las clases de almacenamiento, de capacidad y de recuento de objetos en función del espacio de nombres. Además, utilizando un "límite de rango" Garantiza que las solicitudes de PVC se encuentren dentro de un valor mínimo y máximo antes de reenviar la solicitud al aprovisionador.

Estos valores se definen por espacio de nombres, lo que significa que cada espacio de nombres debe tener valores definidos que se ajustan a los requisitos de sus recursos. Consulte aquí para obtener más información acerca de "cómo aprovechar las cuotas".

Configuración del almacenamiento

Cada plataforma de almacenamiento de la cartera de NetApp tiene unas funciones únicas que benefician a las aplicaciones, en contenedores o no.

Descripción general de la plataforma

Trident funciona con ONTAP y Element. No existe una plataforma que se adapte mejor a todas las aplicaciones y escenarios que otra, sin embargo, las necesidades de la aplicación y el equipo que administra el dispositivo deben tenerse en cuenta al elegir una plataforma.

Debe seguir las prácticas recomendadas de base para el sistema operativo del host con el protocolo que está aprovechando. Opcionalmente, es posible que desee considerar la incorporación de prácticas recomendadas para las aplicaciones, cuando esté disponible, con configuración de back-end, clase de almacenamiento y RVP para optimizar el almacenamiento para aplicaciones específicas.

Prácticas recomendadas para ONTAP y Cloud Volumes ONTAP

Conozca las prácticas recomendadas para configurar ONTAP y Cloud Volumes ONTAP para Trident.

Las siguientes recomendaciones son directrices para configurar ONTAP para cargas de trabajo en contenedores, que consumen volúmenes aprovisionados de forma dinámica por Trident. Cada uno de ellos debe considerarse y evaluarse según la idoneidad de su entorno.

Utilice SVM dedicadas a Trident

Las máquinas virtuales de almacenamiento (SVM) proporcionan separación de tareas administrativas y de aislamiento entre clientes en un sistema ONTAP. Dedicar una SVM a las aplicaciones permite delegar privilegios y aplicar prácticas recomendadas para limitar el consumo de recursos.

Existen varias opciones disponibles para la gestión de la SVM:

- Proporcione la interfaz de gestión del clúster en la configuración del back-end, junto con las credenciales adecuadas, y especifique el nombre de la SVM.
- Cree una interfaz de gestión dedicada para la SVM mediante ONTAP System Manager o la CLI.
- Comparta la función de gestión con una interfaz de datos NFS.

En cada caso, la interfaz debe estar en DNS, y se debe usar el nombre DNS al configurar Trident. Esto permite facilitar algunas situaciones de recuperación ante desastres, por ejemplo, SVM-DR sin retención de identidad de red.

No tiene ninguna preferencia entre tener una LIF de gestión dedicada o compartida para la SVM, sin embargo, debe asegurarse de que las políticas de seguridad de red se alineen con el enfoque que elija. Sin embargo, debería accederse a la LIF de gestión a través de DNS para facilitar la máxima flexibilidad que debería tener "SVM-DR" Se podrá usar en combinación con Trident.

Limite el número máximo de volúmenes

Los sistemas de almacenamiento de ONTAP tienen un número máximo de volúmenes, que varía en función de la versión del software y la plataforma de hardware. Consulte "Hardware Universe de NetApp" Para su plataforma y versión de ONTAP específica, determine los límites exactos. Cuando se agota el número de volúmenes, las operaciones de aprovisionamiento fallan no solo para Trident, sino para todas las solicitudes de almacenamiento.

De Trident `ontap-nas` y.. `ontap-san` Los controladores aprovisionan un FlexVolume para cada volumen persistente (PV) de Kubernetes que se crea. La `ontap-nas-economy` El controlador crea aproximadamente un FlexVolume por cada 200 VP (configurable entre 50 y 300). La `ontap-san-economy` El controlador crea aproximadamente un FlexVolume por cada 100 VP (configurable entre 50 y 200). Para evitar que Trident consuma todos los volúmenes disponibles en el sistema de almacenamiento, debe establecer un límite en la SVM. Puede hacerlo desde la línea de comandos:

```
vserver modify -vserver <svm_name> -max-volumes <num_of_volumes>
```

Valor para `max-volumes` varía en función de varios criterios específicos de su entorno:

- El número de volúmenes existentes en el clúster de ONTAP
- El número de volúmenes que espera aprovisionar fuera de Trident para otras aplicaciones

- El número de volúmenes persistentes que tienen previsto consumir las aplicaciones de Kubernetes

La `max-volumes` El valor es el total de volúmenes aprovisionados en todos los nodos del clúster de ONTAP, no en un nodo ONTAP individual. Como resultado, es posible que encuentre algunas condiciones en las que un nodo de un clúster de ONTAP pueda tener muchos más o menos volúmenes aprovisionados de Trident que otro nodo.

Por ejemplo, un clúster ONTAP de dos nodos tiene la capacidad de alojar un máximo de 2000 FlexVolumes. Tener el recuento de volumen máximo establecido en 1250 parece muy razonable. Sin embargo, si solo "agregados" De un nodo se asigna a la SVM, o los agregados asignados de un nodo no se pueden aprovisionar con (por ejemplo, debido a capacidad), el otro nodo se convierte en el destino para todos los volúmenes aprovisionados de Trident. Esto significa que es posible alcanzar el límite de volumen para ese nodo antes que el `max-volumes` Se alcanza el valor, lo que repercute tanto en Trident como en otras operaciones de volúmenes que utilizan ese nodo. **Puede evitar esta situación asegurándose de que los agregados de cada nodo del clúster están asignados a la SVM que utiliza Trident en los mismos números.**

Limite el tamaño máximo de los volúmenes que ha creado Trident

Para configurar el tamaño máximo de los volúmenes que Trident puede crear `limitVolumeSize` parámetro en la `backend.json` definición.

Además de controlar el tamaño del volumen en la cabina de almacenamiento, también se deben aprovechar las capacidades de Kubernetes.

Configure Trident para utilizar CHAP bidireccional

Puede especificar los nombres de iniciador CHAP y de usuario de destino y las contraseñas en la definición de back-end, y hacer que Trident habilite CHAP en la SVM. Con el `useCHAP` Parámetro en la configuración de back-end, Trident autentica las conexiones iSCSI para los back-ends de ONTAP con CHAP. El soporte CHAP bidireccional está disponible con Trident 20.04 y versiones posteriores.

Cree y utilice una política de calidad de servicio de SVM

Al aprovechar una política de calidad de servicio de ONTAP, aplicada a la SVM, se limita el número de IOPS consumibles por los volúmenes aprovisionados de Trident. Esto ayuda a. "prevenir un matón" O un contenedor fuera de control que no afecta a las cargas de trabajo fuera de la SVM de Trident.

Puede crear una política de calidad de servicio para la SVM en unos pasos. Consulte la documentación de su versión de ONTAP para obtener la información más precisa. El ejemplo siguiente crea una política de calidad de servicio que limita el total de IOPS disponibles para la SVM a 5000.

```
# create the policy group for the SVM
qos policy-group create -policy-group <policy_name> -vserver <svm_name>
-max-throughput 5000iops

# assign the policy group to the SVM, note this will not work
# if volumes or files in the SVM have existing QoS policies
vserver modify -vserver <svm_name> -qos-policy-group <policy_name>
```

Además, si su versión de ONTAP admite esta función, puede considerar el uso de una calidad de servicio

mínima para garantizar un volumen del rendimiento para cargas de trabajo en contenedores. La calidad de servicio adaptativa no es compatible con una política de nivel de SVM.

El número de IOPS dedicado a las cargas de trabajo de los contenedores depende de muchos aspectos. Entre otras cosas, estas incluyen:

- Otras cargas de trabajo que utilizan la cabina de almacenamiento. Si hay otras cargas de trabajo, no relacionadas con la puesta en marcha de Kubernetes, y que utilizan los recursos de almacenamiento, se debe tener cuidado para garantizar que esas cargas de trabajo no se vean afectadas de forma accidental.
- Cargas de trabajo esperadas que se ejecutan en contenedores. Si las cargas de trabajo que tienen requisitos de IOPS elevados se ejecutan en contenedores, una política de calidad de servicio baja resulta en una mala experiencia.

Es importante recordar que una política de calidad de servicio asignada en el nivel de la SVM da como resultado que todos los volúmenes aprovisionados a la SVM compartan el mismo pool de IOPS. Si una, o una cifra pequeña, de las aplicaciones con contenedores tienen un requisito elevado de IOPS, podría convertirse en un problema para las otras cargas de trabajo con contenedores. Si este es el caso, puede que se desee considerar utilizar la automatización externa para asignar políticas de calidad de servicio por volumen.

Debe asignar el grupo de políticas QoS al SVM **only** si la versión de ONTAP es anterior a 9.8.

Cree grupos de políticas de calidad de servicio para Trident

La calidad de servicio garantiza que el rendimiento de las cargas de trabajo críticas no se vea degradado por cargas de trabajo de la competencia. Los grupos de políticas de calidad de servicio de ONTAP proporcionan opciones de calidad de servicio para volúmenes y permiten a los usuarios definir el techo de rendimiento para una o más cargas de trabajo. Para obtener más información sobre la calidad de servicio, consulte "Rendimiento garantizado con QoS".
Puede especificar grupos de políticas de calidad de servicio en el back-end o en un pool de almacenamiento y se aplican a cada volumen creado en ese pool o back-end.

ONTAP tiene dos tipos de grupos de políticas de calidad de servicio: Tradicionales y adaptativos. Los grupos de políticas tradicionales proporcionan un rendimiento máximo (o mínimo, en versiones posteriores) plano en IOPS. La calidad de servicio adaptativa escala automáticamente el rendimiento al tamaño de la carga de trabajo y mantiene la ratio de IOPS en TB|GB a medida que el tamaño de la carga de trabajo cambia. Esto supone una ventaja significativa cuando se gestionan cientos o miles de cargas de trabajo en una puesta en marcha de gran tamaño.

Tenga en cuenta lo siguiente al crear grupos de políticas de calidad de servicio:

- Debe configurar la `qosPolicy` introduzca la `defaults` bloque de la configuración del back-end. Consulte el siguiente ejemplo de configuración del back-end:

```
  ---
version: 1
storageDriverName: ontap-nas
managementLIF: 0.0.0.0
dataLIF: 0.0.0.0
svm: svm0
username: user
password: pass
defaults:
  qosPolicy: standard-pg
storage:
- labels:
    performance: extreme
  defaults:
    adaptiveQosPolicy: extremely-adaptive-pg
- labels:
    performance: premium
  defaults:
    qosPolicy: premium-pg
```

- Debe aplicar los grupos de políticas por volumen, de modo que cada volumen obtenga el rendimiento entero según lo especifique el grupo de políticas. No se admiten los grupos de políticas compartidas.

Para obtener más información sobre los grupos de políticas de calidad de servicio, consulte "Comandos de calidad de servicio de ONTAP 9.8".

Limite el acceso a recursos de almacenamiento a los miembros del clúster de Kubernetes

La limitación del acceso a los volúmenes NFS y a las LUN de iSCSI creadas por Trident es un componente crucial del sistema de seguridad para la puesta en marcha de Kubernetes. Si lo hace, se evita que los hosts que no forman parte del clúster de Kubernetes accedan a los volúmenes y que potencialmente modifiquen los datos de forma inesperada.

Es importante comprender que los espacios de nombres son el límite lógico de los recursos en Kubernetes. Se supone que los recursos del mismo espacio de nombres se pueden compartir; sin embargo, es importante destacar que no existe ninguna funcionalidad entre espacios de nombres. Esto significa que aunque los VP sean objetos globales, cuando están enlazados a una RVP solo pueden acceder a ellos mediante POD que están en el mismo espacio de nombres. **Es fundamental asegurarse de que los espacios de nombres se utilizan para proporcionar la separación cuando sea apropiado.**

La preocupación principal de la mayoría de las organizaciones con respecto a la seguridad de los datos en un contexto de Kubernetes es que un proceso en un contenedor puede acceder al almacenamiento montado en el host, pero que no está destinado al contenedor. "Espacios de nombres" están diseñados para evitar este tipo de compromiso. Sin embargo, hay una excepción: Contenedores privilegiados.

Un contenedor con privilegios es uno que se ejecuta con mucho más permisos de nivel de host de lo normal. No se deniegan de forma predeterminada, por lo que debe desactivar la funcionalidad utilizando "directivas de seguridad de pod".

