Documentación de Astra Trident 24,06

Astra Trident

NetApp
November 13, 2024

Tabla de contenidos

Documentación de Astra Trident 24,06

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.

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

¿Cuáles son las novedades de 24,06

Mejoras

- **IMPORTANTE**: El `limitVolumeSize` parámetro ahora limita el tamaño de qtree/LUN en los controladores económicos de ONTAP. Utilice el parámetro new `limitVolumePoolSize` para controlar los tamaños de FlexVol en esos controladores. ("Problema n.o 341").
- Se ha añadido la capacidad de reparación automática de iSCSI para iniciar análisis de SCSI con un ID de LUN exacto si se están utilizando iGroups obsoletos ("Problema n.o 883").
- Se ha añadido compatibilidad con operaciones de clones de volúmenes y cambio de tamaño que se permite incluso cuando el back-end está en modo suspendido.
- Se ha añadido la capacidad para que los ajustes de registro configurados por el usuario promuevan la controladora Trident a los pods de nodos de Astra Trident.
- Se ha añadido compatibilidad en Astra Trident para utilizar REST DE forma predeterminada, en lugar de ZAPI para las versiones 9.15.1 y posteriores de ONTAP.
- Se ha añadido soporte para nombres de volúmenes y metadatos personalizados en los back-ends de almacenamiento de ONTAP para los nuevos volúmenes persistentes.
- Se ha mejorado `azure-netapp-files` el controlador (ANF) para habilitar automáticamente el directorio Snapshot de forma predeterminada cuando las opciones de montaje de NFS se establecen para utilizar NFS versión 4.x.
- Se ha añadido soporte para Bottlerocket para volúmenes NFS.
- Se ha añadido soporte de previsualización técnica para Google Cloud NetApp Volumes.

Kubernetes

- Añadido soporte para Kubernetes 1,30.
- Se ha añadido la capacidad de Astra Trident DaemonSet para limpiar montajes zombis y archivos de seguimiento residual al inicio ("Problema n.o 883").
- Se ha agregado una anotación de PVC `trident.netapp.io/luksEncryption` para importar dinámicamente volúmenes LUKS ("Problema n.o 849").
- Se añadió el reconocimiento de topología al controlador de ANF.
- Se ha agregado compatibilidad con nodos de Windows Server 2022.

Soluciones

- Se han corregido los fallos de instalación de Astra Trident debido a transacciones obsoletas.
- Se ha corregido el tridentctl para ignorar los mensajes de advertencia de Kubernetes ("Problema n.o 892").
- Se ha cambiado la prioridad de la controladora Astra Trident `SecurityContextConstraint` a `0` ("Problema n.o 887").
- Los controladores ONTAP ahora aceptan tamaños de volumen inferiores a 20MiB ("Problema[#885").
- Se corrigió Astra Trident para evitar que se redujeran los FlexVols durante la operación de cambio de tamaño del controlador ONTAP-SAN.
- Se corrigió un error de importación de volúmenes de ANF con NFS v4,1.

Amortización

- Se ha eliminado el soporte para EOL Windows Server 2019.

Cambios en 24,02

Mejoras

- Se ha añadido soporte para Cloud Identity.
  - AKS con ANF: La identidad de carga de trabajo de Azure se utilizará como identidad de nube.
  - EKS con FSxN - El rol AWS IAM se utilizará como identidad en la nube.
- Soporte añadido para instalar Astra Trident como complemento en el clúster EKS desde la consola de EKS.
- Se ha añadido la capacidad de configurar y deshabilitar la reparación automática de iSCSI ("Problema n.o 864").
- Se ha añadido la personalidad de FSx a los controladores de ONTAP para permitir la integración con IAM y SecretsManager de AWS, y para permitir que Astra Trident elimine volúmenes FSx con backups ("Problema n.o 453").

Kubernetes

- Añadido soporte para Kubernetes 1,29.

Soluciones

- Se corrigieron los mensajes de advertencia de ACP cuando ACP no está activado ("Problema n.o 866").
- Se añadió un retraso de 10 segundos antes de ejecutar una división de clones durante la eliminación de copias de Snapshot para controladores ONTAP cuando se asocia un clon a la copia de Snapshot.

Amortización

- Se ha eliminado el marco de atestaciones in-toto de los manifiestos de imágenes multiplataforma.

Cambios en 23,10

Soluciones

- Expansión de volumen fija si un nuevo tamaño solicitado es menor que el tamaño total del volumen para los controladores de almacenamiento ONTAP-nas y ONTAP-nas-FlexGroup ("Problema n.o 834").
- Tamaño de volumen fijo para mostrar solo el tamaño utilizable del volumen durante la importación para controladores de almacenamiento ONTAP-nas y ONTAP-nas-FlexGroup ("Problema n.o 722").
- Conversión de nombres FlexVol fija para ONTAP-NAS-Economy.
- Se ha solucionado el problema de inicialización de Astra Trident en un nodo de Windows cuando se reinicia el nodo.

Mejoras

Kubernetes

Añadido soporte para Kubernetes 1,28.

Astra Trident

- Soporte añadido para el uso de Azure Managed Identity (AMI) con controlador de almacenamiento de archivos de azure-netapp.
- Se añadió compatibilidad con NVMe over TCP para el controlador ONTAP-SAN.
- Se ha añadido la capacidad para pausar el aprovisionamiento de un volumen cuando el backend está definido en estado suspendido por el usuario ("Problema n.o 558").

Funciones avanzadas disponibles en Astra Control

Con Astra Trident 23,10, un nuevo componente de software llamado Astra Control Provisioning está disponible para los usuarios con licencia de Astra Control. Este aprovisionador ofrece acceso a un superconjunto de funciones avanzadas de aprovisionamiento de almacenamiento y gestión más allá de las que Astra Trident es compatible por sí mismo. Para la versión 23,10, estas funciones incluyen:

- Funcionalidades de backup y restauración para aplicaciones con back-ends de almacenamiento respaldados por controladores económicos de ontap-nas
- Seguridad de back-end del almacenamiento mejorada con cifrado Kerberos 5
- Recuperación de datos mediante un snapshot
- Mejoras de SnapMirror

"Obtén más información sobre el aprovisionador de Astra Control."

Cambios en 23.07.1

Kubernetes: Se ha corregido la eliminación de daemonset para admitir actualizaciones de cero tiempo de inactividad ("Problema n.o 740").

Cambios en 23,07

Soluciones

Kubernetes

- Se ha corregido la actualización de Trident para ignorar los pods antiguos atascados en estado de finalización ("Problema n.o 740").
- Se ha agregado tolerancia a la definición de «transient-Trident-version-pod» ("Problema n.o 795").

Astra Trident

- Se han corregido las solicitudes de ZAPI de ONTAP para garantizar que se consulten los números de serie de LUN al obtener atributos de LUN para identificar y corregir dispositivos iSCSI fantasma durante las operaciones de almacenamiento en caché de nodos.
- Se ha corregido el manejo de errores en el código del controlador de almacenamiento ("Problema n.o 816").
- Se corrigió el cambio de tamaño de la cuota al utilizar controladores ONTAP con use-rest=true.
- Creación de clones LUN fijos en ontap-san-economy.
- Revertir el campo de información de publicación de `rawDevicePath` a `devicePath`; Lógica agregada para rellenar y recuperar (en algunos casos) `devicePath` el campo.

Mejoras

Kubernetes

- Se añadió compatibilidad para importar snapshots aprovisionadas previamente.
- Minimización de la implementación y el inicio de los permisos de linux ("Problema n.o 817").

Astra Trident

- Ya no se notifica el campo de estado para volúmenes y copias Snapshot «en línea».
- Actualiza el estado del backend si el backend de ONTAP está fuera de línea ("Problemas #801", "N.o 543").
- El número de serie de LUN siempre se recupera y se publica durante el flujo de trabajo ControllerVolumePublish.
- Se ha agregado lógica adicional para verificar el tamaño y el número de serie del dispositivo multivía iSCSI.
- Verificación adicional de los volúmenes iSCSI para garantizar que se deja sin almacenar el dispositivo multivía correcto.

Mejora experimental

Se ha añadido soporte de vista previa técnica para NVMe over TCP para el controlador ONTAP-SAN.

Documentación

Se han realizado muchas mejoras organizativas y de formato.

Amortización

Kubernetes

- Se ha eliminado el soporte para las instantáneas v1beta1.

- Se ha eliminado la compatibilidad con los volúmenes previos a CSI y las clases de almacenamiento.
- Se actualizó el mínimo admitido de Kubernetes a 1,22.

Cambios en 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. Forzar desconexión debe estar activado en el momento de la instalación mediante `--enable-force-detach` el indicador de 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 cluster de operador de Trident para estar sincronizados con los permisos de grupo ("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 solucionó el problema por el que el controlador Trident no podía cerrarse 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.
- Se ha corregido el error de compilación para la arquitectura s390 ("Problema n.o 537").
- Se solucionó el 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 determine 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 agregado compatibilidad con los nodos arm64 ("Problema n.o 732").
- Se 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: Los grupos de iniciadores con ámbito de respaldo ya no se crearán al configurar los 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 corrigieron los permisos de rol de cluster de operador de Trident para estar sincronizados con los permisos de grupo "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 reparar las instalaciones de Trident a través de Helm ("Cuestiones #783, #794").

Mejoras

Kubernetes

- Añadido soporte para Kubernetes 1,26.
- Mejora general de la utilización de recursos de control de acceso basado en roles deTrident ("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.
- Los LIF de datos ya no deben especificarse durante la configuración `ontap-san` o `ontap-san-economy` los controladores.

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.
- Ahora, Astra Trident aplica estrictamente el uso de configuración multivía en entornos SAN, con un valor recomendado de `find_multipaths: no` en archivo multipath.conf.

  El uso de una configuración que no sea multivía o el uso `find_multipaths: yes` o `find_multipaths: smart` un valor en el archivo multipath.conf provocará errores de montaje. Trident ha recomendado el uso de `find_multipaths: no` desde la versión 21,07.

Soluciones

- Se ha solucionado el problema específico del backend de ONTAP creado mediante `credentials` el campo que no se conectaba durante la actualización de 22.07.0 ("Problema n.o 759").
- **Docker:** Se ha corregido un problema que provocaba que el plugin de volumen de Docker no se iniciara 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.
  - Añadido soporte para "uso compartido de volúmenes entre espacios de nombres".
- Todos los controladores de almacenamiento de Trident `ontap-*` 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.
- Agregado "Compatibilidad con volúmenes cifrados LUKS" para `ontap-san` y `ontap-san-economy` controladores de almacenamiento.
- Se ha agregado compatibilidad con nodos de Windows Server 2019.
- Se añade "Compatibilidad con volúmenes SMB en nodos de Windows" a través `azure-netapp-files` del 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 la compatibilidad con `yes` las opciones `find_multipaths` y `smart` para la configuración de la multivía del nodo del trabajador 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 las funciones de red al controlador Azure NetApp Files. ("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:** Se actualizó 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 la preparación de nodos a una información de host sencilla y la detección de servicios activos para confirmar que los servicios NFS/iSCSI están disponibles en los nodos de trabajo.

Documentación

Se ha añadido una nueva "Estándares de seguridad de POD"sección (PSS) con detalles de 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 versiones anteriores, consulte "Versiones anteriores de la documentación" .

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

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 de extracción de imágenes de soporte automático al establecer `imageRegistry` el indicador durante la instalación. ("GitHub Número 715")
- Se solucionó el problema en el que el controlador Azure NetApp Files 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.
- Se corrigió el controlador Azure NetApp Files para permitir nombres de recursos duplicados y limitar operaciones a una sola ubicación.
- Se corrigió el controlador Azure NetApp Files para permitir nombres de recursos duplicados y limitar 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 añadió compatibilidad con la opción 'unixPermissions' para volúmenes Azure NetApp Files. ("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:
  - Devuelve el espacio utilizado de un volumen como restoreSize mínimo al crear instantáneas con `ontap-nas` y `ontap-nas-flexgroup` controladores (GitHub número 645).
  - Se ha solucionado el problema por el que `Failed to expand filesystem` se registraba el error después de cambiar el tamaño del volumen (GitHub, número 560).
  - Se ha solucionado el problema por el que un pod podía atascarse en `Terminating` el estado (GitHub número 572).
  - Se ha corregido el caso en el que un `ontap-san-economy` FlexVol estaba lleno de LUN de instantáneas (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 añadido soporte para `unixPermissions` Option to GCP-CVS Performance Volumes.
- 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:
  - Añadido soporte para Kubernetes 1,22.
  - Se ha habilitado el operador de Trident y el gráfico Helm para que funcionen con Kubernetes 1.22 (GitHub, número 628).
  - Imagen de operador agregada al `tridentctl` comando Imágenes (GitHub número 570).

Mejoras experimentales

- Se añadió compatibilidad para la replicación de volúmenes en `ontap-san` el controlador.
- Añadido **tech preview** soporte REST para el `ontap-nas-flexgroup`, `ontap-san` y `ontap-nas-economy` controladores.

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 la versión 1,25 o una versión posterior que tiene Astra Trident instalado, debe actualizar los valores.yaml para establecer `excludePodSecurityPolicy true` o añadir `--set excludePodSecurityPolicy=true` al `helm upgrade` comando antes de actualizar el clúster.
- Astra Trident ahora aplica un espacio en blanco `fsType` (`fsType=""`) para volúmenes que no tengan el `fsType` especificado en su clase de almacenamiento. Al trabajar con Kubernetes 1,17 o una versión posterior, Trident admite proporcionar un espacio vacío `fsType` para volúmenes NFS. Para los volúmenes iSCSI, debe establecer `fsType` en su clase de almacenamiento al aplicar un `fsGroup` contexto de uso de seguridad.
- Cuando se utiliza un back-end en varias instancias de Astra Trident, cada archivo de configuración de back-end debería tener un valor diferente `storagePrefix` para los back-ends de ONTAP o usar otro `TenantName` para los 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` se diferencian o `TenantName` no, es posible que existan colisiones de nombres para los volúmenes creados en el mismo back-end.
- Cuando se instala Astra Trident (mediante `tridentctl` o el operador de Trident) y se utiliza `tridentctl` para gestionar Astra Trident, debes asegurarte de que `KUBECONFIG` se haya definido la variable de entorno. Esto es necesario para indicar el clúster de Kubernetes en `tridentctl` el que debería funcionar. Cuando trabaje con varios entornos de Kubernetes, debe asegurarse de que el `KUBECONFIG` archivo se obtenga 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 se aplica a las

instancias de RHEL/RedHat CoreOS, que requieren `discard` "opción de montaje" ; Asegúrese de que descarte mountOption se incluye en el[`StorageClass`^] para admitir el 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 los objetos basados en Astra Trident `StorageClass` se eliminan de Kubernetes mientras Astra Trident está desconectado, Astra Trident no quita 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 quitar 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 se usa Astra Trident a través de IPv6, debe especificar `managementLIF` y `dataLIF` en la definición del back-end entre corchetes. Por ejemplo, `[fd20:8b1e:b258:2000:f816:3eff:feec:0]`.

> No se puede especificar `dataLIF` en un back-end de SAN de ONTAP. Astra Trident descubre todos los LIF iSCSI disponibles y los utiliza para establecer la sesión multivía.

- Si utiliza `solidfire-san` el 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 está ejecutando Astra Trident 24,06, la documentación de versiones anteriores está disponible según "Ciclo de vida del soporte de Astra Trident"el .

- "Astra Trident 24,02"
- "Astra Trident 23,10"
- "Astra Trident 23,07"
- "Astra Trident 23,04"
- "Astra Trident 23,01"
- "Astra Trident 22,10"
- "Astra Trident 22,07"
- "Astra Trident 22,04"
- "Astra Trident 22,01"
- "Astra Trident 21,10"

Manos a la obra

Obtenga más información sobre Astra Trident

Obtenga más información sobre Astra Trident

Astra Trident es un proyecto de código abierto totalmente compatible y mantenido por NetApp como parte de la "Familia de productos Astra". Se ha diseñado para ayudarle a cumplir las demandas de persistencia de sus aplicaciones en contenedores mediante interfaces estándar del sector, como la Container Storage Interface (CSI).

¿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 en contenedores persistente basado en Astra Trident. Además, ofrece funcionalidades avanzadas de gestión de datos para aplicaciones, como instantáneas, backup y restauración, registros de actividades y clonado activo para protección de datos, recuperación de datos/desastres, auditoría de datos y casos prácticos de migración para cargas de trabajo de Kubernetes.

Más información sobre "Astra o regístrate para una prueba gratuita".

¿Qué es Astra Trident?

Astra Trident permite el consumo y la gestión de recursos de almacenamiento en todas las plataformas de almacenamiento de NetApp más conocidas, tanto en el cloud público como en las instalaciones, incluidos ONTAP (AFF, FAS, Select, Cloud, Amazon FSx para NetApp ONTAP), el software Element (NetApp HCI, SolidFire), el servicio Azure NetApp Files y Cloud Volumes Service en Google Cloud.

Astra Trident es una interfaz de almacenamiento de contenedores (CSI) que ordena el almacenamiento dinámico conforme a la normativa que se integra de forma nativa con "Kubernetes". Astra Trident se ejecuta como un pod de controladora único más un pod de nodo en cada nodo trabajador del clúster. Consulte "Arquitectura de Astra Trident" para obtener más información.

Astra Trident también proporciona integración directa con el ecosistema de Docker para las plataformas de almacenamiento de NetApp. El complemento para volúmenes de Docker de NetApp (nDVP) admite el aprovisionamiento y la gestión de recursos de almacenamiento desde la plataforma de almacenamiento a los hosts de Docker. Consulte "Ponga en marcha Astra Trident para Docker" para obtener más información.

> Si esta es la primera vez que usa Kubernetes, debe familiarizarse con el "Conceptos y herramientas de Kubernetes".

Acepta la prueba de Astra Trident

Para realizar una prueba, solicite acceso a «Puesta en marcha y clonación sencilla de almacenamiento persistente para cargas de trabajo en contenedores» "Versión de prueba de NetApp" mediante una imagen de laboratorio lista para usar. La unidad de prueba proporciona un entorno de pruebas con un clúster de Kubernetes de tres nodos y Astra Trident instalado y configurado. Esta es una excelente manera de familiarizarse con Astra Trident y explorar sus funciones.

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

> No utilice un clúster de Kubernetes creado con estas instrucciones en un entorno de producción. Use las guías de puesta en marcha de producción proporcionadas por su distribución para clústeres listos para la producción.

Integración de Kubernetes con productos de NetApp

La cartera de productos de almacenamiento de NetApp se integra con muchos aspectos de un clúster de Kubernetes, lo que ofrece capacidades de gestión de datos avanzadas que mejoran la funcionalidad, la capacidad, el rendimiento y la disponibilidad de la puesta en marcha de Kubernetes.

Amazon FSX para ONTAP de NetApp

"Amazon FSX para ONTAP de NetApp" Es un servicio AWS totalmente gestionado que le permite iniciar y ejecutar sistemas de archivos con tecnología del sistema operativo de almacenamiento NetApp ONTAP.

Azure NetApp Files

"Azure NetApp Files" Es un servicio de recursos compartidos de archivos de Azure de clase empresarial impulsado por 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 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.

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 NAS por NFS y SMB con un rendimiento all-flash.

Software Element

"Elemento" permite al administrador de almacenamiento consolidar cargas de trabajo garantizando el rendimiento y permitiendo un espacio de almacenamiento simplificado y optimizado.

HCI de NetApp

"HCI de NetApp" simplifica la gestión y el escalado del centro de datos al automatizar las tareas rutinarias y permitir que los administradores de infraestructuras se centren en funciones más importantes.

Astra Trident puede aprovisionar y gestionar dispositivos de almacenamiento para aplicaciones en contenedores directamente en la plataforma de almacenamiento subyacente de NetApp HCI.

ONTAP de NetApp

"ONTAP de NetApp" Es el sistema operativo de almacenamiento unificado multiprotocolo de NetApp que proporciona capacidades 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). Astra Trident es compatible con estos modelos de puesta en marcha de ONTAP.

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"

Arquitectura de Astra Trident

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

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

Astra Trident se pone en marcha como Pod de controladora de Tridentuno o varios Pods de nodos de Trident en el clúster de Kubernetes y utiliza contenedores Sidecar Containers_ estándar de Kubernetes _CSI para simplificar la implementación de los complementos CSI. "Contenedores Sidecar de Kubernetes CSI" Los mantiene la comunidad de Kubernetes Storage.

Kubernetes "selectores de nodos" y "toleraciones y tintes" se utilizan para restringir que un pod se ejecute en un nodo específico o preferido. Durante la instalación de Astra Trident, puede configurar los selectores de nodos y las toleraciones para la controladora y los pods de nodos.

- 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.

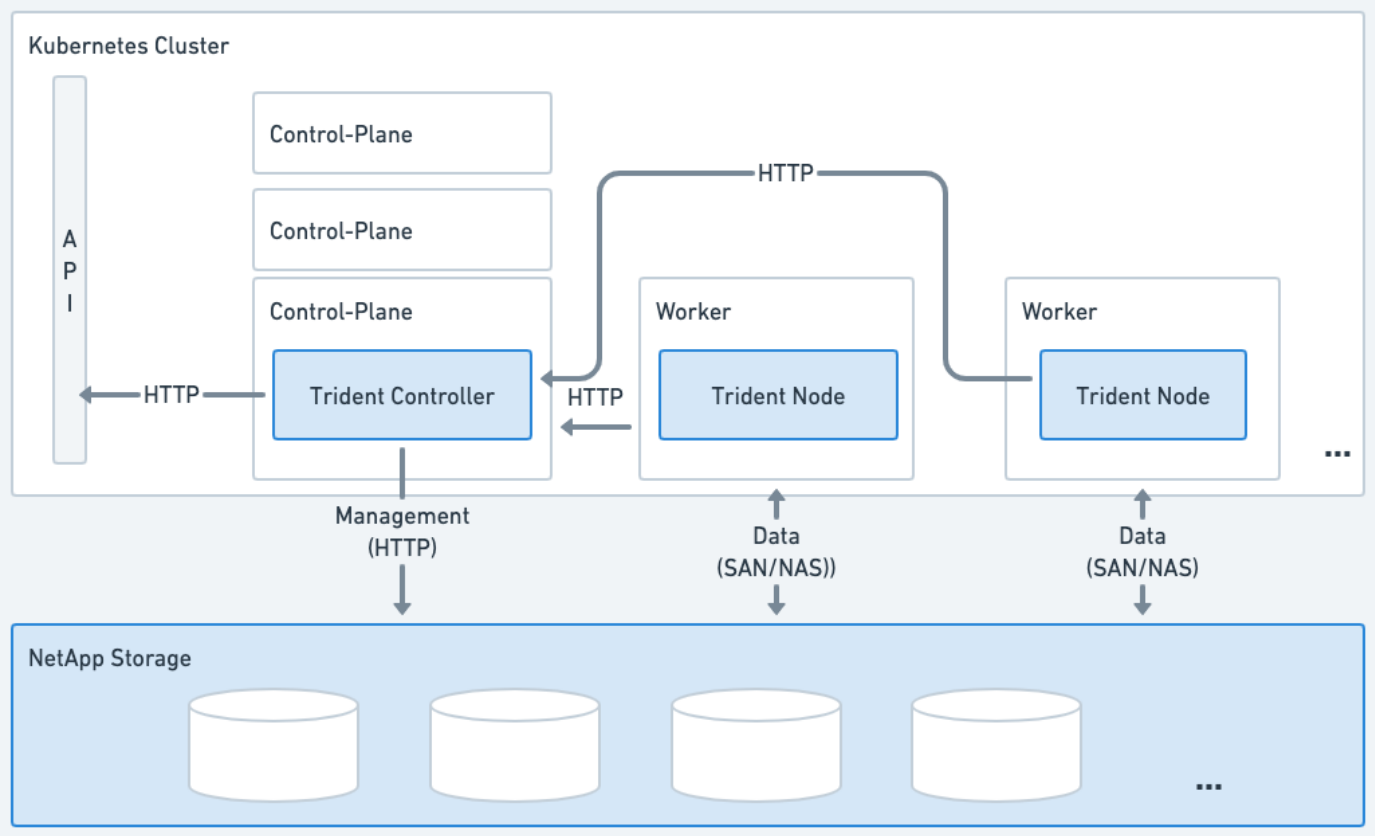


Figura 1. Astra Trident puesto en marcha en el clúster de Kubernetes

Pod de controladora de Trident

Trident Controller Pod es un pod único que ejecuta el complemento CSI Controller.

- Responsable de aprovisionar y gestionar volúmenes en el almacenamiento de NetApp
- Gestionado por una puesta en marcha de Kubernetes
- Se puede ejecutar en el plano de control o en los nodos de trabajo, según los parámetros de instalación.

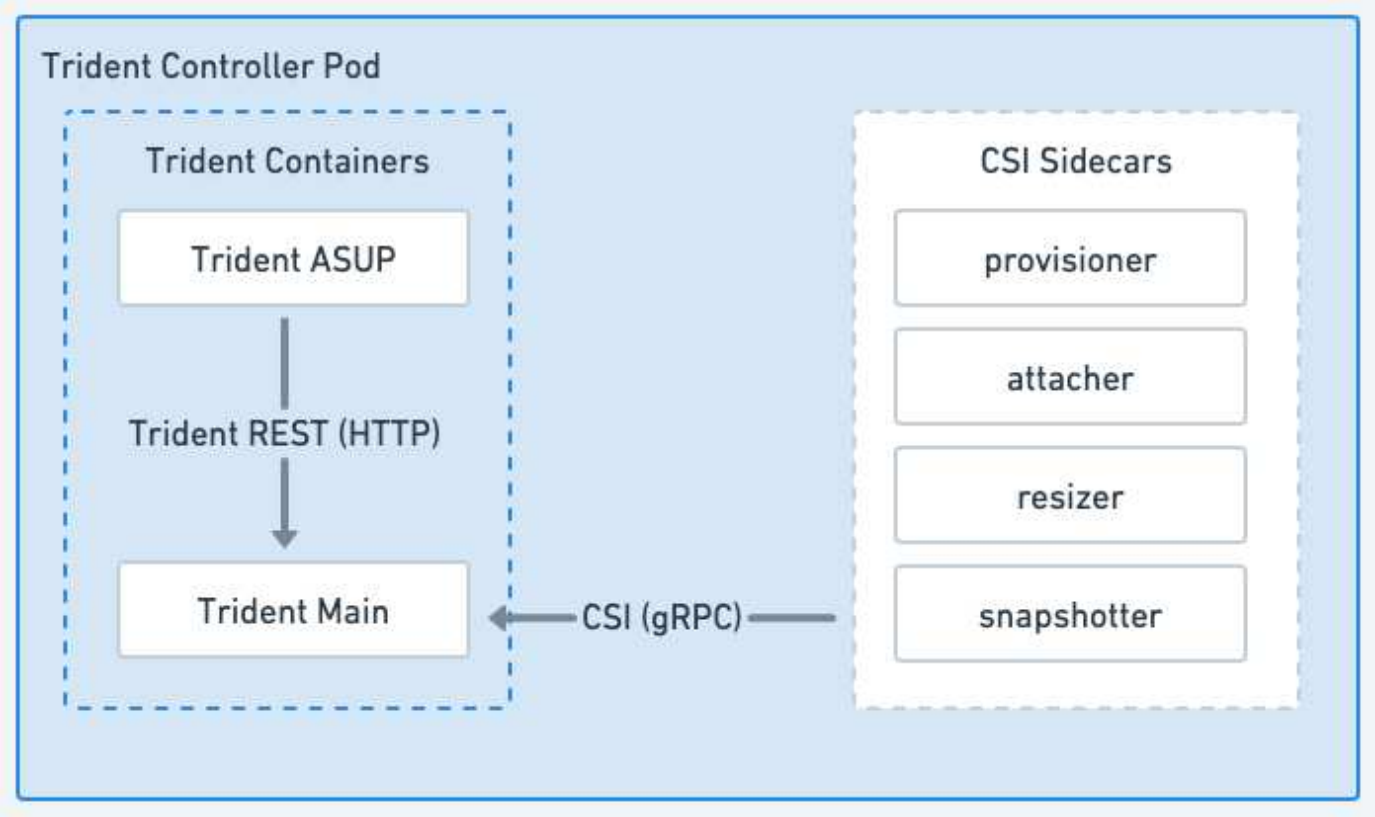


Figura 2. Diagrama de Trident Controller Pod

Pods de nodos de Trident

Los pods de nodos Trident son pods con privilegios que ejecutan el plugin de nodo CSI.

- Responsable del montaje y desmontaje del almacenamiento de los pods que se ejecutan en el host
- Gestionado por un DaemonSet de Kubernetes
- Debe ejecutarse en cualquier nodo que monte el almacenamiento de NetApp

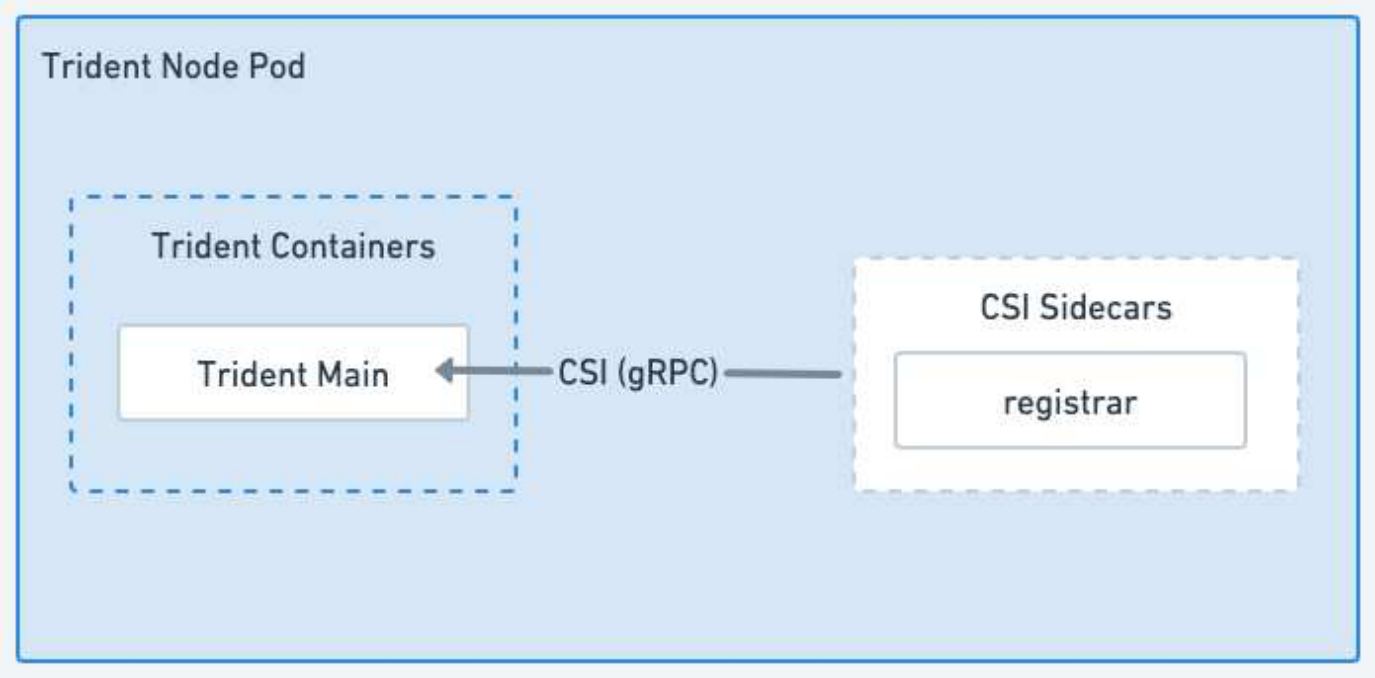


Figura 3. Diagrama de Trident Node Pod

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

Conceptos

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 misma del volumen y requiere la selección de un pool de almacenamiento entre los asociados con la clase de almacenamiento del volumen pendiente.