Para los volúmenes en los que se desea obtener acceso tanto a los hosts de Kubernetes como a los externos,

el almacenamiento se debe gestionar de forma tradicional, con el VP introducido por el administrador, y no gestionado por Trident. Esto garantiza que el volumen de almacenamiento se destruya solo cuando tanto los hosts de Kubernetes como los externos se desconectaron y ya no utilizan el volumen. Además, se puede aplicar una política de exportación personalizada, lo que permite el acceso desde los nodos del clúster de Kubernetes y los servidores objetivo fuera del clúster de Kubernetes.

Para las implementaciones que tienen nodos de infraestructura dedicados (por ejemplo, OpenShift) u otros nodos que no pueden programar aplicaciones de usuario, se deben utilizar directivas de exportación independientes para limitar aún más el acceso a los recursos de almacenamiento. Esto incluye la creación de una directiva de exportación para los servicios que se implementan en dichos nodos de infraestructura (por ejemplo, los servicios de registro y métricas de OpenShift) y aplicaciones estándar que se implementan en nodos que no son de infraestructura.

Usar una política de exportación dedicada

Debe asegurarse de que existe una política de exportación para cada back-end que solo permita el acceso a los nodos presentes en el clúster de Kubernetes. Trident puede crear y gestionar automáticamente políticas de exportación a partir de la versión 20.04. De esta forma, Trident limita el acceso a los volúmenes que aprovisiona a los nodos en el clúster de Kubernetes y simplifica la adición o la eliminación de nodos.

También puede crear una política de exportación manualmente y rellenarla con una o varias reglas de exportación que procesarán cada solicitud de acceso a nodo:

- Utilice la `vserver export-policy create` Comando de la interfaz de línea de comandos de ONTAP para crear la política de exportación.
- Añada reglas a la política de exportación mediante la `vserver export-policy rule create` Comando de la CLI de ONTAP.

Si ejecuta estos comandos, puede restringir el acceso de los nodos de Kubernetes a los datos.

Desactivar `showmount` Para la SVM de la aplicación

La `showmount` Con la función, un cliente NFS puede consultar a la SVM para obtener una lista de exportaciones NFS disponibles. Un pod puesto en marcha en el clúster de Kubernetes puede ejecutar el `showmount -e` Comando en la LIF de datos y recibe una lista de montajes disponibles, incluidos los a los que no tiene acceso. Aunque esto, por sí solo, no supone un compromiso con la seguridad, proporciona información innecesaria, potencialmente que ayuda a un usuario no autorizado a conectarse con una exportación NFS.

Debe desactivar `showmount` Con el comando CLI de ONTAP a nivel de la SVM:

```
vserver nfs modify -vserver <svm_name> -showmount disabled
```

Mejores prácticas para SolidFire

Conozca las prácticas recomendadas para configurar el almacenamiento de SolidFire para Trident.

Crear cuenta de SolidFire

Cada cuenta SolidFire representa un propietario de volumen único y recibe su propio conjunto de credenciales de protocolo de autenticación por desafío mutuo (CHAP). Es posible acceder a los volúmenes asignados a una cuenta mediante el nombre de cuenta y las credenciales CHAP relativas o un grupo de acceso de

volúmenes. Una cuenta puede tener hasta 2000 volúmenes asignados, pero un volumen solo puede pertenecer a una cuenta.

Cree una política de calidad de servicio

Utilice las políticas de calidad de servicio de SolidFire si desea crear y guardar un ajuste de calidad de servicio estandarizado que se puede aplicar a muchos volúmenes.

Puede establecer parámetros de calidad de servicio por cada volumen. El rendimiento de cada volumen se puede garantizar mediante el establecimiento de tres parámetros configurables que definen la calidad de servicio: Min IOPS, Max IOPS y Burst IOPS.

Aquí pueden ver los valores mínimos, máximos y de ráfaga de IOPS en relación con el tamaño de bloque de 4 KB.

Parámetro de IOPS	Definición	Espacio valor	Valor predeterminado	Capacidad Valor (4 KB)
IOPS mín	El nivel garantizado de rendimiento de un volumen.	50	50	15000
Tasa máx. De IOPS	El rendimiento no superará este límite.	50	15000	200.000
IOPS de ráfaga	IOPS máximo permitido en un escenario de ráfaga breve.	50	15000	200.000

Aunque Max IOPS y Burst IOPS se pueden establecer con un valor máximo de 200,000 000, el rendimiento máximo en el mundo real de un volumen se ve limitado por el uso del clúster y el rendimiento por cada nodo.

El tamaño de bloque y el ancho de banda influyen directamente en el número de IOPS. A medida que estos aumenten, el sistema aumentará el ancho de banda hasta el nivel que necesite para procesar los tamaños de bloque más grandes. A medida que aumenta el ancho de banda, se reduce el número de IOPS que el sistema es capaz de conseguir. Consulte "Calidad de servicio de SolidFire" Para obtener más información sobre la calidad de servicio y el rendimiento.

Autenticación SolidFire

Element admite dos métodos para la autenticación: CHAP y grupos de acceso de volumen (VAG). CHAP utiliza el protocolo CHAP para autenticar el host al back-end. Los grupos de acceso de volúmenes controlan el acceso a los volúmenes que aprovisiona. NetApp recomienda utilizar CHAP para la autenticación, ya que es más sencillo y sin límites de escalado.

Trident con el aprovisionador CSI mejorado admite el uso de la autenticación CHAP. Los VAG sólo deben utilizarse en el modo de funcionamiento tradicional no CSI.

La autenticación CHAP (verificación de que el iniciador es el usuario de volumen objetivo) solo se admite con control de acceso basado en la cuenta. Si se utiliza CHAP para la autenticación, hay dos opciones

disponibles: CHAP unidireccional y CHAP bidireccional. CHAP unidireccional autentica el acceso al volumen mediante el nombre de cuenta de SolidFire y el secreto de iniciador. La opción CHAP bidireccional proporciona la manera más segura de autenticar el volumen, ya que el volumen autentica el host a través del nombre de cuenta y el secreto de iniciador, y luego el host autentica el volumen por medio del nombre de cuenta y el secreto de destino.

Sin embargo, si no se puede habilitar CHAP y se requieren los VAG, cree el grupo de acceso y añada los iniciadores de host y los volúmenes al grupo de acceso. Cada IQN que se añade a un grupo de acceso puede acceder a cada volumen del grupo con o sin autenticación CHAP. Si el iniciador de iSCSI está configurado para utilizar la autenticación CHAP, se utiliza el control de acceso basado en cuentas. Si el iniciador iSCSI no está configurado para utilizar la autenticación CHAP, se utiliza el control de acceso del grupo de acceso de volúmenes.

¿Dónde encontrar más información?

A continuación se enumeran algunas de las prácticas recomendadas. Busque en el "Biblioteca de NetApp" para las versiones más actuales.

ONTAP

- "Prácticas recomendadas y guía de implementación de NFS"
- "Guía de administración de SAN" (Para iSCSI)
- "Configuración exprés de iSCSI para RHEL"

Software Element

- "Configuración de SolidFire para Linux"

NetApp HCI

- "Requisitos previos de la implementación de NetApp HCI"
- "Acceda al motor de implementación de NetApp"

Información sobre las prácticas recomendadas de la aplicación

- "Prácticas recomendadas para MySQL en ONTAP"
- "Prácticas recomendadas para MySQL en SolidFire"
- "NetApp SolidFire y Cassandra"
- "Prácticas recomendadas de Oracle en SolidFire"
- "Prácticas recomendadas de PostgreSQL en SolidFire"

No todas las aplicaciones tienen directrices específicas, es importante trabajar con su equipo de NetApp y utilizar el "Biblioteca de NetApp" para encontrar la documentación más actualizada.

Integre Astra Trident

Para integrar Astra Trident, los siguientes elementos de diseño y arquitectura requieren integración: Selección y puesta en marcha de controladores, diseño de clase de almacenamiento, diseño de pools virtuales, efecto de la reclamación de volumen persistente (PVC) en el aprovisionamiento de almacenamiento, las operaciones de

volumen y la puesta en marcha de servicios OpenShift con Astra Trident.

Selección y despliegue del conductor

Seleccione e implemente un controlador de back-end para el sistema de almacenamiento.

Controladores de entorno de administración ONTAP

Los controladores de entorno de administración de ONTAP se diferencian por el protocolo utilizado y cómo se aprovisionan los volúmenes en el sistema de almacenamiento. Por lo tanto, tenga cuidado al decidir qué controlador implementar.

En un nivel superior, si la aplicación cuenta con componentes que necesitan almacenamiento compartido (varios POD que acceden al mismo PVC), los controladores basados en NAS serán la opción predeterminada, mientras que los controladores iSCSI basados en bloques satisfacen las necesidades de almacenamiento no compartido. Elija el protocolo según los requisitos de la aplicación y el nivel de comodidad de los equipos de almacenamiento e infraestructura. Por lo general, existe poca diferencia entre ellas para la mayoría de las aplicaciones, con tanta frecuencia la decisión se basa en si se necesita o no almacenamiento compartido (donde más de un pod necesitará acceso simultáneo).

Los controladores de entorno de administración de ONTAP disponibles son:

- `ontap-nas`: Cada PV aprovisionado es un FlexVolume ONTAP completo.
- `ontap-nas-economy`: Cada PV aprovisionado es un qtree, con un número configurable de qtrees por FlexVolume (el valor predeterminado es 200).
- `ontap-nas-flexgroup`: Cada VP aprovisionado como ONTAP FlexGroup completo y se utilizan todos los agregados asignados a una SVM.
- `ontap-san`: Cada PV aprovisionado es una LUN dentro de su propio FlexVolume.
- `ontap-san-economy`: Cada VP aprovisionado es una LUN, con un número configurable de LUN por FlexVolume (el valor predeterminado es 100).

La elección entre los tres controladores NAS tiene algunas ramificaciones a las funciones, que están disponibles para la aplicación.

Tenga en cuenta que, en las siguientes tablas, no todas las funcionalidades se exponen a través de Astra Trident. El administrador de almacenamiento debe aplicar algunas después del aprovisionamiento si se desea disponer de esta funcionalidad. Las notas al pie de la superíndice distinguen la funcionalidad por característica y controlador.

Unidades NAS de ONTAP	Snapshot	Clones	Políticas de exportación dinámicas	Conexión múltiple	Calidad de servicio	Cambie el tamaño	Replicación
`ontap-nas`	Sí	Sí	Yespie de página:5[]	Sí	Nota de pie de página:1[]	Sí	Nota de pie de página:1[]
`ontap-nas-economy`	Nota de pie de página:3[]	Nota de pie de página:3[]	Yespie de página:5[]	Sí	Nota de pie de página:3[]	Sí	Nota de pie de página:3[]

Unidades NAS de ONTAP	Snapshot	Clones	Políticas de exportación dinámicas	Conexión múltiple	Calidad de servicio	Cambie el tamaño	Replicación
`ontap-nas-flexgroup`	Nota de pie de página:1[]	No	Yespie de página:5[]	Sí	Nota de pie de página:1[]	Sí	Nota de pie de página:1[]

Astra Trident ofrece 2 controladores SAN para ONTAP, cuyas funciones se muestran a continuación.

Controladores para SAN de ONTAP	Snapshot	Clones	Conexión múltiple	CHAP bidireccional	Calidad de servicio	Cambie el tamaño	Replicación
`ontap-san`	Sí	Sí	Nota de pie de página:4[]	Sí	Nota de pie de página:1[]	Sí	Nota de pie de página:1[]
`ontap-san-economy`	Sí	Sí	Nota de pie de página:4[]	Sí	Nota de pie de página:3[]	Sí	Nota de pie de página:3[]

> Nota al pie de las tablas anteriores:
>
> Nota:1[]: No gestionado por Astra Trident
>
> YesFootnote:2[]: Gestionado por Astra Trident, pero no por VP granular
>
> Nota:3[]: No gestionado por Astra Trident y no por VP granular
>
> Yes [4]: Compatible con volúmenes de bloques sin configurar
>
> YesFootnote:5[]: Apoyado por CSI Trident

Las funciones que no son granulares en los VP se aplican a todo el FlexVolume y todos los VP (es decir, qtrees o LUN de FlexVols compartidos) compartirán un programa común.

Como podemos ver en las tablas anteriores, gran parte de la funcionalidad entre `ontap-nas` y.. `ontap-nas-economy` es lo mismo. Sin embargo, porque la `ontap-nas-economy` Esta unidad limita la capacidad de controlar la programación con una granularidad VP, lo que puede afectar a su planificación de backup y recuperación ante desastres en concreto. En el caso de los equipos de desarrollo que desean aprovechar la funcionalidad de clonado de PVC en el almacenamiento de ONTAP, esto solo es posible cuando se utiliza la `ontap-nas`, `ontap-san` o. `ontap-san-economy` de windows

La `solidfire-san` El controlador también es capaz de clonar EVs.

Controladores de entorno de administración Cloud Volumes ONTAP

Cloud Volumes ONTAP proporciona control de datos junto con funciones de almacenamiento empresarial para diversos casos de uso, como recursos compartidos de archivos y almacenamiento a nivel de bloque que presta servicio a protocolos NAS y SAN (NFS, SMB/CIFS e iSCSI). Los controladores compatibles para Cloud Volume ONTAP son `ontap-nas`, `ontap-nas-economy`, `ontap-san` y.. `ontap-san-economy`. Estos son aplicables a Cloud Volume ONTAP para Azure, Cloud Volume ONTAP para GCP.

Controladores de entorno de administración de Amazon FSX para ONTAP

Amazon FSX para ONTAP permite a los clientes aprovechar las funciones, el rendimiento y las funcionalidades administrativas de NetApp con las que ya están familiarizados, a la vez que aprovechan la simplicidad, la agilidad, la seguridad y la escalabilidad de almacenar datos en AWS. FSX para ONTAP es compatible con muchas de las API de administración y las funciones del sistema de archivos de ONTAP. Los controladores compatibles para Cloud Volume ONTAP son `ontap-nas`, `ontap-nas-economy`, `ontap-nas-flexgroup`, `ontap-san` y.. `ontap-san-economy`.

Controladores de back-end de HCI/SolidFire de NetApp

La `solidfire-san` El controlador que se utiliza con las plataformas HCI/SolidFire de NetApp, ayuda al administrador a configurar un back-end de Element para Trident según los límites de calidad de servicio. Si desea diseñar el back-end de modo que establezca los límites de calidad de servicio específicos en los volúmenes aprovisionados por Trident, utilice la `type` parámetro en el archivo back-end. El administrador también puede restringir el tamaño del volumen que podría crearse en el almacenamiento mediante el `limitVolumeSize` parámetro. Actualmente, las funciones de almacenamiento de Element, como el cambio de tamaño de volumen y la replicación de volumen, no se admiten mediante el `solidfire-san` controlador. Estas operaciones se deben realizar manualmente mediante la interfaz de usuario web del software Element.

Controlador SolidFire	Snapshot	Clones	Conexión múltiple	CHAP	Calidad de servicio	Cambie el tamaño	Replicación
`solidfire-san`	Sí	Sí	Nota de pie de página:2[]	Sí	Sí	Sí	Nota de pie de página:1[]

> Pie de página:
>
> Nota:1[]: No gestionado por Astra Trident
>
> Yes [2]: Compatible con volúmenes de bloques sin configurar

Controladores de entorno de administración Azure NetApp Files

Astra Trident utiliza `azure-netapp-files` controlador para administrar "Azure NetApp Files" servicio.

Puede encontrar más información acerca de este controlador y cómo configurarlo en "Configuración de back-end de Astra Trident para Azure NetApp Files".

Controlador Azure NetApp Files	Snapshot	Clones	Conexión múltiple	Calidad de servicio	Expanda	Replicación
`azure-netapp-files`	Sí	Sí	Sí	Sí	Sí	Nota de pie de página:1[]

> Pie de página:
>
> Nota:1[]: No gestionado por Astra Trident

Cloud Volumes Service en el controlador back-end de Google Cloud

Astra Trident utiliza `gcp-cvs` Controlador para vincular con Cloud Volumes Service en Google Cloud.

La `gcp-cvs` La unidad utiliza pools virtuales para abstraer el back-end y permitir a Astra Trident determinar la ubicación del volumen. El administrador define los pools virtuales en `backend.json` archivos. Las clases de almacenamiento utilizan selectores para identificar los pools virtuales por etiqueta.

- Si se definen pools virtuales en el back-end, Astra Trident intentará crear un volumen en los pools de almacenamiento de Google Cloud a los que están limitados esos pools virtuales.
- Si no se definen pools virtuales en el back-end, Astra Trident selecciona un pool de almacenamiento de Google Cloud de los pools de almacenamiento disponibles en la región.

Para configurar el back-end de Google Cloud en Astra Trident, debe especificar `projectNumber`, `apiRegion`, y. `apiKey` en el archivo de fondo. Puede encontrar el número de proyecto en la consola de Google Cloud. La clave API se obtiene del archivo de claves privadas de la cuenta de servicio que creó al configurar el acceso de API para Cloud Volumes Service en Google Cloud.

Para obtener más información sobre Cloud Volumes Service en los tipos de servicio y niveles de servicio de Google Cloud, consulte "Obtenga más información sobre la compatibilidad de Astra Trident con CVS para GCP".

Controlador de Cloud Volumes Service para Google Cloud	Snapshot	Clones	Conexión múltiple	Calidad de servicio	Expanda	Replicación
`gcp-cvs`	Sí	Sí	Sí	Sí	Sí	Disponible solo en el tipo de servicio CVS-Performance.

Notas de replicación

- Astra Trident no gestiona la replicación.
- El clon se creará en el mismo pool de almacenamiento que el volumen de origen.

Diseño de clase de almacenamiento

Las clases de almacenamiento individuales deben configurarse y aplicarse para crear un objeto de clase de almacenamiento Kubernetes. En esta sección se analiza cómo diseñar una clase de almacenamiento para su aplicación.

Utilización de back-end específica

El filtrado se puede usar en un objeto de clase de almacenamiento específico para determinar el pool o conjunto de pools de almacenamiento que se utilizarán con esa clase de almacenamiento específica. Se pueden establecer tres conjuntos de filtros en la clase de almacenamiento: `storagePools`, `additionalStoragePools`, y/o. `excludeStoragePools`.

La `storagePools` el parámetro ayuda a restringir el almacenamiento al conjunto de pools que coinciden con cualquier atributo especificado. La `additionalStoragePools` El parámetro se utiliza para ampliar el conjunto de pools que utilizará Astra Trident para el aprovisionamiento junto con el conjunto de pools seleccionados por los atributos y. `storagePools` parámetros. Es posible usar un parámetro de forma independiente o ambos juntos para garantizar que se seleccione el conjunto adecuado de pools de almacenamiento.

La `excludeStoragePools` el parámetro se utiliza para excluir específicamente el conjunto de pools enumerado que coincide con los atributos.

Emular las políticas de calidad de servicio

Si desea diseñar clases de almacenamiento para emular políticas de calidad de servicio, cree una clase de almacenamiento con la `media` atributo como `hdd` o. `ssd`. Según la `media` Atributo mencionado en la clase de almacenamiento, Trident seleccionará el back-end apropiado `hdd` o. `ssd` agregados para coincidir con el atributo de medios y, a continuación, dirigir el aprovisionamiento de los volúmenes al agregado específico. Por tanto, podemos crear UNA CLASE PREMIUM DE almacenamiento que tendría `media` atributo establecido como `ssd` Las cuales pueden clasificarse como política DE calidad DE servicio PREMIUM. Podemos crear otro ESTÁNDAR de clase de almacenamiento que tenga el conjunto de atributos de medios como "hdd", que podría clasificarse como política DE calidad DE servicio ESTÁNDAR. También podríamos usar el atributo "IOPS" en la clase de almacenamiento para redirigir el aprovisionamiento a un dispositivo Element que se puede definir como una Política de calidad de servicio.

Utilizar back-end basado en funciones específicas

Las clases de almacenamiento se pueden diseñar para dirigir el aprovisionamiento de volúmenes en un entorno de administración específico, donde se habilitan funciones como thin provisioning y thick, copias Snapshot, clones y cifrado. Para especificar qué almacenamiento se debe utilizar, cree clases de almacenamiento que especifiquen el back-end adecuado con la función necesaria habilitada.

Pools virtuales

Hay pools virtuales disponibles para todos los back-ends de Astra Trident. Puede definir pools virtuales para cualquier back-end a través de cualquier controlador que proporcione Astra Trident.

Los pools virtuales permiten a un administrador crear un nivel de abstracción sobre los back-ends que se puede hacer referencia a través de las clases de almacenamiento, para obtener mayor flexibilidad y colocación eficiente de los volúmenes en back-ends. Pueden definirse distintos back-ends con la misma clase de servicio. Es más, es posible crear varios pools de almacenamiento en el mismo back-end, pero con características diferentes. Cuando se configura una clase de almacenamiento con un selector con las etiquetas específicas, Astra Trident elige un back-end que coincide con todas las etiquetas de selector para

colocar el volumen. Si las etiquetas del selector de clase de almacenamiento coinciden con varios pools de almacenamiento, Astra Trident elegirá una de ellas para aprovisionar el volumen desde.

Diseño de pool virtual

Al crear un back-end, generalmente puede especificar un conjunto de parámetros. Era imposible que el administrador creara otro back-end con las mismas credenciales de almacenamiento y con un conjunto de parámetros diferente. Con la introducción de pools virtuales, este problema se ha aliviado. Los pools virtuales son una abstracción de niveles introducida entre el back-end y la clase de almacenamiento de Kubernetes de modo que el administrador puede definir parámetros junto con etiquetas a las que se puede hacer referencia a través de las clases de almacenamiento de Kubernetes como selector, de forma independiente del back-end. Es posible definir pools virtuales para todos los back-ends de NetApp compatibles con Astra Trident. Esta lista incluye HCI de SolidFire/NetApp, ONTAP, Cloud Volumes Service en GCP y Azure NetApp Files.

> Al definir los pools virtuales, se recomienda no intentar reorganizar el orden de los grupos virtuales existentes en una definición de backend. También es aconsejable no editar/modificar atributos para un pool virtual existente y definir un nuevo pool virtual en su lugar.

Emulación de distintos niveles de servicio/calidad de servicio

Se pueden diseñar pools virtuales para emular clases de servicio. Al utilizar la implementación de pools virtuales para el servicio Cloud Volume para Azure NetApp Files, examinemos cómo podemos configurar distintas clases de servicio. Configure el backend ANF con varias etiquetas, que representan diferentes niveles de rendimiento. Configurado `servicelevel` aspecto al nivel de rendimiento apropiado y agregar otros aspectos requeridos en cada etiqueta. Ahora cree diferentes clases de almacenamiento de Kubernetes que se asignarán a diferentes pools virtuales. Con el `parameters.selector` Campo, cada clase de almacenamiento llama a qué pools virtuales se pueden utilizar para alojar un volumen.

Asignación de un conjunto específico de aspectos

Se pueden diseñar varios pools virtuales con un conjunto específico de aspectos a partir de un único back-end de almacenamiento. Para ello, configure el backend con varias etiquetas y defina los aspectos necesarios en cada etiqueta. Ahora cree diferentes clases de almacenamiento de Kubernetes usando `parameters.selector` campo que se asignará a diferentes pools virtuales. Los volúmenes que se aprovisionan en el back-end tendrán los aspectos definidos en el pool virtual elegido.

Las características de PVC que afectan al aprovisionamiento de almacenamiento

Algunos parámetros que superen la clase de almacenamiento solicitada pueden afectar al proceso de decisión de aprovisionamiento de Astra Trident al crear una RVP.

Modo de acceso

Al solicitar un almacenamiento a través de un PVC, uno de los campos obligatorios es el modo de acceso. El modo deseado puede afectar el back-end seleccionado para alojar la solicitud de almacenamiento.

Astra Trident intentará igualar el protocolo de almacenamiento que se utiliza con el método de acceso especificado según la siguiente matriz. Es independiente de la plataforma de almacenamiento subyacente.

	ReadWriteOnce	ReadOnlyMany	ReadWriteMany
ISCSI	Sí	Sí	Sí (bloque sin formato)
NFS	Sí	Sí	Sí

Si se solicita un PVC ReadWriteMany enviado a una implementación de Trident sin un back-end de NFS configurado, no se aprovisionará ningún volumen. Por este motivo, el solicitante debe usar el modo de acceso adecuado para su aplicación.

Operaciones de volumen

Modifique los volúmenes persistentes

Los volúmenes persistentes son, con dos excepciones, objetos inmutables en Kubernetes. Una vez creada, la política de reclamaciones y el tamaño se pueden modificar. Sin embargo, esto no impide que se modifiquen algunos aspectos del volumen fuera de Kubernetes. Esto puede ser deseable para personalizar el volumen para aplicaciones específicas, con el fin de garantizar que la capacidad no se consume accidentalmente, o simplemente mover el volumen a una controladora de almacenamiento diferente por cualquier motivo.