Asociación de clase de almacenamiento

La asociación de pools de almacenamiento de backend con una clase de almacenamiento depende tanto de los atributos solicitados de la clase de almacenamiento como de sus `storagePools` listas ,

`additionalStoragePools` y. `excludeStoragePools` 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 `additionalStoragePools` la lista a ese conjunto, incluso si sus atributos no cumplen todos o alguno de los atributos solicitados de la clase de almacenamiento. Debe usar la `excludeStoragePools` lista para anular y quitar pools de almacenamiento del uso para 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 los `ontap-nas` `gcp-cvs` controladores , , , `ontap-san` y `azure-netapp-files`, cada volumen persistente (PV) se asigna a un 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 la tecnología snapshot de la competencia. Estas copias Snapshot son extremadamente eficientes, tanto en el tiempo necesario para crearlas como en el espacio de almacenamiento.
- Para `ontap-nas-flexgroup` el controlador, cada volumen persistente (VP) se asigna a un 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 la tecnología snapshot de la competencia. Estas copias Snapshot son extremadamente eficientes, tanto en el tiempo necesario para crearlas como en el espacio de almacenamiento.
- Para `ontap-san-economy` el controlador, los VP se asignan a LUN creadas en FlexVols compartidos. Las copias Snapshot Volumede VP realizan FlexClones del LUN asociado. La tecnología FlexClone de ONTAP permite crear copias de incluso los conjuntos de datos más grandes casi al instante. Las copias comparten bloques de datos con sus padres, sin consumir almacenamiento, excepto lo que se necesita para los metadatos.
- Para `solidfire-san` el controlador, cada VP se asigna a una LUN creada en el clúster de

software/NetApp HCI de NetApp Element. 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.

- Cuando trabajan con los `ontap-nas` controladores y `ontap-san`, las copias Snapshot de ONTAP son copias puntuales de la 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.

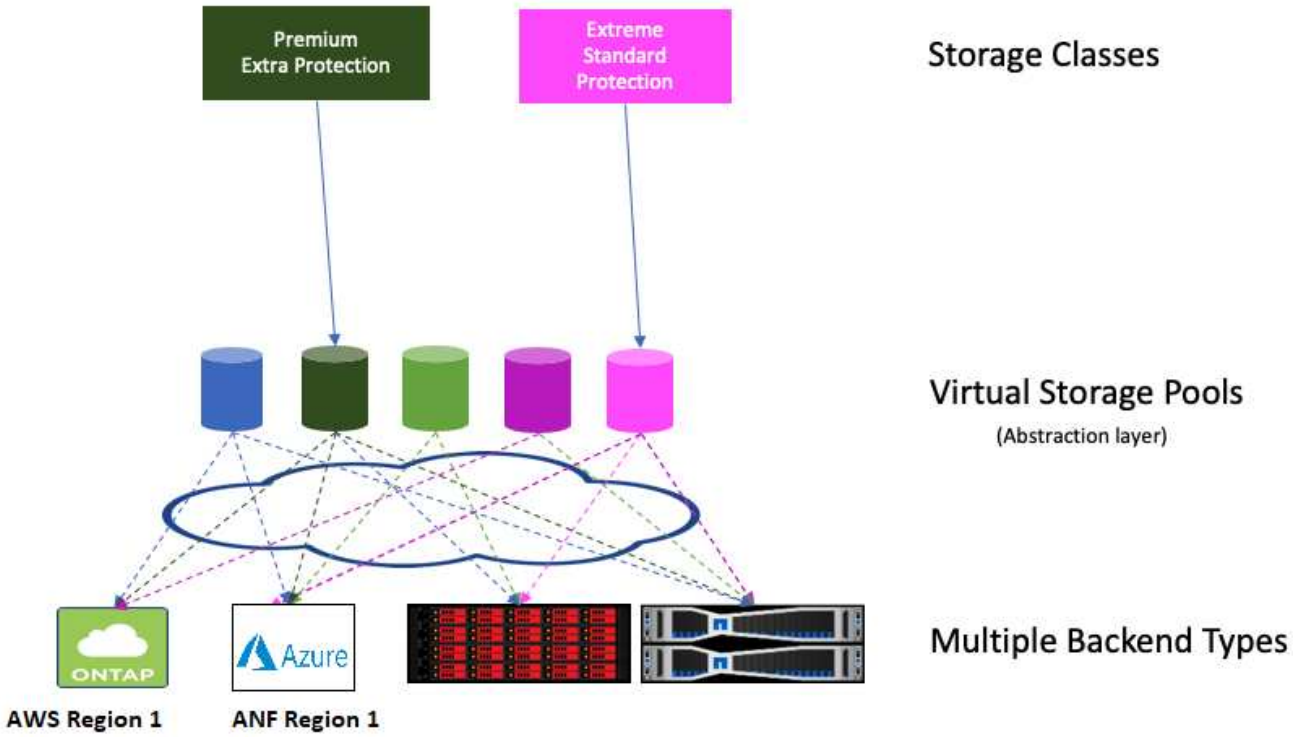
Con Astra Trident, puede usar VolumeSnapshots para crear nuevos VP a partir de ellos. La creación de VP a partir de estas snapshots se realiza usando la tecnología FlexClone para los back-ends de ONTAP y CVS compatibles. Cuando se crea un volumen persistente a partir de una copia Snapshot, el volumen de backup es FlexClone del volumen principal de la copia Snapshot. `solidfire-san`El controlador usa clones de volúmenes del software Element para crear VP a partir de snapshots. 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 para cada backend de una manera común e independiente del backend sin `StorageClass` especificar qué backend físico, pool de backend o tipo de backend utilizar para cumplir con 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 la coincidencia en `StorageClasses`. En su lugar, el administrador define una o más 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.

A `StorageClass` identifica el pool virtual que se va a 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

Obtén más información sobre cómo utiliza 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, espera encontrar un grupo de acceso llamado `trident` a menos que especifique uno o más ID de grupo de acceso en la configuración.

Aunque Astra Trident asocia nuevos volúmenes con los grupos de acceso configurados, no crea ni gestiona ellos mismos 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 iSCSI de cada nodo del clúster de Kubernetes, que podrían montar los volúmenes aprovisionados por 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 conocer los límites más recientes sobre los grupos de acceso de volúmenes, consulte "aquí" .

Si está modificando la configuración de una que esté utilizando el grupo de acceso predeterminado `trident` a otra que también use otras, incluya el ID del `trident` grupo de acceso en la lista.

Inicio rápido para Astra Trident

Puedes instalar Astra Trident y empezar a gestionar los recursos de almacenamiento en unos pocos pasos. Antes de empezar, revise "Requisitos de Astra Trident".

> Para Docker, consulte "Astra Trident para Docker".

1 Instala Astra Trident

Astra Trident ofrece varios métodos de instalación y modos optimizados para una variedad de entornos y organizaciones.

"Instale 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.

"Prepare el nodo de trabajo"

Cree un backend

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.

"Configurar un backend" de su sistema de almacenamiento

Cree una clase de almacenamiento de Kubernetes

El objeto Kubernetes StorageClass especifica Astra Trident como el aprovisionador y le permite crear una clase de almacenamiento para aprovisionar volúmenes con atributos personalizables. Astra Trident crea una clase de almacenamiento coincidente para los objetos de Kubernetes que especifica el aprovisionador de Astra Trident.

"Cree una clase de almacenamiento"

Aprovisione un volumen

Un *PersistentVolume* (PV) es un recurso de almacenamiento físico aprovisionado por el administrador del clúster en un clúster de Kubernetes. *PersistentVolumeClaim* (RVP) es una solicitud para acceder al volumen persistente en el clúster.

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

"Aprovisione un volumen"

El futuro

Ahora puede añadir back-ends adicionales, gestionar clases de almacenamiento, gestionar back-ends y realizar operaciones de volumen.

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

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

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

- Kubernetes 1,31 ahora es compatible con Astra Trident. Actualiza Astra Trident antes de actualizar Kubernetes.
- Astra Trident aplica estrictamente el uso de configuración multivía en entornos SAN, con un valor recomendado de `find_multipaths: no` en archivo multipath.conf.

  El uso de una configuración que no sea multivía o el uso `find_multipaths: yes` o `find_multipaths: smart` un valor en el archivo multipath.conf provocará errores de montaje. Astra 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 1,16 básico
- Kubernetes 1,24 - 1,31
- OpenShift 4,10 - 4,16

El operador Trident es compatible con las siguientes versiones:

- Anthos on-premises (VMware) y Anthos en 1,16 básico
- Kubernetes 1,24 - 1,31
- OpenShift 4,10 - 4,16

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

Astra Trident y ONTAP se pueden utilizar como proveedor de almacenamiento de"Virt de KubeVirt".

Antes de actualizar un clúster de Kubernetes de 1,24 a 1,25 o una versión posterior que tenga Astra Trident instalado, consulte "Actualizar una instalación Helm"el .

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?
Astra Trident	1,24 - 1,31	No
Snapshots de volumen	1,24 - 1,31	No
RVP desde snapshots de volumen	1,24 - 1,31	No
Cambio de tamaño del VP de iSCSI	1,24 - 1,31	No
CHAP bidireccional de ONTAP	1,24 - 1,31	No
Políticas de exportación dinámicas	1,24 - 1,31	No
Operador de Trident	1,24 - 1,31	No
Topología CSI	1,24 - 1,31	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)

  NVMe/TCP requiere RHEL 9 o posterior.

- Ubuntu 22,04 o posterior (AMD64 y ARM64)
- Windows Server 2022

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` utilidad 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, iSCSI o NVMe según la selección de controladores.

"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 `tridentctl images` el comando para verificar la lista de imágenes de contenedor necesarias.

Versiones de Kubernetes	Imagen de contenedor
v1.24.0, v1.25.0, v1.26.0, v1.27.0, v1.28.0, v1.29.0, v1.30.0, v1.31.0	• docker.io/netapp/trident:24.06.0 • docker.io/netapp/trident-autosupport:24,06 • registry.k8s.io/sig-storage/csi-provisioner:v4,0.1 • registry.k8s.io/sig-storage/csi-attacher:v4,6.0 • registry.k8s.io/sig-storage/csi-resizer:v1.11.0 • registry.k8s.io/sig-storage/csi-snapshotter:v7,0.2 • registry.k8s.io/sig-storage/csi-node-driver-registrador:v2.10.0 • docker.io/netapp/trident-operator:24.06.0 (opcional)

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. Puedes instalar Astra Trident usando el operador Trident (manualmente o usando Helm) o con `tridentctl`. En este tema se proporciona información importante para seleccionar el proceso de instalación adecuado.

Información vital sobre Astra Trident 24,06

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

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

- Kubernetes 1,31 ahora es compatible con Astra Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident aplica estrictamente el uso de configuración multivía en entornos SAN, con un valor recomendado de `find_multipaths: no` en archivo multipath.conf.

  El uso de una configuración que no sea multivía o el uso `find_multipaths: yes` o `find_multipaths: smart` un valor en el archivo multipath.conf provocará errores 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. Consulte 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.
- `KUBECONFIG`La variable de entorno establecida para que apunte a la configuración de su clúster de Kubernetes.
- Si utiliza Kubernetes con Docker Enterprise, "Siga sus pasos para habilitar el acceso a la CLI".

Si no te has familiarizado con el"conceptos básicos", ahora es un buen momento para hacerlo.

Elija el método de instalación

Seleccione el método de instalación adecuado. También debe revisar las consideraciones para "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"utilizar los atributos en el `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

**Contabilidad de recursos **

El operador Trident en el ámbito del clúster gestiona los recursos asociados con una instalación de Astra Trident en el nivel del clúster. Esto mitiga los errores que pueden producirse al mantener los recursos de ámbito de cluster mediante un operador de ámbito de espacio de nombres. Esto es esencial para la reparación automática y la aplicación de parches.

** 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. `trident-operator-<generated-id>`Se crea un pod que asocia una `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 es necesario editar el `TridentOrchestrator` CR para realizar actualizaciones en una instalación.

Por ejemplo, piense en una situación en la que necesita habilitar Astra Trident para generar registros de depuración. Para ello, aplique un parche `TridentOrchestrator` a para establecer `spec.debug` en `true`:

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

Después de `TridentOrchestrator` actualizarse, el operador procesa las actualizaciones y aplica parches a la instalación existente. Esto podría desencadenar la creación de nuevos pods para modificar la instalación en consecuencia.

Clean reinstallation

El operador Trident en el ámbito del clúster permite eliminar sin problemas los recursos del ámbito del clúster. Los usuarios pueden desinstalar completamente Astra Trident y reinstalarlo fácilmente.

**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.

Utilizando `tridentctl`

Si tiene un despliegue existente que debe actualizarse o si está buscando personalizar altamente su despliegue, debe considerar . Este es el método convencional de puesta en marcha de Astra Trident.

Puede generar los manifiestos para 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 22,04 se pueden 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 utiliza registros predeterminados para almacenar (`registry.k8s.io`las imágenes Trident (`docker.io`) y CSI) necesarias.

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:

- Implementa la versión adecuada de `kubectl` en el equipo remoto desde donde quieras implementar Astra Trident.
- Copie los archivos de configuración del clúster de Kubernetes y establezca la `KUBECONFIG` variable de entorno en el equipo remoto.
- Inicie `kubectl get nodes` un comando para verificar que puede conectarse al clúster de Kubernetes requerido.
- 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 del `tridentctl` binario para desinstalar Astra Trident. Del mismo modo, si va a implementar con el operador, debe editar el `TridentOrchestrator` CR y configurar `spec.uninstall=true` la desinstalación de Astra Trident.
- Si tiene una puesta en marcha basada en operadores que desea quitar y utilizar en su lugar `tridentctl` para implementar Astra Trident, primero debe editar `TridentOrchestrator` y establecer `spec.uninstall=true` la desinstalación de Astra Trident. A continuación, suprima `TridentOrchestrator` y despliegue del operador. A continuación, puede instalar utilizando `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 tiene un `tridentctl` despliegue basado en Helm, puede utilizar el despliegue basado en Helm sin tener 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.
- El `--kubelet-dir` indicador debe establecerse en la ubicación del directorio kubelet. Por defecto, esto es `/var/vcap/data/kubelet`.

  Se sabe que la especificación de la ubicación de kubelet mediante `--kubelet-dir` funciona para operadores, Helm e implementaciones de Trident `tridentctl`.

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 tiene un registro de imágenes privado, utilice el "proceso de puesta en marcha sin conexión".

Información vital sobre Astra Trident 24,06

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

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

- Kubernetes 1,31 ahora es compatible con Astra Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident aplica estrictamente el uso de configuración multivía en entornos SAN, con un valor recomendado de `find_multipaths: no` en archivo multipath.conf.

  El uso de una configuración que no sea multivía o el uso `find_multipaths: yes` o `find_multipaths: smart` un valor en el archivo multipath.conf provocará errores 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

Revise "descripción general de la instalación"para asegurarse de que cumple los requisitos previos de la instalación y ha seleccionado la opción de instalación correcta para su entorno.

Antes de empezar

Antes de comenzar la instalación, inicie sesión en el host Linux y compruebe que está gestionando un funcionamiento "Clúster de Kubernetes compatible"y que dispone de la Privileges necesaria.

> Con OpenShift, use `oc` en lugar de en `kubectl` 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 última versión del instalador de Trident de "La sección *Assets* de GitHub".

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

Paso 2: Crear el `TridentOrchestrator` CRD

Cree la `TridentOrchestrator` definición de recursos personalizados (CRD). Creará `TridentOrchestrator` recursos personalizados más adelante. Utilice la versión YAML de CRD adecuada en `deploy/crds` para crear la `TridentOrchestrator` CRD.

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

Paso 3: 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, 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 despliega el operador en `trident` el espacio de nombres. Si el `trident` espacio de nombres no existe, créelo con:

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

- Para desplegar el operador en un espacio de nombres distinto `trident` del espacio de nombres, actualice `serviceaccount.yaml` `clusterrolebinding.yaml` y `operator.yaml` genere el archivo de grupo mediante el `kustomization.yaml`.

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

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

  b. Compile el paquete con el siguiente comando donde *<bundle.yaml>* está `bundle_pre_1_25.yaml` o `bundle_post_1_25.yaml` basado en 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: Crear `TridentOrchestrator` e instalar Trident

Ahora puedes crear `TridentOrchestrator` e instalar Astra Trident. Opcionalmente, puede "Personalice su instalación de Trident" utilizar los atributos de la `TridentOrchestrator` especificación.

```
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:24.06
    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:24.06.0
  Message:                  Trident installed  Namespace:
trident
  Status:                   Installed
  Version:                  v24.06.0
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.

Utilizando `TridentOrchestrator` el estado

El estado de `TridentOrchestrator` indica si la instalación se ha realizado correctamente y muestra la

versión de Trident instalada. Durante la instalación, el estado de `TridentOrchestrator` cambia de `Installing` a `Installed`. Si observa `Failed` el estado y el operador es incapaz de recuperarse por sí mismo, "compruebe los registros".

Estado	Descripción
Instalación	El operador está instalando Astra Trident con este `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.
Con errores	El operador no pudo instalar, aplicar parches, actualizar o desinstalar Astra Trident; el operador intentará recuperarse automáticamente de este estado. Si este estado continúa, necesitará solucionar problemas.
Actualizando	El operador está actualizando una instalación existente.
Error	`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
```

Utilizando `tridentctl`

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

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 24.06.0        | 24.06.0        |
+----------------+----------------+
```

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 el "proceso de implementación estándar".

Información vital sobre Astra Trident 24,06

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

información bï¿½tico sobre Astra Tridbï¿½tico

- Kubernetes 1,31 ahora es compatible con Astra Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident aplica estrictamente el uso de configuración multivía en entornos SAN, con un valor recomendado de `find_multipaths: no` en archivo multipath.conf.

  El uso de una configuración que no sea multivía o el uso `find_multipaths: yes` o `find_multipaths: smart` un valor en el archivo multipath.conf provocará errores 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

Revise "descripción general de la instalación"para asegurarse de que cumple los requisitos previos de la instalación y ha seleccionado la opción de instalación correcta para su entorno.

Antes de empezar

Inicie sesión en el host Linux y compruebe que está gestionando un funcionamiento y "Clúster de Kubernetes compatible"que dispone de la Privileges necesaria.

Con OpenShift, use `oc` en lugar de en `kubectl` 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 última versión del instalador de Trident de "La sección *Assets* de GitHub".

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

Paso 2: Crear el `TridentOrchestrator` CRD

Cree la `TridentOrchestrator` definición de recursos personalizados (CRD). Creará `TridentOrchestrator` recursos personalizados más adelante. Utilice la versión YAML de CRD adecuada en `deploy/crds` para crear el `TridentOrchestrator` CRD:

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

Paso 3: Actualice la ubicación del registro en el operador

En `/deploy/operator.yaml`, actualice `image: docker.io/netapp/trident-operator:24.06.0` para reflejar la ubicación del registro de imágenes. "Imágenes Trident y CSI"Puede estar ubicado en un registro o registros diferentes, pero todas las imágenes CSI deben estar ubicadas en el mismo registro. Por ejemplo:

- `image: <your-registry>/trident-operator:24.06.0` si todas sus imágenes están ubicadas en un registro.

- `image: <your-registry>/netapp/trident-operator:24.06.0` 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, 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 despliega el operador en `trident` el espacio de nombres. Si el `trident` espacio de nombres no existe, créelo con:

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

- Para desplegar el operador en un espacio de nombres distinto `trident` del espacio de nombres, actualice `serviceaccount.yaml` `clusterrolebinding.yaml` y `operator.yaml` genere el archivo de grupo mediante el `kustomization.yaml`.

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

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

  b. Compile el paquete con el siguiente comando donde *<bundle.yaml>* está `bundle_pre_1_25.yaml` o `bundle_post_1_25.yaml` basado en 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 5: Actualice la ubicación del registro de imágenes en el `TridentOrchestrator`

"Imágenes Trident y CSI"Puede estar ubicado en un registro o registros diferentes, pero todas las imágenes CSI deben estar ubicadas en el mismo registro. Actualice `deploy/crds/tridentorchestrator_cr.yaml` para agregar las especificaciones de ubicación adicionales basadas en la configuración del registro.

Imágenes en un registro

```
imageRegistry: "<your-registry>"
autosupportImage: "<your-registry>/trident-autosupport:24.06"
tridentImage: "<your-registry>/trident:24.06.0"
```

Imágenes en diferentes registros

Debe anexar `sig-storage` a `imageRegistry` para utilizar distintas ubicaciones de registro.

```
imageRegistry: "<your-registry>/sig-storage"
autosupportImage: "<your-registry>/netapp/trident-autosupport:24.06"
tridentImage: "<your-registry>/netapp/trident:24.06.0"
```

Paso 6: Crear `TridentOrchestrator` e instalar Trident

Ahora puedes crear `TridentOrchestrator` e instalar Astra Trident. Si lo desea, puede "Personalice su instalación de Trident"utilizar los atributos de la `TridentOrchestrator` especificación. 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:24.06
  Debug:              true
  Image Registry:     <your-registry>/sig-storage
  Namespace:          trident
  Trident Image:      <your-registry>/netapp/trident:24.06.0
Status:
  Current Installation Params:
    IPv6:                       false
    Autosupport Hostname:
    Autosupport Image:          <your-registry>/netapp/trident-
autosupport:24.06
    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:24.06.0
  Message:                Trident installed
  Namespace:              trident
  Status:                 Installed
  Version:                v24.06.0
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.

Utilizando `TridentOrchestrator` el estado

El estado de `TridentOrchestrator` indica si la instalación se ha realizado correctamente y muestra la versión de Trident instalada. Durante la instalación, el estado de `TridentOrchestrator` cambia de `Installing` a `Installed`. Si observa `Failed` el estado y el operador es incapaz de recuperarse por sí mismo, "compruebe los registros".

Estado	Descripción
Instalación	El operador está instalando Astra Trident con este `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.
Con errores	El operador no pudo instalar, aplicar parches, actualizar o desinstalar Astra Trident; el operador intentará recuperarse automáticamente de este estado. Si este estado continúa, necesitará solucionar problemas.
Actualizando	El operador está actualizando una instalación existente.
Error	`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
```

Utilizando `tridentctl`

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

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 24.06.0        | 24.06.0        |
+----------------+----------------+
```

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 tiene un registro de imágenes privado, utilice el "proceso de puesta en marcha sin conexión".

Información vital sobre Astra Trident 24,06

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

información bï¿½tico sobre Astra Tridbï¿½tico

- Kubernetes 1,31 ahora es compatible con Astra Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident aplica estrictamente el uso de configuración multivía en entornos SAN, con un valor recomendado de `find_multipaths: no` en archivo multipath.conf.

  El uso de una configuración que no sea multivía o el uso `find_multipaths: yes` o `find_multipaths: smart` un valor en el archivo multipath.conf provocará errores 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

Con Trident "Carta del timón"puede desplegar el operador Trident e instalar Trident en un solo paso.

Revise "descripción general de la instalación"para asegurarse de que cumple los requisitos previos de la instalación y ha seleccionado la opción de instalación correcta para su 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. Utilice `helm install` y especifique un nombre para la implementación como en el ejemplo siguiente, donde `100.2404.0` se encuentra la versión de Astra Trident que está instalando.

```
helm install <name> netapp-trident/trident-operator --version 100.2406.0
--create-namespace --namespace <trident-namespace>
```

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

Puede utilizar `helm list` para revisar detalles de instalación como nombre, espacio de nombres, gráfico, estado, versión de la aplicación, y 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 el siguiente `--set` comando, donde `100.2406.0` es la versión de Astra Trident que está instalando:

```
helm install <name> netapp-trident/trident-operator --version 100.2406.0
--create-namespace --namespace trident --set tridentDebug=true
```

Opciones de configuración

Esta tabla y el `values.yaml` archivo, que forma parte del diagrama Helm, proporcionan 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	

Opción	Descripción	Predeterminado
`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 información.	
`tridentControllerPluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más información.	
`tridentNodePluginNodeSelector`	Selectores de nodos adicionales para POD. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más información.	
`tridentNodePluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más información.	
`imageRegistry`	Identifica el registro de `trident-operator` , `trident` y otras imágenes. Déjelo vacío para aceptar el valor predeterminado.	`""`
`imagePullPolicy`	Establece la política de extracción de imágenes para la `trident-operator`.	`IfNotPresent`
`imagePullSecrets`	Define los secretos de extracción de imágenes para `trident-operator` , `trident` y otras imágenes.	
`kubeletDir`	Permite anular la ubicación del host del estado interno de kubelet.	`"/var/lib/kubelet"`
`operatorLogLevel`	Permite definir el nivel de log 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 para `trident-operator`.	`""`
`operatorImageTag`	Permite sobrescribir la etiqueta de la `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, `0s` siendo 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>`

Opción	Descripción	Predeterminado
`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 definir 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 en `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`
`cloudProvider`	Establecer como `"Azure"` cuando se utilizan identidades gestionadas o una identidad de nube en un clúster de AKS. Establecer en «AWS» cuando se utiliza una identidad de nube en un clúster de EKS.	`""`
`cloudIdentity`	Defina la identidad de carga de trabajo («azure.workload.identity/client-id: Xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx») cuando utilice la identidad de cloud en un clúster de AKS. Establezca el rol de AWS IAM ('eks.amazonaws.com/role-arn: arn:aws:iam::123456:role/astratrident-role') cuando utilice la identidad de nube en un clúster de EKS.	`""`
`iscsiSelfHealingInterval`	El intervalo en el que se invoca la reparación automática de iSCSI.	`5m0s`

Opción	Descripción	Predeterminado
`iscsiSelfHealingWaitTime`	La duración después del cual la reparación automática de iSCSI inicia un intento de resolver una sesión obsoleta realizando un cierre de sesión y un inicio de sesión posterior.	`7m0s`

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 que un pod se ejecute en un nodo específico o preferido. Usando el`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.

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 el "proceso de implementación estándar".

Información vital sobre Astra Trident 24,06

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

información bï¿½tico sobre Astra Tridbï¿½tico

- Kubernetes 1,31 ahora es compatible con Astra Trident. Actualizar Trident antes de actualizar Kubernetes.
- Astra Trident aplica estrictamente el uso de configuración multivía en entornos SAN, con un valor recomendado de `find_multipaths: no` en archivo multipath.conf.

  El uso de una configuración que no sea multivía o el uso `find_multipaths: yes` o `find_multipaths: smart` un valor en el archivo multipath.conf provocará errores 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

Con Trident "Carta del timón"puede desplegar el operador Trident e instalar Trident en un solo paso.

Revise "descripción general de la instalación"para asegurarse de que cumple los requisitos previos de la instalación y ha seleccionado la opción de instalación correcta para su 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. Utilice `helm install` y especifique un nombre para el despliegue y la ubicación del registro de imágenes. "Imágenes Trident y CSI"Puede estar ubicado en un registro o registros diferentes, pero todas las imágenes CSI deben estar ubicadas en el mismo registro. En los ejemplos, `100.2406.0` es la versión de Astra Trident que está instalando.

Imágenes en un registro

```
helm install <name> netapp-trident/trident-operator --version
100.2406.0 --set imageRegistry=<your-registry> --create-namespace
--namespace <trident-namespace>
```

Imágenes en diferentes registros

Debe anexar `sig-storage` a `imageRegistry` para utilizar distintas ubicaciones de registro.

```
helm install <name> netapp-trident/trident-operator --version
100.2406.0 --set imageRegistry=<your-registry>/sig-storage --set
operatorImage=<your-registry>/netapp/trident-operator:24.06.0 --set
tridentAutosupportImage=<your-registry>/netapp/trident-
autosupport:24.06 --set tridentImage=<your-
registry>/netapp/trident:24.06.0 --create-namespace --namespace
<trident-namespace>
```

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

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

Opción	Descripción
`--set`	Especifique anulaciones en la línea de comandos.

Por ejemplo, para cambiar el valor predeterminado de `debug`, ejecute el siguiente `--set` comando, donde `100.2406.0` es la versión de Astra Trident que está instalando:

```
helm install <name> netapp-trident/trident-operator --version 100.2406.0
--create-namespace --namespace trident --set tridentDebug=true
```

Opciones de configuración

Esta tabla y el `values.yaml` archivo, que forma parte del diagrama Helm, proporcionan 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 información.	
`tridentControllerPluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte "Descripción de los pods de la controladora y los pods de nodo" para obtener más información.	
`tridentNodePluginNodeSelector`	Selectores de nodos adicionales para POD. Consulte "Descripción de los pods de la controladora y los pods de nodo" para obtener más información.	
`tridentNodePluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte "Descripción de los pods de la controladora y los pods de nodo" para obtener más información.	

Opción	Descripción	Predeterminado
`imageRegistry`	Identifica el registro de `trident-operator` , `trident` y otras imágenes. Déjelo vacío para aceptar el valor predeterminado.	""
`imagePullPolicy`	Establece la política de extracción de imágenes para la `trident-operator`.	`IfNotPresent`
`imagePullSecrets`	Define los secretos de extracción de imágenes para `trident-operator` , `trident` y otras imágenes.	
`kubeletDir`	Permite anular la ubicación del host del estado interno de kubelet.	`"/var/lib/kubelet"`
`operatorLogLevel`	Permite definir el nivel de log 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 para `trident-operator`.	""
`operatorImageTag`	Permite sobrescribir la etiqueta de la `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, `0s` siendo 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>`

Opción	Descripción	Predeterminado
`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 definir 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 en `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`

Personalice la instalación del operador de Trident

El operador Trident te permite personalizar la instalación de Astra Trident con los atributos de la `TridentOrchestrator` especificación. Si desea personalizar la instalación más allá de lo que `TridentOrchestrator` permiten los argumentos, considere utilizar `tridentctl` para generar manifiestos YAML personalizados para 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 que un pod se ejecute en un nodo específico o preferido. Usando el`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 donde está instalado Astra Trident. Este parámetro **no se puede actualizar después de instalar Astra Trident**. Si lo intenta, el `TridentOrchestrator` estado cambia a `Failed`. Astra Trident no está pensado para la migración entre espacios de nombres.

Esta tabla detalla `TridentOrchestrator` los atributos.

Parámetro	Descripción	Predeterminado
`namespace`	Espacio de nombres para instalar Astra Trident en	`"default"`
`debug`	Habilite la depuración para Astra Trident	`false`
`enableForceDetach`	`ontap-san` y `ontap-san-economy` solo. 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.	`false`
`windows`	La configuración `true` para activa la instalación en los nodos de trabajador de Windows.	`false`
`cloudProvider`	Establecer como `"Azure"` cuando se utilizan identidades gestionadas o una identidad de nube en un clúster de AKS. Establecer en «AWS» cuando se utiliza una identidad de nube en un clúster de EKS.	`""`
`cloudIdentity`	Defina la identidad de carga de trabajo («azure.workload.identity/client-id: Xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx») cuando utilice la identidad de cloud en un clúster de AKS. Establezca el rol de AWS IAM ('eks.amazonaws.com/role-arn: arn:aws:iam::123456:role/astratrident-role) cuando utilice la identidad de la nube en un clúster de EKS.	`""`
`IPv6`	Instale Astra Trident sobre IPv6	falso

Parámetro	Descripción	Predeterminado
`k8sTimeout`	Tiempo de espera para las operaciones de Kubernetes	`30sec`
`silenceAutosupport`	No envíe paquetes AutoSupport a NetApp automáticamente	`false`
`autosupportImage`	La imagen contenedora para telemetría AutoSupport	`"netapp/trident-autosupport:24.06"`
`autosupportProxy`	La dirección/puerto de un proxy para enviar telemetría AutoSupport	`"http://proxy.example.com:8888"`
`uninstall`	Una Marca utilizada para desinstalar Astra Trident	`false`
`logFormat`	Formato de registro de Astra Trident para utilizar [text,json]	`"text"`
`tridentImage`	Imagen de Astra Trident para instalar	`"netapp/trident:24.06"`
`imageRegistry`	Ruta al registro interno, del formato `<registry FQDN>[:port][/subpath]`	`"k8s.gcr.io/sig-storage"` (Kubernetes 1,19 o posterior) o `"quay.io/k8scsi"`
`kubeletDir`	Ruta al directorio kubelet del host	`"/var/lib/kubelet"`
`wipeout`	Una lista de recursos para eliminar y 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. Los valores válidos son: `Always` Para extraer siempre la imagen. `IfNotPresent` para extraer la imagen solo si aún no existe en el nodo. `Never` para no extraer nunca 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 sólo para `ontap-san` y. `ontap-san-economy` 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 LAS NGN de Kubernetes, no elimine `out-of-service` los daños de un nodo en mal estado hasta que se reprogramen 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 de clúster de Kubernetes haya aplicado la `node.kubernetes.io/out-of-service=nodeshutdown:NoExecute` tinte al nodo y `enableForceDetach` se establezca 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).

> El controlador 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 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 las 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 un `cleanable` estado (se ha quitado la intonía de fuera de servicio y el nodo está en `Ready` estado) y todas las rutas obsoletas publicadas están limpias, Astra Trident readmitirá el nodo a medida que `clean` se admitan nuevos volúmenes publicados al nodo.

Configuraciones de ejemplo

Puede utilizar los atributos de Opciones de configuraciónal definir `TridentOrchestrator` para personalizar la instalación.

Configuración personalizada básica

Este es un ejemplo de una instalación 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
```

Selectores de nodos

Este ejemplo instala Astra Trident con selectores de nodos.

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

Nodos de trabajo de Windows

En este ejemplo se instala Astra Trident 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
```

Identidades administradas en un cluster AKS

En este ejemplo se instala Astra Trident para habilitar identidades gestionadas en un clúster de AKS.

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

Identidad de nube en un clúster de AKS

En este ejemplo se instala Astra Trident para usarlo con una identidad de cloud en un clúster de AKS.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  cloudProvider: "Azure"
  cloudIdentity: 'azure.workload.identity/client-id: xxxxxxxx-xxxx-
xxxx-xxxx-xxxxxxxxxxx'
```

Identidad de nube en un clúster de EKS

En este ejemplo se instala Astra Trident para usarlo con una identidad de cloud en un clúster de AKS.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  cloudProvider: "AWS"
  cloudIdentity: "'eks.amazonaws.com/role-arn:
arn:aws:iam::123456:role/astratrident-role'"
```

Instale utilizando trimentctl

Instale utilizando trimentctl

Puede instalar Astra Trident mediante `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 `tridentctl` el despliegue, consulte "Personalice la implementación trimentctl".

Información vital sobre Astra Trident 24,06

- 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 aplica estrictamente el uso de configuración multivía en entornos SAN, con un valor recomendado de `find_multipaths: no` en archivo multipath.conf.

  El uso de una configuración que no sea multivía o el uso `find_multipaths: yes` o `find_multipaths: smart` un valor en el archivo multipath.conf provocará errores de montaje. Trident ha recomendado el uso de `find_multipaths: no` desde la versión 21,07.

Instala Astra Trident mediante `tridentctl`

Revise "descripción general de la instalación"para asegurarse de que cumple los requisitos previos de la instalación y ha seleccionado la opción de instalación correcta para su entorno.

Antes de empezar

Antes de comenzar la instalación, inicie sesión en el host Linux y compruebe que está gestionando un

funcionamiento "Clúster de Kubernetes compatible"y que dispone de la Privileges necesaria.

> Con OpenShift, use `oc` en lugar de en `kubectl` 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 última versión del instalador de Trident de "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/v24.06.0/trident-
installer-24.06.0.tar.gz
tar -xf trident-installer-24.06.0.tar.gz
cd trident-installer
```

Paso 2: Instale Astra Trident

Instala Astra Trident en el espacio de nombres deseado ejecutando `tridentctl install` el 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:24.06 --trident
-image <your-registry>/trident:24.06.0
```

Imágenes en diferentes registros

Debe anexar `sig-storage` a `imageRegistry` para utilizar distintas ubicaciones de registro.

```
./tridentctl install -n trident --image-registry <your-registry>/sig-
storage --autosupport-image <your-registry>/netapp/trident-
autosupport:24.06 --trident-image <your-
registry>/netapp/trident:24.06.0
```

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=24.06.0
INFO Trident installation succeeded.
....
```

Compruebe la instalación

Puede verificar la instalación utilizando el estado de creación del 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 **running**, la plataforma no se instaló. Utilice `-d` para "activa el modo de depuración"solucionar el problema.

Utilizando `tridentctl`

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

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 24.06.0        | 24.06.0        |
+----------------+----------------+
```

Configuraciones de ejemplo

Los siguientes ejemplos proporcionan configuraciones de ejemplo para instalar Astra Trident utilizando `tridentctl`.

Nodos de Windows

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

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

Forzar separación

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

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

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 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 de sidecar CSI necesarias en un repositorio privado, es preferible especificar la ubicación de ese repositorio mediante el `--image-registry` switch, que toma el formulario `<registry FQDN>[:port]`.

Si está utilizando una distribución de Kubernetes, donde `kubelet` mantiene sus datos en una ruta distinta a 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` parámetro se crean los siguientes archivos YAML en el directorio del instalador `setup`:

- `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 generar estos archivos, puede modificarlos según sus necesidades y luego utilizarlos `--use-custom-yaml` para instalar el despliegue personalizado.

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

Utilice Astra Trident

Prepare el nodo de trabajo

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

Seleccionar las herramientas adecuadas

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

Herramientas de NFS

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

Herramientas iSCSI

"Instale las herramientas iSCSI" si está utilizando `ontap-san`: `ontap-san-economy`, , `solidfire-san`.

Herramientas de NVMe

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

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

Detección del servicio de nodos

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

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

Revisar los eventos

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

```
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 NFS

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

RHEL 8 O POSTERIOR

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

Ubuntu

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

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

Volúmenes iSCSI

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

Funcionalidades de reparación automática de iSCSI

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

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

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

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

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

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

- Faltan sesiones iSCSI
- Faltan LUN

Puntos a tener en cuenta antes de actualizar Trident

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

Instale las herramientas iSCSI

Instale las herramientas iSCSI mediante los comandos del sistema operativo.

Antes de empezar

- Cada nodo del clúster de Kubernetes debe tener un IQN único. **Este es un requisito previo necesario**.
- Si utiliza RHCOS versión 4,5 o posterior, u otra distribución de Linux compatible con RHEL, con `solidfire-san` el controlador y Element OS 12,5 o anterior, asegúrese de que el algoritmo de autenticación CHAP se haya configurado en MD5 en `/etc/iscsi/iscsid.conf`. Los algoritmos CHAP seguros compatibles con FIPS SHA1, SHA-256 y SHA3-256 están disponibles con Element 12,7.

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

- Cuando se utilicen los nodos de trabajo que ejecutan RHEL/RedHat CoreOS con VP iSCSI, especifique `discard` mountOption en StorageClass para realizar la recuperación de espacio en línea. Consulte "Documentación de redhat".

RHEL 8 O POSTERIOR

1. Instale los siguientes paquetes del sistema:

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

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

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

3. Configure el escaneo en manual:

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

4. Activar accesos múltiples:

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

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

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

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

6. Activar e iniciar `iscsi`:

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

Ubuntu

1. Instale los siguientes paquetes del sistema:

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

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

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

3. Configure el escaneo en manual:

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

4. Activar accesos múltiples:

```
sudo tee /etc/multipath.conf <<-'EOF
defaults {
    user_friendly_names yes
    find_multipaths no
}
EOF
sudo systemctl enable --now multipath-tools.service
sudo service multipath-tools restart
```

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

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

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

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

Configure o deshabilite la reparación automática de iSCSI

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

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

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

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

Timón

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

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

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

tridentctl

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

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

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

Volúmenes NVMe/TCP

Instale las herramientas NVMe mediante los comandos de su sistema operativo.

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

RHEL 9

```
sudo yum install nvme-cli
sudo yum install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

Ubuntu

```
sudo apt install nvme-cli
sudo apt -y install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

Verifique la instalación

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

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

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

Configurar y gestionar back-ends

Configurar los back-ends

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

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

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

Azure NetApp Files

Configure un back-end de Azure NetApp Files

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

Información del controlador de Azure NetApp Files

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

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

Consideraciones

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

Identidades administradas para AKS

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

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

Operador de Trident

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

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

Timón

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

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

<code>tridentctl</code>

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

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

Identidad de nube para AKS

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

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

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

Operador de Trident

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

Por ejemplo:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "Azure"
  *cloudIdentity: 'azure.workload.identity/client-id: xxxxxxxx-xxxx-
xxxx-xxxx-xxxxxxxxxxx'*
```

Timón

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

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

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

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

<code>tridentctl</code>

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

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

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

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

Prepárese para configurar un back-end de Azure NetApp Files

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

Requisitos previos para volúmenes NFS y SMB

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

Para configurar y utilizar un "Azure NetApp Files" backend, necesita lo siguiente:

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

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

Rol de contribuyente personalizado

```
{
    "id": "/subscriptions/<subscription-
id>/providers/Microsoft.Authorization/roleDefinitions/<role-
definition-id>",
    "properties": {
        "roleName": "custom-role-with-limited-perms",
        "description": "custom role providing limited
permissions",
        "assignableScopes": [
            "/subscriptions/<subscription-id>"
        ],
        "permissions": [
            {
                "actions": [

"Microsoft.NetApp/netAppAccounts/capacityPools/read",

"Microsoft.NetApp/netAppAccounts/capacityPools/write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/read",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/delete",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
read",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
delete",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/MountTarge
ts/read",
                    "Microsoft.Network/virtualNetworks/read",

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

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
```

```
ions/write",

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

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

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

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

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

Requisitos adicionales para volúmenes SMB

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

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

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

- Proxy CSI configurado como servicio de Windows. Para configurar un `csi-proxy`, consulte "GitHub:

Proxy CSI"o "GitHub: Proxy CSI para Windows" para los nodos de Kubernetes que se ejecutan en Windows.

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

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

Opciones de configuración del back-end

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

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

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

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

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

Parámetro	Descripción	Predeterminado
`debugTraceFlags`	Indicadores de depuración que se deben usar para la solución de problemas. Ejemplo, `\{"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 nulas. El valor predeterminado es nulo en volúmenes de NFS.	`nfs`
`supportedTopologies`	Representa una lista de regiones y zonas soportadas por este backend. Para obtener más información, consulte "Utilice Topología CSI".	

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

Permisos y recursos necesarios

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

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

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

Tipo	Formato
Grupo de recursos	<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 información.

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

Configuraciones de ejemplo

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

Configuración mínima

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

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

```
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
  tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
  clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
  clientSecret: SECRET
  location: eastus
```

Identidades administradas para AKS

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

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  capacityPools: ["ultra-pool"]
  resourceGroups: ["aks-ami-eastus-rg"]
  netappAccounts: ["smb-na"]
  virtualNetwork: eastus-prod-vnet
  subnet: eastus-anf-subnet
```

Identidad de nube para AKS

Esta configuración de backend omite `tenantID`, `clientID`, y `clientSecret`, que son opcionales cuando se utiliza una identidad de nube.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  capacityPools: ["ultra-pool"]
  resourceGroups: ["aks-ami-eastus-rg"]
  netappAccounts: ["smb-na"]
  virtualNetwork: eastus-prod-vnet
  subnet: eastus-anf-subnet
  location: eastus
  subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
```

Configuración de niveles de servicio específica con filtros de pools de capacidad

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

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

Configuración avanzada

Esta configuración de back-end reduce aún más el alcance de la ubicación de volúmenes en una única subred y también modifica algunos valores predeterminados de aprovisionamiento de volúmenes.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
serviceLevel: Ultra
capacityPools:
- application-group-1/account-1/ultra-1
- application-group-1/account-1/ultra-2
virtualNetwork: my-virtual-network
subnet: my-subnet
networkFeatures: Standard
nfsMountOptions: vers=3,proto=tcp,timeo=600
limitVolumeSize: 500Gi
defaults:
  exportRule: 10.0.0.0/24,10.0.1.0/24,10.0.2.100
  snapshotDir: 'true'
  size: 200Gi
  unixPermissions: '0777'
```

Configuración de pool virtual

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

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
resourceGroups:
- application-group-1
networkFeatures: Basic
nfsMountOptions: vers=3,proto=tcp,timeo=600
labels:
  cloud: azure
storage:
- labels:
    performance: gold
  serviceLevel: Ultra
  capacityPools:
  - ultra-1
  - ultra-2
  networkFeatures: Standard
- labels:
    performance: silver
  serviceLevel: Premium
  capacityPools:
  - premium-1
- labels:
    performance: bronze
  serviceLevel: Standard
  capacityPools:
  - standard-1
  - standard-2
```

Configuración de topologías admitidas

Astra Trident facilita el aprovisionamiento de volúmenes para cargas de trabajo según regiones y zonas de disponibilidad. El `supportedTopologies` bloque en esta configuración de backend se utiliza para proporcionar una lista de regiones y zonas por backend. Los valores de región y zona especificados aquí deben coincidir con los valores de región y zona de las etiquetas de cada nodo de clúster de Kubernetes. Estas regiones y zonas representan la lista de valores permitidos que se pueden proporcionar en una clase de almacenamiento. Para las clases de almacenamiento que contengan un subconjunto de las regiones y zonas proporcionadas en un back-end, Astra Trident creará volúmenes en la región y la zona mencionadas. Para obtener más información, consulte "Utilice Topología CSI".

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

Definiciones de clases de almacenamiento

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

Ejemplo de definiciones utilizando `parameter.selector` el campo

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

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=gold"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: silver
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: bronze
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=bronze"
allowVolumeExpansion: true
```

Definiciones de ejemplo de volúmenes SMB

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

Configuración básica en el espacio de nombres predeterminado

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

Uso de diferentes secretos por espacio de nombres

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

Uso de diferentes secretos por volumen

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: ${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

`nasType: smb` Filtros para pools que admiten volúmenes SMB. `nasType: nfs` O `nasType: null` filtros para pools NFS.

Cree el back-end

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

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

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

```
tridentctl logs
```

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

NetApp Volumes para Google Cloud

Configura un back-end de Google Cloud NetApp Volumes

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

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

Detalles del controlador de Google Cloud NetApp Volumes

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

Controlador	Protocolo	VolumeMode	Modos de acceso compatibles	Sistemas de archivos compatibles
`google-cloud-netapp-volumes`	NFS	Sistema de archivos	RWO, ROX, RWX, RWOP	`nfs`

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

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

Requisitos previos para volúmenes de NFS

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

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

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

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

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

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

Opciones de configuración del back-end

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

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

Parámetro	Descripción	Predeterminado
`location`	La ubicación de Google Cloud, donde Astra Trident crea volúmenes de GCNV. Al crear clústeres de Kubernetes entre regiones, los volúmenes creados en un `location` se pueden usar en cargas de trabajo programadas en nodos de varias regiones de Google Cloud. El tráfico entre regiones conlleva un coste adicional.	
`apiKey`	La clave de la API para la cuenta de servicio de Google Cloud con `netappcloudvolumes.admin` el rol. Incluye el contenido en formato JSON del archivo de clave privada de una cuenta de servicio de Google Cloud (copiado literal en el archivo de configuración de back-end). `apiKey`Debe incluir pares clave-valor para las siguientes claves `type:,,,,project_id client_email client_id,,auth_uri token_uri auth_provider_x509_cert_url`, y `client_x509_cert_url`.	
`nfsMountOptions`	Control preciso de las opciones de montaje NFS.	"nfsvers=3"
`limitVolumeSize`	No se puede aprovisionar si el tamaño del volumen solicitado es superior a este valor.	"" (no se aplica de forma predeterminada)
`serviceLevel`	El nivel de servicio de un pool de almacenamiento y sus volúmenes. Los valores son `flex`, `standard`, `premium`, o `extreme`.	
`network`	La red de Google Cloud utilizada para los volúmenes GCNV.	
`debugTraceFlags`	Indicadores de depuración que se deben usar para la solución de problemas. Ejemplo, `{"api":false, "method":true}`. No lo utilice a menos que esté solucionando problemas y necesite un volcado de registro detallado.	nulo
`supportedTopologies`	Representa una lista de regiones y zonas soportadas por este backend. Para obtener más información, consulte "Utilice Topología CSI". Por ejemplo: `supportedTopologies: - topology.kubernetes.io/region: europe-west6 topology.kubernetes.io/zone: europe-west6-b`	

Opciones de aprovisionamiento de volúmenes

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

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

Configuraciones de ejemplo

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

Configuración mínima

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

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

```
---

apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: 'f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec'
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    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-----

---

apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: '123455380079'
  location: europe-west6
  serviceLevel: premium
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: '103346282737811234567'
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Configuración con filtro StoragePools

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: 'f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec'
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    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-----

---

apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
```

```
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: '123455380079'
  location: europe-west6
  serviceLevel: premium
  storagePools:
  - premium-pool1-europe-west6
  - premium-pool2-europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: '103346282737811234567'
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Configuración de pool virtual

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

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

Para obtener más información, consulte "Pools virtuales".

```
---

apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: 'f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec'
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    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-----

---

apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: '123455380079'
  location: europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: '103346282737811234567'
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
  defaults:
    snapshotReserve: '10'
    exportRule: 10.0.0.0/24
  storage:
    - labels:
        performance: extreme
      serviceLevel: extreme
      defaults:
        snapshotReserve: '5'
        exportRule: 0.0.0.0/0
    - labels:
        performance: premium
      serviceLevel: premium
```

```
    - labels:
        performance: standard
      serviceLevel: standard
```

El futuro

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

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

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

```
kubectl get tridentbackendconfig

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

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

```
tridentctl logs
```

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

Más ejemplos

Ejemplos de definición de la clase de almacenamiento

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

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

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

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

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: extreme-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=extreme"
  backendType: "google-cloud-netapp-volumes"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: premium-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=premium"
  backendType: "google-cloud-netapp-volumes"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: standard-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=standard"
  backendType: "google-cloud-netapp-volumes"
```

Para obtener más información sobre las clases de almacenamiento, consulte "Cree una clase de almacenamiento".

Ejemplo de definición de PVC

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

Para verificar si la RVP está vinculada, ejecute el siguiente comando:

```
kubectl get pvc gcnv-nfs-pvc

NAME          STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS AGE
gcnv-nfs-pvc  Bound    pvc-b00f2414-e229-40e6-9b16-ee03eb79a213  100Gi
RWX            gcnv-nfs-sc  1m
```

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

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

Detalles del controlador de Google Cloud

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

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

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

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

- **CVS-Performance**: El tipo de servicio predeterminado Astra Trident. Este tipo de servicio optimizado para el rendimiento es más adecuado para cargas de trabajo de producción que valoran el rendimiento. El tipo

de servicio CVS-Performance es una opción de hardware que admite volúmenes con un tamaño mínimo de 100 GIB. Puede elegir una de "tres niveles de servicio"las opciones:

  - `standard`
  - `premium`
  - `extreme`

- **CVS**: El tipo de servicio CVS proporciona una alta disponibilidad zonal con niveles de rendimiento limitados a moderados. El tipo de servicio CVS es una opción de software que usa pools de almacenamiento para admitir volúmenes de solo 1 GIB. El pool de almacenamiento puede contener hasta 50 volúmenes en los que todos los volúmenes comparten la capacidad y el rendimiento del pool. Puede elegir una de "dos niveles de servicio"las opciones:

  - `standardsw`
  - `zoneredundantstandardsw`

Lo que necesitará

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

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

Opciones de configuración del back-end

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

Parámetro	Descripción	Predeterminado
`version`		Siempre 1
`storageDriverName`	Nombre del controlador de almacenamiento	"gcp-cvs"
`backendName`	Nombre personalizado o el back-end de almacenamiento	Nombre de controlador + "_" + parte de la clave de API
`storageClass`	Parámetro opcional utilizado para especificar el tipo de servicio CVS. Se utiliza `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.	

Parámetro	Descripción	Predeterminado
`hostProjectNumber`	Se requiere si se utiliza una red VPC compartida. En este escenario, `projectNumber` es el proyecto de servicio y `hostProjectNumber` es el proyecto host.	
`apiRegion`	Región de Google Cloud en la que Astra Trident crea volúmenes de Cloud Volumes Service. Al crear clústeres de Kubernetes entre regiones, los volúmenes creados en un `apiRegion` se pueden usar en cargas de trabajo programadas en nodos de varias regiones de Google Cloud. El tráfico entre regiones conlleva un coste adicional.	
`apiKey`	La clave de la API para la cuenta de servicio de Google Cloud con `netappcloudvolumes.admin` el rol. Incluye el contenido en formato JSON del archivo de clave privada de una cuenta de servicio de Google Cloud (copiado literal en el archivo de configuración de back-end).	
`proxyURL`	URL de proxy si se requiere servidor proxy para conectarse a la cuenta CVS. El servidor proxy puede ser un proxy HTTP o HTTPS. En el caso de un proxy HTTPS, se omite la validación de certificados para permitir el uso de certificados autofirmados en el servidor proxy. No se admiten los servidores proxy con autenticación habilitada.	
`nfsMountOptions`	Control preciso de las opciones de montaje NFS.	"nfsvers=3"
`limitVolumeSize`	No se puede aprovisionar si el tamaño del volumen solicitado es superior a este valor.	"" (no se aplica de forma predeterminada)
`serviceLevel`	El nivel de servicio CVS-Performance o CVS para nuevos volúmenes. Los valores de rendimiento de CVS son `standard` `premium` , o `extreme`. Los valores de 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 entre regiones, su definición de StorageClass para `allowedTopologies` debe incluir todas las regiones. Por ejemplo:
	`- key: topology.kubernetes.io/region`
	`values:`
	`- us-east1`
	`- europe-west1`	

Opciones de aprovisionamiento de volúmenes

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

Parámetro	Descripción	Predeterminado
`exportRule`	Las reglas de exportación de nuevos volúmenes. Debe ser una lista separada por comas con cualquier combinación de direcciones IPv4 o subredes IPv4 en notación CIDR.	"0.0.0.0/0"
`snapshotDir`	Acceso al `.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-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: '123456789012345678901'
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
```

Ejemplo 2: Configuración de nivel de servicio

Este ejemplo muestra las opciones de configuración del back-end, incluidos el nivel de servicio y los valores predeterminados de volumen.

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
apiRegion: us-west2
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: '123456789012345678901'
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
proxyURL: http://proxy-server-hostname/
nfsMountOptions: vers=3,proto=tcp,timeo=600
limitVolumeSize: 10Ti
serviceLevel: premium
defaults:
  snapshotDir: 'true'
  snapshotReserve: '5'
  exportRule: 10.0.0.0/24,10.0.1.0/24,10.0.2.100
  size: 5Ti
```

Ejemplo 3: Configuración de pool virtual

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

Aquí, los valores predeterminados específicos se establecen para todos los pools virtuales, que establecen `snapshotReserve` el valor en 5% y el `exportRule` en 0,0.0,0/0. Los pools virtuales se definen en la `storage` sección. Cada pool virtual individual define su propio `serviceLevel` y algunos pools sobrescriben los valores por defecto. Las etiquetas de pool virtual se utilizaron para diferenciar los pools en función de `performance` y. `protection`

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
apiRegion: us-west2
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: '123456789012345678901'
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
nfsMountOptions: vers=3,proto=tcp,timeo=600
defaults:
  snapshotReserve: '5'
  exportRule: 0.0.0.0/0
labels:
  cloud: gcp
region: us-west2
storage:
- labels:
    performance: extreme
    protection: extra
  serviceLevel: extreme
```

```
  defaults:
    snapshotDir: 'true'
    snapshotReserve: '10'
    exportRule: 10.0.0.0/24
- labels:
    performance: extreme
    protection: standard
  serviceLevel: extreme
- labels:
    performance: premium
    protection: extra
  serviceLevel: premium
  defaults:
    snapshotDir: 'true'
    snapshotReserve: '10'
- labels:
    performance: premium
    protection: standard
  serviceLevel: premium
- labels:
    performance: standard
  serviceLevel: standard
```

Definiciones de clases de almacenamiento

Las siguientes definiciones de StorageClass se aplican al ejemplo de configuración de pool virtual. Con `parameters.selector`, puede especificar para cada clase de almacenamiento el pool virtual utilizado para alojar un volumen. Los aspectos definidos en el pool elegido serán el volumen.

Ejemplo de clase de almacenamiento

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

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

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

Ejemplos de tipo de servicio CVS

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

Ejemplo 1: Configuración mínima

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

```
---
version: 1
storageDriverName: gcp-cvs
projectNumber: '012345678901'
storageClass: software
apiRegion: us-east4
apiKey:
  type: service_account
  project_id: my-gcp-project
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
  client_id: '123456789012345678901'
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
serviceLevel: standardsw
```

Ejemplo 2: Configuración del pool de almacenamiento

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

```
---
version: 1
storageDriverName: gcp-cvs
backendName: gcp-std-so-with-pool
projectNumber: '531265380079'
apiRegion: europe-west1
apiKey:
  type: service_account
  project_id: cloud-native-data
  private_key_id: "<id_value>"
  private_key: |-
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@cloud-native-
data.iam.gserviceaccount.com
  client_id: '107071413297115343396'
  auth_uri: https://accounts.google.com/o/oauth2/auth
  token_uri: https://oauth2.googleapis.com/token
  auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
  client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/cloudvolumes-admin-
sa%40cloud-native-data.iam.gserviceaccount.com
storageClass: software
zone: europe-west1-b
network: default
storagePools:
- 1bc7f380-3314-6005-45e9-c7dc8c2d7509
serviceLevel: Standardsw
```

El futuro

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

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

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

```
tridentctl logs
```

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

Configure un back-end de NetApp HCI o SolidFire

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

Detalles del controlador de elementos

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

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

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

Antes de empezar

Necesitarás lo siguiente antes de crear un backend de elemento.

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

Opciones de configuración del back-end

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

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

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

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

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

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

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

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

Astra Trident copia las etiquetas presentes en un pool de almacenamiento a la LUN de almacenamiento del entorno de administración al aprovisionar. Para mayor comodidad, los administradores de almacenamiento pueden definir etiquetas por pool virtual y agrupar volúmenes por etiqueta.

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

```
---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0
```

```
SVIP: "<svip>:3260"
TenantName: "<tenant>"
UseCHAP: true
Types:
- Type: Bronze
  Qos:
    minIOPS: 1000
    maxIOPS: 2000
    burstIOPS: 4000
- Type: Silver
  Qos:
    minIOPS: 4000
    maxIOPS: 6000
    burstIOPS: 8000
- Type: Gold
  Qos:
    minIOPS: 6000
    maxIOPS: 8000
    burstIOPS: 10000
type: Silver
labels:
  store: solidfire
  k8scluster: dev-1-cluster
region: us-east-1
storage:
- labels:
    performance: gold
    cost: '4'
  zone: us-east-1a
  type: Gold
- labels:
    performance: silver
    cost: '3'
  zone: us-east-1b
  type: Silver
- labels:
    performance: bronze
    cost: '2'
  zone: us-east-1c
  type: Bronze
- labels:
    performance: silver
    cost: '1'
  zone: us-east-1d
```

Las siguientes definiciones de StorageClass se refieren a los pools virtuales anteriores. En

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

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

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-gold-four
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=gold; cost=4"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-three
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver; cost=3"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-bronze-two
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=bronze; cost=2"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-one
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver; cost=1"
  fsType: "ext4"
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=silver"
  fsType: "ext4"
```

Obtenga más información

- "Los grupos de acceso de volúmenes"

Controladores para SAN de ONTAP

Información general del controlador de SAN de ONTAP

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

Información sobre el controlador de SAN de ONTAP

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

> Si utiliza Astra Control para protección, recuperación y movilidad, lea Compatibilidad de controladores Astra Control.

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

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

Compatibilidad de controladores Astra Control

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

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

Permisos de usuario

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

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

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

Consideraciones adicionales para NVMe/TCP

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

que se incluye:

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

Astra Trident no es compatible:

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

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

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

Requisitos

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

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

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

Autentique el backend de ONTAP

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

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

Puede actualizar los back-ends existentes para moverse entre métodos basados en credenciales y basados en certificados. Sin embargo, solo se admite un método de autenticación a la vez. Para cambiar a un método de autenticación diferente, debe eliminar el método existente de la configuración del back-end.

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

Habilite la autenticación basada en credenciales

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

Una definición de backend de ejemplo tendrá este aspecto:

YAML

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

JSON

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

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

Habilite la autenticación basada en certificados

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

- ClientCertificate: Valor codificado en base64 del certificado de cliente.
- ClientPrivateKey: Valor codificado en base64 de la clave privada asociada.
- TrustedCACertificate: Valor codificado en base64 del certificado de CA de confianza. Si se utiliza una CA

de confianza, se debe proporcionar este parámetro. Esto se puede ignorar si no se utiliza ninguna CA de confianza.

Un flujo de trabajo típico implica los pasos siguientes.

Pasos

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

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=admin"
```

2. Añada un certificado de CA de confianza al clúster ONTAP. Es posible que ya sea gestionado por el administrador de almacenamiento. Ignore si no se utiliza ninguna CA de confianza.

```
security certificate install -type server -cert-name <trusted-ca-cert-
name> -vserver <vserver-name>
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled
true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca
<cert-authority>
```

3. Instale el certificado y la clave de cliente (desde el paso 1) en el clúster ONTAP.

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

4. Confirme que el rol de inicio de sesión de seguridad de ONTAP admite `cert` el método de autenticación.

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

5. Probar la autenticación mediante un certificado generado. Reemplace <LIF de gestión de ONTAP> y <vserver name> por la IP de LIF de gestión y el nombre de SVM.

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns="http://www.netapp.com/filer/admin" version="1.21"
vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. Codifique certificados, claves y certificados de CA de confianza con Base64.

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

7. Cree un backend utilizando los valores obtenidos del paso anterior.

```
cat cert-backend.json
{
"version": 1,
"storageDriverName": "ontap-san",
"backendName": "SanBackend",
"managementLIF": "1.2.3.4",
"svm": "vserver_test",
"clientCertificate": "Faaaakkkkeeee...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 ejecutar `tridentctl backend update`.

```
cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-san",
"backendName": "SanBackend",
"managementLIF": "1.2.3.4",
"svm": "vserver_test",
"username": "vsadmin",
"password": "password",
"storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend SanBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

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

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

Autentica conexiones con CHAP bidireccional

Astra Trident puede autenticar sesiones iSCSI con CHAP bidireccional para `ontap-san` los controladores y. `ontap-san-economy` Esto requiere la activación de `useCHAP` la opción en la definición de backend. Cuando se establece en `true`, Astra Trident configura la seguridad del iniciador predeterminado de la máquina virtual de almacenamiento como CHAP bidireccional y establece el nombre de usuario y los secretos del archivo back-end. NetApp recomienda utilizar CHAP bidireccional para autenticar las conexiones. Consulte la siguiente configuración de ejemplo:

```
---
version: 1
storageDriverName: ontap-san
backendName: ontap_san_chap
managementLIF: 192.168.0.135
svm: ontap_iscsi_svm
useCHAP: true
username: vsadmin
password: password
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
```

El `useCHAP` parámetro es una opción booleana que puede configurarse una sola vez. De forma predeterminada, se establece en FALSE. Después de configurarlo en true, no puede establecerlo en false.

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

Cómo funciona

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

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

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

Rotar las credenciales y actualizar los back-ends

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

Al actualizar los secretos CHAP para un backend, debe utilizar `tridentctl` para actualizar el backend. No actualice las credenciales en el clúster de almacenamiento a través de la interfaz de usuario de CLI/ONTAP, ya que Astra Trident no podrá recoger estos cambios.