Los aprovisionadores de árbol de Kubernetes no admiten las operaciones de cambio de tamaño de volumen para NFS o iSCSI VP en este momento. Astra Trident admite la ampliación de volúmenes NFS e iSCSI.

Los detalles de conexión del VP no se pueden modificar una vez creado.

Cree snapshots de volumen bajo demanda

Astra Trident admite la creación de instantáneas de volumen bajo demanda y la creación de EVs a partir de instantáneas utilizando el marco CSI. Las copias Snapshot proporcionan un método cómodo de mantener copias de un momento específico de los datos y poseen un ciclo de vida independiente del VP de origen de Kubernetes. Estas instantáneas se pueden utilizar para clonar EVs.

Crear volúmenes a partir de snapshots

Astra Trident también admite la creación de volúmenes PersistentVolumes a partir de snapshots de volúmenes. Para ello, sólo tiene que crear una reclamación de volumen persistente y mencionar la `datasource` como la snapshot necesaria a partir de la que se debe crear el volumen. Astra Trident se encargará de gestionar esta RVP mediante la creación de un volumen con los datos presentes en la snapshot. Con esta función, es posible duplicar datos entre regiones, crear entornos de prueba, reemplazar un volumen de producción dañado o dañado en su totalidad, o recuperar archivos y directorios específicos y transferirlos a otro volumen adjunto.

Mueva volúmenes al clúster

Los administradores de almacenamiento pueden mover volúmenes entre agregados y controladoras en el clúster de ONTAP de forma no disruptiva al consumidor de almacenamiento. Esta operación no afecta al clúster Astra Trident o Kubernetes, siempre y cuando el agregado de destino sea el que utilice la SVM a la que Astra Trident tenga acceso. Lo que es importante: Si el agregado se ha añadido recientemente a la SVM, deberá actualizar el back-end añadiendo de nuevo a Astra Trident. Esto hará que Astra Trident vuelva a realizar el inventario de las SVM para que se reconozca el nuevo agregado.

Sin embargo, Astra Trident no admite automáticamente la transferencia de volúmenes entre back-ends. Esto incluye entre SVM en el mismo clúster, entre clústeres o en una plataforma de almacenamiento diferente (incluso si ese sistema de almacenamiento está conectado a Astra Trident).

Si se copia un volumen en otra ubicación, es posible utilizar la función de importación de volúmenes para importar los volúmenes actuales a Astra Trident.

Expanda los volúmenes

Astra Trident admite el cambio de tamaño de VP iSCSI y NFS. De este modo, los usuarios pueden cambiar el tamaño de sus volúmenes directamente desde la capa de Kubernetes. La expansión de volumen es posible para las principales plataformas de almacenamiento de NetApp, como ONTAP, HCI de SolidFire/NetApp y back-ends de Cloud Volumes Service. Para permitir una posible expansión más adelante, establezca `allowVolumeExpansion` para `true` En el tipo de almacenamiento asociado con el volumen. Siempre que sea necesario cambiar el tamaño del volumen persistente, edite el `spec.resources.requests.storage` Anotación en la reclamación de volumen persistente al tamaño de volumen requerido. Trident se ocupa automáticamente de ajustar el tamaño del volumen en el clúster de almacenamiento.

Importe un volumen existente en Kubernetes

La importación de volúmenes ofrece la posibilidad de importar un volumen de almacenamiento existente en un entorno de Kubernetes. Actualmente es compatible con `ontap-nas`, `ontap-nas-flexgroup`, `solidfire-san`, `azure-netapp-files`, y. `gcp-cvs` de windows Esta función es útil cuando se pasa una aplicación existente a Kubernetes o durante escenarios de recuperación ante desastres.

Cuando utilice las ONTAP y. `solidfire-san` controladores, utilice el comando `tridentctl import volume <backend-name> <volume-name> -f /path/pvc.yaml` Para importar un volumen existente a Kubernetes y que Astra Trident gestione. El archivo PVC YLMA o JSON que se usa en el comando import volume señala a una clase de almacenamiento que identifica a Astra Trident como el aprovisionador. Cuando se utiliza un back-end de HCI/SolidFire de NetApp, asegúrese de que los nombres de los volúmenes sean únicos. Si los nombres de los volúmenes se duplican, clone el volumen en un nombre único de modo que la función de importación de volumen pueda distinguir entre ellos.

Si la `azure-netapp-files` o. `gcp-cvs` se utiliza el controlador, utilice el comando `tridentctl import volume <backend-name> <volume path> -f /path/pvc.yaml` Para importar el volumen a Kubernetes que gestiona Astra Trident. Esto garantiza una referencia de volumen única.

Una vez ejecutado el comando anterior, Astra Trident encontrará el volumen en el back-end y leerá su tamaño. Agregará automáticamente (y sobrescribirá si es necesario) el tamaño del volumen del PVC configurado. A continuación, Astra Trident crea el nuevo VP y Kubernetes enlaza la RVP con el VP.

Si se puso en marcha un contenedor de modo que requería la RVP específica importada, este permanecería en estado pendiente hasta que el par PVC/VP se enlaza a través del proceso de importación del volumen. Una vez enlazados el par PVC/PV, el contenedor debería aparecer, siempre que no haya otros problemas.

Implementar servicios OpenShift

Los servicios de clúster de valor añadido de OpenShift proporcionan una funcionalidad importante a los administradores de clúster y a las aplicaciones que se alojan. Sin embargo, el almacenamiento que utilizan estos servicios puede aprovisionarse con los recursos locales de nodos, esto limita con frecuencia la capacidad, el rendimiento, la capacidad de recuperación y la sostenibilidad del servicio. Sin embargo, al aprovechar una cabina de almacenamiento empresarial para ofrecer la capacidad de estos servicios se puede mejorar considerablemente el servicio. Al igual que sucede con todas las aplicaciones, OpenShift y los administradores de almacenamiento deberían trabajar estrechamente para determinar cuáles son las mejores opciones para cada uno de ellos. La documentación de Red Hat debe utilizarse en gran medida para determinar los requisitos y garantizar que se satisfagan las necesidades de tamaño y rendimiento.

Servicio de registro

Se ha documentado en la implementación y administración del almacenamiento para el registro "netapp.io" en la "blog".

Servicio de registro

Al igual que otros servicios OpenShift, el servicio de registro se pone en marcha con Ansible, con parámetros de configuración suministrados por el archivo de inventario, también conocido como los hosts, que se proporcionan al libro de estrategia. Hay dos métodos de instalación que se tratarán: Implementar el registro durante la instalación inicial de OpenShift y desplegar el registro después de que OpenShift haya sido implementado
instalado.

> A partir de Red Hat OpenShift versión 3.9, la documentación oficial recomienda contra NFS para el servicio de registro debido a problemas relacionados con la corrupción de datos. Esto se basa en las pruebas de Red Hat de sus productos. El servidor NFS de ONTAP no tiene estos problemas y puede realizar fácilmente una implementación de registro. Finalmente, la elección del protocolo para el servicio de registro depende de usted; simplemente sabe que ambos funcionarán bien cuando usen las plataformas de NetApp y no hay motivos para evitar NFS si eso es lo que prefiere.

Si decide utilizar NFS con el servicio de registro, tendrá que establecer la variable Ansible `openshift_enable_unsupported_configurations` para `true` para evitar que el instalador falle.

Manos a la obra

Opcionalmente, el servicio de registro puede implementarse tanto para aplicaciones como para las operaciones principales del propio clúster OpenShift. Si decide implementar el registro de operaciones, especificando la variable `openshift_logging_use_ops` como `true`, se crearán dos instancias del servicio. Las variables que controlan la instancia de registro de las operaciones contienen "OPS" en ellas, mientras que la instancia de las aplicaciones no.

Es importante configurar las variables de Ansible según el método de puesta en marcha para garantizar que los servicios subyacentes utilizan el almacenamiento correcto. Veamos las opciones de cada uno de los métodos de implementación.

> Las siguientes tablas sólo contienen las variables que son relevantes para la configuración de almacenamiento, ya que están relacionadas con el servicio de registro. Puede encontrar otras opciones en "Documentación de registro de RedHat OpenShift" que deben revisarse, configurarse y utilizarse en función de la puesta en marcha.

Las variables de la siguiente tabla harán que el libro de estrategia de Ansible cree un VP y una RVP para el servicio de registro con los detalles proporcionados. Este método es significativamente menos flexible que usar la tableta playbook de instalación de componentes después de la instalación de OpenShift; sin embargo, si tiene volúmenes existentes disponibles, es una opción.

Variable	Detalles
`openshift_logging_storage_kind`	Establezca en `nfs` Para que el instalador cree un PV de NFS para el servicio de registro.
`openshift_logging_storage_host`	El nombre de host o la dirección IP del host NFS. Esto debe configurarse en la LIF de datos de su máquina virtual.

Variable	Detalles
`openshift_logging_storage_nfs_directory`	La ruta de montaje para la exportación NFS. Por ejemplo, si el volumen se juntan como `/openshift_logging`, utilizaría esa ruta de acceso para esta variable.
`openshift_logging_storage_volume_name`	El nombre, por ejemplo `pv_ose_logs`, Del PV que se va a crear.
`openshift_logging_storage_volume_size`	Por ejemplo, el tamaño de la exportación NFS `100Gi`.

Si su clúster OpenShift ya se está ejecutando y, por lo tanto, Trident se ha implementado y configurado, el instalador puede utilizar el aprovisionamiento dinámico para crear los volúmenes. Será necesario configurar las siguientes variables.

Variable	Detalles
`openshift_logging_es_pvc_dynamic`	Establezca esta opción en true para usar volúmenes aprovisionados dinámicamente.
`openshift_logging_es_pvc_storage_class_name`	El nombre de la clase de almacenamiento que se utilizará en la RVP.
`openshift_logging_es_pvc_size`	El tamaño del volumen solicitado en la RVP.
`openshift_logging_es_pvc_prefix`	Prefijo para los EVs que utiliza el servicio de registro.
`openshift_logging_es_ops_pvc_dynamic`	Establezca en `true` para utilizar volúmenes aprovisionados de forma dinámica para la instancia de registro de operaciones.
`openshift_logging_es_ops_pvc_storage_class_name`	Nombre de la clase de almacenamiento para la instancia de registro de operaciones.
`openshift_logging_es_ops_pvc_size`	El tamaño de la solicitud de volumen para la instancia de operaciones.
`openshift_logging_es_ops_pvc_prefix`	Prefijo para las RVP de instancia de OPS.

Despliegue la pila de registro

Si va a implementar el registro como parte del proceso de instalación inicial de OpenShift, sólo tendrá que seguir el proceso de implementación estándar. Ansible configurará y pondrá en marcha los servicios y los objetos de OpenShift necesarios para que el servicio esté disponible tan pronto como finalice Ansible.

No obstante, si se pone en marcha después de la instalación inicial, Ansible deberá usar el libro de estrategia de los componentes. Este proceso puede cambiar ligeramente con diferentes versiones de OpenShift, así que asegúrese de leer y seguir "Documentación de Red Hat OpenShift Container Platform 3.11" para su versión.

Servicio de métricas

El servicio de métricas proporciona al administrador información valiosa sobre el estado, la utilización de recursos y la disponibilidad del clúster OpenShift. También es necesaria para la funcionalidad de escala automática en pod y muchas organizaciones usan datos del servicio de mediciones para su cargo y/o para mostrar aplicaciones.

Al igual que sucede con el servicio de registro y OpenShift en su conjunto, Ansible se utiliza para poner en marcha el servicio de métricas. Además, al igual que el servicio de registro, el servicio de mediciones se puede implementar durante una configuración inicial del clúster o después de su funcionamiento mediante el método de instalación de componentes. Las siguientes tablas contienen las variables importantes a la hora de configurar el almacenamiento persistente para el servicio de métricas.

> Las siguientes tablas solo contienen las variables relevantes para la configuración del almacenamiento en cuanto se relaciona con el servicio de mediciones. Hay muchas otras opciones en la documentación que se deben revisar, configurar y utilizar de acuerdo con su implementación.