```
cat backend-san.json
{
    "version": 1,
    "storageDriverName": "ontap-san",
    "backendName": "ontap_san_chap",
    "managementLIF": "192.168.0.135",
    "svm": "ontap_iscsi_svm",
    "useCHAP": true,
    "username": "vsadmin",
    "password": "password",
    "chapInitiatorSecret": "cl9qxUpDaTeD",
    "chapTargetInitiatorSecret": "rqxigXgkeUpDaTeD",
    "chapTargetUsername": "iJF4heBRT0TCwxyz",
    "chapUsername": "uh2aNCLSd6cNwxyz",
}

./tridentctl update backend ontap_san_chap -f backend-san.json -n trident
+----------------+----------------+--------------------------------------
+--------+---------+
|   NAME         | STORAGE DRIVER |                 UUID                 |
STATE   | VOLUMES |
+----------------+----------------+--------------------------------------
+--------+---------+
| ontap_san_chap | ontap-san      | aa458f3b-ad2d-4378-8a33-1a472ffbeb5c |
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 SAN de ONTAP

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

Opciones de configuración del back-end

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

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

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

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

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

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

Parámetro	Descripción	Predeterminado
`spaceAllocation`	Asignación de espacio para las LUN	verdadero
`spaceReserve`	Modo de reserva de espacio; «ninguno» (fino) o «volumen» (grueso)	ninguno
`snapshotPolicy`	Política de Snapshot que se debe usar	ninguno
`qosPolicy`	Grupo de políticas de calidad de servicio que se asignará a los volúmenes creados. Elija uno de 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	""

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

Ejemplos de aprovisionamiento de volúmenes

Aquí hay un ejemplo con los valores predeterminados definidos:

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: trident_svm
username: admin
password: <password>
labels:
  k8scluster: dev2
  backend: dev2-sanbackend
storagePrefix: alternate-trident
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: standard
  spaceAllocation: 'false'
  snapshotPolicy: default
  snapshotReserve: '10'
```

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

Para los back-ends que definen `snapshotReserve`, Astra Trident calcula el tamaño de los volúmenes de la siguiente manera:

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

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

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

Ejemplos de configuración mínima

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

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

Ejemplo de SAN ONTAP

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

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
labels:
  k8scluster: test-cluster-1
  backend: testcluster1-sanbackend
username: vsadmin
password: <password>
```

Ejemplo de economía de SAN ONTAP

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

1. ejemplo

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

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

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

Ejemplo de autenticación basada en certificados

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

```
---
version: 1
storageDriverName: ontap-san
backendName: DefaultSANBackend
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
```

Ejemplos de CHAP bidireccional

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

Ejemplo de CHAP de SAN de ONTAP

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
labels:
  k8scluster: test-cluster-1
  backend: testcluster1-sanbackend
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
username: vsadmin
password: <password>
```

Ejemplo de CHAP de economía de SAN ONTAP

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

Ejemplo de NVMe/TCP

Debe tener una SVM configurada con NVMe en el back-end de ONTAP. Esta es una configuración de back-end básica para NVMe/TCP.

```
---
version: 1
backendName: NVMeBackend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_nvme
username: vsadmin
password: password
sanType: nvme
useREST: true
```

Ejemplo de configuración de backend con nameTemplate

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

Ejemplos de back-ends con pools virtuales

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

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

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

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'
```

Ejemplo de NVMe/TCP

```
---
version: 1
storageDriverName: ontap-san
sanType: nvme
managementLIF: 10.0.0.1
svm: nvme_svm
username: vsadmin
password: <password>
useREST: true
defaults:
  spaceAllocation: 'false'
  encryption: 'true'
storage:
- labels:
    app: testApp
    cost: '20'
  defaults:
    spaceAllocation: 'false'
    encryption: 'false'
```

Asigne los back-ends a StorageClass

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

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

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

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

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

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

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

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

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
```

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

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

- `my-test-app-sc`StorageClass se asignará al `testAPP` pool virtual del `ontap-san` controlador con `sanType: nvme`. Esta es la única oferta de piscina `testApp`.

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

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

Controladores para NAS de ONTAP

Información general del controlador de NAS de ONTAP

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

Información sobre el controlador de NAS de ONTAP

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

> Si utiliza Astra Control para protección, recuperación y movilidad, lea Compatibilidad de controladores Astra Control.

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

Compatibilidad de controladores Astra Control

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

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

Permisos de usuario

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

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

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

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

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

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

Requisitos

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

Autentique el backend de ONTAP

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

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

Puede actualizar los back-ends existentes para moverse entre métodos basados en credenciales y basados en certificados. Sin embargo, solo se admite un método de autenticación a la vez. Para cambiar a un método de autenticación diferente, debe eliminar el método existente de la configuración del back-end.

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

Habilite la autenticación basada en credenciales

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

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

Una definición de backend de ejemplo tendrá este aspecto:

YAML

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

JSON

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

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

Habilite la autenticación basada en certificados

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

- ClientCertificate: Valor codificado en base64 del certificado de cliente.
- ClientPrivateKey: Valor codificado en base64 de la clave privada asociada.
- TrustedCACertificate: Valor codificado en base64 del certificado de CA de confianza. Si se utiliza una CA de confianza, se debe proporcionar este parámetro. Esto se puede ignorar si no se utiliza ninguna CA de confianza.

Un flujo de trabajo típico implica los pasos siguientes.

Pasos

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

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=vsadmin"
```

2. Añada un certificado de CA de confianza al clúster ONTAP. Es posible que ya sea gestionado por el administrador de almacenamiento. Ignore si no se utiliza ninguna CA de confianza.

```
security certificate install -type server -cert-name <trusted-ca-cert-
name> -vserver <vserver-name>
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled
true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca
<cert-authority>
```

3. Instale el certificado y la clave de cliente (desde el paso 1) en el clúster ONTAP.

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

4. Confirme que el rol de inicio de sesión de seguridad de ONTAP admite `cert` el método de autenticación.

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

5. Probar la autenticación mediante un certificado generado. Reemplace <LIF de gestión de ONTAP> y <vserver name> por la IP de LIF de gestión y el nombre de SVM. Debe asegurarse de que el LIF tenga su política de servicio establecida en `default-data-management`.

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns="http://www.netapp.com/filer/admin" version="1.21"
vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. Codifique certificados, claves y certificados de CA de confianza con Base64.

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

7. Cree un backend utilizando los valores obtenidos del paso anterior.

```
cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "NasBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"clientCertificate": "Faaaakkkkeeee...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 ejecutar `tridentctl update backend`.

```
cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "NasBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"username": "vsadmin",
"password": "password",
"storagePrefix": "myPrefix_"
}

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

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

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

Gestione las políticas de exportación de NFS

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

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

- Astra Trident puede gestionar dinámicamente la propia política de exportación; en este modo de funcionamiento, el administrador de almacenamiento especifica una lista de bloques CIDR que representan direcciones IP admisibles. Astra Trident agrega automáticamente las IP de nodo que se incluyen en estos rangos a la directiva de exportación. Como alternativa, cuando no se especifican CIDR,

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

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

Gestione de forma dinámica políticas de exportación

Astra Trident proporciona la capacidad de gestionar dinámicamente las políticas de exportación para los back-ends de ONTAP. De este modo, el administrador de almacenamiento puede especificar un espacio de direcciones permitido para las IP de nodos de trabajo, en lugar de definir reglas explícitas de forma manual. Simplifica en gran medida la gestión de políticas de exportación; las modificaciones de la política de exportación ya no requieren intervención manual en el clúster de almacenamiento. Además, esto ayuda a restringir el acceso al clúster de almacenamiento solo a nodos de trabajo con IP en el rango especificado, lo que permite una gestión automatizada y de gran granularidad.

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

Ejemplo

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

```
---
version: 1
storageDriverName: ontap-nas
backendName: ontap_nas_auto_export
managementLIF: 192.168.0.135
svm: svm1
username: vsadmin
password: password
autoExportCIDRs:
- 192.168.0.0/24
autoExportPolicy: true
```

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

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

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

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

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

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

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

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

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

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

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

> Si se elimina un back-end con políticas de exportación gestionadas automáticamente, se eliminará la política de exportación creada de forma dinámica. Si se vuelve a crear el back-end, se trata como un nuevo back-end y dará lugar a la creación de una nueva política de exportación.

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

Prepárese para aprovisionar los volúmenes de SMB

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

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

Antes de empezar

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

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

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

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

Pasos

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

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

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

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

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

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

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

c. Compruebe que se ha creado el recurso compartido:

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

Consulte "Cree un recurso compartido de SMB"para obtener información detallada.

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

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

Opciones y ejemplos de configuración NAS de ONTAP

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

Opciones de configuración del back-end

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

Parámetro	Descripción	Predeterminado
`version`		Siempre 1
`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 de controlador + «_» + LIF de datos
`managementLIF`	Dirección IP de un clúster o una LIF de gestión de SVM Se puede especificar un nombre de dominio completo (FQDN). Puede configurarse para que utilice direcciones IPv6 si Astra Trident se instaló mediante la marca IPv6. Las direcciones IPv6 deben definirse entre corchetes, `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` como . Para una conmutación de sitios MetroCluster fluida, consulte [mcc-best].	“10.0.0.1”, “[2001:1234:abcd::fefe]”
`dataLIF`	Dirección IP de LIF de protocolo. Recomendamos especificar `dataLIF`. En caso de no proporcionar esta información, Astra Trident busca las LIF de datos desde la SVM. Puede especificar un nombre de dominio completo (FQDN) para las operaciones de montaje de NFS, lo que permite crear un DNS round-robin para lograr el equilibrio de carga entre varios LIF de datos. Se puede cambiar después del ajuste inicial. Consulte . Puede configurarse para que utilice direcciones IPv6 si Astra Trident se instaló mediante la marca IPv6. Las direcciones IPv6 deben definirse entre corchetes, `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` como . **Omitir para MetroCluster.** Consulte la [mcc-best].	Dirección especificada o derivada de la SVM, si no se especifica (no recomendada)
`svm`	Máquina virtual de almacenamiento para usar **Omitir para MetroCluster.** Consulte la [mcc-best].	Derivada si se especifica una SVM `managementLIF`
`autoExportPolicy`	Habilite la creación y actualización automática de la política de exportación [Boolean]. Mediante las `autoExportPolicy` opciones y `autoExportCIDRs`, Astra Trident puede gestionar automáticamente las políticas de exportación.	falso
`autoExportCIDRs`	Lista de CIDRs para filtrar las IP del nodo de Kubernetes contra cuando `autoExportPolicy` se habilita. Mediante las `autoExportPolicy` opciones y `autoExportCIDRs`, Astra Trident puede gestionar automáticamente las políticas de exportación.	[«0,0.0,0/0», «:/0»]

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

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

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

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

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

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

Ejemplos de aprovisionamiento de volúmenes

Aquí hay un ejemplo con los valores predeterminados definidos:

```
---
version: 1
storageDriverName: ontap-nas
backendName: customBackendName
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
labels:
  k8scluster: dev1
  backend: dev1-nasbackend
svm: trident_svm
username: cluster-admin
password: <password>
limitAggregateUsage: 80%
limitVolumeSize: 50Gi
nfsMountOptions: nfsvers=4
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: premium
  exportPolicy: myk8scluster
  snapshotPolicy: default
  snapshotReserve: '10'
```

For `ontap-nas` and `ontap-nas-flexgroups`, Astra Trident utiliza ahora un nuevo cálculo para garantizar que el tamaño del FlexVol se ajuste correctamente con el porcentaje de reserva de SnapVault y la RVP. Cuando el usuario solicita una RVP, Astra Trident crea el FlexVol original con más espacio mediante el nuevo cálculo. Este cálculo garantiza que el usuario recibe el espacio de escritura que solicitó en el PVC y no menos espacio que el que solicitó. Antes de v21.07, cuando el usuario solicita una RVP (por ejemplo, 5GIB) con el 50 por ciento de snapshotReserve, solo obtiene 2,5 GIB de espacio editable. Esto se debe a que lo que el usuario solicitó es todo el volumen y `snapshotReserve` es un porcentaje de ello. Con Trident 21,07, lo que el usuario solicita es el espacio de escritura y Astra Trident define `snapshotReserve` la cantidad como el porcentaje de todo el volumen. Esto no se aplica a `ontap-nas-economy`. Vea el siguiente ejemplo para ver cómo funciona:

El cálculo es el siguiente:

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

Para snapshotReserve = 50 % y la solicitud de RVP = 5 GIB, el tamaño total del volumen es 2/.5 = 10 GIB y el tamaño disponible es de 5 GIB, lo que es lo que solicitó el usuario en la solicitud de RVP. El `volume show` comando debería mostrar resultados similares a este ejemplo:

Los back-ends existentes de instalaciones anteriores aprovisionan volúmenes como se explicó anteriormente al actualizar Astra Trident. En el caso de los volúmenes que creó antes de actualizar, debe cambiar el tamaño de sus volúmenes para que se observe el cambio. Por ejemplo, una RVP de 2GiB GB con `snapshotReserve=50` versiones anteriores dio como resultado un volumen que proporciona 1GiB GB de espacio editable. Cambiar el tamaño del volumen a 3 GIB, por ejemplo, proporciona a la aplicación 3 GIB de espacio editable en un volumen de 6 GIB.

Ejemplos de configuración mínima

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

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

Ejemplo de economía de NAS ONTAP

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

Ejemplo de FlexGroup NAS de ONTAP

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

Ejemplo de MetroCluster

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

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

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

Ejemplo de volúmenes de SMB

```
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
nasType: smb
securityStyle: ntfs
unixPermissions: ""
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

Ejemplo de autenticación basada en certificados

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

```
---
version: 1
backendName: DefaultNASBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.15
svm: nfs_svm
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

Ejemplo de política de exportación automática

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

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

Ejemplo de direcciones IPv6

Este ejemplo se muestra `managementLIF` usando una dirección IPv6.

```
---
version: 1
storageDriverName: ontap-nas
backendName: nas_ipv6_backend
managementLIF: "[5c5d:5edf:8f:7657:bef8:109b:1b41:d491]"
labels:
  k8scluster: test-cluster-east-1a
  backend: test1-ontap-ipv6
svm: nas_ipv6_svm
username: vsadmin
password: password
```

Ejemplo de Amazon FSx para ONTAP mediante volúmenes de bloque de mensajes del servidor

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

```
---
version: 1
backendName: SMBBackend
storageDriverName: ontap-nas
managementLIF: example.mgmt.fqdn.aws.com
nasType: smb
dataLIF: 10.0.0.15
svm: nfs_svm
smbShare: smb-share
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

Ejemplo de configuración de backend con nameTemplate

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

Ejemplos de back-ends con pools virtuales

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

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

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

Ejemplo de NAS ONTAP

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
svm: svm_nfs
username: admin
password: <password>
nfsMountOptions: nfsvers=4
defaults:
  spaceReserve: none
  encryption: 'false'
  qosPolicy: standard
labels:
  store: nas_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
- labels:
    app: msoffice
    cost: '100'
  zone: us_east_1a
  defaults:
    spaceReserve: volume
    encryption: 'true'
    unixPermissions: '0755'
    adaptiveQosPolicy: adaptive-premium
- labels:
    app: slack
    cost: '75'
  zone: us_east_1b
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0755'
- labels:
    department: legal
    creditpoints: '5000'
  zone: us_east_1b
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0755'
- labels:
    app: wordpress
```

```
    cost: '50'
  zone: us_east_1c
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0775'
- labels:
    app: mysqldb
    cost: '25'
  zone: us_east_1d
  defaults:
    spaceReserve: volume
    encryption: 'false'
    unixPermissions: '0775'
```

Ejemplo de FlexGroup NAS de ONTAP

```
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: <password>
defaults:
  spaceReserve: none
  encryption: 'false'
labels:
  store: flexgroup_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
- labels:
    protection: gold
    creditpoints: '50000'
  zone: us_east_1a
  defaults:
    spaceReserve: volume
    encryption: 'true'
    unixPermissions: '0755'
- labels:
    protection: gold
    creditpoints: '30000'
  zone: us_east_1b
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0755'
- labels:
    protection: silver
    creditpoints: '20000'
  zone: us_east_1c
  defaults:
    spaceReserve: none
    encryption: 'true'
    unixPermissions: '0775'
- labels:
    protection: bronze
    creditpoints: '10000'
  zone: us_east_1d
  defaults:
```

```
    spaceReserve: volume
    encryption: 'false'
    unixPermissions: '0775'
```

Ejemplo de economía de NAS ONTAP

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

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

Asigne los back-ends a StorageClass

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

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

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

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

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

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

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

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

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
```

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

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

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

`dataLIF`Actualice tras la configuración inicial

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

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

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

Amazon FSX para ONTAP de NetApp

Utilice Astra Trident con Amazon FSX para ONTAP de NetApp

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

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

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

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

Requisitos

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

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

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

Consideraciones

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

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

Autenticación

Astra Trident ofrece dos modos de autenticación.

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

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

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

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

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

Obtenga más información

- "Documentación de Amazon FSX para ONTAP de NetApp"
- "Publicación del blog en Amazon FSX para ONTAP de NetApp"

Cree un rol de IAM y AWS Secret

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

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

Crear secreto de AWS Secret Manager

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

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

Crear política de IAM

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

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

Policy JSON archivo:

```
policy.json:
{
    "Statement": [
        {
            "Action": [
                "fsx:DescribeFileSystems",
                "fsx:DescribeVolumes",
                "fsx:CreateVolume",
                "fsx:RestoreVolumeFromSnapshot",
                "fsx:DescribeStorageVirtualMachines",
                "fsx:UntagResource",
                "fsx:UpdateVolume",
                "fsx:TagResource",
                "fsx:DeleteVolume"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": "secretsmanager:GetSecretValue",
            "Effect": "Allow",
            "Resource": "arn:aws:secretsmanager:<aws-region>:<aws-account-
id>:secret:<aws-secret-manager-name>"
        }
    ],
    "Version": "2012-10-17"
}
```

Rol Crear e IAM para la cuenta de servicio

En el siguiente ejemplo, se crea un rol de IAM para la cuenta de servicio en EKS:

```
eksctl create iamserviceaccount --name trident-controller --namespace trident
--cluster <my-cluster> --role-name <AmazonEKS_FSxN_CSI_DriverRole> --role-only
--attach-policy-arn arn:aws:iam::aws:policy/service-
role/AmazonFSxNCSIDriverPolicy --approve
```

Instale Astra Trident

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

Puedes instalar Astra Trident mediante uno de los siguientes métodos:

- Timón

- Complemento EKS

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

Instala Astra Trident mediante helm

1. Descargue el paquete de instalador de Astra Trident

   El paquete de instalación de Astra Trident incluye todo lo necesario para poner en marcha al operador de Trident e instalar Astra Trident. Descargue y extraiga la última versión del instalador de Astra Trident de la sección Activos de GitHub.
   ```
   wget https://github.com/NetApp/trident/releases/download/v24.06.0/trident-
   installer-24.06.0.tar.gz
   tar -xf trident-installer-24.06.0.tar.gz
   cd trident-installer
   ```

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

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

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

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

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

   ```
   helm list -n trident
   ```

   ```
   NAME                NAMESPACE   REVISION    UPDATED
   STATUS       CHART                          APP VERSION

   trident-operator    trident     1           2024-10-14 14:31:22.463122
   +0300 IDT    deployed     trident-operator-100.2406.1    24.06.1
   ```

Instala Astra Trident a través del complemento EKS

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

Requisitos previos

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

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

Habilita el complemento Astra Trident para AWS

Clúster EKS

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

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

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

Consola de gestión

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

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

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

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

CLI DE AWS

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

```
add-on.json
{

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

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

2. Instala el complemento de Astra Trident EKS»

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

Actualiza el complemento EKS de Astra Trident

Clúster EKS

- Compruebe la versión actual de su complemento FSxN Trident CSI. Sustituya `my-cluster` por el nombre del clúster.
  `eksctl get addon --name netapp_trident-operator --cluster my-cluster`

  Ejemplo de salida:

```
NAME                          VERSION              STATUS    ISSUES
IAMROLE    UPDATE AVAILABLE    CONFIGURATION VALUES
netapp_trident-operator    v24.6.1-eksbuild.1    ACTIVE    0
{"cloudIdentity":"'eks.amazonaws.com/role-arn:
arn:aws:iam::139763910815:role/AmazonEKS_FSXN_CSI_DriverRole'"}
```

- Actualice el complemento a la versión devuelta bajo ACTUALIZACIÓN DISPONIBLE en la salida del paso anterior.
  `eksctl update addon --name netapp_trident-operator --version v24.6.1-eksbuild.1 --cluster my-cluster --force`

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

Consola de gestión

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

CLI DE AWS

El siguiente ejemplo actualiza el complemento EKS:

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

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

Desinstale/quite el complemento de Astra Trident EKS

Tienes dos opciones para eliminar un complemento de Amazon EKS:

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

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

Clúster EKS

El siguiente comando desinstala el complemento EKS de Astra Trident:
`eksctl delete addon --cluster K8s-arm --name netapp_trident-operator`

Consola de gestión

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

CLI DE AWS

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

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

Configure el backend de almacenamiento

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

YAML

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

JSON

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

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

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

FSX para ONTAP detalles del controlador

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

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

Para obtener información detallada sobre el conductor, consulte "Controladores de NAS"y "Controladores de SAN".

Configuraciones de ejemplo

Configuración para AWS FSx para ONTAP con administrador secreto

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

Configuración de la clase de almacenamiento para volúmenes SMB

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

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

Configuración avanzada de backend y ejemplos

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

Parámetro	Descripción	Ejemplo
`version`		Siempre 1
`storageDriverName`	Nombre del controlador de almacenamiento	`ontap-nas`, , , , `ontap-nas-economy` `ontap-nas-flexgroup` `ontap-san` , `ontap-san-economy`
`backendName`	Nombre personalizado o el back-end de almacenamiento	Nombre del conductor + “_” + dataLIF

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

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

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

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

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

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

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

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

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

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

Prepárese para aprovisionar los volúmenes de SMB

Puede aprovisionar volúmenes SMB con `ontap-nas` el controlador. Antes de completar Integración de controladores ONTAP SAN y NASlos siguientes pasos.

Antes de empezar

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

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

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

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

Pasos

1. Cree recursos compartidos de SMB. Puede crear los recursos compartidos de administrador de SMB de dos maneras mediante el "Consola de administración de Microsoft"complemento Carpetas compartidas o mediante la CLI de ONTAP. Para crear los recursos compartidos de SMB mediante la CLI de ONTAP:
   a. Si es necesario, cree la estructura de ruta de acceso de directorio para el recurso compartido.

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

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

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

c. Compruebe que se ha creado el recurso compartido:

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

Consulte "Cree un recurso compartido de SMB"para obtener información detallada.

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

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

Configure una clase de almacenamiento y la RVP

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

Cree una clase de almacenamiento

Configurar un objeto de Kubernetes StorageClass

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

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

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

Cree una clase de almacenamiento

Pasos

1. Se trata de un objeto de Kubernetes, así que utilícelo `kubectl` para crearlo en Kubernetes.

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

2. Ahora debería ver una clase de almacenamiento * Basic-csi* tanto en Kubernetes como en Astra Trident, y Astra Trident debería haber descubierto las piscinas en el back-end.

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

Cree el VP y la RVP

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

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

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

Manifiestos de muestra

Manifiesto de muestra de volumen persistente

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

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

Manifiestos de muestra de PersistentVolumeClaim

Estos ejemplos muestran opciones básicas de configuración de PVC.

PVC con acceso RWO

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

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

PVC con NVMe/TCP

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

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

Cree el VP y la RVP

Pasos

1. Cree el VP.

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

2. Compruebe el estado de PV.

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

3. Cree la RVP.

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

4. Verifique el estado de la RVP.

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

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

Atributos de Astra Trident

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

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

Atributo	Tipo	Valores	Oferta	Solicitud	Admitido por
Tipo de backendType	cadena	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san, gcp-cvs, azure-netapp-files, ontap-san-economy	Pool pertenece a este tipo de backend	Backend especificado	Todos los conductores
snapshot	bool	verdadero, falso	El pool admite volúmenes con Snapshot	Volumen con snapshots habilitadas	ontap-nas, ontap-san, solidfire-san y gcp-cvs
clones	bool	verdadero, falso	Pool admite el clonado de volúmenes	Volumen con clones habilitados	ontap-nas, ontap-san, solidfire-san y gcp-cvs
cifrado	bool	verdadero, falso	El pool admite volúmenes cifrados	Volumen con cifrado habilitado	ontap-nas, ontap-nas-economy, ontap-nas-flexgroups, ontap-san
IOPS	int	entero positivo	El pool es capaz de garantizar IOPS en este rango	El volumen garantizado de estas IOPS	solidfire-san

Esta versión 1: No es compatible con sistemas ONTAP Select

Despliegue la aplicación de muestra

Despliegue la aplicación de muestra.

Pasos

1. Monte el volumen en un pod.

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

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

```
kind: Pod
apiVersion: v1
metadata:
  name: pv-pod
spec:
  volumes:
    - name: pv-storage
      persistentVolumeClaim:
       claimName: basic
  containers:
    - name: pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: pv-storage
```

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

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

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

```
Filesystem                                                      Size
Used Avail Use% Mounted on
192.168.188.78:/trident_pvc_ae45ed05_3ace_4e7c_9080_d2a83ae03d06  1.1G
320K  1.0G   1% /my/mount/path
```

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

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

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

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

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

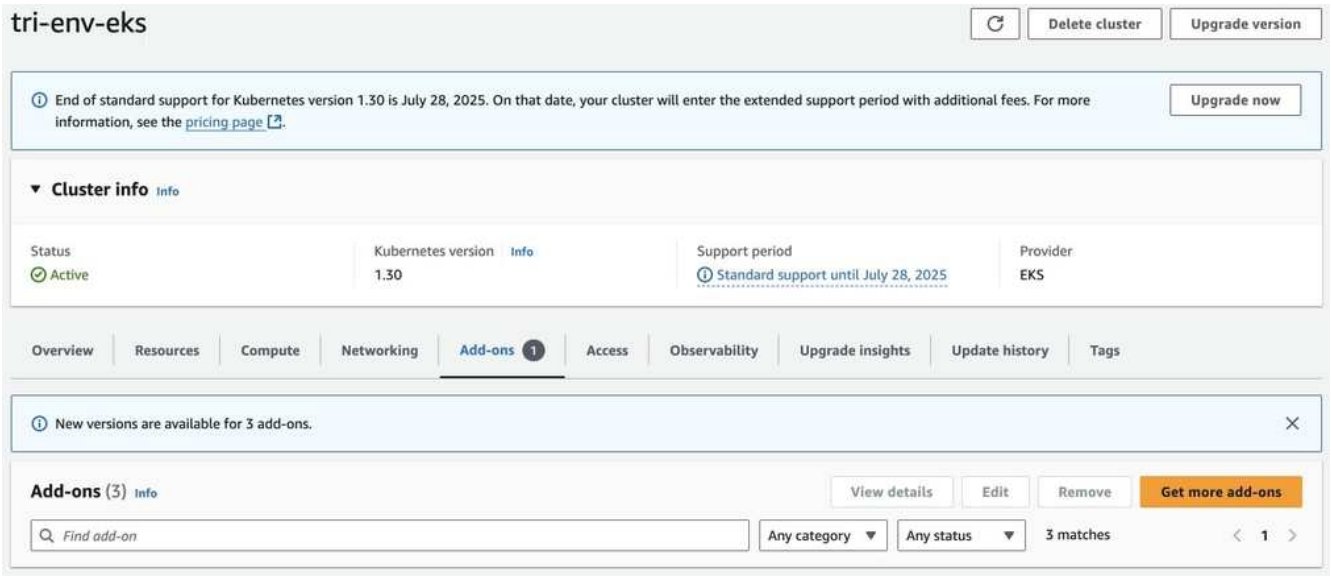
Requisitos previos

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

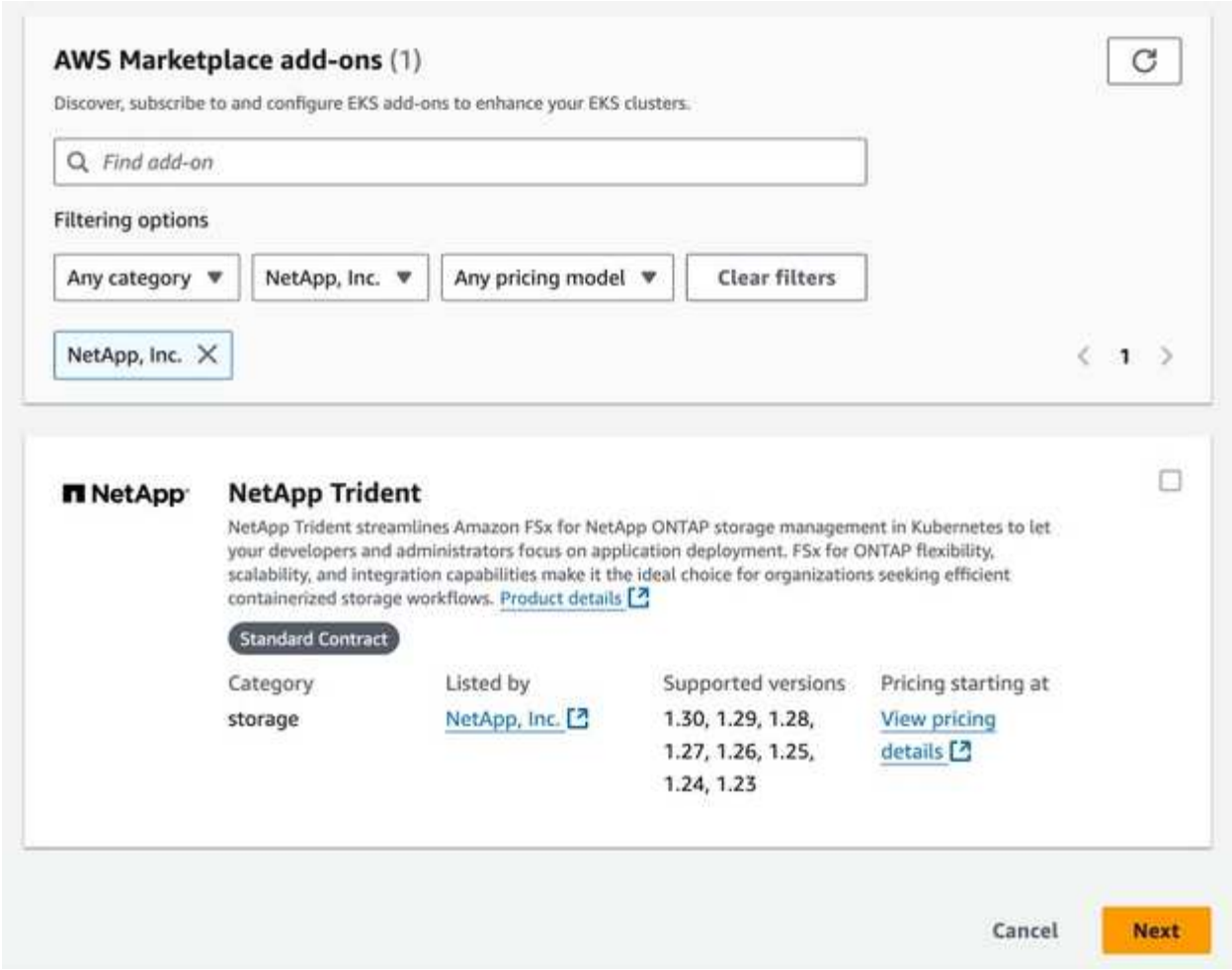
- Una cuenta de clúster de Amazon EKS con suscripción complementaria
- Permisos de AWS para AWS Marketplace:
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- Tipo de AMI: Amazon Linux 2 (AL2_x86_64) o Amazon Linux 2 ARM(AL2_ARM_64)
- Tipo de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

Pasos

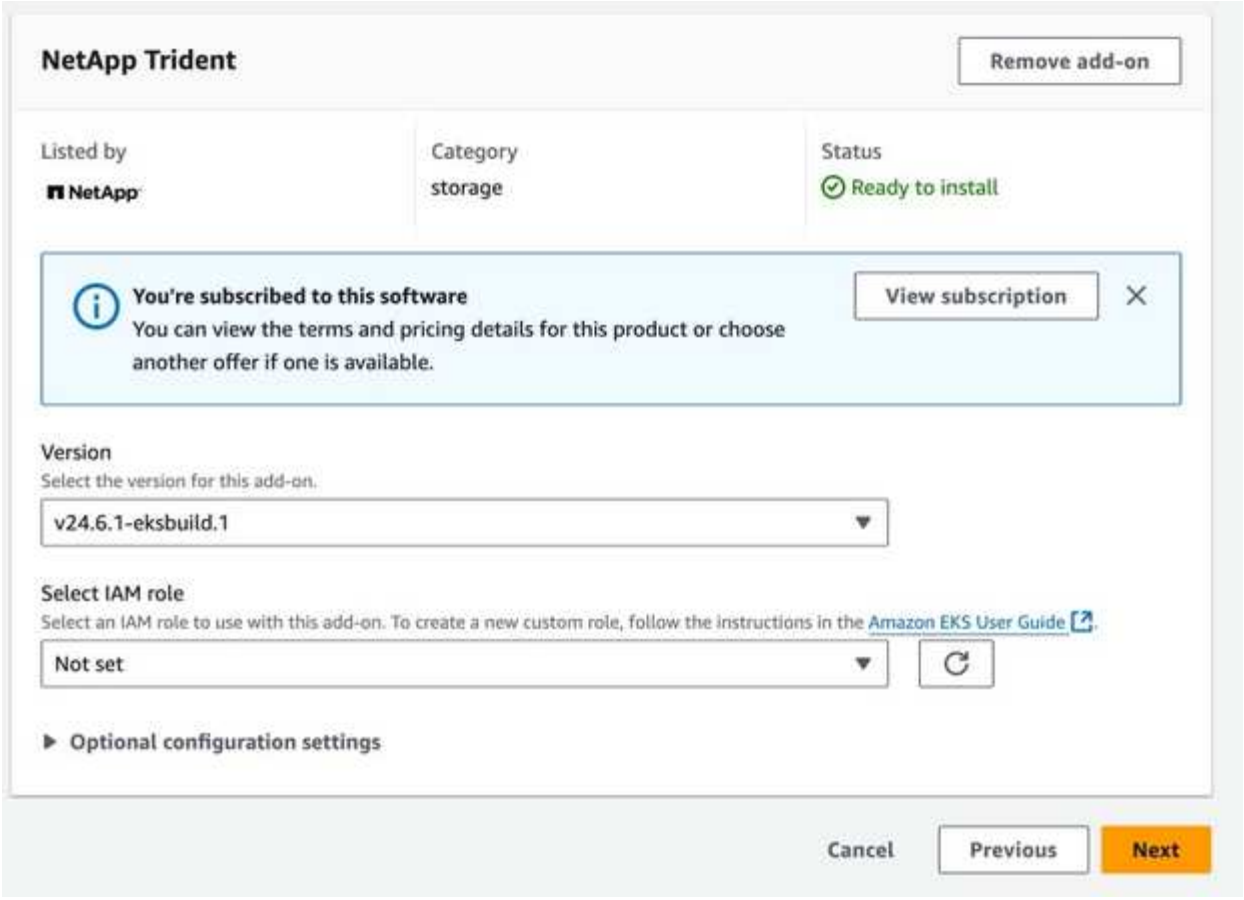
1. En tu clúster de EKS Kubernetes, navega a la pestaña **Add-ons**.



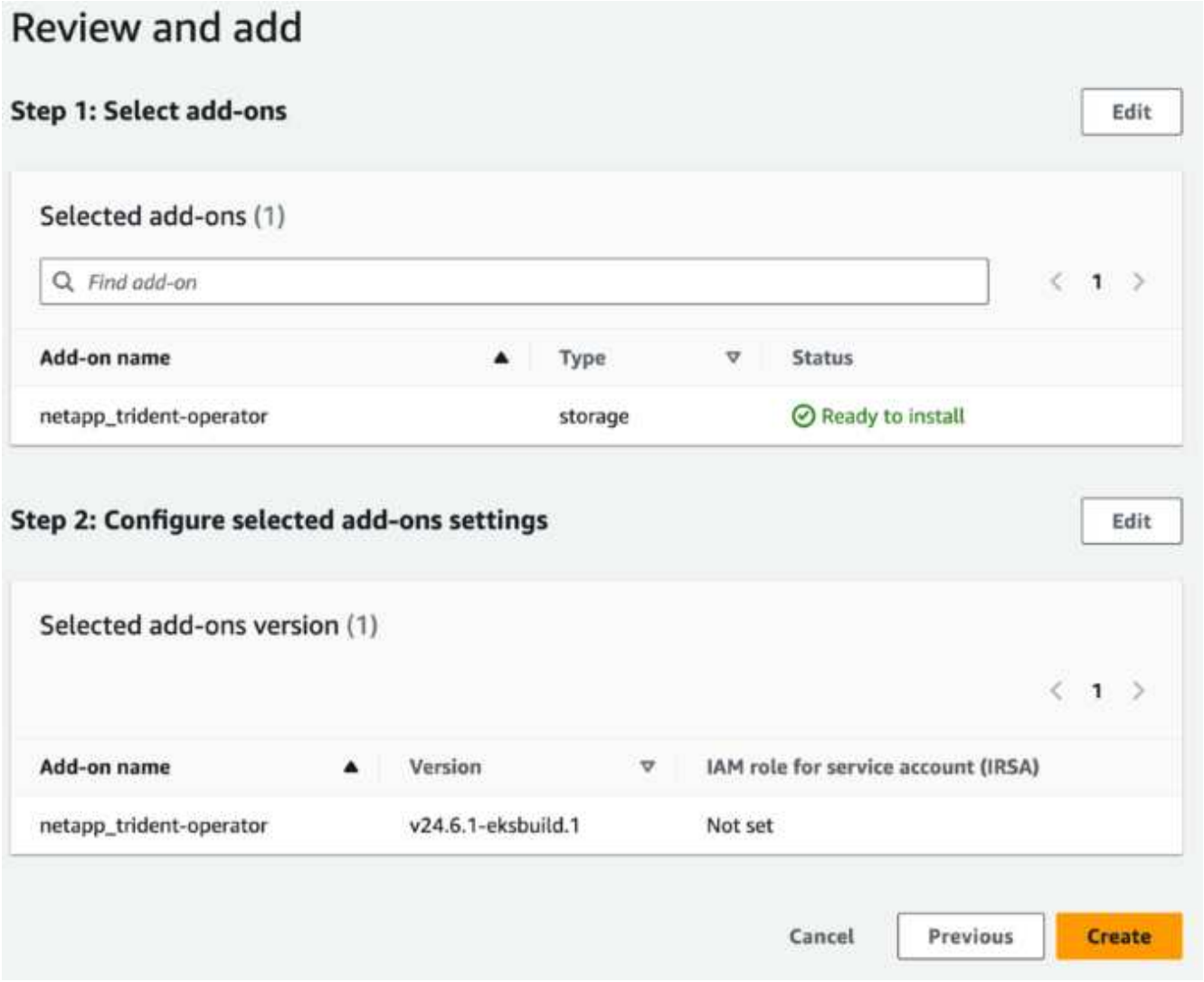
2. Vaya a **AWS Marketplace add-ons** y elija la categoría *storage*.



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



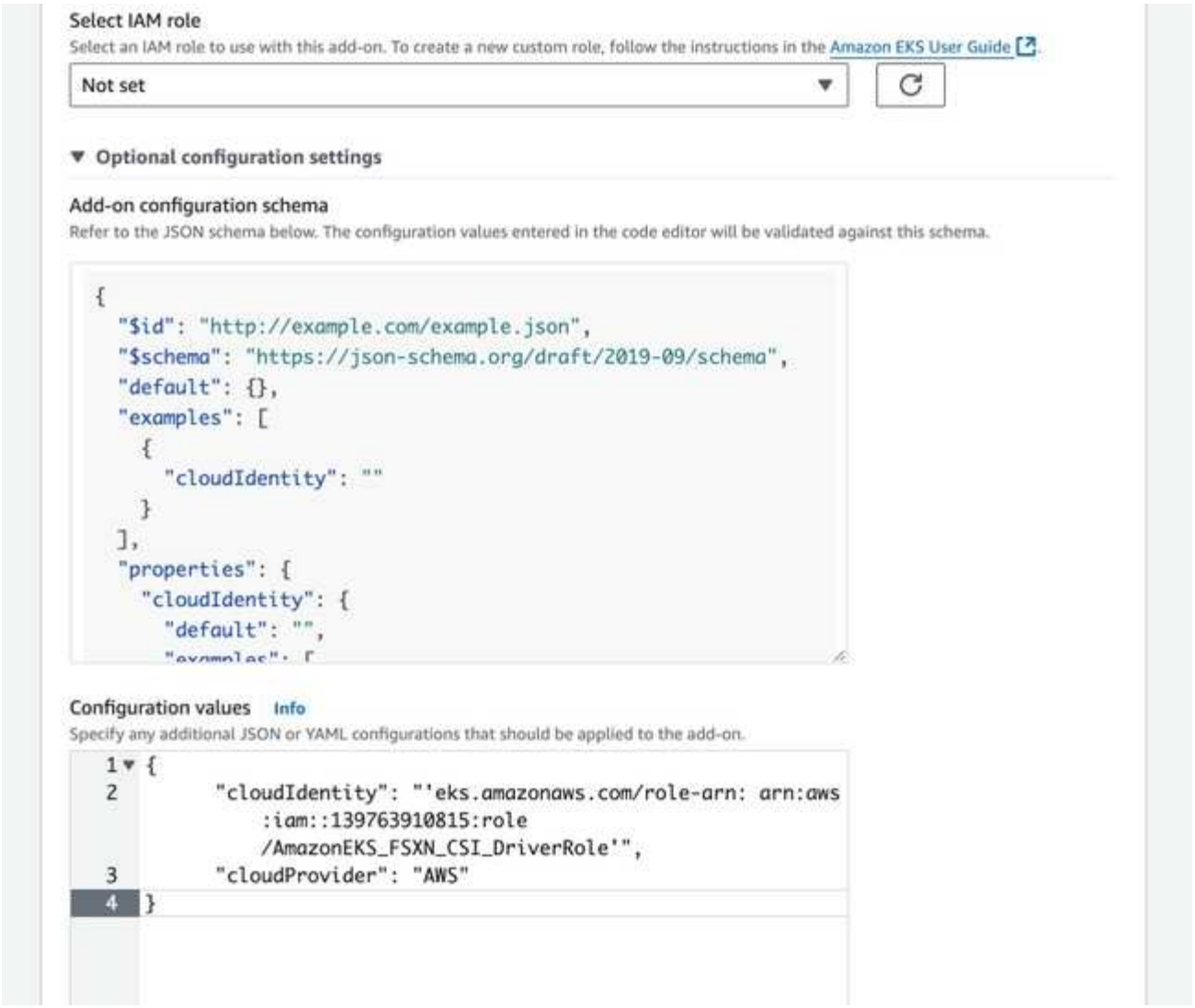
5. Seleccione la opción Rol IAM que desea heredar del nodo.



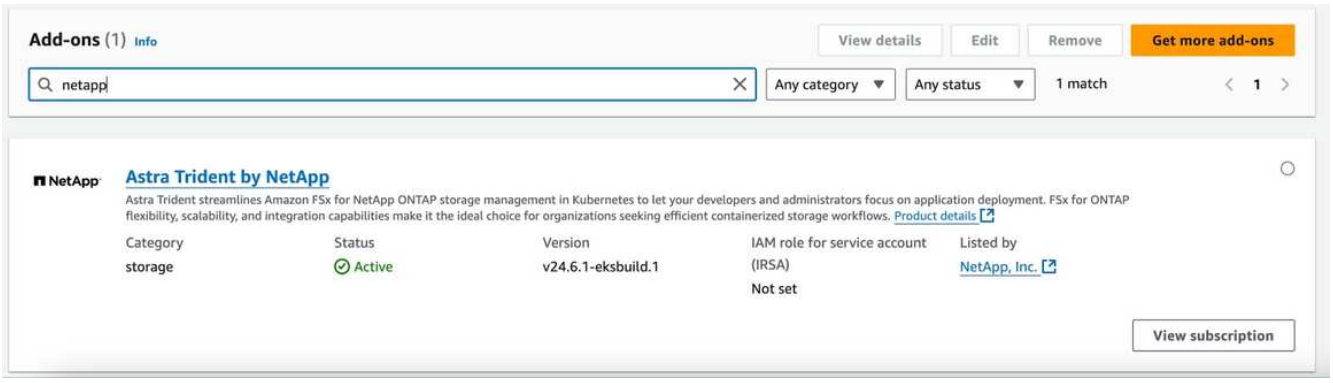
6. (Opcional) Configure cualquier configuración opcional según sea necesario y seleccione **Siguiente**.

   Siga el esquema de configuración **Add-On** y establezca el parámetro configurationValues en la sección **Valores de configuración** en el Role-arn que creó en el paso anterior (el valor debe tener el siguiente formato `eks.amazonaws.com/role-arn: arn:aws:iam::464262061435:role/AmazonEKS_FSXN_CSI_DriverRole`:). Si selecciona Sustituir para el método de resolución de conflictos, una o más de las configuraciones del complemento existente se pueden sobrescribir con la configuración del complemento Amazon EKS. Si no habilita esta opción y existe un conflicto con la configuración existente, se producirá un error en la operación. Puede utilizar el mensaje de error resultante para solucionar el conflicto. Antes de seleccionar esta opción, asegúrese de que el complemento de Amazon EKS no gestiona la configuración que necesita para autogestionar.

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



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



Instalar/desinstalar el complemento Astra Trident EKS mediante la CLI

Instale el complemento Astra Trident EKS mediante la CLI:

El siguiente comando de ejemplo instala el complemento EKS de Astra Trident:
`eksctl create addon --cluster K8s-arm --name netapp_trident-operator --version v24.6.1-eksbuild`

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

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

Desinstale el complemento Astra Trident EKS mediante la CLI:

El siguiente comando desinstala el complemento EKS de Astra Trident:
`eksctl delete addon --cluster K8s-arm --name netapp_trident-operator`

Cree back-ends con kubectl

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

`TridentBackendConfig`

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

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

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

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

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

Consulte el siguiente ejemplo para conocer el formato de `TridentBackendConfig` la CR:

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

También puede echar un vistazo a los ejemplos "instalador de trident" del directorio para ver ejemplos de configuraciones para el servicio/plataforma de almacenamiento que desee.

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

La `spec` sección también incluye `credentials` campos y `deletionPolicy`, que se han introducido recientemente en el `TridentBackendConfig` CR:

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

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

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

Descripción general de los pasos

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

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

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

Paso 1: Cree un secreto de Kubernetes

Cree un secreto que contenga las credenciales de acceso para el back-end. Esto es único para cada servicio/plataforma de almacenamiento. Veamos un ejemplo:

```
kubectl -n trident create -f backend-tbc-ontap-san-secret.yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-san-secret
type: Opaque
stringData:
  username: cluster-admin
  password: t@Ax@7q(>
```

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

Descripción de campos secretos de la plataforma de almacenamiento	Secreto	Descripción de los campos
Azure NetApp Files	ID del Cliente	El ID de cliente de un registro de aplicación
Cloud Volumes Service para GCP	id_clave_privada	ID de la clave privada. Parte de la clave API de la cuenta de servicio de GCP con el rol de administrador CVS

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

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

Paso 2: Crear el `TridentBackendConfig` CR

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

```
kubectl -n trident create -f backend-tbc-ontap-san.yaml
```

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

Paso 3: Verifique el estado de la `TridentBackendConfig` CR

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

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

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

La fase puede tomar uno de los siguientes valores:

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

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

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

- `Lost`: El backend asociado con `TridentBackendConfig` el CR se eliminó accidental o deliberadamente y el `TridentBackendConfig` CR todavía tiene una referencia al backend eliminado. La `TridentBackendConfig` CR se puede eliminar independientemente del `deletionPolicy` valor.
- `Unknown`: Astra Trident no puede determinar el estado o la existencia del backend asociado con el `TridentBackendConfig` CR. Por ejemplo, si el servidor API no responde o si falta el `tridentbackends.trident.netapp.io` CRD. Esto puede requerir intervención.

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

(Opcional) Paso 4: Obtener más detalles

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

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

```
NAME                    BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8   Bound   Success   ontap-san        delete
```

Además, también puede obtener un volcado YAML/JSON de `TridentBackendConfig`.

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

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

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

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

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

Gestionar back-ends

Realice la gestión del entorno de administración con kubectl

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

Eliminar un back-end

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

Ejecute el siguiente comando:

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

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

Ver los back-ends existentes

Ejecute el siguiente comando:

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

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

Actualizar un back-end

Puede haber varias razones para actualizar un back-end:

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

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

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

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

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

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

- Si falla una actualización de back-end, el back-end continúa en su última configuración conocida. Puede ver los logs para determinar la causa ejecutando `kubectl get tbc <tbc-name> -o yaml -n trident` o `kubectl describe tbc <tbc-name> -n trident`.
- Después de identificar y corregir el problema con el archivo de configuración, puede volver a ejecutar el comando update.

Realizar la administración de back-end con trimentctl

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

Cree un back-end

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

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

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

```
tridentctl logs -n trident
```

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

Eliminar un back-end

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

1. Recupere el nombre del backend:

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

2. Eliminar el back-end:

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

Si Astra Trident ha 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` genera los objetos backend. Esto utiliza la `jq` utilidad, que necesita instalar.

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

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

Pasar entre las opciones de administración del back-end

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

Opciones para gestionar back-ends

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

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

Gestionar `tridentctl` back-ends utilizando `TridentBackendConfig`

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

Esto se aplica a las siguientes situaciones:

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

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

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

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

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

Paso 0: Identificar el back-end

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

```
tridentctl get backend ontap-nas-backend -n trident
+---------------------+----------------
```

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

cat ontap-nas-backend.json

{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "managementLIF": "10.10.10.1",
    "dataLIF": "10.10.10.2",
    "backendName": "ontap-nas-backend",
    "svm": "trident_svm",
    "username": "cluster-admin",
    "password": "admin-password",

    "defaults": {
        "spaceReserve": "none",
        "encryption": "false"
    },
    "labels":{"store":"nas_store"},
    "region": "us_east_1",
    "storage": [
        {
            "labels":{"app":"msoffice", "cost":"100"},
            "zone":"us_east_1a",
            "defaults": {
                "spaceReserve": "volume",
                "encryption": "true",
                "unixPermissions": "0755"
            }
        },
        {
            "labels":{"app":"mysqldb", "cost":"25"},
            "zone":"us_east_1d",
            "defaults": {
                "spaceReserve": "volume",
                "encryption": "false",
                "unixPermissions": "0775"
            }
```

```
        }
    ]
}
```

Paso 1: Cree un secreto de Kubernetes

Cree un secreto que contenga las credenciales del back-end, como se muestra en este ejemplo:

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

apiVersion: v1
kind: Secret
metadata:
  name: ontap-nas-backend-secret
type: Opaque
stringData:
  username: cluster-admin
  password: admin-password

kubectl create -f tbc-ontap-nas-backend-secret.yaml -n trident
secret/backend-tbc-ontap-san-secret created
```

Paso 2: Crear un `TridentBackendConfig` CR

El siguiente paso consiste en crear un `TridentBackendConfig` CR que se enlazará automáticamente a la preexistente `ontap-nas-backend` (como en este ejemplo). Asegurarse de que se cumplen los siguientes requisitos:

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

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

```
cat backend-tbc-ontap-nas.yaml
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-ontap-nas-backend
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.10.10.1
  dataLIF: 10.10.10.2
  backendName: ontap-nas-backend
  svm: trident_svm
  credentials:
    name: mysecret
  defaults:
    spaceReserve: none
    encryption: 'false'
  labels:
    store: nas_store
  region: us_east_1
  storage:
  - labels:
      app: msoffice
      cost: '100'
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: 'true'
      unixPermissions: '0755'
  - labels:
      app: mysqldb
      cost: '25'
    zone: us_east_1d
    defaults:
      spaceReserve: volume
      encryption: 'false'
      unixPermissions: '0775'

kubectl create -f backend-tbc-ontap-nas.yaml -n trident
tridentbackendconfig.trident.netapp.io/tbc-ontap-nas-backend created
```

Paso 3: Verifique el estado de la `TridentBackendConfig` CR

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

```
kubectl get tbc tbc-ontap-nas-backend -n trident
NAME                   BACKEND NAME          BACKEND UUID
PHASE   STATUS
tbc-ontap-nas-backend  ontap-nas-backend     52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7   Bound   Success

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

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

Gestionar `TridentBackendConfig` back-ends utilizando `tridentctl`

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

Paso 0: Identificar el back-end

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

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                    BACKEND NAME        BACKEND UUID
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 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

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

Paso 1: Confirme `deletionPolicy` **que está establecido en** `retain`

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

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                    BACKEND NAME        BACKEND UUID
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` esté establecido en `retain`.

Paso 2: Eliminar el `TridentBackendConfig` CR

El paso final es eliminar la `TridentBackendConfig` CR. Después de confirmar que el `deletionPolicy` está definido en `retain`, puede continuar con la eliminación:

```
kubectl delete tbc backend-tbc-ontap-san -n trident
tridentbackendconfig.trident.netapp.io "backend-tbc-ontap-san" deleted

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

Tras la eliminación `TridentBackendConfig` del objeto, Astra Trident simplemente lo elimina sin eliminar el propio back-end.

Crear y gestionar clases de almacenamiento

Cree una clase de almacenamiento

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

Configurar un objeto de Kubernetes StorageClass

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

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

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

Cree una clase de almacenamiento

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

Pasos

1. Se trata de un objeto de Kubernetes, así que utilícelo `kubectl` para crearlo en Kubernetes.

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

2. Ahora debería ver una clase de almacenamiento * Basic-csi* tanto en Kubernetes como en Astra Trident, y Astra Trident debería haber descubierto las piscinas en el back-end.

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

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

Muestras de clase de almacenamiento

Astra Trident "definiciones simples de clase de almacenamiento para back-ends específicos" proporciona .

Como alternativa, puede editar `sample-input/storage-class-csi.yaml.templ` el archivo que viene con el instalador y reemplazarlo *`BACKEND_TYPE`* por el nombre del controlador de almacenamiento.

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

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

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

Gestione las clases de almacenamiento

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

Consulte las clases de almacenamiento existentes

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

```
kubectl get storageclass
```

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

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

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

```
tridentctl get storageclass
```

- Para ver la información detallada de la clase de almacenamiento sincronizado de Astra Trident, ejecute el siguiente comando:

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

Establecer una clase de almacenamiento predeterminada

Kubernetes 1.6 añadió la capacidad de establecer un tipo de almacenamiento predeterminado. Esta es la clase de almacenamiento que se usará para aprovisionar un volumen persistente si un usuario no especifica una en una solicitud de volumen persistente (PVC).

- Defina una clase de almacenamiento predeterminada estableciendo la anotación `storageclass.kubernetes.io/is-default-class` en TRUE en la definición de la clase de almacenamiento. Según la especificación, cualquier otro valor o ausencia de la anotación se interpreta como falso.
- Puede configurar una clase de almacenamiento existente para que sea la clase de almacenamiento predeterminada mediante el siguiente comando:

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

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

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

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

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

Identifique el back-end para una clase de almacenamiento

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

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

Elimine una clase de almacenamiento

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

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

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

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

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

Aprovisione y gestione volúmenes

Aprovisione un volumen

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

Descripción general

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

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

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

Manifiestos de muestra

Manifiesto de muestra de volumen persistente

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

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

Manifiestos de muestra de PersistentVolumeClaim

Estos ejemplos muestran opciones básicas de configuración de PVC.

PVC con acceso RWO

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

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

PVC con NVMe/TCP

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

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

Muestras de manifiesto de POD

Estos ejemplos muestran configuraciones básicas para conectar la RVP a un pod.

Configuración básica

```
kind: Pod
apiVersion: v1
metadata:
  name: pv-pod
spec:
  volumes:
    - name: pv-storage
      persistentVolumeClaim:
       claimName: basic
  containers:
    - name: pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: pv-storage
```

Configuración de NVMe/TCP básica

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

Cree el VP y la RVP

Pasos

1. Cree el VP.

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

2. Compruebe el estado de PV.

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

3. Cree la RVP.

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

4. Verifique el estado de la RVP.

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

5. Monte el volumen en un pod.

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

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

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

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

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

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

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

Expanda los volúmenes

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

Expanda un volumen iSCSI

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

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

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

Edite la definición de StorageClass para definir `allowVolumeExpansion` el campo en `true`.

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

Para un StorageClass ya existente, edítelo para incluir el `allowVolumeExpansion` parámetro.

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

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

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

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

```
kubectl get pvc
NAME      STATUS   VOLUME                                     CAPACITY
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

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

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

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

```
 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: Expande el PV

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

```
kubectl edit pvc san-pvc
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: "2019-10-10T17:32:29Z"
  finalizers:
  - kubernetes.io/pvc-protection
  name: san-pvc
  namespace: default
  resourceVersion: "16609"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/san-pvc
  uid: 8a814d62-bd58-4253-b0d1-82f2885db671
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
 ...
```

Paso 5: Validar la expansión

Para validar que la ampliación ha funcionado correctamente, compruebe el tamaño del volumen PVC, PV y Astra Trident:

```
kubectl get pvc san-pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi
RWO            ontap-san      11m
kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM              STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi        RWO
Delete           Bound    default/san-pvc    ontap-san               12m
tridentctl get volumes -n trident
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
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 es compatible con la expansión de volumen para VP 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 PV de NFS, en primer lugar, el administrador debe configurar la clase de almacenamiento para permitir la expansión del volumen estableciendo `allowVolumeExpansion` el campo en `true`:

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

Si ya creó un tipo de almacenamiento sin esta opción, puede editar el tipo de almacenamiento existente mediante el uso `kubectl edit storageclass` de para permitir la expansión de volumen.

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

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

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

```
kubectl get pvc
NAME           STATUS   VOLUME
CAPACITY     ACCESS MODES   STORAGECLASS    AGE
ontapnas20mb   Bound    pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   20Mi
RWO            ontapnas        9s

kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM                  STORAGECLASS    REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   20Mi       RWO
Delete           Bound    default/ontapnas20mb   ontapnas
2m42s
```

Paso 3: Expande el PV

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

```
kubectl edit pvc ontapnas20mb
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: 2018-08-21T18:26:44Z
  finalizers:
  - kubernetes.io/pvc-protection
  name: ontapnas20mb
  namespace: default
  resourceVersion: "1958015"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/ontapnas20mb
  uid: c1bd7fa5-a56f-11e8-b8d7-fa163e59eaab
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
...
```

Paso 4: Validar la expansión

Puede validar que el tamaño de la configuración ha funcionado correctamente comprobando el tamaño del volumen PVC, PV y Astra Trident:

```
kubectl get pvc ontapnas20mb
NAME           STATUS   VOLUME
CAPACITY   ACCESS MODES   STORAGECLASS    AGE
ontapnas20mb   Bound    pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   1Gi
RWO            ontapnas        4m44s

kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM                  STORAGECLASS    REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   1Gi        RWO
Delete           Bound    default/ontapnas20mb   ontapnas
5m35s

tridentctl get volume pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 -n trident
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
| pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 | 1.0 GiB | ontapnas      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | true    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

Importar volúmenes

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

Descripción general y consideraciones

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

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

Consideraciones

Antes de importar un volumen, revise las siguientes consideraciones.

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

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

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

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

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

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

Importe un volumen

Puede usar `tridentctl import` para importar un volumen.

Pasos

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

   A continuación se muestra un ejemplo de una especificación mínima:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: my_claim
  namespace: my_namespace
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: my_storage_class
```

No incluya parámetros adicionales, como el nombre del VP o el tamaño del volumen. Esto puede provocar un error en el comando de importación.

2. Utilice `tridentctl import` el comando para especificar el nombre del back-end de Astra Trident que contiene el volumen y el nombre que identifica de forma única el volumen en el almacenamiento (por ejemplo: ONTAP FlexVol, Element Volume, ruta Cloud Volumes Service). El `-f` argumento es necesario para especificar la ruta al archivo PVC.

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

Ejemplos

Revise los siguientes ejemplos de importación de volúmenes para los controladores compatibles.

NAS de ONTAP y NAS FlexGroup de ONTAP

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

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

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

Ejemplos de NAS de ONTAP

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

Volumen gestionado

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

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

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

Volumen no gestionado

Cuando se utiliza `--no-manage` el argumento, Astra Trident no cambia el nombre del volumen.

El siguiente ejemplo importa `unmanaged_volume` en el `ontap_nas` backend:

```
tridentctl import volume nas_blog unmanaged_volume -f <path-to-pvc-
file> --no-manage

+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | 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 con `ontap-san` el controlador. La importación de volúmenes no es compatible con `ontap-san-economy` el controlador.

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

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 FlexVol al `pvc-<uuid>` formato y la LUN dentro de FlexVol a `lun0`.

El siguiente ejemplo importa el `ontap-san-managed` FlexVol que está presente en `ontap_san_default` el backend:

```
tridentctl import volume ontapsan_san_default ontap-san-managed -f pvc-
basic-import.yaml -n trident -d

+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
| pvc-d6ee4f54-4e40-4454-92fd-d00fc228d74a | 20 MiB | basic         |
block    | cd394786-ddd5-4470-adc3-10c5ce4ca757 | online | true    |
+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
```

Volumen no gestionado

El siguiente ejemplo importa `unmanaged_example_volume` en el `ontap_san` backend:

```
tridentctl import volume -n trident san_blog unmanaged_example_volume
-f pvc-import.yaml --no-manage
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
| pvc-1fc999c9-ce8c-459c-82e4-ed4380a4b228 | 1.0 GiB | san-blog      |
block    | e3275890-7d80-4af6-90cc-c7a0759f555a | online | false   |
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
```

Si tiene 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 `solidfire-san` el controlador.

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

Ejemplo de elemento

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

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

+------------------------------------------+--------+--------------
+----------+------------------------------------------+--------+---------+
|                   NAME                   |  SIZE  | STORAGE CLASS |
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 con `gcp-cvs` el controlador.

> Para importar un volumen respaldado por NetApp Cloud Volumes Service en Google Cloud Platform, identifique el volumen según la ruta de volumen. La ruta del volumen es la parte de la ruta de exportación del volumen después de `:/` . Por ejemplo, si la ruta de exportación es `10.0.0.1:/adroit-jolly-swift`, la ruta del volumen es `adroit-jolly-swift`.