Variable	Detalles
`openshift_metrics_storage_kind`	Establezca en `nfs` Para que el instalador cree un PV de NFS para el servicio de registro.
`openshift_metrics_storage_host`	El nombre de host o la dirección IP del host NFS. Esto debe configurarse en el LIF de datos de su SVM.
`openshift_metrics_storage_nfs_directory`	La ruta de montaje para la exportación NFS. Por ejemplo, si el volumen se juntan como `/openshift_metrics`, utilizaría esa ruta de acceso para esta variable.
`openshift_metrics_storage_volume_name`	El nombre, ej `pv_ose_metrics`, Del PV que se va a crear.
`openshift_metrics_storage_volume_size`	Por ejemplo, el tamaño de la exportación NFS `100Gi`.

Si su clúster OpenShift ya se está ejecutando y, por lo tanto, Trident se ha implementado y configurado, el instalador puede utilizar el aprovisionamiento dinámico para crear los volúmenes. Será necesario configurar las siguientes variables.

Variable	Detalles
`openshift_metrics_cassandra_pvc_prefix`	Prefijo que se utiliza para las RVP de métricas.
`openshift_metrics_cassandra_pvc_size`	El tamaño de los volúmenes que se van a solicitar.
`openshift_metrics_cassandra_storage_type`	El tipo de almacenamiento que se utilizará para las métricas, debe establecerse una dinámica para que Ansible cree RVP con la clase de almacenamiento adecuada.
`openshift_metrics_cassanda_pvc_storage_class_name`	El nombre de la clase de almacenamiento que se va a utilizar.

Implementar el servicio de métricas

Con las variables de Ansible definidas en el archivo de hosts/inventario, ponga en marcha el servicio con Ansible. Si va a implementar en el momento de la instalación de OpenShift, el PV se creará y utilizará automáticamente. Si pone en marcha usando los libros de estrategia de los componentes, después de la instalación de OpenShift, Ansible creará las RVP necesarias y, una vez que Astra Trident ha aprovisionado el almacenamiento para ellos, pondrá en marcha el servicio.

Las variables anteriores y el proceso de implementación pueden cambiar con cada versión de OpenShift.

Asegúrese de revisar y seguir "Guía de implementación de OpenShift de redhat" para su versión de modo que esté configurada para su entorno.

Protección de datos y recuperación ante desastres

Obtén más información sobre las opciones de protección y recuperación para Astra Trident y los volúmenes creados con Astra Trident. Debería tener una estrategia de protección y recuperación de datos para cada aplicación con un requisito de persistencia.

Replicación y recuperación de Astra Trident

Puede crear un backup para restaurar Astra Trident en caso de un desastre.

Replicación de Astra Trident

Astra Trident utiliza CRD de Kubernetes para almacenar y gestionar su propio estado y el clúster ETCD de Kubernetes para almacenar sus metadatos.

Pasos

1. Realice una copia de seguridad del clúster etcd de Kubernetes mediante "Kubernetes: Realizar backups de un clúster etcd".
2. Coloque los artefactos de backup en un FlexVol.

> Le recomendamos que proteja la SVM en la que reside FlexVol con una relación de SnapMirror con otra SVM.

Recuperación de Astra Trident

Con los CRD de Kubernetes y la snapshot etcd del clúster de Kubernetes, puedes recuperar Astra Trident.

Pasos

1. Desde la SVM de destino, monte el volumen que contiene los certificados y archivos de datos ETCD de Kubernetes en el host que se configurará como nodo maestro.
2. Copie todos los certificados necesarios correspondientes al clúster de Kubernetes en `/etc/kubernetes/pki` y los archivos del miembro etcd debajo de `/var/lib/etcd`.
3. Restaure el clúster de Kubernetes desde el backup etcd mediante "Kubernetes: Restaurar un clúster ETCD".
4. Ejecución `kubectl get crd` Para verificar que todos los recursos personalizados de Trident han surgido y recuperado los objetos de Trident para verificar que todos los datos están disponibles.

Replicación y recuperación de SVM

Astra Trident no puede configurar relaciones de replicación; sin embargo, el administrador de almacenamiento puede utilizar "SnapMirror de ONTAP" Para replicar una SVM.

En caso de desastre, puede activar la SVM de destino de SnapMirror para empezar a servir datos. Puede volver al primario cuando se restauran los sistemas.

Acerca de esta tarea

Tenga en cuenta lo siguiente al usar la función de replicación de SVM de SnapMirror:

- Debe crear un back-end distinto para cada SVM con la función SVM-DR habilitada.
- Configure las clases de almacenamiento para seleccionar los back-ends replicados solo cuando sea necesario para evitar tener volúmenes que no necesitan replicación aprovisionados en los back-ends que admitan SVM-DR.
- Los administradores de aplicaciones deben comprender el coste y la complejidad adicionales asociados con la replicación y estudiar detenidamente su plan de recuperación antes de iniciar este proceso.

Replicación de SVM

Puede utilizar "ONTAP: Replicación de SnapMirror SVM" Para crear la relación de replicación de SVM.

SnapMirror le permite configurar opciones para controlar lo que se va a replicar. Necesitará saber qué opciones seleccionó al realizar la preformación Recuperación de SVM mediante Astra Trident.

- "-identity-preserve true" Replica toda la configuración de la SVM.
- "-descarte-configs red" Excluye las LIF y la configuración de red relacionada.
- "-identity-preserve false" replica solo los volúmenes y la configuración de seguridad.

Recuperación de SVM mediante Astra Trident

Astra Trident no detecta automáticamente fallos de SVM. En caso de desastre, el administrador puede iniciar manualmente la conmutación por error de Trident en la nueva SVM.

Pasos

1. Cancelar las transferencias programadas y continuas de SnapMirror, interrumpir la relación de replicación, detener la SVM de origen y, a continuación, activar la SVM de destino de SnapMirror.
2. Si especificó `-identity-preserve false` o. `-discard-config network` Al configurar la replicación de SVM, actualice el `managementLIF` y.. `dataLIF` En el archivo de definición de back-end de Trident.
3. Confirme `storagePrefix` Está presente en el archivo de definición de back-end de Trident. Este parámetro no puede cambiarse. Omitiendo `storagePrefix` provocará que la actualización de backend falle.
4. Actualice todos los back-ends requeridos para reflejar el nuevo nombre de la SVM de destino mediante:

```
./tridentctl update backend <backend-name> -f <backend-json-file> -n
<namespace>
```

5. Si especificó `-identity-preserve false` o. `discard-config network`, debe rebotar todos los pods de aplicación.

> Si especificó `-identity-preserve true`, Todos los volúmenes aprovisionados por Astra Trident comienzan a servir datos cuando se activa la SVM de destino.

Replicación y recuperación de volúmenes

Astra Trident no puede configurar las relaciones de replicación de SnapMirror; sin embargo, el administrador de almacenamiento puede utilizar "Replicación y recuperación SnapMirror de ONTAP" Para replicar volúmenes creados por Astra Trident.

Luego, podrá importar los volúmenes recuperados a Astra Trident mediante "importación de volumen tridentctl".

> La importación no es compatible con `ontap-nas-economy`, `ontap-san-economy`, o. `ontap-flexgroup-economy` de windows

Protección de datos Snapshot

Puede proteger y restaurar datos con:

- Un controlador snapshot externo y CRD para crear snapshots de volúmenes de Kubernetes de volúmenes persistentes (VP).

  "Copias de Snapshot de volumen"

- Snapshots de ONTAP para restaurar el contenido completo de un volumen o para recuperar archivos o LUN individuales.

  "Snapshots de ONTAP"

Replicación de aplicaciones de Astra Control Center

Con Astra Control, puede replicar datos y cambios de aplicaciones de un clúster a otro mediante las funcionalidades de replicación asíncrona de SnapMirror.

"Astra Control: Replique aplicaciones en un sistema remoto mediante la tecnología SnapMirror"

Seguridad

Seguridad

Utilice las recomendaciones que se enumeran aquí para asegurarse de que su instalación de Astra Trident es segura.

Ejecute Astra Trident en su propio espacio de nombres

Es importante evitar que las aplicaciones, los administradores de aplicaciones, los usuarios y las aplicaciones de gestión accedan a las definiciones de objetos de Astra Trident o a los pods para garantizar un almacenamiento fiable y bloquear la potencial actividad maliciosa.

Para separar el resto de aplicaciones y usuarios de Astra Trident, instale siempre Astra Trident en su propio espacio de nombres Kubernetes (`trident`). Si coloca Astra Trident en su propio espacio de nombres, solo el personal administrativo de Kubernetes tiene acceso al pod de la Astra Trident y los artefactos (como los secretos CHAP y de back-end, si corresponde) almacenados en los objetos de CRD named.
Debe asegurarse de permitir que solo los administradores tengan acceso al espacio de nombres de Astra Trident y, por lo tanto, tengan acceso a `tridentctl` cliente más.

Utilice la autenticación CHAP con los back-ends DE SAN de ONTAP

Astra Trident admite la autenticación basada en CHAP para las cargas de trabajo SAN de ONTAP (mediante el `ontap-san` y.. `ontap-san-economy` de windows). NetApp recomienda utilizar CHAP bidireccional con Astra Trident para la autenticación entre un host y el back-end de almacenamiento.

En el caso de los back-ends de ONTAP que utilizan controladores de almacenamiento SAN, Astra Trident puede configurar CHAP bidireccional y gestionar los nombres de usuario y los secretos CHAP a través de `tridentctl`.
Consulte "aquí" Para comprender cómo Astra Trident configura CHAP en los back-ends de ONTAP.

> La compatibilidad CON CHAP para los back-ends de ONTAP está disponible con Trident 20.04 y versiones posteriores.

Utilice la autenticación CHAP con NetApp HCI y back-ends de SolidFire

NetApp recomienda poner en marcha CHAP bidireccional para garantizar la autenticación entre un host y los back-ends de NetApp HCI y SolidFire. Astra Trident utiliza un objeto secreto que incluye dos contraseñas CHAP por inquilino. Cuando Trident se instala como aprovisionador CSI, gestiona los secretos CHAP y los almacena en un `tridentvolume` Objeto CR para el PV correspondiente. Cuando se crea un VP, CSI Astra Trident utiliza los secretos CHAP para iniciar una sesión iSCSI y comunicarse con el sistema NetApp HCI y SolidFire a través de CHAP.

> Los volúmenes creados por CSI Trident no están asociados con ningún grupo de acceso de volúmenes.

En el frontend que no sea CSI, Kubernetes gestiona la conexión de volúmenes como dispositivos en los nodos de trabajo. Tras crear un volumen, Astra Trident realiza una llamada API al sistema HCI/SolidFire de NetApp para recuperar los secretos si ese secreto no existe ya. A continuación, Astra Trident pasa los secretos a Kubernetes. La kuelet que se encuentra en cada nodo accede a los secretos a través de la API de Kubernetes y los utiliza para ejecutar y habilitar CHAP entre cada nodo que accede al volumen y el sistema HCI/SolidFire de NetApp donde están ubicados los volúmenes.

Utilice Astra Trident con NVE y NAE

ONTAP de NetApp proporciona cifrado de datos en reposo para proteger los datos confidenciales en el caso de robo, devolución o reasignación de un disco. Para obtener más información, consulte "Configure la información general de cifrado de volúmenes de NetApp".

- Si NAE está habilitado en el back-end, cualquier volumen aprovisionado en Astra Trident se habilitará para NAE.
- Si NAE no está habilitado en el back-end, todos los volúmenes aprovisionados en Astra Trident tendrán el cifrado NVE habilitado a menos que establezca el indicador NVE en `false` en la configuración de back-end.

Los volúmenes que se crean en Astra Trident en un back-end con la NAE habilitada deben ser NVE o NAE cifrados.

- Puede establecer el indicador NVE Encryption como `true` En la configuración del back-end de Trident, con el fin de anular el cifrado NAE y utilizar una clave de cifrado específica por volumen.
- Establecer la Marca NVE Encryption como `false` En un back-end habilitado para NAE se creará un volumen habilitado para NAE. No puede deshabilitar el cifrado NAE mediante la Marca NVE Encryption a. `false`.

- Es posible crear manualmente un volumen NVE en Astra Trident mediante la definición explícita de la Marca NVE a. `true`.

Para obtener más información sobre las opciones de configuración del back-end, consulte:

- "Opciones de configuración DE SAN de ONTAP"
- "Opciones de configuración de NAS de ONTAP"

Configuración de clave unificada de Linux (LUKS)

Puede habilitar Unified Key Setup (LUKS) de Linux para cifrar los volúmenes DE ECONOMÍA SAN de ONTAP y SAN DE ONTAP en Astra Trident. Astra Trident admite la rotación de claves de acceso y la expansión de volumen para volúmenes cifrados con LUKS.