Ejemplo de Google Cloud Platform

En el ejemplo siguiente se importa `gcp-cvs` un volumen en el back-end `gcpcvs_YEppr` con la ruta de volumen de `adroit-jolly-swift` .

```
tridentctl import volume gcpcvs_YEppr adroit-jolly-swift -f <path-to-pvc-
file> -n trident

+------------------------------------------+--------+--------------
+----------+--------------------------------------+--------+---------+
|                   NAME                   |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+--------+--------------
+----------+--------------------------------------+--------+---------+
| pvc-a46ccab7-44aa-4433-94b1-e47fc8c0fa55 | 93 GiB | gcp-storage   | file
| e1a6e65b-299e-4568-ad05-4f0a105c888f | online | true    |
+------------------------------------------+--------+--------------
+----------+--------------------------------------+--------+---------+
```

Azure NetApp Files

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

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

Ejemplo de Azure NetApp Files

En el ejemplo siguiente se importa `azure-netapp-files` un volumen en el back-end `azurenetappfiles_40517` con la ruta de 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    |
+------------------------------------------+---------+--------------
+----------+--------------------------------------+--------+---------+
```

Personalizar nombres y etiquetas de volúmenes

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

Antes de empezar

Nombres de volumen y etiquetas personalizables admiten:

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

Limitaciones

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

Comportamientos clave de los nombres de volúmenes personalizables

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

Ejemplos de configuración de backend con plantilla de nombre y etiquetas

Las plantillas de nombre personalizado se pueden definir en el nivel raíz y/o de grupo.

Ejemplo de nivel raíz

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

Ejemplo de nivel de pool

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

Ejemplos de plantillas de nombres

Ejemplo 1:

```
"nameTemplate": "{{ .config.StoragePrefix }}_{{ .volume.Name }}_{{
.config.BackendName }}"
```

Ejemplo 2:

```
"nameTemplate": "pool_{{ .config.StoragePrefix }}_{{ .volume.Name }}_{{
slice .volume.RequestName 1 5 }}""
```

Puntos que considerar

1. En el caso de las importaciones de volúmenes, las etiquetas se actualizan solo si el volumen existente tiene etiquetas en un formato específico. Por ejemplo `{"provisioning":{"Cluster":"ClusterA", "PVC": "pvcname"}}`.
2. En el caso de importaciones de volúmenes gestionados, el nombre del volumen sigue a la plantilla de nombres definida en el nivel raíz en la definición de backend.
3. Astra Trident no admite el uso de un operador de segmentos con el prefijo de almacenamiento.
4. Si las plantillas no dan como resultado nombres de volúmenes únicos, Astra Trident añadirá algunos caracteres aleatorios para crear nombres de volumen únicos.
5. Si el nombre personalizado de un volumen de ahorro de NAS supera los 64 caracteres de longitud, Astra Trident asignará el nombre a los volúmenes según la convención de nomenclatura existente. Para el resto de los controladores ONTAP, si el nombre del volumen supera el límite de nombre, se produce un error en el proceso de creación de volúmenes.

Comparta un volumen NFS en espacios de nombres

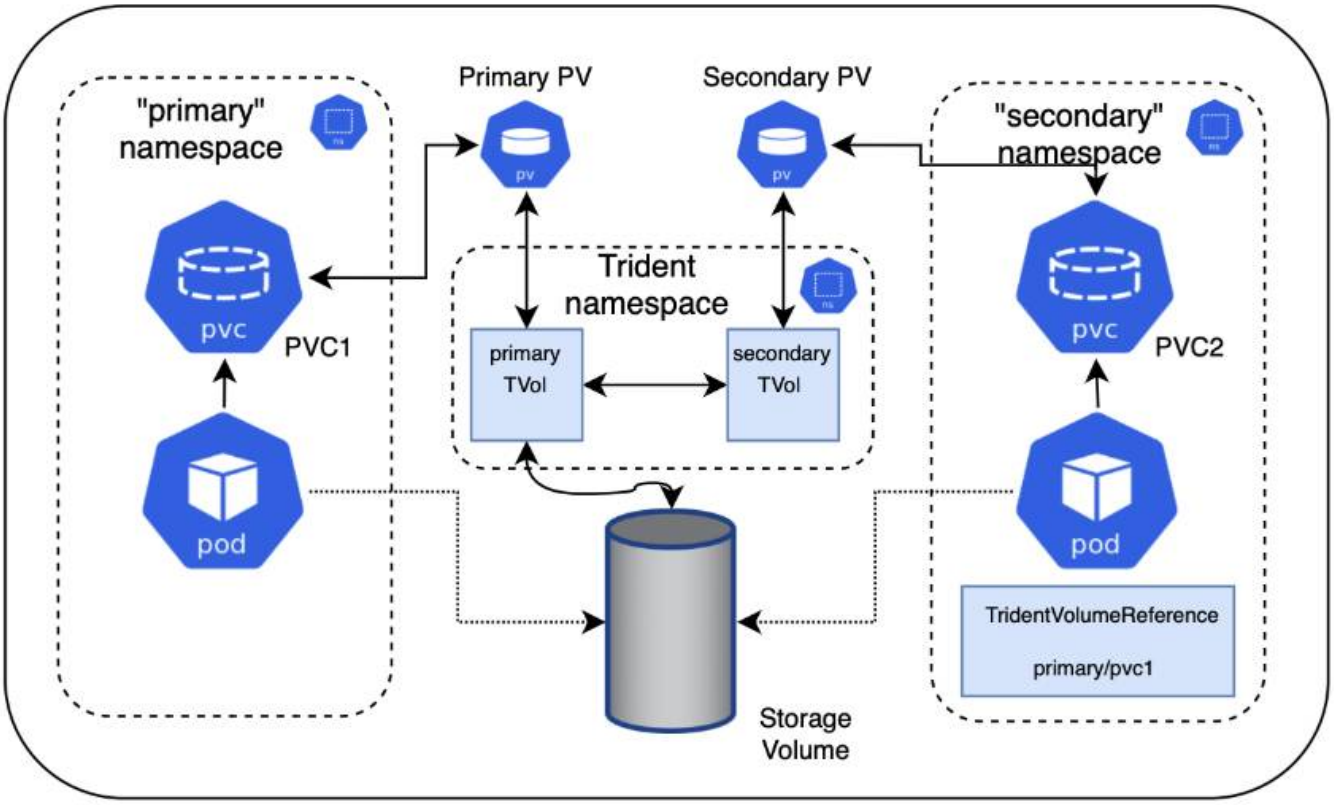
Con Astra Trident, se puede crear un volumen en un espacio de nombres primario y compartirlo en uno o más espacios de nombres secundarios.

Funciones

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

- Varios niveles de control de acceso para garantizar la seguridad
- Funciona con todos los controladores de volúmenes NFS de Trident
- No depende de 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 PVC de origen para compartir el volumen**

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

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

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

3 **Cree TridentVolumeReference en el espacio de nombres de destino**

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

4 **Cree la RVP subordinada en el espacio de nombres de destino**

El propietario del espacio de nombres de destino crea el PVC subordinado para utilizar el origen de datos desde el PVC de origen.

Configurar los espacios de nombres de origen y destino

Para garantizar la seguridad, el uso compartido entre espacios de nombres requiere la colaboración y la acción del propietario del espacio de nombres de origen, el administrador de clúster y el propietario del espacio de nombres de destino. La función de usuario se designa en cada paso.

Pasos

1. **Propietario del espacio de nombres de origen:** Crear el PVC (`pvc1`) en el espacio de nombres de origen que otorga permiso para compartir con el espacio de nombres de destino (`namespace2`) usando la `shareToNamespace` anotación.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1
  namespace: namespace1
  annotations:
    trident.netapp.io/shareToNamespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

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

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

2. **Administrador de clúster:** cree la función personalizada y kubeconfig para conceder permiso al propietario del espacio de nombres de destino para crear el sistema TridentVolumeReference CR en el espacio de nombres de destino.
3. **Propietario del espacio de nombres de destino:** Crear un CR de TridentVolumeReference en el espacio de nombres de destino que se refiere al espacio de nombres de origen `pvc1`.

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. **Propietario del espacio de nombres de destino:** Crear un PVC (`namespace2`)(`pvc2` en el espacio de nombres de destino) Utilizando la `shareFromPVC` anotación para designar el PVC de origen.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  annotations:
    trident.netapp.io/shareFromPVC: namespace1/pvc1
  name: pvc2
  namespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

El tamaño del PVC de destino debe ser menor o igual que el PVC de origen.

Resultados

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

Elimine un volumen compartido

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

Se utiliza `tridentctl get` para consultar los volúmenes subordinados

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

options].

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

Indicadores:

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

Limitaciones

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

Si quiere más información

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

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

Utilice Topología CSI

Astra Trident puede crear y conectar volúmenes a los nodos que están presentes en un clúster de Kubernetes de forma selectiva utilizando el "Función de topología CSI".

Descripción general

Con la función de topología CSI, el acceso a los volúmenes puede limitarse a un subconjunto de nodos, en función de regiones y zonas de disponibilidad. En la actualidad, los proveedores de cloud permiten a los administradores de Kubernetes generar nodos basados en zonas. Los nodos se pueden ubicar en diferentes zonas de disponibilidad dentro de una región o en varias regiones. Para facilitar el aprovisionamiento de volúmenes para cargas de trabajo en una arquitectura de varias zonas, Astra Trident utiliza la topología CSI.

Obtenga más información sobre la función Topología de CSI "aquí" .

Kubernetes ofrece dos modos de enlace de volúmenes únicos:

- Con `VolumeBindingMode` la opción establecida en `Immediate`, Astra Trident crea el volumen sin tener en cuenta la topología. La vinculación de volúmenes y el aprovisionamiento dinámico se manejan cuando

se crea la RVP. Este es el valor por defecto `VolumeBindingMode` y es adecuado para clusters que no aplican restricciones de topología. Los volúmenes persistentes se crean sin depender de los requisitos de programación del pod solicitante.

- Con `VolumeBindingMode` establecido en `WaitForFirstConsumer`, la creación y vinculación de un volumen persistente para una RVP se retrasa hasta que se programe y cree un pod que utilice la RVP. De esta forma, se crean volúmenes con el fin de cumplir las restricciones de programación que se aplican en los requisitos de topología.

> `WaitForFirstConsumer`El modo de enlace no requiere etiquetas de topología. Esto se puede utilizar independientemente de la característica de topología CSI.

Lo que necesitará

Para utilizar la topología CSI, necesita lo siguiente:

- Un clúster de Kubernetes que ejecuta un "Compatible con la versión de Kubernetes"

```
kubectl version
Client Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:50:19Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:41:49Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
```

- Los nodos del clúster deben tener etiquetas que introduzcan el reconocimiento de topología (`topology.kubernetes.io/region`y `topology.kubernetes.io/zone`). Estas etiquetas * deben estar presentes en los nodos del clúster* antes de instalar Astra Trident para que Astra Trident tenga en cuenta la topología.

```
kubectl get nodes -o=jsonpath='{range .items[*]}[{.metadata.name},
{.metadata.labels}]{"\n"}{end}' | grep --color "topology.kubernetes.io"
[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 backend puede llevar un bloque opcional `supportedTopologies` que representa una lista de zonas y regiones soportadas. En el caso de StorageClasses que utilizan dicho back-end, solo se creará un volumen si lo solicita una aplicación programada en una región/zona admitida.

A continuación se muestra un ejemplo de definición de backend:

YAML

```
---
version: 1
storageDriverName: ontap-san
backendName: san-backend-us-east1
managementLIF: 192.168.27.5
svm: iscsi_svm
username: admin
password: password
supportedTopologies:
- topology.kubernetes.io/region: us-east1
  topology.kubernetes.io/zone: us-east1-a
- topology.kubernetes.io/region: us-east1
  topology.kubernetes.io/zone: us-east1-b
```

JSON

```
{
 "version": 1,
 "storageDriverName": "ontap-san",
 "backendName": "san-backend-us-east1",
 "managementLIF": "192.168.27.5",
 "svm": "iscsi_svm",
 "username": "admin",
 "password": "password",
 "supportedTopologies": [
{"topology.kubernetes.io/region": "us-east1",
"topology.kubernetes.io/zone": "us-east1-a"},
{"topology.kubernetes.io/region": "us-east1",
"topology.kubernetes.io/zone": "us-east1-b"}
]
}
```

`supportedTopologies` se utiliza para proporcionar una lista de regiones y zonas por backend. Estas regiones y zonas representan la lista de valores permitidos que se pueden proporcionar en un StorageClass. En el caso de StorageClasses que contienen un subconjunto de las regiones y zonas proporcionadas en un back-end, Astra Trident creará un volumen en el back-end.

También puede definir `supportedTopologies` por pool de almacenamiento. Consulte el siguiente ejemplo:

```
---
version: 1
storageDriverName: ontap-nas
backendName: nas-backend-us-central1
managementLIF: 172.16.238.5
svm: nfs_svm
username: admin
password: password
supportedTopologies:
- topology.kubernetes.io/region: us-central1
  topology.kubernetes.io/zone: us-central1-a
- topology.kubernetes.io/region: us-central1
  topology.kubernetes.io/zone: us-central1-b
storage:
- labels:
    workload: production
  supportedTopologies:
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-a
- labels:
    workload: dev
  supportedTopologies:
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-b
```

En este ejemplo, `region` las etiquetas y `zone` representan la ubicación del pool de almacenamiento. `topology.kubernetes.io/region` `topology.kubernetes.io/zone` y dicte dónde se pueden consumir los pools de almacenamiento.

Paso 2: Defina las clases de almacenamiento que tienen en cuenta la topología

En función de las etiquetas de topología que se proporcionan a los nodos del clúster, se puede definir StorageClasse para que contenga información de topología. Esto determinará los pools de almacenamiento que sirven como candidatos para las solicitudes de RVP y el subconjunto de nodos que pueden usar los volúmenes aprovisionados mediante Trident.

Consulte el siguiente ejemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
name: netapp-san-us-east1
provisioner: csi.trident.netapp.io
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
- matchLabelExpressions:
- key: topology.kubernetes.io/zone
  values:
  - us-east1-a
  - us-east1-b
- key: topology.kubernetes.io/region
  values:
  - us-east1
parameters:
  fsType: "ext4"
```

En la definición de StorageClass proporcionada anteriormente, `volumeBindingMode` se establece en `WaitForFirstConsumer`. Las RVP solicitadas con este tipo de almacenamiento no se verán en cuestión hasta que se mencionan en un pod. Y, `allowedTopologies` proporciona las zonas y la región que se van a utilizar. `netapp-san-us-east1`StorageClass creará EVs en el `san-backend-us-east1` backend definido anteriormente.

Paso 3: Cree y utilice un PVC

Con el clase de almacenamiento creado y asignado a un back-end, ahora puede crear RVP.

Vea el ejemplo `spec` a continuación:

```
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: pvc-san
spec:
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: netapp-san-us-east1
```

La creación de una RVP con este manifiesto daría como resultado lo siguiente:

```
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 en los nodos que están presentes en `us-east1` la región y que elija entre cualquier nodo que esté presente en las `us-east1-a` zonas o. `us-east1-b`

Consulte la siguiente salida:

```
kubectl get pods -o wide
NAME        READY   STATUS    RESTARTS   AGE   IP               NODE
NOMINATED NODE   READINESS GATES
app-pod-1   1/1     Running   0          19s   192.168.25.131   node2
<none>           <none>
kubectl get pvc -o wide
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS          AGE   VOLUMEMODE
pvc-san   Bound    pvc-ecb1e1a0-840c-463b-8b65-b3d033e2e62b   300Mi
RWO            netapp-san-us-east1   48s   Filesystem
```

Actualice los back-ends que se van a incluir `supportedTopologies`

Los back-ends preexistentes se pueden actualizar para incluir una lista de `supportedTopologies` uso `tridentctl backend update`. Esto no afectará a los volúmenes que ya se han aprovisionado, y sólo se utilizarán en las siguientes CVP.

Obtenga más información

- "Gestione recursos para contenedores"
- "Selector de nodos"
- "Afinidad y anti-afinidad"
- "Tolerancias y taints"

Trabajar con instantáneas

Las snapshots de volúmenes de Kubernetes de Persistent Volumes (VP) permiten copias puntuales de volúmenes. Es posible crear una copia Snapshot de un volumen creado con Astra Trident, importar una copia de Snapshot creada fuera de Astra Trident, crear un volumen nuevo a partir de una copia de Snapshot existente y recuperar datos de volumen de copias Snapshot.

Descripción general

La instantánea de volumen es compatible con `ontap-nas`, , , `ontap-nas-flexgroup` `ontap-san` `ontap-san-economy` `solidfire-san` , , , `gcp-cvs`, y `azure-netapp-files` los conductores.

Antes de empezar

Debe tener un controlador de instantánea externo y definiciones de recursos personalizados (CRD) para trabajar con instantáneas. Esta es la responsabilidad del orquestador de Kubernetes (por ejemplo: Kubeadm, GKE, OpenShift).

Si su distribución de Kubernetes no incluye el controlador de instantáneas y los CRD, consulte Implemente una controladora Snapshot de volumen.

> No cree una controladora Snapshot si crea instantáneas de volumen bajo demanda en un entorno de GKE. GKE utiliza un controlador de instantáneas oculto integrado.

Cree una copia de Snapshot de volumen

Pasos

1. Cree un `VolumeSnapshotClass`. Para obtener más información, consulte "VolumeSnapshotClass".
   - `driver`Los puntos al controlador CSI de Astra Trident.
   - `deletionPolicy` puede ser `Delete` o. `Retain` Cuando se establece en `Retain`, la instantánea física subyacente del clúster de almacenamiento se conserva incluso si se elimina el `VolumeSnapshot` objeto.

     Ejemplo

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

2. Crear una instantánea de una RVP existente.

   Ejemplos

   - En este ejemplo, se crea una copia Snapshot de una RVP existente.

     ```
     cat snap.yaml
     apiVersion: snapshot.storage.k8s.io/v1
     kind: VolumeSnapshot
     metadata:
       name: pvc1-snap
     spec:
       volumeSnapshotClassName: csi-snapclass
       source:
         persistentVolumeClaimName: pvc1
     ```

   - En este ejemplo, se crea un objeto Snapshot de volumen para una RVP denominada `pvc1` y el nombre de la Snapshot se establece en `pvc1-snap`. Una Snapshot de volumen es similar a una RVP y se asocia con `VolumeSnapshotContent` un objeto que representa la snapshot real.

     ```
     kubectl create -f snap.yaml
     volumesnapshot.snapshot.storage.k8s.io/pvc1-snap created

     kubectl get volumesnapshots
     NAME                    AGE
     pvc1-snap               50s
     ```

- Es posible identificar `VolumeSnapshotContent` el objeto de `pvc1-snap` la Snapshot de volumen describiéndolo. El `Snapshot Content Name` identifica el objeto VolumeSnapshotContent que sirve para esta snapshot. `Ready To Use`El parámetro indica que la snapshot se puede usar para crear una nueva RVP.

```
kubectl describe volumesnapshots pvc1-snap
Name:         pvc1-snap
Namespace:    default
.
.
.
Spec:
  Snapshot Class Name:    pvc1-snap
  Snapshot Content Name:  snapcontent-e8d8a0ca-9826-11e9-9807-
525400f3f660
  Source:
    API Group:
    Kind:       PersistentVolumeClaim
    Name:       pvc1
Status:
  Creation Time:  2019-06-26T15:27:29Z
  Ready To Use:   true
  Restore Size:   3Gi
.
.
```

Cree una RVP a partir de una snapshot de volumen

Puede usar `dataSource` para crear una RVP con una VolumeSnapshot llamada `<pvc-name>` como origen de los datos. Una vez creada la RVP, se puede conectar a un pod y utilizarla como cualquier otro PVC.

> La RVP se creará en el mismo back-end que el volumen de origen. Consulte "KB: La creación de una RVP a partir de una snapshot de RVP de Trident no se puede crear en un back-end alternativo".

En el siguiente ejemplo se crea la RVP utilizando `pvc1-snap` como origen de datos.

```
cat pvc-from-snap.yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: golden
  resources:
    requests:
      storage: 3Gi
  dataSource:
    name: pvc1-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

Importe una copia de Snapshot de volumen

Astra Trident admite "Proceso de snapshot aprovisionado previamente de Kubernetes"para permitir al administrador de clúster crear `VolumeSnapshotContent` un objeto e importar copias de Snapshot creadas fuera de Astra Trident.

Antes de empezar

Astra Trident debe haber creado o importado el volumen principal del snapshot.

Pasos

1. **Cluster admin:** Crear un `VolumeSnapshotContent` objeto que haga referencia a la instantánea backend. Esto inicia el flujo de trabajo de las copias Snapshot en Astra Trident.
   - Especifique el nombre de la instantánea de backend en `annotations` como `trident.netapp.io/internalSnapshotName: <"backend-snapshot-name">`.
   - Especificar `<name-of-parent-volume-in-trident>/<volume-snapshot-content-name>` en `snapshotHandle`. Esta es la única información proporcionada a Astra Trident por el Snapshotter externo en la `ListSnapshots` llamada.

     `<volumeSnapshotContentName>`No siempre puede coincidir con el nombre de instantánea de backend debido a restricciones de nomenclatura de CR.

     Ejemplo

     En el siguiente ejemplo se crea un `VolumeSnapshotContent` objeto que hace referencia a la instantánea backend `snap-01` .

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotContent
metadata:
  name: import-snap-content
  annotations:
    trident.netapp.io/internalSnapshotName: "snap-01"  # This is the
name of the snapshot on the backend
spec:
  deletionPolicy: Retain
  driver: csi.trident.netapp.io
  source:
    snapshotHandle: pvc-f71223b5-23b9-4235-bbfe-e269ac7b84b0/import-
snap-content # <import PV name or source PV name>/<volume-snapshot-
content-name>
```

2. **Cluster admin:** Crear el `VolumeSnapshot` CR que hace referencia al `VolumeSnapshotContent` objeto. Esto solicita acceso para utilizar `VolumeSnapshot` en un espacio de nombres determinado.

   Ejemplo

   En el siguiente ejemplo se crea una `VolumeSnapshot` CR denominada `import-snap` que hace referencia a la `VolumeSnapshotContent` `import-snap-content` .

   ```
   apiVersion: snapshot.storage.k8s.io/v1
   kind: VolumeSnapshot
   metadata:
     name: import-snap
   spec:
     # volumeSnapshotClassName: csi-snapclass (not required for pre-
   provisioned or imported snapshots)
     source:
       volumeSnapshotContentName: import-snap-content
   ```

3. **Procesamiento interno (no se requiere acción):** El Snapshotter externo reconoce el recién creado `VolumeSnapshotContent` y ejecuta la `ListSnapshots` llamada. Astra Trident crea el `TridentSnapshot`.
   - El dispositivo de instantáneas externo establece el `VolumeSnapshotContent` en `readyToUse` y el `VolumeSnapshot` en `true`.
   - Trident devuelve `readyToUse=true`.
4. **Cualquier usuario:** Crear un `PersistentVolumeClaim` para hacer referencia al nuevo `VolumeSnapshot`, donde el `spec.dataSource` nombre (o `spec.dataSourceRef`) es el nombre `VolumeSnapshot`.

   Ejemplo

   En el siguiente ejemplo se crea una RVP que hace referencia a la `VolumeSnapshot` llamada `import-`

snap.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: simple-sc
  resources:
    requests:
      storage: 1Gi
  dataSource:
    name: import-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

Recuperar datos de volumen mediante copias Snapshot

El directorio de snapshots está oculto de forma predeterminada para facilitar la máxima compatibilidad de los volúmenes aprovisionados mediante los `ontap-nas` controladores y. `ontap-nas-economy` Permita que `.snapshot` el directorio recupere datos de snapshots directamente.

Use la interfaz de línea de comandos de ONTAP para restaurar un volumen en un estado registrado en una snapshot anterior.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

> Cuando se restaura una copia Snapshot, se sobrescribe la configuración de volúmenes existente. Se pierden los cambios que se hagan en los datos del volumen después de crear la copia Snapshot.

Eliminar un VP con snapshots asociadas

Cuando se elimina un volumen persistente con instantáneas asociadas, el volumen Trident correspondiente se actualiza a un “estado de eliminación”. Quite las snapshots de volumen para eliminar el volumen de Astra Trident.

Implemente una controladora Snapshot de volumen

Si su distribución de Kubernetes no incluye el controlador de snapshots y los CRD, puede implementarlos de la siguiente manera.

Pasos

1. Crear CRD de snapshot de volumen.

```
cat snapshot-setup.sh
#!/bin/bash
# Create volume snapshot CRDs
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshotclasses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshotcontents.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 actualice `namespace` su espacio de nombres.

Enlaces relacionados

- "Copias de Snapshot de volumen"
- "VolumeSnapshotClass"

Gestione y supervise Astra Trident

Actualice Astra Trident

Actualice Astra Trident

A partir del lanzamiento de la versión 24,02, Astra Trident sigue una cadencia de lanzamiento de cuatro meses y entrega tres versiones principales cada año. Cada nueva versión se basa en versiones anteriores y proporciona nuevas funciones, mejoras de rendimiento, 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.
- Astra Trident 23,07 y versiones posteriores requieren v1 copias Snapshot de volumen y ya no admite copias Snapshot alfa o beta.
- Si has creado Cloud Volumes Service para Google Cloud en "Tipo de servicio CVS", debes actualizar la configuración de back-end para utilizar `standardsw` el nivel de servicio o `zoneredundantstandardsw` al actualizar desde Astra Trident 23,01. Si no se actualiza el `serviceLevel` en el backend, se podrían producir errores en los volúmenes. Consulte "Muestras de tipo de servicio CVS" para obtener más información.
- Cuando se realiza la actualización, es importante que `parameter.fsType` `StorageClasses` utilices Astra Trident. Puede eliminar y volver a crear `StorageClasses` sin interrumpir los volúmenes existentes anteriores.
  - Este es un **requisito** de la aplicación de "contextos de seguridad" volúmenes SAN.
  - El directorio sample input contiene ejemplos, como `storage-class-basic.yaml.templ` y link:https://github.com/NetApp/Trident/blob/master/Trident-installer/sample-input/storage-class-bronze-class[`storage-class-bronze-default.yaml` ^.
  - Para obtener más información, consulte "Problemas conocidos".

Paso 1: Seleccione una versión

Las versiones de Astra Trident siguen una convención de nomenclatura basada en fechas `YY.MM`, donde «YY» son los dos últimos dígitos del año y «MM» es el mes. Las versiones de DOT siguen `YY.MM.X` una 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 directamente de 23,04 (o cualquier versión de 23,04 puntos) a 24,06.
- Si va a actualizar desde una versión fuera de la ventana de cuatro versiones, realice una actualización de varios pasos. Utilice las instrucciones de actualización de la "versión anterior" que va a actualizar para actualizar a la versión más reciente que se ajuste a la ventana de cuatro versiones. Por ejemplo, si utiliza

22,01 y desea actualizar a la versión 24,06:

a. Primera actualización de 22,07 a 23,04.

b. A continuación, actualice de 23,04 a 24,06.

> 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. Para obtener más información, consulte la "Detalles del problema en GitHub".

Paso 2: Determine el método de instalación original

Para determinar qué versión solías instalar originalmente Astra Trident:

1. Se utiliza `kubectl get pods -n trident` para examinar los pods.
   - Si no hay ningún pod de operador, se instaló Astra Trident 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 de operador, utilícelo `kubectl describe torc` 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

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

- "Actualice con el operador Trident"

> Le sugerimos que lo revise "Comprender el flujo de trabajo de actualización del operador" antes de actualizar con el operador.

*

Actualizar con el operador

Comprender el flujo de trabajo de actualización del operador

Antes de usar el operador Trident para actualizar Astra Trident, debe comprender los procesos en segundo plano que se producen durante la actualización. Esto incluye cambios en la controladora Trident, en el pod de controladora y en los pods de nodos, así como en el DaemonSet de nodos que permiten actualizaciones graduales.

Manejo de actualizaciones del operador Trident

Uno de los muchos "Ventajas del uso del operador Trident" que instalan y actualizan Astra Trident es la gestión automática de los objetos de Astra Trident y Kubernetes sin interrumpir los volúmenes montados existentes. De esta forma, Astra Trident puede admitir renovaciones sin tiempos de inactividad o

"*actualizaciones sucesivas*". En concreto, el operador Trident se comunica con el clúster de Kubernetes para:

- Elimine y vuelva a crear la implementación de Trident Controller y DaemonSet de nodos.
- Sustituya el pod de la controladora de Trident y los pods de nodos de Trident por nuevas versiones.
  - Si no se actualiza un nodo, no impide que se actualicen los nodos restantes.
  - Solo los nodos con un nodo de Trident en ejecución pueden montar volúmenes.

Para obtener más información sobre la arquitectura de Astra Trident en el clúster de Kubernetes, consulte "Arquitectura de Astra Trident".

Flujo de trabajo de actualización del operador

Cuando inicie una actualización con el operador Trident:

1. El operador **Trident**:
   a. Detecta la versión instalada actualmente de Astra Trident (versión *n*).
   b. Actualiza todos los objetos de Kubernetes, incluidos CRD, RBAC y Trident SVC.
   c. Elimina la implementación de Trident Controller para la versión *n*.
   d. Crea la implementación de Trident Controller para la versión *n+1*.
2. **Kubernetes** crea Trident Controller Pod para *n+1*.
3. El operador **Trident**:
   a. Elimina el conjunto de cambios de nodo Trident para *n*. El operador no espera la terminación del Node Pod.
   b. Crea el inicio del demonio del nodo Trident para *n+1*.
4. **Kubernetes** crea pods de nodos Trident en nodos que no ejecutan Trident Node Pod *n*. De este modo se garantiza que nunca haya más de un pod de nodo de Trident, de ninguna versión, en un nodo.

Actualiza una instalación de Astra Trident mediante el operador Trident o Helm

Puede actualizar Astra Trident de forma manual o mediante Helm mediante el operador Trident. Puede actualizar de una instalación del operador de Trident a otra instalación del operador de Trident o actualizar de una `tridentctl` instalación a una versión del operador de Trident. Revise "Seleccione un método de actualización" antes de actualizar una instalación del operador Trident.

Actualizar una instalación manual

Puede actualizar desde una instalación de operadores Trident en el ámbito del clúster a otra 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.

Para actualizar desde Astra Trident que se ha instalado con el operador de espacio de nombres (de la 20,07 a la 20,10), utiliza las instrucciones de actualización de "la versión instalada"Astra Trident.

Acerca de esta tarea

Trident proporciona un archivo de paquete que se puede utilizar para instalar el operador y crear objetos

asociados para la versión de Kubernetes.

- Para los clústeres que ejecutan Kubernetes 1,24, 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

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 23,07, ejecute el siguiente comando:

```
kubectl delete -f 23.07.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

3. Si personalizó la instalación inicial mediante `TridentOrchestrator` atributos, puede editar el `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 del paquete correcto para su entorno, donde *<bundle.yaml>* está `bundle_pre_1_25.yaml` o `bundle_post_1_25.yaml` se basa en su versión de Kubernetes. Por ejemplo, si está instalando Astra Trident 24,06, ejecute el siguiente comando:

```
kubectl create -f 24.06.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

Actualizar una instalación Helm

Puede actualizar una instalación de Astra Trident Helm.

> Cuando actualice un clúster de Kubernetes de 1,24 a la versión 1,25 o una versión posterior que tiene Astra Trident instalado, debe actualizar los valores.yaml para establecer `excludePodSecurityPolicy` `true` o añadir `--set excludePodSecurityPolicy=true` al `helm upgrade` comando antes de actualizar el clúster.

Pasos

1. Si "Instalar Astra Trident mediante Helm"utiliza `helm upgrade trident netapp-trident/trident-operator --version 100.2406.0` para actualizar en un solo paso. Si no ha añadido el repositorio Helm o no puede utilizarlo para actualizar:
   a. Descargue la última versión de Astra Trident de "La sección *Assets* de GitHub".

b. Utilice `helm upgrade` el comando donde `trident-operator-24.06.0.tgz` refleja la versión a la que desea actualizar.

```
helm upgrade <name> trident-operator-24.06.0.tgz
```

> Si establece opciones personalizadas durante la instalación inicial (como la especificación de registros privados y reflejados para imágenes Trident y CSI), agregue el `helm upgrade` comando Using `--set` para asegurarse de que esas opciones se incluyen en el comando UPGRADE; de lo contrario, los valores se restablecerán a los valores predeterminados.

2. Ejecute `helm list` para verificar que la tabla y la versión de la aplicación se han actualizado. Ejecute `tridentctl logs` para revisar los mensajes de depuración.

Actualización de una `tridentctl` instalación al operador Trident

Puede actualizar a la última versión del operador Trident desde una `tridentctl` instalación. Los back-ends y EVs existentes estarán disponibles automáticamente.

> Antes de cambiar entre los métodos de instalación, revise "Moverse entre los métodos de instalación".

Pasos

1. Descargue la última versión de Astra Trident.

```
# Download the release required [24.060.0]
mkdir 24.06.0
cd 24.06.0
wget
https://github.com/NetApp/trident/releases/download/v24.06.0/trident-
installer-24.06.0.tar.gz
tar -xf trident-installer-24.06.0.tar.gz
cd trident-installer
```

2. Cree el `tridentorchestrator` CRD a partir 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-name.yaml>

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

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

4. Crea un `TridentOrchestrator` CR 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:                v24.06.0
```

Actualice con trimentctl

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

Acerca de esta tarea

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

Revise "Seleccione un método de actualización" antes de actualizar con `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.

Gestión de Astra Trident con tridentctl

`https://github.com/NetApp/trident/releases["Paquete de instalación de Trident"^]Incluye la `tridentctl` utilidad de línea de comandos para proporcionar un acceso simple a Astra Trident. Los usuarios de Kubernetes que cuentan con suficientes privilegios pueden utilizarlo para instalar Astra Trident o gestionar el espacio de nombres que contiene el pod de Astra Trident.`

Comandos e indicadores globales

Puede ejecutar `tridentctl help` para obtener una lista de comandos disponibles `tridentctl` o agregar el `--help` indicador a cualquier comando para obtener una lista de opciones y indicadores para ese comando específico.

```
tridentctl [command] [--optional-flag]
```

La utilidad Astra Trident `tridentctl` admite los siguientes comandos y avisos globales.