En Astra Trident, los volúmenes cifrados por LUKS utilizan el cifer y el modo AES-xts-Capellanía, como recomienda "NIST".

Antes de empezar

- Los nodos de trabajo deben tener instalado cryptsetup 2.1 o superior (pero inferior a 3.0). Si desea más información, visite "Gitlab: Cryptsetup".
- Por motivos de rendimiento, recomendamos que los nodos de trabajo admitan las nuevas instrucciones estándar de cifrado avanzado (AES-ni). Para verificar el soporte de AES-ni, ejecute el siguiente comando:

```
grep "aes" /proc/cpuinfo
```

Si no se devuelve nada, su procesador no admite AES-ni. Para obtener más información sobre AES-ni, visite: "Intel: Instrucciones estándar de cifrado avanzado (AES-ni)".

Active el cifrado LUKS

Puede habilitar el cifrado por volumen en el lado del host usando la configuración de clave unificada de Linux (LUKS) para SAN de ONTAP y volúmenes DE ECONOMÍA SAN de ONTAP.

Pasos

1. Defina los atributos de cifrado LUKS en la configuración de backend. Para obtener más información sobre las opciones de configuración del back-end para SAN de ONTAP, consulte "Opciones de configuración DE SAN de ONTAP".

```
"storage": [
    {
        "labels":{"luks": "true"},
        "zone":"us_east_1a",
        "defaults": {
            "luksEncryption": "true"
        }
    },
    {
        "labels":{"luks": "false"},
        "zone":"us_east_1a",
        "defaults": {
            "luksEncryption": "false"
        }
    },
]
```

2. Uso `parameters.selector` Para definir los pools de almacenamiento mediante el cifrado LUKS. Por ejemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: netapp.io/trident
parameters:
  selector: "luks=true"
  csi.storage.k8s.io/node-stage-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

3. Cree un secreto que contenga la frase de paso LUKS. Por ejemplo:

```
kubectl -n trident create -f luks-pvc1.yaml
apiVersion: v1
kind: Secret
metadata:
  name: luks-pvc1
stringData:
  luks-passphrase-name: A
  luks-passphrase: secretA
```

Limitaciones

Los volúmenes cifrados LUKS no pueden aprovechar la deduplicación y la compresión de ONTAP.

Configuración de backend para importar volúmenes LUKS

Para importar un volumen LUKS, debe establecer `luksEncryption` para(`true` en el backend. La `luksEncryption` La opción indica a Astra Trident si el volumen cumple con la normativa LUKS (`true`) O no cumple con LUKS (`false`) como se muestra en el siguiente ejemplo.

```
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: trident_svm
username: admin
password: password
defaults:
  luksEncryption: 'true'
  spaceAllocation: 'false'
  snapshotPolicy: default
  snapshotReserve: '10'
```

Gire una frase de paso LUKS

Puede girar la frase de paso de LUKS y confirmar la rotación.

> No olvide una clave de acceso hasta que haya verificado que ya no hace referencia a ningún volumen, snapshot o secreto. Si se pierde una clave de acceso de referencia, es posible que no se pueda montar el volumen y los datos seguirán estando cifrados e inaccesibles.

Acerca de esta tarea

LA rotación DE la frase de paso LUKS se produce cuando se crea un pod que monta el volumen después de especificar una nueva frase de paso LUKS. Cuando se crea un nuevo pod, Astra Trident compara la frase de paso de LUKS del volumen con la frase de paso activa en el secreto.

- Si la clave de acceso del volumen no coincide con la clave de acceso activa en el secreto, se produce la rotación.
- Si la clave de acceso del volumen coincide con la clave de acceso activa en el secreto, el `previous-luks-passphrase` se ignora el parámetro.

Pasos

1. Añada el `node-publish-secret-name` y.. `node-publish-secret-namespace` Parámetros de StorageClass. Por ejemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-san
provisioner: csi.trident.netapp.io
parameters:
  trident.netapp.io/backendType: "ontap-san"
  csi.storage.k8s.io/node-stage-secret-name: luks
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
  csi.storage.k8s.io/node-publish-secret-name: luks
  csi.storage.k8s.io/node-publish-secret-namespace: ${pvc.namespace}
```

2. Identifique las bases de datos passhrases existentes en el volumen o la snapshot.

Volumen

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames:["A"]
```

Snapshot

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames:["A"]
```

3. Actualice el secreto LUKS del volumen para especificar las passphrases nuevas y anteriores. Asegúrese `previous-luke-passphrase-name` y.. `previous-luks-passphrase` coincidir con la frase de contraseña anterior.

```
apiVersion: v1
kind: Secret
metadata:
  name: luks-pvc1
stringData:
  luks-passphrase-name: B
  luks-passphrase: secretB
  previous-luks-passphrase-name: A
  previous-luks-passphrase: secretA
```

4. Cree un nuevo pod montando el volumen. Esto es necesario para iniciar la rotación.
5. Compruebe que se ha girado la frase de paso.

Volumen

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames:["B"]
```

Snapshot

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames:["B"]
```

Resultados

La frase de contraseña se giró cuando solo se devuelve la nueva frase de contraseña en el volumen y la instantánea.

> Si se devuelven dos passphrasas, por ejemplo `luksPassphraseNames: ["B", "A"]`, la rotación está incompleta. Puede activar un nuevo pod para intentar completar la rotación.

Habilite la expansión de volumen

Es posible habilitar la ampliación de volumen en un volumen cifrado LUKS.

Pasos

1. Habilite el `CSINodeExpandSecret` puerta de características (beta 1.25+). Consulte "Kubernetes 1.25: Use Secrets for Node-Driven Expansion of CSI Volumes" para obtener más detalles.
2. Añada el `node-expand-secret-name` y.. `node-expand-secret-namespace` Parámetros de StorageClass. Por ejemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: netapp.io/trident
parameters:
  selector: "luks=true"
  csi.storage.k8s.io/node-stage-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
  csi.storage.k8s.io/node-expand-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-expand-secret-namespace: ${pvc.namespace}
allowVolumeExpansion: true
```

Resultados

Al iniciar la ampliación de almacenamiento en línea, el kubelet pasa las credenciales adecuadas al controlador.

Referencia

Puertos Astra Trident

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

Puertos Astra Trident

Astra Trident se comunica mediante 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 nivel de gravedad/preparación se puede cambiar durante la instalación utilizando el `--probe-port` bandera. Es importante asegurarse de que este puerto no esté siendo utilizado por otro proceso en los nodos de trabajo.

API DE REST de Astra Trident

Aunque "comandos y opciones de trimentctl" Son la forma más sencilla de interactuar con la API REST de Astra Trident, puede utilizar el extremo REST directamente si lo prefiere.

Cuándo utilizar la API DE REST

La API REST es útil en las instalaciones avanzadas que usan Astra Trident como binario independiente en las puestas en marcha sin Kubernetes.

Para una mayor seguridad, la Astra Trident `REST API` se restringe a localhost de forma predeterminada cuando se ejecuta dentro de un pod. Para cambiar este comportamiento, debe configurar Astra Trident's. `-address` en su configuración del pod.

Uso de la API DE REST

La API funciona de la siguiente manera:

`GET`

- `GET <trident-address>/trident/v1/<object-type>`: Enumera todos los objetos de ese tipo.
- `GET <trident-address>/trident/v1/<object-type>/<object-name>`: Obtiene los detalles del objeto con nombre.

`POST`

`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 obtener información sobre la especificación de cada tipo de objeto, consulte LINK:tridentctl.html[`tridentctl` comandos y opciones].
- 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.

`DELETE`

`DELETE <trident-address>/trident/v1/<object-type>/<object-name>`: Elimina 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 el enlace:tridentctl.html[`tridentctl` comandos y opciones].

Para obtener ejemplos de cómo se llama a estas API, pase la depuración (`-d`) bandera. Para obtener más información, consulte el enlace:tridentctl.html[`tridentctl` comandos y opciones].

Opciones de línea de comandos

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

Registro

- `-debug`: Habilita 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 la compatibilidad con 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 la compatibilidad con 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 ponga en marcha fuera de un pod; sin embargo, solo admite conexiones no seguras con el servidor API. Para conectarse con seguridad, implemente Trident en un pod con el `-k8s_pod` opción.
- `-k8s_config_path <file>`: Necesario; debe especificar esta ruta de acceso a un archivo KubeConfig.

Docker

- `-volume_driver <name>`: Nombre del controlador utilizado al registrar el complemento Docker. De forma predeterminada es `netapp`.

- `-driver_port <port-number>`: Escucha en este puerto en lugar de un socket de dominio UNIX.
- `-config <file>`: Necesario; debe especificar esta ruta de acceso a un archivo de configuración de back-end.

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. Uso `-address ""` Para hacer que la interfaz DE 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 escuchar el servidor REST de Trident. El valor predeterminado es 8000.
- `-rest`: Activa la interfaz DE REPOSO. El valor predeterminado es TRUE.

Los productos de NetApp están integrados con Kubernetes

La cartera de productos de almacenamiento de NetApp se integra con muchos aspectos diferentes de un clúster de Kubernetes, por lo que proporciona funcionalidades de gestión de datos avanzadas que mejoran la funcionalidad, la funcionalidad, el rendimiento y la disponibilidad de la puesta en marcha de Kubernetes.

Astra

"Astra" Facilita a las empresas la gestión, protección y movimiento de sus cargas de trabajo en contenedores con gran cantidad de datos que se ejecutan en Kubernetes en los clouds públicos y en las instalaciones. Astra aprovisiona y proporciona un almacenamiento en contenedores persistente mediante Trident de la cartera de almacenamiento probada y amplia de NetApp en el cloud público y en las instalaciones. También ofrece un conjunto amplio de funcionalidades avanzadas de gestión de datos para aplicaciones, como snapshots, backups y restauración, registros de actividades y clonado activo para la protección de datos, recuperación ante desastres/datos, auditoría de datos y casos de uso de migración para cargas de trabajo de Kubernetes.

ONTAP

ONTAP es el sistema operativo de almacenamiento unificado multiprotocolo de NetApp que proporciona funcionalidades avanzadas de gestión de datos para cualquier aplicación. Los sistemas ONTAP tienen configuraciones all-flash, híbridas o all-HDD y ofrecen muchos modelos de puesta en marcha diferentes, como hardware a medida (FAS y AFF), unidad genérica (ONTAP Select) y solo cloud (Cloud Volumes ONTAP).

> Trident es compatible con todos los modelos de puesta en marcha de ONTAP mencionados anteriormente.

Cloud Volumes ONTAP

"Cloud Volumes ONTAP" Es un dispositivo de almacenamiento exclusivamente de software que ejecuta el software para la gestión de datos ONTAP en el cloud. Puede utilizar Cloud Volumes ONTAP para cargas de

trabajo de producción, recuperación ante desastres, DevOps, recursos compartidos de archivos y gestión de bases de datos. Amplía el almacenamiento empresarial al cloud ofreciendo eficiencias del almacenamiento, alta disponibilidad, replicación de datos, organización en niveles de los datos y consistencia de las aplicaciones.

Amazon FSX para ONTAP de NetApp

"Amazon FSX para ONTAP de NetApp" Es un servicio AWS totalmente gestionado que permite a los clientes iniciar y ejecutar sistemas de archivos con tecnología del sistema operativo de almacenamiento ONTAP de NetApp. FSX para ONTAP permite a los clientes aprovechar las funciones, el rendimiento y las funcionalidades administrativas de NetApp que ya conocen y, al mismo tiempo, aprovechar la simplicidad, la agilidad, la seguridad y la escalabilidad del almacenamiento de datos en AWS. FSX para ONTAP es compatible con muchas de las API de administración y las funciones del sistema de archivos de ONTAP.

Software Element

"Elemento" permite al administrador de almacenamiento consolidar cargas de trabajo garantizando el rendimiento y haciendo posible un espacio de almacenamiento simplificado y optimizado. Junto con una API para permitir la automatización de todos los aspectos de la gestión del almacenamiento, Element permite a los administradores de almacenamiento hacer más con menos esfuerzo.

NetApp HCI

"NetApp HCI" simplifica la gestión y el escalado del centro de datos mediante la automatización de las tareas rutinarias y permite que los administradores de la infraestructura se centren en funciones más importantes.

Trident es totalmente compatible con NetApp HCI. Trident puede aprovisionar y gestionar dispositivos de almacenamiento para aplicaciones en contenedores directamente en la plataforma de almacenamiento subyacente de NetApp HCI.

Azure NetApp Files

"Azure NetApp Files" Es un servicio de recursos compartidos de archivos de Azure de clase empresarial con la tecnología de NetApp. Puede ejecutar sus cargas de trabajo basadas en archivos más exigentes de forma nativa en Azure, con el rendimiento y la gestión de datos enriquecidos que espera de NetApp.

Cloud Volumes Service para Google Cloud

"Cloud Volumes Service de NetApp para Google Cloud" Es un servicio de archivos nativo del cloud que proporciona volúmenes de NAS en NFS y SMB con rendimiento all-flash. Este servicio permite que se ejecute cualquier carga de trabajo, incluidas las aplicaciones heredadas, en la nube de GCP. Proporciona un servicio totalmente gestionado que ofrece alto rendimiento consistente, clonado instantáneo, protección de datos y acceso seguro a instancias de Google Compute Engine (GCE).

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` solicitando un nuevo `PersistentVolume` De un tamaño concreto de un Kubernetes `StorageClass` previamente configurado por el administrador.
2. Kubernetes `StorageClass` Identifica a Trident como su aprovisionador y incluye los parámetros que indican a Trident cómo aprovisionar un volumen para la clase solicitada.
3. Trident analiza sus propios recursos `StorageClass` con el mismo nombre que identifica la coincidencia `Backends` y.. `StoragePools` que puede usar para aprovisionar volúmenes para la clase.
4. Trident aprovisiona el almacenamiento en un back-end coincidente y crea dos objetos: Un `PersistentVolume` En Kubernetes, donde se indica cómo encontrar, montar y tratar el volumen, y un volumen en Trident que conserva la relación entre `PersistentVolume` y el almacenamiento real.
5. Kubernetes enlaza con el `PersistentVolumeClaim` a los nuevos `PersistentVolume`. Pods que incluyen `PersistentVolumeClaim` monte ese volumen persistente en cualquier host en el que se ejecute.
6. Un usuario crea un `VolumeSnapshot` De un PVC existente, utilizando un `VolumeSnapshotClass` Eso es lo 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` Esto 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 conjunto de pasos involucrados en la creación de un `PersistentVolume` y un `Volume`.

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

Kubernetes `PersistentVolumeClaim` objetos

Un Kubernetes `PersistentVolumeClaim` El objeto es una solicitud de almacenamiento que realiza un usuario de 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

Anotación	Opción de volumen	Controladores compatibles
trident.netapp.io/exportPolicy	Política de exportoPolicy	ontap-nas ontap-nas-economy, 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, ontap-nas-flexgroup
trident.netapp.io/unixPermissions	Permisos univalados	ontap-nas ontap-nas-economy, ontap-nas-flexgroup
trident.netapp.io/blockSize	Tamaño del bloque	solidfire-san

Si el VP creado tiene el `Delete` Reclamar política, 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 utiliza `Retain` Política, Trident ignora la operación y asume que el administrador la limpiará desde Kubernetes y el back-end, lo que permitirá realizar un backup o la inspección del volumen antes de su eliminación. Tenga en cuenta que al eliminar el VP, Trident no eliminará el volumen de backup. Debe quitarlo usando 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. Eche un vistazo `On-Demand Volume Snapshots` para ver cómo funcionaría.

Trident también proporciona la `cloneFromPVC` y.. `splitOnClone` anotaciones para crear clones. Puede usar estas anotaciones para clonar una RVP sin tener que utilizar la implementación de CSI (en Kubernetes 1,13 y versiones anteriores), o bien si la versión de Kubernetes no es compatible con snapshots de volúmenes beta (Kubernetes 1,16 y versiones anteriores). Tenga en cuenta que Trident 19.10 admite el flujo de trabajo CSI para clonar desde un PVC.

Puede utilizar el `cloneFromPVC` y.. `splitOnClone` Anotaciones con CSI Trident así como el frontend tradicional no CSI.

A continuación se muestra un ejemplo: Si un usuario ya tiene una RVP llamada `mysql`, El usuario puede crear un nuevo PVC llamado `mysqlclone` mediante la anotación, por ejemplo `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 la `ontap-nas` y.. `ontap-san` Controladores, es posible que sea conveniente establecer la anotación de PVC `trident.netapp.io/splitOnClone` en conjunto con `trident.netapp.io/cloneFromPVC`. Con `trident.netapp.io/splitOnClone` establezca en `true`, Trident divide el volumen clonado del volumen principal y, por lo tanto, separa completamente el ciclo de vida del volumen clonado de su principal a costa de perder alguna eficiencia de almacenamiento. No está configurado `trident.netapp.io/splitOnClone` o establecerlo en `false` provoca una reducción del consumo de espacio en el back-end a costa de crear dependencias entre los volúmenes principal y clonado, de modo que no se pueda eliminar el volumen principal, 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.

La `sample-input` el directorio contiene ejemplos de definiciones de PVC para utilizarlas con Trident. Consulte los objetos de Trident Volume para obtener una descripción completa de los parámetros y la configuración asociados con Trident Volumes.

Kubernetes `PersistentVolume` objetos

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

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

Cuando se crea una RVP que hace referencia a un sistema basado en Trident `StorageClass`, Trident aprovisiona un nuevo volumen utilizando 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.

Kubernetes `StorageClass` objetos

Kubernetes `StorageClass` los objetos se especifican por nombre en `PersistentVolumeClaims` para aprovisionar el almacenamiento con una serie 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 Kubernetes `StorageClass` Objeto 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
Pools de almacenamiento	Map[string]StringList	no	Mapa de nombres de backend a listas de pools de almacenamiento dentro
AdicionalStoragePools	Map[string]StringList	no	Mapa de nombres de backend a listas de pools de almacenamiento de
ExcludeStoragePools	Map[string]StringList	no	Asignación de nombres de backend a. listas de pools de almacenamiento en

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

Atributo	Tipo	Valores	Oferta	Solicitud	Admitido por
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
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 es que pueda usar `attributes` solo para modelar las cualidades del almacenamiento que necesita para satisfacer las necesidades de una clase particular. Trident detecta y selecciona automáticamente pools de almacenamiento que coincidan *all* del `attributes` que especifique.

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

Puede utilizar el `storagePools` el parámetro para restringir aún más el conjunto de pools que coinciden con cualquier especificado `attributes`. En otras palabras, Trident utiliza la intersección de pools identificados por el `attributes` y.. `storagePools` parámetros 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 cualquier pool que seleccione `attributes` y.. `storagePools` parámetros.

Puede utilizar 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 la `storagePools` y.. `additionalStoragePools` parámetros, cada entrada toma el formulario `<backend>:<storagePoolList>`, donde `<storagePoolList>` es una lista de pools de almacenamiento separados por comas para el back-end especificado. Por ejemplo, un valor para `additionalStoragePools` puede parecer `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 los 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	Kubernetes Versión
Tipo fstype	cadena	ext4, ext3, xfs, etc.	Tipo de sistema de archivos para el bloque volúmenes	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

- La `fsType` El parámetro se utiliza para controlar el tipo de sistema de archivos deseado para las LUN DE SAN. Además, Kubernetes utiliza también 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 la `fsGroup` contexto de seguridad de un pod solo if `fsType` está configurado. Consulte "Kubernetes: Configure un contexto de seguridad para un Pod o contenedor" para obtener información general sobre la configuración de la propiedad del volumen con `fsGroup` contexto. Kubernetes aplicará el `fsGroup` valor solo si:

  - `fsType` se establece 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 aún debe especificar un `fsType`. Puede configurarlo 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 usar con Trident en `sample-input/storage-class-*.yaml`. Al eliminar una clase de almacenamiento Kubernetes, también se elimina el tipo de almacenamiento Trident correspondiente.

Kubernetes `VolumeSnapshotClass` objetos

Kubernetes `VolumeSnapshotClass` los objetos son similares `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.

1. `VolumeSnapshotClass` debe ser definido por un administrador para crear snapshots. 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
```

La `driver` Especifica a Kubernetes que solicitudes de snapshots de volumen del `csi-snapclass` Trident gestiona la clase. La `deletionPolicy` especifica la acción que se debe realizar cuando se debe eliminar

una instantánea. Cuando `deletionPolicy` se establece en `Delete`, los objetos de instantánea del volumen, así como la instantánea subyacente en el clúster de almacenamiento, se eliminan cuando se elimina una instantánea. Como alternativa, establecerlo en `Retain` significa eso `VolumeSnapshotContent` y se conserva la snapshot física.

Kubernetes `VolumeSnapshot` objetos

Un Kubernetes `VolumeSnapshot` objeto es una solicitud para crear una copia de 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 llega una solicitud Snapshot de volumen, Trident gestiona automáticamente la creación de la snapshot para el volumen en el back-end y expone la snapshot creando un único `VolumeSnapshotContent` objeto. 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.

Kubernetes `VolumeSnapshotContent` objetos

Un Kubernetes `VolumeSnapshotContent` object representa una snapshot tomada de un volumen ya aprovisionado. Es similar a un `PersistentVolume` y significa una instantánea aprovisionada en el clúster de almacenamiento. Similar a. `PersistentVolumeClaim` y.. `PersistentVolume` los objetos, cuando se crea una snapshot, el `VolumeSnapshotContent` object mantiene una asignación de uno a uno `VolumeSnapshot` objeto, que solicitó la creación de la snapshot.

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

La `VolumeSnapshotContent` el objeto contiene detalles que identifican de manera única la instantánea, como la `snapshotHandle`. Este `snapshotHandle` Es una combinación única del nombre del PV y el nombre del `VolumeSnapshotContent` objeto.

Cuando llega una solicitud de Snapshot, Trident crea la snapshot en el back-end. Una vez creada la copia de Snapshot, Trident configura un `VolumeSnapshotContent` Objeto y, por lo tanto, expone la snapshot a la API de Kubernetes.

Kubernetes `CustomResourceDefinition` objetos

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.

A partir del lanzamiento de la versión 19.07, Trident utiliza una serie de `CustomResourceDefinition` Objetos que conservan la identidad de objetos de Trident, como los back-ends de Trident, las clases de almacenamiento de 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 simple, cuando se crea un back-end usando `tridentctl`, a correspondiente `tridentbackends` El objeto CRD se crea para el consumo por parte de 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 actualizar desde una versión anterior de Trident (una que utilizó `etcd` Para mantener el estado), el instalador de Trident migra los datos del `etcd` Almacén de datos clave-valor y crea los objetos CRD correspondientes.
- Al desinstalar Trident mediante la `tridentctl uninstall` Comando, los pods de Trident se eliminan, pero los CRD creados no se borran. Consulte "Desinstale Trident" Para comprender cómo Trident se puede eliminar por completo y volver a configurar desde cero.

Trident `StorageClass` objetos

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

> Con Kubernetes, estos objetos se crean automáticamente cuando se crea un Kubernetes `StorageClass` Que usa Trident como aprovisionador está registrado.

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, en el caso de las puestas en marcha de Kubernetes, esperamos que se creen al registrar el nuevo Kubernetes `StorageClass` objetos.

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 Kubernetes `StorageClass` objeto.

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

Trident `StoragePool` objetos

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.

Trident `Volume` objetos

Los volúmenes son la unidad básica de aprovisionamiento y constan de extremos back-end, como recursos compartidos de NFS y LUN iSCSI. En Kubernetes, se corresponden directamente con `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

Atributo	Tipo	Obligatorio	Descripción
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
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

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

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

Trident `Snapshot` objetos

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, se corresponden directamente con `VolumeSnapshotContent` objetos. Cada copia de Snapshot se asocia con un volumen, que es el origen de los datos de la copia de Snapshot.

Cada uno `Snapshot` object incluye las propiedades que se enumeran 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
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 un Kubernetes `VolumeSnapshot` Se crea la solicitud del objeto, Trident funciona mediante la creación de un objeto Snapshot en el sistema de almacenamiento que realiza backups. La `internalName` de este objeto snapshot se genera combinando el prefijo `snapshot-` con la `UID` de la `VolumeSnapshot` objeto (por ejemplo, `snapshot-e8d8a0ca-9826-11e9-9807-525400f3f660`). `volumeName` y.. `volumeInternalName` se rellenan obteniendo los detalles del respaldo volumen.

Astra Trident `ResourceQuota` objeto

El inicio de Trident consume un `system-node-critical` Clase de prioridad, la clase de prioridad más alta disponible en Kubernetes, para garantizar que Astra Trident pueda identificar y limpiar volúmenes durante un apagado correcto de nodos y permitir que Trident demonset pods prevea las cargas de trabajo con una prioridad menor en clústeres donde hay una alta presión en los recursos.

Para conseguirlo, Astra Trident utiliza una `ResourceQuota` Objeto garantizar que se cumple una clase prioritaria "system-node-Critical" en el demonset de Trident. Antes de la puesta en marcha y la creación de demonset, Astra Trident busca la `ResourceQuota` object y, si no se detecta, lo aplica.

Si necesita más control sobre la cuota de recursos predeterminada y la clase de prioridad, puede generar una `custom.yaml` o configure 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 acerca de las cuotas de recursos, consulte "Kubernetes: Cuotas de recursos".

Limpie `ResourceQuota` si la instalación falla

En el raro caso en que la instalación falle después del `ResourceQuota` se crea el objeto, primero se intenta "desinstalando" y, a continuación, vuelva a instalar.

Si esto no funciona, quite manualmente la `ResourceQuota` objeto.

Quitar `ResourceQuota`

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

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

comandos y opciones de trimentctl

La "Paquete de instalación de Trident" incluye una utilidad de línea de comandos, `tridentctl`, Que proporciona un acceso sencillo a Astra Trident. Los usuarios de Kubernetes con suficientes privilegios pueden usarlo para instalar Astra Trident y también para interactuar con ella directamente para gestionar el espacio de nombres que contiene el pod Astra Trident.

Comandos y opciones disponibles

Para obtener información de uso, ejecute `tridentctl --help`.

Los comandos disponibles y las opciones globales son:

```
Usage:
  tridentctl [command]
```

Comandos disponibles:

- `create`: Añadir un recurso a Astra Trident.
- `delete`: Elimine uno o más recursos de Astra Trident.
- `get`: Obtenga uno o más recursos de Astra Trident.
- `help`: Ayuda sobre cualquier comando.
- `images`: Imprime una tabla de las imágenes de contenedores que necesita Astra Trident.
- `import`: Importe un recurso existente a Astra Trident.
- `install`: Instalar Astra Trident.
- `logs`: Imprime los registros de Astra Trident.
- `send`: Envíe un recurso desde Astra Trident.
- `uninstall`: Desinstalar Astra Trident.
- `update`: Modificar un recurso en Astra Trident.
- `upgrade`: Actualizar un recurso en Astra Trident.
- `version`: Imprime la versión de Astra Trident.

Indicadores:

- `-d, --debug`: Salida de depuración.
- `-h, --help`: Ayuda para `tridentctl`.
- `-n, --namespace string`: Espacio de nombres de la implementación de Astra Trident.
- `-o, --output string`: Formato de salida. Uno de json|yaml|name|Wide|ps (predeterminado).
- `-s, --server string`: Dirección/puerto de la interfaz REST de Astra Trident.

> 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).

> 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).

`create`

Puede utilizar ejecutar el `create` Comando para añadir un recurso a Astra Trident.

```
Usage:
  tridentctl create [option]
```

Opción disponible:
`backend`: Añadir un back-end a Astra Trident.

delete

Puede ejecutar el `delete` Comando para eliminar uno o más recursos de Astra Trident.

```
Usage:
  tridentctl delete [option]
```

Opciones disponibles:

- `backend`: Elimine uno o más back-ends de almacenamiento de Astra Trident.
- `snapshot`: Elimine una o más instantáneas de volumen de Astra Trident.
- `storageclass`: Elimine una o varias clases de almacenamiento de Astra Trident.
- `volume`: Elimine uno o varios volúmenes de almacenamiento de Astra Trident.

get

Puede ejecutar el `get` Comando para obtener uno o más recursos de Astra Trident.

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

Opciones disponibles:

- `backend`: Obtenga uno o más back-ends de almacenamiento de Astra Trident.
- `snapshot`: Obtiene una o más instantáneas de Astra Trident.
- `storageclass`: Obtenga una o más clases de almacenamiento de Astra Trident.
- `volume`: Obtenga uno o más volúmenes de Astra Trident.

`volume` indicadores:
* `-h, --help`: Ayuda para volúmenes.
* `--parentOfSubordinate string`: Limite la consulta al volumen de origen subordinado.
* `--subordinateOf string`: Limite la consulta a las subordinadas del volumen.

images

Puede ejecutar el `images` Indicador para imprimir una tabla de las imágenes de contenedor que necesita

Astra Trident.

```
Usage:
  tridentctl images [flags]
```

Indicadores:
* `-h, --help`: Help for images.
* -V, --k8s-version string': Versión semántica del cluster de Kubernetes.

import volume

Puede ejecutar el `import volume` Comando para importar un volumen existente a Astra Trident.

```
Usage:
  tridentctl import volume <backendName> <volumeName> [flags]
```

Alias:
`volume, v`

Indicadores:

- `-f, --filename string`: Ruta al archivo YLMA o JSON PVC.
- `-h, --help`: Ayuda para el volumen.
- `--no-manage`: Cree sólo PV/PVC. No asuma que se gestiona el ciclo de vida de los volúmenes.

install

Puede ejecutar el `install` Banderas para instalar Astra Trident.

```
Usage:
  tridentctl install [flags]
```

Indicadores:

- `--autosupport-image string`: La imagen contenedora del sistema de telemetría AutoSupport (valor predeterminado: "netapp/trident autosupport:20.07.0").
- `--autosupport-proxy string`: La dirección/puerto de un proxy para enviar telemetría AutoSupport.
- `--csi`: Instalar CSI Trident (reemplazar sólo para Kubernetes 1.13, requiere puertas de funciones).
- `--enable-node-prep`: Intente instalar los paquetes necesarios en los nodos.
- `--generate-custom-yaml`: Genere archivos YAML sin instalar nada.
- `-h, --help`: Ayuda para instalar.
- `--http-request-timeout`: Anule el tiempo de espera de la solicitud HTTP para la API DE REST de

la controladora Trident (por defecto 1m30s).

- `--image-registry string`: La dirección/puerto de un registro de imagen interna.
- `--k8s-timeout duration`: El tiempo de espera para todas las operaciones de Kubernetes (por defecto 3 m0s).
- `--kubelet-dir string`: La ubicación del host del estado interno de Kubelet (predeterminado "/var/lib/kubelet").
- `--log-format string`: El formato de registro de Astra Trident (texto, json) (por defecto "text").
- `--pv string`: El nombre del PV heredado utilizado por Astra Trident, se asegura de que esto no existe (por defecto "trident").
- `--pvc string`: El nombre del PVC heredado utilizado por Astra Trident, se asegura de que esto no exista (por defecto “tridente”).
- `--silence-autosupport`: No envíe los paquetes AutoSupport a NetApp automáticamente (valor predeterminado: TRUE).
- `--silent`: Desactiva la mayoría de la salida durante la instalación.
- `--trident-image string`: La imagen de Astra Trident que se va a instalar.
- `--use-custom-yaml`: Utilice cualquier archivo YAML existente en el directorio de instalación.
- `--use-ipv6`: Utilice IPv6 para la comunicación de Astra Trident.

logs

Puede ejecutar el `logs` Indicadores para imprimir los registros de Astra Trident.

```
Usage:
  tridentctl logs [flags]
```

Indicadores:

- `-a, --archive`: Cree un archivo de soporte con todos los registros a menos que se especifique lo contrario.
- `-h, --help`: Ayuda para registros.
- `-l, --log string`: Mostrar el registro de Astra Trident. Uno de trident|auto|trident-operator|All (valor predeterminado "auto").
- `--node string`: El nombre del nodo Kubernetes del que se van a recopilar registros del nodo pod.
- `-p, --previous`: Obtiene los registros de la instancia anterior del contenedor si existe.
- `--sidecars`: Obtener los registros de los contenedores sidecar.

send

Puede ejecutar el `send` Para enviar un recurso desde Astra Trident.

```
Usage:
  tridentctl send [option]
```

Opción disponible:
`autosupport`: Enviar un fichero AutoSupport a NetApp.

uninstall

Puede ejecutar el `uninstall` Indicadores para desinstalar Astra Trident.

```
Usage:
  tridentctl uninstall [flags]
```

Indicadores:
* `-h, --help`: Ayuda para la desinstalación.
* `--silent`: Desactiva la mayoría de la salida durante la desinstalación.

update

Puede ejecutar el `update` Comandos para modificar un recurso en Astra Trident.

```
Usage:
  tridentctl update [option]
```

Opciones disponibles:
`backend`: Actualizar un back-end en Astra Trident.

upgrade

Puede ejecutar el `upgrade` Comandos para actualizar un recurso en Astra Trident.

```
Usage:
tridentctl upgrade [option]
```

Opción disponible:
`volume`: Actualice uno o más volúmenes persistentes de NFS/iSCSI a CSI.

version

Puede ejecutar el `version` indicadores para imprimir la versión de `tridentctl` Y el servicio Trident que se ejecuta.

```
Usage:
  tridentctl version [flags]
```

Indicadores:
* `--client`: Sólo versión de cliente (no se necesita ningún servidor).
* `-h, --help`: Ayuda para la versión.

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, Astra Trident habilita ciertos permisos durante la instalación. En las siguientes secciones se detallan los permisos establecidos por Astra 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 los montajes iSCSI y utiliza la conexión a redes host para comunicarse con el daemon iSCSI.
IPC de host	NFS utiliza la comunicación entre procesos (IPC) para comunicarse con NFSD.
PID del host	Necesario para comenzar `rpc-statd` Para NFS. Astra Trident consulta los procesos de host para determinar si `rpc-statd` Se ejecuta antes de montar volúmenes NFS.

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

Permiso	Descripción
Seccomp	Seccomp Profile siempre es "no confinado" en contenedores con privilegios; por lo tanto, no se puede activar en Astra Trident.
SELinux	En OpenShift, los contenedores con privilegios se ejecutan en `spc_t` El dominio ("contenedor superprivilegiado") y los contenedores sin privilegios se ejecutan en el `container_t` dominio. Encendido `containerd`, con `container-selinux` instalado, todos los contenedores se ejecutan en el `spc_t` Dominio, que desactiva SELinux de forma efectiva. Por lo tanto, Astra Trident no añade `seLinuxOptions` 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 la etiqueta de espacio de nombres para `privileged` se ha cambiado. 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

Campo	Descripción	Predeterminado
`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 "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 que no sea seguro `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 volúmenes NFS requiere que el IPC del host se comunique con `nfsd`	`true`
`hostNetwork`	Iscsiadm requiere que la red del host se comunique con el demonio iSCSI.	`true`
`hostPID`	Se requiere el PID del host para comprobar si `rpc-statd` está ejecutándose en el nodo.	`true`
`hostPorts`	Trident no utiliza puertos de host.	Vacío
`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`

Campo	Descripción	Predeterminado
`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 está configurado `seLinuxOptions` Debido a que actualmente existen diferencias en el modo en que los tiempos de ejecución de contenedores y las distribuciones de Kubernetes se encargan de 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`.	`true`
`allowHostNetwork`	Iscsiadm requiere que la red del host se comunique con el demonio iSCSI.	`true`
`allowHostPID`	Se requiere el PID del host para comprobar si `rpc-statd` está ejecutándose en el nodo.	`true`
`allowHostPorts`	Trident no utiliza puertos de host.	`false`
`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`

Etiquetas	Descripción	Predeterminado
`allowedUnsafeSysctls`	Trident no requiere que no sea seguro `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 está configurado `seLinuxOptions` Debido a que actualmente existen diferencias en el modo en que los tiempos de ejecución de contenedores y las distribuciones de Kubernetes se encargan de SELinux.	Vacío
`seccompProfiles`	Los contenedores privilegiados siempre funcionan "sin confinar".	Vacío
`supplementalGroups`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`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`

Avisos legales

Los avisos legales proporcionan acceso a las declaraciones de copyright, marcas comerciales, patentes y mucho más.

Derechos de autor

"https://www.netapp.com/company/legal/copyright/"

Marcas comerciales

NETAPP, el logotipo de NETAPP y las marcas enumeradas en la página de marcas comerciales de NetApp son marcas comerciales de NetApp, Inc. Los demás nombres de empresas y productos son marcas comerciales de sus respectivos propietarios.

"https://www.netapp.com/company/legal/trademarks/"

Estadounidenses

Puede encontrar una lista actual de las patentes propiedad de NetApp en:

https://www.netapp.com/pdf.html?item=/media/11887-patentspage.pdf

Política de privacidad

"https://www.netapp.com/company/legal/privacy-policy/"

Código abierto

Puede revisar las licencias y los derechos de autor de terceros que se utilizan en el software de NetApp para Astra Trident en el archivo de avisos de cada versión en https://github.com/NetApp/trident/.