Comandos

`create`
Añade un recurso a Astra Trident.

`delete`
Quita uno o varios recursos de Astra Trident.

`get`
Obtén uno o más recursos de Astra Trident.

`help`
Ayuda sobre cualquier comando.

`images`
Imprime una tabla de las imágenes de contenedores que Astra Trident necesita.

`import`
Importar un recurso existente a Astra Trident.

`install`
Instala Astra Trident.

`logs`
Imprime los registros desde Astra Trident.

`send`
Enviar un recurso desde Astra Trident.

`uninstall`
Desinstale Astra Trident.

`update`
Modificar un recurso en Astra Trident.

`update backend state`
Suspender temporalmente las operaciones de backend.

`upgrade`
Actualiza un recurso en Astra Trident.

`version`
Imprime la versión de Astra Trident.

Indicadores globales

-d, --debug
Salida de depuración.

-h, --help
Ayuda para `tridentctl`.

-k, --kubeconfig string
Especifique `KUBECONFIG` la ruta para ejecutar comandos localmente o desde un clúster de Kubernetes a otro.

También puede exportar la `KUBECONFIG` variable para que apunte a un clúster de Kubernetes específico y emitir `tridentctl` comandos a ese clúster.

-n, --namespace string
Espacio de nombres de puesta en marcha 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).

Opciones de comando y indicadores

cree

Utilice `create` el comando para añadir un recurso a Astra Trident.

```
tridentctl create [option]
```

Opciones

`backend`: Añadir un backend a Astra Trident.

eliminar

Usa `delete` el comando para quitar uno o varios recursos de Astra Trident.

```
tridentctl delete [option]
```

Opciones

`backend`: Eliminar uno o varios back-ends de almacenamiento de Astra Trident.
`snapshot`: Elimina una o varias instantáneas de volumen de Astra Trident.
`storageclass`: Elimina una o varias clases de almacenamiento de Astra Trident.

`volume`: Elimina uno o varios volúmenes de almacenamiento de Astra Trident.

obtenga

Usa `get` el comando para obtener uno o varios recursos de Astra Trident.

```
tridentctl get [option]
```

Opciones

`backend`: Obtenga uno o varios back-ends de almacenamiento de Astra Trident.
`snapshot`: Obtenga una o más instantáneas de Astra Trident.
`storageclass`: Obtenga una o varias clases de almacenamiento de Astra Trident.
`volume`: Obtenga uno o varios volúmenes de Astra Trident.

Indicadores

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

imágenes

Utiliza `images` banderas para imprimir una tabla de las imágenes de contenedores que Astra Trident necesita.

```
tridentctl images [flags]
```

Indicadores

`-h`, `--help`: Ayuda para imágenes.
`-v`, `--k8s-version string`: Versión semántica del clúster de Kubernetes.

importe volumen

Use `import volume` el comando para importar un volumen existente a Astra Trident.

```
tridentctl import volume <backendName> <volumeName> [flags]
```

Alias

`volume`, `v`

Indicadores

`-f`, `--filename string`: Ruta a YAML o archivo JSON PVC.
`-h`, `--help`: Ayuda para el volumen.
`--no-manage`: Crear PV/PVC solamente. No asuma que se gestiona el ciclo de vida de los volúmenes.

instale

Utiliza `install` los indicadores para instalar Astra Trident.

```
tridentctl install [flags]
```

Indicadores

`--autosupport-image string`: La imagen del contenedor para la telemetría AutoSupport (por defecto "NetApp/Trident AutoSupport:<current-version>").
`--autosupport-proxy string`: La dirección/puerto de un proxy para enviar telemetría AutoSupport.
`--enable-node-prep`: Intenta instalar los paquetes necesarios en los nodos.
`--generate-custom-yaml`: Generar archivos YAML sin instalar nada.
`-h, --help`: Ayuda para la instalación.
`--http-request-timeout`: Sustituya el tiempo de espera de la solicitud HTTP para la API REST del controlador Trident (por defecto 1m30s).
`--image-registry string`: La dirección/puerto de un registro interno de imágenes.
`--k8s-timeout duration`: El tiempo de espera para todas las operaciones de Kubernetes (por defecto 3m0s).
`--kubelet-dir string`: La ubicación del anfitrión del estado interno de kubelet (por defecto "/var/lib/kubelet").
`--log-format string`: El formato de registro de Astra Trident (texto, json) (texto predeterminado).
`--pv string`: El nombre del PV heredado utilizado por Astra Trident, se asegura de que no exista (por defecto, «Trident»).
`--pvc string`: El nombre del PVC heredado utilizado por Astra Trident, se asegura de que esto no exista (por defecto "Trident").
`--silence-autosupport`: No enviar paquetes AutoSupport a NetApp automáticamente (predeterminado true).
`--silent`: Deshabilitar la mayoría de la salida durante la instalación.
`--trident-image string`: La imagen de Astra Trident para instalar.
`--use-custom-yaml`: Utilice cualquier archivo YAML existente que exista en el directorio de configuración.
`--use-ipv6`: Utilice IPv6 para la comunicación de Astra Trident.

registros

Utiliza `logs` marcas para imprimir los registros de Astra Trident.

```
tridentctl logs [flags]
```

Indicadores

`-a, --archive`: Crear un archivo de soporte con todos los registros a menos que se especifique lo contrario.
`-h, --help`: Ayuda para registros.
`-l, --log string`: Registro de Astra Trident para mostrar. Uno de Trident|auto|Trident-operator|all (automático por defecto).
`--node string`: El nombre del nodo de Kubernetes desde el que se recopilan los registros de pod de nodo.
`-p, --previous`: Obtenga los logs de la instancia de contenedor anterior si existe.
`--sidecars`: Obtenga los registros para los contenedores sidecar.

enviar

Utilice `send` el comando para enviar un recurso desde Astra Trident.

```
tridentctl send [option]
```

Opciones

`autosupport`: Enviar un archivo AutoSupport a NetApp.

desinstalar

Utiliza `uninstall` flags para desinstalar Astra Trident.

```
tridentctl uninstall [flags]
```

Indicadores

`-h, --help`: Ayuda para desinstalar.
`--silent`: Deshabilitar la mayoría de la salida durante la desinstalación.

actualizar

Utilice `update` el comando para modificar un recurso en Astra Trident.

```
tridentctl update [option]
```

Opciones

`backend`: Actualizar un backend en Astra Trident.

actualizar estado de backend

Utilice `update backend state` el comando para suspender o reanudar operaciones de back-end.

```
tridentctl update backend state <backend-name> [flag]
```

Puntos que considerar

- Si se crea un backend con un TridentBackendConfig (tbc), el backend no se puede actualizar con un `backend.json` archivo.
- Si el `userState` se ha establecido en una tbc, no se puede modificar mediante el `tridentctl update backend state <backend-name> --user-state suspended/normal` comando.
- Para recuperar la capacidad de establecer el `userState` tridentctl vía una vez que se ha establecido a través de tbc, el `userState` campo debe eliminarse del tbc. Esto se puede hacer usando `kubectl edit tbc` el comando. Una vez eliminado el `userState` campo, puede utilizar `tridentctl update backend state` el comando para cambiar el `userState` de un backend.
- Utilice el `tridentctl update backend state` para cambiar la `userState`. También puede actualizar el `userState` archivo Using `TridentBackendConfig` OR `backend.json` ; esto desencadena una reinicialización completa del backend y puede llevar mucho tiempo.

  #### Indicadores

  `-h, --help`: Ayuda para el estado de backend.
  `--user-state`: Establecer `suspended` para pausar las operaciones de backend. Establezca esta opción `normal` para reanudar las operaciones de backend. Cuando se establece en `suspended`:

- `AddVolume` `Import Volume` y se ponen en pausa.
- `CloneVolume`, , , `ResizeVolume` `PublishVolume` `UnPublishVolume`, , `CreateSnapshot`, `GetSnapshot` `RestoreSnapshot`, , , `DeleteSnapshot` `RemoveVolume`, , `GetVolumeExternal`,

`ReconcileNodeAccess` seguir estando disponible.

También puede actualizar el estado del backend utilizando `userState` el campo en el archivo de configuración de backend `TridentBackendConfig` o `backend.json`. Para obtener más información, consulte "Opciones para gestionar back-ends" y. "Realice la gestión del entorno de administración con kubectl"

Ejemplo:

JSON

Siga estos pasos para actualizar el `userState` utilizando el `backend.json` archivo:

1. Edite el `backend.json` archivo para incluir el `userState` campo con su valor establecido en 'SUSPENded'.
2. Actualice el backend con el `tridentctl backend update` comando y la ruta de acceso al archivo actualizado `backend.json` .

   Ejemplo: `tridentctl backend update -f /<path to backend JSON file>/backend.json`

```
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "managementLIF": "<redacted>",
    "svm": "nas-svm",
    "backendName": "customBackend",
    "username": "<redacted>",
    "password": "<redacted>",
    "userState": "suspended",
}
```

YAML

Puede editar el tbc después de que se haya aplicado con el `kubectl edit <tbc-name> -n <namespace>` comando. En el ejemplo siguiente se actualiza el estado del back-end para suspender con la `userState: suspended` opción:

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-ontap-nas
spec:
  version: 1
  backendName: customBackend
  storageDriverName: ontap-nas
  managementLIF: <redacted>
  svm: nas-svm
userState: suspended
  credentials:
    name: backend-tbc-ontap-nas-secret
```

versión

Utilice `version` indicadores para imprimir la versión de `tridentctl` y el servicio Trident en ejecución.

```
tridentctl version [flags]
```

Indicadores

`--client`: Solo versión de cliente (no se requiere servidor).
`-h, --help`: Ayuda para la versión.

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 muestran en el puerto de destino `8001` en el `/metrics` punto final. 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" o usted puede elegir ejecutar Prometeo como un "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. Esto "blog" explica cómo puedes usar Prometheus y Grafana con Astra Trident para recuperar métricas. El blog explica cómo puede ejecutar Prometheus como operador en su clúster de Kubernetes y la creación de un ServiceMonitor para obtener métricas de Astra Trident.

Paso 2: Cree un Prometheus ServiceMonitor

Para consumir las métricas de Trident, debe crear un ServiceMonitor de Prometheus que vigile `trident-csi` el servicio y escuche en 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 el `trident-csi` servicio y busca específicamente el `metrics` punto final del servicio. Como resultado, Prometheus ahora está configurado para comprender las métricas de Astra Trident.

Además de las métricas disponibles directamente desde Astra Trident, kubelet expone muchas `kubelet_volume_*` métricas a través de su propio punto final 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 evitar que Astra Trident envíe métricas de Prometheus e información de back-end básica a NetApp, pase el `--silence-autosupport` indicador durante la instalación de Astra Trident.
- Astra Trident también puede enviar registros de contenedor a Soporte de NetApp bajo demanda a través `tridentctl send autosupport`de . Deberá activar Astra Trident para cargar los registros. Antes de enviar registros, debe aceptar NetApphttps://www.netapp.com/company/legal/privacy-policy/["política de privacidad"^].
- A menos que se especifique lo contrario, Astra Trident recupera los registros de las últimas 24 horas.
- Puede especificar el plazo de retención del registro con `--since` el indicador. Por ejemplo `tridentctl send autosupport --since=1h`: . Esta información se recopila y se envía a través de un `trident-autosupport` contenedor instalado 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. Incluye una "CLUF" que no es aplicable a la propia imagen del contenedor de 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 de contenedor, puede utilizar el `--image-registry` indicador.
- También puede configurar direcciones URL proxy generando los archivos YLMA de instalación. Esto se puede hacer `tridentctl install --generate-custom-yaml` usando para crear los archivos YAML y agregando el `--proxy-url` argumento para el `trident-autosupport` contenedor en `trident-deployment.yaml`.

Deshabilite las métricas de Astra Trident

Para **desactivar las métricas** de ser reportadas, debe generar YAML personalizados (usando el `--generate-custom-yaml` indicador) y editarlos para eliminar el `--metrics` indicador de ser invocado para el `trident-main` contenedor.

Desinstale Astra Trident

Debe utilizar el mismo método para desinstalar Astra Trident que utilizó para instalar Astra Trident.

Acerca de esta tarea

- Si necesita una corrección de los errores observados después de una actualización, problemas de dependencia o una actualización fallida o incompleta, debe desinstalar Astra Trident y volver a instalar la versión anterior siguiendo las instrucciones específicas para ese "versión". Esta es la única forma recomendada de *downgrade* a una versión anterior.
- Para facilitar la actualización y la reinstalación, la desinstalación de Astra Trident no quita los CRD ni los objetos relacionados que ha creado Astra Trident. Si necesita eliminar por completo Astra Trident y todos sus datos, consulte "Elimina por completo Astra Trident y CRD"el .

Antes de empezar

Si va a decomisionar clústeres de Kubernetes, debe eliminar todas las aplicaciones que usan volúmenes creados por Astra Trident antes de desinstalar. De este modo se garantiza la eliminación de las RVP en los nodos de Kubernetes antes de que se eliminen.

Determine el método de instalación original

Debe usar el mismo método para desinstalar Astra Trident que utilizó para instalarlo. Antes de la desinstalación, compruebe la versión que usaba para instalar Astra Trident originalmente.

1. Se utiliza `kubectl get pods -n trident` para examinar los pods.
   - Si no hay ningún pod de operador, se instaló Astra Trident 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 de operador, utilícelo `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.

Desinstale una instalación del operador Trident

Puede desinstalar una instalación de operador trident manualmente o usando Helm.

Desinstale la instalación manual

Si instaló Astra Trident mediante el operador, puede desinstalarlo siguiendo una de las siguientes acciones:

1. **Editar `TridentOrchestrator` CR y establecer el indicador de desinstalación**:

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

   Cuando el `uninstall` indicador está definido en `true`, el operador Trident desinstala Trident, pero no elimina el propio TridentOrchestrator. Debe limpiar el TridentOrchestrator y crear uno nuevo si desea volver a instalar Trident.

2. **Eliminar `TridentOrchestrator`**: Al eliminar el `TridentOrchestrator` CR que se utilizó para implementar Astra Trident, le indicas al operador que desinstale Trident. El operador procesa la eliminación `TridentOrchestrator` y procede a eliminar la implementación de Astra Trident y el inicio de datos, eliminando los pods de Trident que había creado como parte de la instalación.

   ```
   kubectl delete -f deploy/<bundle.yaml> -n <namespace>
   ```

Desinstale la instalación de Helm

Si ha instalado Astra Trident mediante Helm, puede desinstalarlo utilizando `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 una `tridentctl` instalación

Use `uninstall` el comando en `tridentctl` para quitar todos los recursos asociados con Astra Trident, excepto los CRD y los objetos relacionados:

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

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 el despliegue cumple con todos los "requisitos".
- Compruebe que tiene instalada una versión compatible de Docker. Si la versión de Docker no está actualizada, "instálelo o actualícelo".

```
docker --version
```

- Comprobar que los requisitos previos del protocolo están instalados y configurados en el host.

Herramientas de NFS

Instale las herramientas de NFS mediante los comandos del sistema operativo.

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.

Herramientas iSCSI

Instale las herramientas iSCSI mediante los comandos del sistema operativo.

RHEL 8 O POSTERIOR

1. Instale los siguientes paquetes del sistema:

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

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

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

3. Configure el escaneo en manual:

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

4. Activar accesos múltiples:

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

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

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

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

6. Activar e iniciar `iscsi`:

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

Ubuntu

1. Instale los siguientes paquetes del sistema:

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

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

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

3. Configure el escaneo en manual:

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

4. Activar accesos múltiples:

```
sudo tee /etc/multipath.conf <<-'EOF'
defaults {
    user_friendly_names yes
    find_multipaths no
}
EOF
sudo systemctl enable --now multipath-tools.service
sudo service multipath-tools restart
```

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

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

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

Herramientas de NVMe

Instale las herramientas NVMe mediante los comandos de su sistema operativo.

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

RHEL 9

```
sudo yum install nvme-cli
sudo yum install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

Ubuntu

```
sudo apt install nvme-cli
sudo apt -y install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

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.

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 la "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 está desactualizada, "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 especificando la `config` opción con el nombre de archivo. El archivo de configuración debe estar ubicado en `/etc/netappdvp` el directorio del 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. Reemplace `<version>` por la versión del plugin (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/v24.10.0/trident-
installer-24.06.0.tar.gz
tar zxf trident-installer-24.06.0.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 binario y crear el archivo de configuración, inicie el daemon de Trident con el archivo de configuración que desee.

```
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 unidad de ejemplo para sistemas basados en systemd en `contrib/trident.service.example` el repositorio de Git. 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 según el nombre del 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
```

> Cada vez que modifique el archivo de unidad, ejecute `systemctl daemon-reload` el 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 período en el que `docker volume` los comandos dirigidos al plugin no tendrán éxito, y las aplicaciones no podrán montar volúmenes hasta que el plugin vuelva a ejecutarse. 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 de la `netapp/ndvp-plugin` imagen a la `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 deshabilitar el plugin, luego quitar el plugin y, a continuación, instalar la versión de Astra Trident requerida 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 quitar volúmenes fácilmente mediante los comandos estándar

`docker volume` con el nombre de controlador de Astra Trident especificado cuando sea 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 utilizarán los valores por defecto 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 como potencias 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
```

> Al utilizar `solidfire-san` el controlador, el ejemplo anterior elimina y purga el volumen.

Realice los siguientes pasos para actualizar Astra Trident for Docker.

Clonar un volumen

Si utiliza `ontap-nas` , , , `ontap-san`, `solidfire-san` y `gcp-cvs storage drivers`, Astra Trident puede clonar volúmenes. Al utilizar `ontap-nas-flexgroup` los controladores o `ontap-nas-economy`, 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

Puedes acceder a dispositivos de bloques creados externamente (o a sus clones) mediante contenedores usando Trident **only** si no tienen particiones y si su sistema de archivos es compatible con Astra Trident (por ejemplo: Un `ext4`-formateado `/dev/sdc1` no será accesible 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 mediante `-o` el 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 de 1 GIB.
`spaceReserve`	Aprovisione el volumen de manera thin o thick, de manera predeterminada, es thin. Los valores válidos son `none` (thin provisioning) y `volume` (thick provisioning).
`snapshotPolicy`	Esto establecerá la política de instantáneas en el valor deseado. El valor predeterminado es `none`, lo que significa que no se crearán instantáneas automáticamente 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 si se navega hasta `.snapshot` el directorio de cualquier directorio del volumen.
`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.

Opción	Descripción
`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, el clonado de una base de datos vacía puede ofrecer un gran ahorro de tiempo y un ahorro reducido en espacio de almacenamiento, por lo que es mejor dividir el clon de inmediato.
`encryption`	Habilite el cifrado de volúmenes de NetApp (NVE) en el nuevo volumen; los valores predeterminados son `false`. Para usar esta opción, debe tener una licencia para NVE y habilitarse en el clúster. Si NAE está habilitado en el back-end, cualquier volumen aprovisionado en Astra Trident estará habilitado para NAE. Para obtener más información, consulte: "Cómo funciona Astra Trident con NVE y NAE".
`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` serán el propietario. El texto o el formato numérico funcionará.
`snapshotDir`	Si configura esta opción en `true`, `.snapshot` el directorio será visible para los clientes que accedan al volumen. El valor por defecto es `false`, lo que significa que la visibilidad del `.snapshot` directorio está desactivada por defecto. Algunas imágenes, por ejemplo, la imagen oficial de MySQL, no funcionan como se esperaba cuando el `.snapshot` directorio es visible.

Opción	Descripción
`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`	Si se configura en `false`, se desactivará la función de asignación de espacio de la LUN. El valor predeterminado es `true`, lo que significa que 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 instantáneas y la política de instantáneas 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 a él se especifica mediante `-o type=service_level` la nomenclatura.

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, el valor predeterminado es 1GiB o la entrada de configuración… Valores predeterminados: {«size»: «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 (es decir, mediante `docker plugin` comandos), véalos 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 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 el host, los registros están disponibles en el directorio del host `/var/log/netappdvp`. Para activar el registro de depuración, especifique `-debug` cuándo se ejecuta 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 gestor de paquetes necesario para el sistema operativo host y compruebe `rpcbind` si se está ejecutando. Puede comprobar el estado del servicio rpcbind ejecutando a `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 múltiples instancias es darles diferentes nombres usando la `--alias` opción con el plugin en contenedor, u `--volume-driver` opción al instanciar 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`

Opción	Descripción	Ejemplo
`storagePrefix`	Prefijo opcional para los nombres de volúmenes. Predeterminado `netappdvp_`.	`staging_`
`limitVolumeSize`	Restricción opcional de los tamaños de volumen. Valor por defecto: '' (no forzado)	`10g`

No utilice `storagePrefix` (incluido el valor predeterminado) para los back-ends de elementos. De forma predeterminada, el `solidfire-san` controlador ignorará esta configuración 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. `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. Astra Trident utiliza "Asignación de LUN selectiva de ONTAP" para descubrir las LIF iSCI necesarias para establecer una sesión multivía. Se genera una advertencia si `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 el `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 solo para los `ontap-nas` conductores y `ontap-nas-economy`**. "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 solo para `ontap-san` el conductor**.	`netappdvp`
`limitVolumeSize`	Tamaño máximo de volumen que se puede solicitar.	`300g`
`qtreesPerFlexvol`	El número máximo de qtrees por FlexVol debe estar comprendido entre [50, 300], y el valor predeterminado es 200. **Para el `ontap-nas-economy` controlador, esta opción permite personalizar el número máximo de qtrees por FlexVol**.	`300`
`sanType`	**Compatible solo para `ontap-san` el conductor.** Utilice para seleccionar `iscsi` iSCSI o `nvme` NVMe/TCP.	`iscsi` si está en blanco
`limitVolumePoolSize`	**Compatible `ontap-san-economy` `ontap-san-economy` solo para conductores y.** Limita el tamaño de FlexVol en los controladores económicos de ONTAP ONTAP-nas y ONTAP-SAN.	`300g`

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` (thin provisioning) o `volume` (grueso)	`none`
`snapshotPolicy`	La política de Snapshot que se va a utilizar, el valor predeterminado es `none`	`none`
`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`	Habilita el cifrado de volúmenes de NetApp (NVE) en el nuevo volumen; se establece de forma predeterminada en `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 al `.snapshot` directorio, por defecto a. `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 y `unix` estilos de seguridad de NFS `mixed`. El valor predeterminado es `unix`.	`unix`
`fileSystemType`	Opción SAN para seleccionar el tipo de sistema de archivos, de forma predeterminada a. `ext4`	`xfs`

Opción	Descripción	Ejemplo
`tieringPolicy`	Política de organización en niveles que se debe utilizar, la opción predeterminada es `none`; `snapshot-only` para la configuración de SVM-DR anterior a ONTAP 9.5	`none`

Opciones de escala

Los `ontap-nas` controladores y `ontap-san` 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 volumen de Docker se ajustan a esa limitación, `ontap-nas` el controlador es la solución NAS preferida por las funciones adicionales que ofrece FlexVols, como las copias Snapshot granulares de volumen Docker y el clonado.

Si necesita más volúmenes de Docker de los que se pueden acomodar según los límites de FlexVol, elija el `ontap-nas-economy` o `ontap-san-economy` el controlador.

`ontap-nas-economy`El controlador crea volúmenes de Docker como qtrees de ONTAP dentro de un pool de volúmenes FlexVol 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. `ontap-nas-economy`El controlador no admite copias Snapshot granulares de volumen Docker ni clonado.

`ontap-nas-economy`El controlador no es compatible actualmente con Docker Swarm, porque Swarm no orquesta la creación de volúmenes en varios nodos.

`ontap-san-economy`El controlador crea volúmenes de Docker como LUN de ONTAP dentro de un pool compartido de volúmenes FlexVol 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 `ontap-nas-flexgroup` el controlador para aumentar el paralelismo hacia un único volumen que pueda crecer hasta alcanzar el rango 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 de miembro/volumen de FlexGroup recomendado es de 100GiB.
- No se admite la clonado en volúmenes de FlexGroup.

Para obtener información sobre las instancias de FlexGroup y las cargas de trabajo adecuadas para las instancias de FlexGroup, consulte la "Prácticas recomendadas y guía de implementación de los volúmenes FlexGroup de NetApp".

Para obtener características avanzadas y una gran escala en el mismo entorno, puede ejecutar varias instancias del complemento Docker Volume Plugin, con una usando `ontap-nas` y otra usando `ontap-nas-economy`.

Archivos de configuración de ONTAP de ejemplo

Ejemplo de NFS para el controlador <code> ONTAP-nas</code>

```
{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "username": "vsadmin",
    "password": "password",
    "aggregate": "aggr1",
    "defaults": {
      "size": "10G",
      "spaceReserve": "none",
      "exportPolicy": "default"
    }
}
```

Ejemplo de NFS para el controlador <code> ONTAP-nas-FlexGroup </code>

```
{
    "version": 1,
    "storageDriverName": "ontap-nas-flexgroup",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "username": "vsadmin",
    "password": "password",
    "defaults": {
      "size": "100G",
      "spaceReserve": "none",
      "exportPolicy": "default"
    }
}
```

Ejemplo de NFS para el controlador <code> ONTAP-nas-economy</code>

```
{
    "version": 1,
    "storageDriverName": "ontap-nas-economy",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "username": "vsadmin",
    "password": "password",
    "aggregate": "aggr1"
}
```

Ejemplo de iSCSI para el controlador <code> ONTAP-san</code>

```
{
    "version": 1,
    "storageDriverName": "ontap-san",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.3",
    "svm": "svm_iscsi",
    "username": "vsadmin",
    "password": "password",
    "aggregate": "aggr1",
    "igroupName": "netappdvp"
}
```

Ejemplo de NFS para el controlador <code> ONTAP-san-economy</code>

```
{
    "version": 1,
    "storageDriverName": "ontap-san-economy",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.3",
    "svm": "svm_iscsi_eco",
    "username": "vsadmin",
    "password": "password",
    "aggregate": "aggr1",
    "igroupName": "netappdvp"
}
```

Ejemplo de NVMe/TCP para el controlador <code> ONTAP-san</code>

```
{
  "version": 1,
  "backendName": "NVMeBackend",
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "svm": "svm_nvme",
  "username":"vsadmin",
  "password":"password",
  "sanType": "nvme",
  "useREST": true
}
```

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 realizó una actualización con volúmenes existentes, deberá configurar este valor para acceder a los volúmenes antiguos que se asignaron a través del método de nombre del volumen.	`netappdvp-`

`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 parámetro extra `config`.

```
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. Debido a la forma en que funciona ONTAP y cómo funcionan los `ontap-nas` controladores y `ontap-san`, son los únicos que 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.

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" proporcionar separación administrativa entre diferentes aplicaciones y constituyen 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. "mecanismo de cuotas de almacenamiento"Permite al administrador implementar límites de consumo globales y específicos de clase de almacenamiento, de capacidad y de recuento de objetos por espacio de nombres. Además, el uso de un "límite de rango" garantiza que las solicitudes RVP se encuentren dentro de un valor mínimo y máximo antes de que la solicitud se reenvíe 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 información sobre "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. Independientemente, el LIF de gestión debería ser accesible mediante DNS, lo que para facilitar la máxima flexibilidad debería "SVM-DR" ser usado 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 "NetApp Hardware Universe" para conocer su plataforma específica y la versión de ONTAP para determinar 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.

Los controladores y `ontap-san` Trident `ontap-nas` aprovisionan un FlexVolume para cada volumen persistente (VP) de Kubernetes que se crea. `ontap-nas-economy`El controlador crea aproximadamente un FlexVolume por cada 200 VP (configurable entre 50 y 300). `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>
```

El 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

El `max-volumes` 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 desde un nodo no se pueden aprovisionar con respecto (por ejemplo, debido a la capacidad), el otro nodo se convierte en el destino para todos los volúmenes aprovisionados de Trident. Esto significa que se puede alcanzar el límite de volumen para ese nodo antes de `max-volumes` alcanzar el valor, lo que afecta tanto al Trident como a otras operaciones de volumen que usan 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 para los volúmenes que puede crear Trident, use el `limitVolumeSize` parámetro en su `backend.json` definición.

Además de controlar el tamaño del volumen en la cabina de almacenamiento, también se deben aprovechar las capacidades de Kubernetes.

Limite el tamaño máximo de FlexVols creados por Trident

Para configurar el tamaño máximo para FlexVols utilizados como pools para los controladores ONTAP-san-economy y ONTAP-nas-economy, utilice el `limitVolumePoolSize` parámetro en su `backend.json` definición.

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. Cuando se usa `useCHAP` el parámetro en la configuración de back-end, Trident autentica las conexiones iSCSI para back-ends de ONTAP con CHAP.

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" que el contenedor esté fuera de control o que pueda afectar 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 QoS, 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 definir la `qosPolicy` clave en el `defaults` bloque de la configuración de backend. 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 acerca de los grupos de políticas de QoS, 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. Estos no se rechazan por defecto, así que asegúrese de desactivar la capacidad mediante el uso "directivas de seguridad de pod"de .

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. 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 `vserver export-policy create` el comando CLI de ONTAP para crear la política de exportación.
- Añada reglas a la política de exportación mediante `vserver export-policy rule create` el comando de la CLI de ONTAP.

Si ejecuta estos comandos, puede restringir el acceso de los nodos de Kubernetes a los datos.

Deshabilite `showmount` para la SVM de aplicaciones

```
`showmount`La función permite que un cliente NFS consulte a la SVM para obtener una lista de exportaciones NFS disponibles. Un pod puesto en marcha en el clúster de Kubernetes puede emitir `showmount -e` el comando contra la LIF de datos y recibir 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 deshabilitarlo mediante `showmount` el comando CLI de ONTAP a nivel de 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	Valor mínimo	Valor predeterminado	Valor máx. (4KB)
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 calidad de servicio y 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 las versiones más recientes en la "Biblioteca de NetApp".

ONTAP

- "Prácticas recomendadas y guía de implementación de NFS"
- "Guía de administración de los sistemas 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 "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 VP aprovisionado es un volumen flexible de ONTAP completo.
- `ontap-nas-economy`: Cada VP aprovisionado es un qtree, con un número configurable de qtrees por FlexVolume (el valor predeterminado es 200).
- `ontap-nas-flexgroup`: Cada VP aprovisionado como un ONTAP FlexGroup completo, y se utilizan todos los agregados asignados a una SVM.
- `ontap-san`: Cada VP aprovisionado es una LUN con su propio FlexVolume.
- `ontap-san-economy`: Cada VP aprovisionado es un 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.

Controladores para NAS de ONTAP	Snapshot	Clones	Políticas de exportación dinámicas	Conexión múltiple	Calidad de servicio	Cambie el tamaño	Replicación
`ontap-nas`	Sí	Sí	Nota de pie 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[]	Nota de pie de página:5[]	Sí	Nota de pie de página:3[]	Sí	Nota de pie de página:3[]
`ontap-nas-flexgroup`	Nota de pie de página:1[]	No	Nota de pie 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: YesFootnote:1[]: No gestionado por Astra Trident YesFootnote:2[]: Gestionado por Astra Trident, pero no por PV YesFootnote granular:3[]: No gestionado por Astra Trident y no por PV nota al pie de página granular:4[]: Soportado para volúmenes de bloque bruto YesFootnote:5[]: Admitido por Astra 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 la misma. Sin embargo, dado que `ontap-nas-economy` la controladora limita la capacidad para controlar la programación con una granularidad por VP, esto puede afectar a la recuperación ante desastres y a la planificación de backup en particular. Para los equipos de desarrollo que desean aprovechar la funcionalidad de clon de RVP en el almacenamiento de ONTAP, esto solo es posible cuando se utilizan los `ontap-nas` `ontap-san` controladores o. `ontap-san-economy`

`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 NetApp ONTAP te permite aprovechar las funciones, el rendimiento y las funcionalidades administrativas de NetApp que ya conoces, a la vez que aprovechas la simplicidad, la agilidad, la seguridad y la escalabilidad de almacenar datos en AWS. FSX para ONTAP es compatible con muchas funciones del sistema de archivos ONTAP y API de administración. 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

`solidfire-san`El controlador que se utiliza con las plataformas NetApp HCI/SolidFire ayuda al administrador a configurar un back-end de Element para Trident basándose en los límites de calidad de servicio. Si desea diseñar su backend de modo que establezca los límites específicos de QoS en los volúmenes aprovisionados por Trident, utilice el `type` parámetro en el archivo backend. El administrador también puede restringir el tamaño del volumen que se puede crear en el almacenamiento con `limitVolumeSize` el parámetro. Actualmente, las funciones de almacenamiento de Element como el cambio de tamaño de volúmenes y la replicación de volúmenes no se admiten mediante `solidfire-san` el 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[]

> Nota al pie: Yesfonote:1[]: No administrado por Astra Trident Yes [2]: Admitido para volúmenes de bloque RAW

Controladores de entorno de administración Azure NetApp Files

Astra Trident utiliza `azure-netapp-files` el controlador para gestionar "Azure NetApp Files"el servicio.

Puede encontrar más información sobre 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: Yesfonote:1[]: No administrado por Astra Trident

Cloud Volumes Service en el controlador back-end de Google Cloud

Astra Trident usa `gcp-cvs` el controlador para asociarse con Cloud Volumes Service en Google Cloud.

```
`gcp-cvs`El controlador utiliza pools virtuales para abstraer el back-end y permitir que Astra Trident determine la ubicación del volumen. El administrador define los pools virtuales de los `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 back-end. 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 los tipos de servicio y los niveles de servicio de Cloud Volumes Service en 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 definir tres conjuntos de filtros en la clase de almacenamiento `storagePools:`, `additionalStoragePools` Y/o `excludeStoragePools`.

```
`storagePools`El parámetro ayuda a restringir el almacenamiento al
conjunto de pools que coinciden con los atributos especificados.
`additionalStoragePools`El parámetro se utiliza para ampliar el conjunto
de pools que Astra Trident utilizará 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.
```

El `excludeStoragePools` parámetro se utiliza para excluir específicamente el juego de pools mostrado 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 el `media` atributo `hdd` como o. `ssd` En función del `media` atributo mencionado en la clase de almacenamiento, Trident seleccionará el back-end adecuado que sirve `hdd` o `ssd` agrega para que coincida con el atributo de medio y, a continuación, dirigirá el aprovisionamiento de los volúmenes al agregado concreto. Por lo tanto, podemos crear una clase PREMIUM de almacenamiento que tendría `media` un conjunto de atributos que `ssd` podría clasificarse como la política de calidad de servicio DE 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 de Azure NetApp Files con varias etiquetas, que representan diferentes niveles de rendimiento. Establezca `servicelevel` Aspect en el nivel de rendimiento adecuado y agregue otros aspectos requeridos en cada etiqueta. Ahora cree diferentes clases de almacenamiento de Kubernetes que se asignarán a diferentes pools virtuales. En este `parameters.selector` campo, cada StorageClass llama la atención sobre los pools virtuales que se pueden usar 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 utilizando `parameters.selector` el campo que se asignaría 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 algunos aspectos del volumen se modifiquen 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, solo tiene que crear una reclamación de volumen persistente y mencionar la `datasource` 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` en `true` en el StorageClass asociado con el volumen. Siempre que sea necesario cambiar el tamaño del volumen persistente, edite la `spec.resources.requests.storage` anotación en la reclamación Volumen persistente al tamaño de volumen deseado. 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` `azure-netapp-files` los controladores , , `ontap-nas-flexgroup`, , `solidfire-san` y. `gcp-cvs` Esta función es útil cuando se pasa una aplicación existente a Kubernetes o durante escenarios de recuperación ante desastres.

Cuando uses la ONTAP y `solidfire-san` los controladores, utiliza el comando `tridentctl import volume <backend-name> <volume-name> -f /path/pvc.yaml` para importar un volumen existente a Kubernetes que gestione Astra Trident. 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 `azure-netapp-files` se utiliza el controlador o `gcp-cvs`, utilice el comando `tridentctl import volume <backend-name> <volume path> -f /path/pvc.yaml` para importar el volumen a Kubernetes que gestionará 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. Añadirá (y sobrescribirá automáticamente si es necesario) el tamaño de volumen de la RVP configurada. 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

La implementación y gestión del almacenamiento para el registro se ha documentado en "netapp.io"la "blog".

Servicio de registro

Al igual que otros servicios de OpenShift, el servicio de registro se implementa mediante Ansible con los parámetros de configuración suministrados por el archivo de inventario, también conocido como 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 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 puesta en marcha 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, deberá establecer la variable Ansible `openshift_enable_unsupported_configurations` para `true` 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 desplegar 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.

Configurar las variables de Ansible de acuerdo con el método de puesta en marcha es importante para garantizar que los servicios subyacentes utilizan el almacenamiento correcto. Veamos las opciones para cada uno de los métodos de despliegue.

Las siguientes tablas solo incluyen las variables relevantes para la configuración del almacenamiento en relación con el servicio de registro. Puede encontrar otras opciones en "Documentación de registro de RedHat OpenShift" las que se deben revisar, configurar y utilizar según su implementación.

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 NFS para el servicio de registro.

Variable	Detalles
`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.
`openshift_logging_storage_nfs_directory`	La ruta de montaje para la exportación NFS. Por ejemplo, si el volumen se une como `/openshift_logging`, usaría esa ruta para esta variable.
`openshift_logging_storage_volume_name`	El nombre, por ejemplo `pv_ose_logs`, del VP que se va a crear.
`openshift_logging_storage_volume_size`	El tamaño de la exportación NFS, por ejemplo `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 esta opción `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 métricas se puede implementar durante una configuración inicial del cluster o después de su funcionamiento utilizando 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 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 une como `/openshift_metrics`, usaría esa ruta para esta variable.
`openshift_metrics_storage_volume_name`	El nombre, por ejemplo `pv_ose_metrics`, del VP que se va a crear.
`openshift_metrics_storage_volume_size`	El tamaño de la exportación NFS, por ejemplo `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 va a poner en marcha mediante los libros de estrategia de componentes, después de la instalación de OpenShift, Ansible creará cualquier RVP necesario y, después de que Astra Trident haya

aprovisionado 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"la versión de modo que esté configurada para el 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. Haga una copia de seguridad del clúster etcd de Kubernetes con "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 de miembros etcd en `/var/lib/etcd`.
3. Restaure el clúster de Kubernetes desde el backup etcd con "Kubernetes: Restaurar un clúster ETCD".
4. Ejecutar `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 backend de Trident.
3. Confirme que `storagePrefix` está presente en el archivo de definición de backend de Trident. Este parámetro no puede cambiarse. Si se omite `storagePrefix`, se producirá un error en la actualización del backend.
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 ha especificado `-identity-preserve false` o. `discard-config network`, debe devolver todos los pods de aplicación.

> Si ha especificado `-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 los volúmenes creados por Astra Trident.

Luego, es posible importar los volúmenes recuperados a Astra Trident con "importación de volumen tridentctl".

> La importación no está soportada en `ontap-nas-economy` los controladores , , `ontap-san-economy`o. `ontap-flexgroup-economy`

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 de 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. Debes asegurarte de permitir solo el acceso de los administradores al espacio de nombres de Astra Trident y, por lo tanto, el acceso a `tridentctl` la aplicación.

Utilice la autenticación CHAP con los back-ends DE SAN de ONTAP

Astra Trident admite la autenticación basada en CHAP para cargas de trabajo SAN de ONTAP (mediante `ontap-san` y `ontap-san-economy` controladores). NetApp recomienda utilizar CHAP bidireccional con Astra Trident para la autenticación entre un host y el back-end de almacenamiento.

Para los back-ends ONTAP que utilizan controladores de almacenamiento SAN, Astra Trident puede configurar CHAP bidireccional y gestionar nombres de usuario y secretos CHAP a través de `tridentctl`. Consulte ""para comprender cómo Astra Trident configura CHAP en back-ends de ONTAP.

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 se instala Astra Trident, gestiona los secretos CHAP y los almacena en `tridentvolume` un objeto CR para el VP correspondiente. Al crear un VP, Astra Trident utiliza los secretos de CHAP para iniciar una sesión iSCSI y comunicarse con el sistema NetApp HCI y SolidFire a través de CHAP.

Los volúmenes que crea Astra Trident no están asociados con ningún grupo de acceso de volumen.

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, cualquier volumen aprovisionado en Astra Trident se habilitará NVE a menos que establezca la marca de cifrado de NVE en `false` 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 de cifrado de NVE `true` en la configuración de back-end de Trident para anular el cifrado NAE y usar una clave de cifrado específica por volumen.
- Si se establece la marca de cifrado de NVE `false` en un back-end con NAE habilitado, se creará un volumen con la función NAE habilitada. No se puede deshabilitar el cifrado NAE mediante la marca de cifrado de NVE en `false`.

- Si desea crear manualmente un volumen de NVE en Astra Trident, debe establecer explícitamente la marca de cifrado de NVE en `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 con LUKS utilizan el cifrado y el modo aes-xts-plain64, como recomienda "NIST".

Antes de empezar

- Los nodos de trabajo deben tener instalado cryptsetup 2.1 o superior (pero inferior a 3.0). Para obtener 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 de backend 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. Se utiliza `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: csi.trident.netapp.io
parameters:
  selector: "luks=true"
  csi.storage.k8s.io/node-stage-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

3. 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` en(`true` en el backend. `luksEncryption`La opción indica a Astra Trident si el volumen es compatible con LUKS (`true`) o no 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'
```

Configuración de PVC para importar volúmenes LUKS

Para importar volúmenes LUKS dinámicamente, establezca la anotación `trident.netapp.io/luksEncryption` en `true` e incluya una clase de almacenamiento habilitada para LUKS en la RVP como se muestra en este ejemplo.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: luks-pvc
  namespace: trident
  annotations:
    trident.netapp.io/luksEncryption: "true"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: luks-sc
```

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, `previous-luks-passphrase` se omite el parámetro.

Pasos

1. Añada `node-publish-secret-name` los parámetros y `node-publish-secret-namespace` 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 de que `previous-luke-passphrase-name` `previous-luks-passphrase`coincide 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 contraseñas, 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. Active la `CSINodeExpandSecret` puerta de función (beta 1,25+). Consulte "Kubernetes 1.25: Use Secrets for Node-Driven Expansion of CSI Volumes" para obtener más información.
2. Añada `node-expand-secret-name` los parámetros y `node-expand-secret-namespace` StorageClass. Por ejemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: csi.trident.netapp.io
parameters:
  selector: "luks=true"
  csi.storage.k8s.io/node-stage-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
  csi.storage.k8s.io/node-expand-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-expand-secret-namespace: ${pvc.namespace}
allowVolumeExpansion: true
```

Resultados

Al iniciar la ampliación de almacenamiento en línea, el kubelet pasa las credenciales adecuadas al controlador.

Conocimiento y apoyo

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?

A partir del lanzamiento de la versión 24,02, Astra Trident se lanza cada cuatro meses: Febrero, junio y octubre.

¿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?

Utiliza `tridentctl get` el comando para obtener más información sobre la configuración de Astra Trident.

¿Puedo obtener mediciones sobre cómo aprovisiona Astra Trident el almacenamiento?

Sí. Extremos de Prometheus que se pueden utilizar para recopilar información sobre la operación de Astra Trident, como el número de back-ends gestionados, el número de volúmenes aprovisionados, bytes consumidos, etc. También puede "Cloud Insights" utilizar para supervisión y análisis.

¿Cambia la experiencia del usuario al utilizar Astra Trident como aprovisionador CSI?

No. No hay cambios en lo que respecta a la experiencia del usuario y las funcionalidades. El nombre de aprovisionador utilizado 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

¿Admite Astra Trident una instalación sin conexión desde un registro privado?

Sí, Astra Trident se puede instalar sin conexión. Consulte "Obtenga más información sobre la instalación de Astra Trident".

¿Puedo instalar Astra Trident de forma remota?

Sí. Astra Trident 18,10 y versiones posteriores admiten la funcionalidad de instalación remota desde cualquier máquina que tenga `kubectl` acceso al clúster. Después de `kubectl` verificar el acceso (por ejemplo, inicie un `kubectl get nodes` comando desde el equipo remoto para verificar), siga las instrucciones de instalación.

¿Puedo configurar la alta disponibilidad con Astra Trident?

Astra Trident se instala como puesta en marcha de Kubernetes (ReplicaSet) en una instancia, por lo que tiene HA incorporada. No debería 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 ClusterRole de Kubernetes, que tiene acceso específico a los recursos PersistentVolume, PersistentVolumeClaim, StorageClass y Secret del clúster de Kubernetes. Consulte "Personalice la instalación trimentctl".

¿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 "Personalice la instalación trimentctl".

¿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 y/o especifique un parámetro único `StoragePrefix` en el `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. Para iniciar un caso de soporte, póngase en contacto con "Soporte de NetApp".

¿Cómo se genera un bundle del registro de soporte?

Puede crear un paquete de soporte ejecutando `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?

Crea un problema "Astra Trident Github" y menciona **RFE** en el asunto y la descripción del problema.

¿Dónde puedo elevar un defecto?

Crear 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 tienes alguna pregunta, problema o solicitud, ponte en contacto con nosotros a través de Astra "Canal de discordia" o GitHub.

La contraseña de mi sistema de almacenamiento ha cambiado y Astra Trident ya no funciona. ¿Cómo me recupero?

Actualice la contraseña del backend con `tridentctl update backend myBackend -f </path/to_new_backend.json> -n trident`. Reemplace `myBackend` en el ejemplo con su nombre de backend y `` `/path/to_new_backend.json `` con la ruta al archivo correcto `backend.json`.

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?

Si necesita una corrección de los errores observados después de una actualización, problemas de dependencia o una actualización incorrecta o incompleta, debe "Desinstale Astra Trident"volver a instalar la versión anterior siguiendo las instrucciones específicas para esa versión. Esta es la única forma recomendada de cambiar a una versión anterior.

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. Astra Trident utiliza "Asignación de LUN selectiva de ONTAP" para descubrir las LIF iSCI necesarias para establecer una sesión multivía. Se genera una advertencia si `dataLIF` se define explícitamente. Consulte "Opciones y ejemplos de configuración SAN de ONTAP" para obtener más información.
- ONTAP nas: 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 información

¿Puede Astra Trident configurar CHAP para los back-ends de ONTAP?

Sí. Astra Trident es compatible con CHAP bidireccional para back-ends de ONTAP. Esto requiere configuración `useCHAP=true` en la configuración de backend.

¿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.

¿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 los back-ends NAS de ONTAP.
- `managementLIF` Para back-ends de SAN de ONTAP. No se puede especificar `dataLIF` en un back-end de SAN de ONTAP.

Astra Trident debe instalarse con el flag `--use-ipv6` (para `tridentctl` la instalación), `IPv6` (para el operador Trident) o `tridentTPv6` (para la instalación Helm) para que funcione en IPv6.

¿Se puede actualizar la LIF de gestión en el back-end?

Sí, es posible actualizar la LIF de gestión de back-end con `tridentctl update backend` el comando.

¿Es posible actualizar la LIF de datos en el back-end?

Solo puede actualizar la LIF de datos en `ontap-nas` y `ontap-nas-economy`.

¿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 de backend no se pueden utilizar para seleccionar automáticamente los pools correctos para una clase, los `storagePools` parámetros y `additionalStoragePools` se utilizan para seleccionar un juego específico de pools.

¿Cómo puedo asegurarme de que Astra Trident no se provisione desde un back-end específico?

```
`excludeStoragePools`El parámetro se utiliza para filtrar el conjunto de pools que Astra Trident utilizará para el aprovisionamiento y quitará los pools que coincidan.
```

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 según los parámetros presentes en `StorageClass` y `PersistentVolumeClaim`. Por ejemplo, si hay varios back-ends de controlador ONTAP-nas, Astra Trident intenta hacer coincidir los parámetros en `StorageClass` y `PersistentVolumeClaim` combinado y hacer coincidir un backend que pueda entregar 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?

`ontap-nas-economy`El controlador crea hasta 200 Qtrees en el mismo FlexVol (configurable entre 50 y 300), 100.000 Qtrees por nodo del clúster y 2,4m por clúster. Cuando introduce un nuevo `PersistentVolumeClaim` que recibe servicio del controlador de economía, el conductor busca ver si ya existe un 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 de Kubernetes, siga el ejemplo proporcionado "aquí".

¿Cómo se configuran los volúmenes aprovisionados en una política de exportación específica?

Para permitir que los hosts adecuados accedan a un volumen, utilice 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?

Utilice `StorageClasses` para implementar la calidad de servicio para 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 provisionamiento grueso, debe configurar el archivo de definición de backend o el `StorageClass`. Si ambos están configurados, `StorageClass` tiene prioridad. Configure lo siguiente para ONTAP:

1. Activado `StorageClass`, defina el `provisioningType` atributo como grueso.
2. En el archivo de definición de back-end, habilite los volúmenes gruesos configurando `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.

¿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.
```

¿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 activado la puerta de función.

¿Cuáles son los controladores compatibles con las instantáneas de volumen de Astra Trident?

A partir de hoy, la asistencia de instantáneas bajo demanda está disponible para nuestro `ontap-nas`, , `ontap-nas-flexgroup`, , , `ontap-san` `ontap-san-economy` , `solidfire-san` `gcp-cvs`, y `azure-netapp-files` controladores de backend.

¿Cómo puedo realizar un backup con Snapshot de un volumen aprovisionado por Astra Trident con ONTAP?

Está disponible en `ontap-nas` `ontap-san` los controladores , y. `ontap-nas-flexgroup` También puede especificar un `snapshotPolicy` para `ontap-san-economy` el controlador en el nivel de FlexVol.

También está disponible en `ontap-nas-economy` los controladores, pero en la granularidad de FlexVol, no en la de qtree. Para habilitar la capacidad de los volúmenes Snapshot aprovisionados por Astra Trident, se debe establecer la opción del parámetro backend `snapshotPolicy` con la política Snapshot que se desee tal

y como 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, también puedes reservar un porcentaje específico de espacio en disco para almacenar las copias snapshot mediante Astra Trident estableciendo `snapshotReserve` el atributo en el archivo de definición de back-end. Si ha configurado `snapshotPolicy` y `snapshotReserve` en el archivo de definición de backend, el porcentaje de reserva de instantánea se establece de acuerdo con el `snapshotReserve` porcentaje mencionado en el archivo backend. Si no se menciona el `snapshotReserve` número de porcentaje, ONTAP toma por defecto el porcentaje de reserva de instantáneas como 5. Si la `snapshotPolicy` opción se define en none, el porcentaje de reserva de instantáneas se establece en 0.

¿Puedo acceder directamente al directorio de snapshot del volumen y copiar los archivos?

Sí, puede acceder al directorio snapshot en el volumen aprovisionado por Trident mediante la configuración del `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.

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.

Resolución de problemas generales

- Si el pod de Trident no funciona correctamente (por ejemplo, cuando el pod de Trident está atascado `ContainerCreating` en la fase con menos de dos contenedores listos), se está ejecutando `kubectl -n trident describe deployment trident` y `kubectl -n trident describe pod trident-**` puede proporcionar información adicional. Obtener registros de kubelet (por ejemplo, vía `journalctl -xeu kubelet`) también puede ser útil.
- Si no hay información suficiente en los registros de Trident, puede intentar habilitar el modo de depuración para Trident pasando la `-d` marca al parámetro install según su opción de instalación.

  A continuación, confirme que la depuración se ha definido 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 comprobar esto observando la columna 'AGE' en la salida de `kubectl get pod -n trident`.

  Para usar 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 backend incluyendo `debugTraceFlags` en su definición de backend. Por ejemplo, incluya `debugTraceFlags: {“api”:true, “method”:true,}` para obtener llamadas a la API y recorridos de métodos en los registros de Trident. Los back-ends existentes se pueden `debugTraceFlags` configurar con un `tridentctl backend update`.
- Cuando utilice RedHat CoreOS, asegúrese de que `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 con el que se puede encontrar cuando se utiliza Trident "Azure NetApp Files" es cuando los secretos del inquilino y del cliente provienen de un registro de la aplicación con permisos

insuficientes. Para ver una lista completa de los requisitos de Trident, consulte "Azure NetApp Files"la configuración.

- Si hay problemas con el montaje de un PV en un contenedor, asegúrese de que `rpcbind` está instalado y en ejecución. Utilice el gestor de paquetes necesario para el sistema operativo host y compruebe `rpcbind` si se está ejecutando. Puede comprobar el estado `rpcbind` del servicio ejecutando un `systemctl status rpcbind` o su equivalente.
- Si un back-end de Trident informa de que está en `failed` estado a pesar de haber trabajado anteriormente, es probable que se deba al cambio de las credenciales de SVM/admin asociadas al back-end. La actualización de la información de backend mediante `tridentctl update backend` el pod de Trident o el reinicio solucionará este problema.
- Si encuentra problemas de permiso al instalar Trident con Docker como tiempo de ejecución del contenedor, intente instalar Trident con el `--in cluster=false` indicador. Esto no utilizará un pod de instalador y evitará problemas de permisos que se ven debido al `trident-installer` usuario.
- Utilice el `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, ejecute primero `tridentctl uninstall` el comando para quitar Trident. Descargue el deseado "Versión de Trident" e instálelo con `tridentctl install` el comando.
- Tras una instalación correcta, si una RVP se atasca en `Pending` la fase, en ejecución `kubectl describe pvc` se puede proporcionar información adicional sobre por qué Trident no pudo aprovisionar un VP para esta RVP.

Implementación incorrecta de Trident con el operador

Si está desplegando Trident mediante el operador, el estado de `TridentOrchestrator` cambia de `Installing` a `Installed`. Si observa `Failed` el estado y el operador no puede recuperarse por sí mismo, 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 entender por qué la instalación de Trident no se ha realizado correctamente, debe echar un vistazo al `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. Dado que cada clúster de Kubernetes solo puede tener una instancia de Trident, el operador se asegura de que en un momento dado solo haya una activa `TridentOrchestrator` que pueda crear.

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 las vainas no pueden inicializarse completamente porque no se obtuvieron una o más imágenes contenedoras.

Para solucionar el problema, debe editar el `TridentOrchestrator` CR. Como alternativa, puede suprimir `TridentOrchestrator` y crear uno nuevo con la definición modificada y precisa.

Puesta en marcha de Trident incorrecta mediante `tridentctl`

Para ayudar a averiguar qué salió mal, podría ejecutar el instalador de nuevo usando el `-d` argumento, que activará el modo de depuración y le ayudará a entender cuál es el problema:

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

Después de resolver el problema, puede limpiar la instalación del modo siguiente y, a continuación, ejecutar `tridentctl install` el comando de nuevo:

```
./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.
```

Elimina por completo Astra Trident y CRD

Puedes quitar por completo Astra Trident y todos los CRD creados y los recursos personalizados asociados.

Esta acción no se puede deshacer. No hagas esto a menos que quieras una instalación completamente nueva de Astra Trident. Para desinstalar Astra Trident sin eliminar CRD, consulte "Desinstale Astra Trident".

Operador de Trident

Para desinstalar Astra Trident y eliminar completamente CRD mediante el operador Trident:

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

Timón

Para desinstalar Astra Trident y eliminar por completo CRD mediante Helm:

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

<code>tridentctl</code>

Para eliminar completamente los CRD después de desinstalar Astra Trident usando `tridentctl`

```
tridentctl obliviate crd
```

Se produce un error al anular el almacenamiento en caché del nodo de NVMe con espacios de nombres de bloque sin configurar RWX o Kubernetes 1,26

Si ejecuta Kubernetes 1,26, la anulación del almacenamiento provisional del nodo puede fallar cuando se usa NVMe/TCP con espacios de nombres de bloque sin configurar de RWX. Los siguientes escenarios proporcionan una solución alternativa al fallo. También puede actualizar Kubernetes a 1,27.

Se ha eliminado el espacio de nombres y el pod

Piensa en un escenario en el que tienes un espacio de nombres gestionado por Astra Trident (volumen persistente NVMe) conectado a un pod. Si elimina el espacio de nombres directamente desde el backend de ONTAP, el proceso de anulación del almacenamiento provisional se bloquea después de intentar eliminar el pod. Este escenario no afecta al clúster de Kubernetes ni a otro funcionamiento.

Solución alternativa

Desmonte el volumen persistente (que corresponde al espacio de nombres) del nodo correspondiente y elimínelo.

LIF de datos bloqueadas

```
 If you block (or bring down) all the dataLIFs of the NVMe Astra Trident
backend, the unstaging process gets stuck when you attempt to delete the
pod. In this scenario, you cannot run any NVMe CLI commands on the
Kubernetes node.
.Solución alternativa
Abra dataLIFS para restaurar la funcionalidad completa.
```

Se ha eliminado la asignación de espacio de nombres

```
 If you remove the `hostNQN` of the worker node from the corresponding
subsystem, the unstaging process gets stuck when you attempt to delete the
pod. In this scenario, you cannot run any NVMe CLI commands on the
Kubernetes node.
.Solución alternativa
Vuelva a agregar el `hostNQN` al subsistema.
```

Soporte técnico

NetApp ofrece compatibilidad con Astra Trident de muchas maneras. Hay disponibles amplias opciones de soporte gratuito las 24 horas del día, los 7 días de la semana, como artículos de la base de conocimiento (KB) y un canal Discord.

Ciclo de vida del soporte de Astra Trident

Astra Trident proporciona tres niveles de soporte en función de su versión. Consulte "Compatibilidad con la versión del software NetApp para definiciones".

Soporte completo
: Astra Trident proporciona soporte completo durante doce meses a partir de la fecha de lanzamiento.

Soporte limitado
: Astra Trident ofrece un soporte limitado durante los meses entre 13 y 24 desde la fecha de lanzamiento.

Autosoporte
: La documentación de Astra Trident está disponible durante los meses 25 - 36, desde la fecha de lanzamiento.

Versión	Soporte completo	Soporte limitado	Autosoporte
"24,06"	Junio de 2025	Junio de 2026	Junio de 2027
"24,02"	Febrero de 2025	Febrero de 2026	Febrero de 2027
"23,10"	Octubre de 2024	Octubre de 2025	Octubre de 2026

Versión	Soporte completo	Soporte limitado	Autosoporte
"23,07"	Julio de 2024	Julio de 2025	Julio de 2026
"23,04"	—	Abril de 2025	Abril de 2026
"23,01"	—	A enero de 2025	A enero de 2026
"22,10"	—	Octubre de 2024	Octubre de 2025
"22,07"	—	Julio de 2024	Julio de 2025
"22,04"	—	—	Abril de 2025
"22,01"	—	—	A enero de 2025
"21,10"	—	—	Octubre de 2024

Autosoporte

Para obtener una lista completa de 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".

Soporte de la comunidad

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.

Soporte técnico de NetApp

Si necesita ayuda con Astra Trident, cree un paquete de soporte con `tridentctl logs -a -n trident` y envíelo a `NetApp Support <Getting Help>`.

Si quiere más información

- "Blogs de Astra"
- "Blogs de Astra Trident"
- "Kubernetes Hub"
- "NetApp.io"

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

API DE REST de Astra Trident

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

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, Astra Trident `REST API` está restringido al host local de forma predeterminada cuando se ejecuta dentro de un pod. Para cambiar este comportamiento, debes establecer el argumento de Astra Trident `-address` en su configuración de pod.

Uso de la API DE REST

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

La API funciona de la siguiente manera:

OBTENGA

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

Muestra todos los objetos de ese tipo.

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

Obtiene los detalles del objeto con nombre.

PUBLICAR

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

Crea un objeto del tipo especificado.

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

ELIMINAR

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

Suprime el recurso con nombre.

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

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`

Activa la salida de depuración.

`-loglevel <level>`

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

Kubernetes

`-k8s_pod`

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

-k8s_api_server <insecure-address:insecure-port>

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

Docker

-volume_driver <name>

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

-driver_port <port-number>

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

-config <file>

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

DESCANSO

-address <ip-or-host>

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

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

-port <port-number>

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

-rest

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

Objetos de Kubernetes y Trident

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

¿Cómo interactúan los objetos entre sí?

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

1. Un usuario crea un `PersistentVolumeClaim` pedido nuevo de `PersistentVolume` un tamaño concreto a partir de un Kubernetes `StorageClass` que había configurado previamente el administrador.
2. El Kubernetes `StorageClass` identifica a Trident como su aprovisionador e incluye parámetros que indican a Trident cómo aprovisionar un volumen para la clase solicitada.
3. Trident mira por sí mismo `StorageClass` con el mismo nombre que identifica la coincidencia `Backends` y `StoragePools` que puede utilizar para aprovisionar volúmenes para la clase.
4. Trident aprovisiona almacenamiento en un back-end coincidente y crea dos objetos: Un `PersistentVolume` en Kubernetes que indica a Kubernetes cómo encontrar, montar y tratar el volumen, así como un volumen en Trident que conserva la relación entre `PersistentVolume` el y el almacenamiento real.
5. Kubernetes enlaza los `PersistentVolumeClaim` a la nueva `PersistentVolume`. Pods que incluyen el `PersistentVolumeClaim` montaje de Volume persistente en cualquier host en el que se ejecute.
6. Un usuario crea `VolumeSnapshot` un de un RVP existente, utilizando un `VolumeSnapshotClass` que apunta a Trident.
7. Trident identifica el volumen asociado con la RVP y crea una copia Snapshot del volumen en su back-end. También crea un `VolumeSnapshotContent` que le indica a Kubernetes cómo identificar la snapshot.
8. Un usuario puede crear un `PersistentVolumeClaim` uso `VolumeSnapshot` como origen.
9. Trident identifica la instantánea necesaria y realiza el mismo juego de pasos involucrados en la creación de un `PersistentVolume` y `Volume` un .

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

`PersistentVolumeClaim`Objetos de Kubernetes

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

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

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

Anotación	Opción de volumen	Controladores compatibles
trident.netapp.io/snapshotReserve	Reserva de copias Snapshot	ontap-nas, ontap-nas-flexgroup, ontap-san, gcp-cvs
trident.netapp.io/snapshotDirectory	Snapshot shotDirectory	ontap-nas, ontap-nas-economy y ontap-nas-flexgroup
trident.netapp.io/unixPermissions	Permisos univalados	ontap-nas, ontap-nas-economy y ontap-nas-flexgroup
trident.netapp.io/blockSize	Tamaño del bloque	solidfire-san

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

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

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

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

Considere los siguientes puntos:

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

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

`PersistentVolume`Objetos de Kubernetes

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

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

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

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

`StorageClass`Objetos de Kubernetes

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

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

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

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

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

Los parámetros de la clase de almacenamiento son:

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

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

Atributos de selección del pool de almacenamiento

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

Atributo	Tipo	Valores	Oferta	Solicitud	Admitido por
media 1	cadena	hdd, híbrido, ssd	Pool contiene medios de este tipo; híbrido significa ambos	Tipo de medios especificado	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san y solidfire-san
AprovisionaciónTipo	cadena	delgado, grueso	El pool admite este método de aprovisionamiento	Método de aprovisionamiento especificado	grueso: all ONTAP; thin: all ONTAP y solidfire-san
Tipo de backendType	cadena	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san, gcp-cvs, azure-netapp-files, ontap-san-economy	Pool pertenece a este tipo de backend	Backend especificado	Todos los conductores

Atributo	Tipo	Valores	Oferta	Solicitud	Admitido por
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 sería que pueda utilizar `attributes` solo para modelar las cualidades del almacenamiento que necesita para satisfacer las necesidades de una clase determinada. Trident detecta y selecciona automáticamente los pools de almacenamiento que coinciden con *todos* de los `attributes` especificados.

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

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

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

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

En los `storagePools` parámetros y `additionalStoragePools`, cada entrada toma el formato `<backend>:<storagePoolList>`, donde `<storagePoolList>` es una lista separada por comas de pools de almacenamiento para el backend especificado. Por ejemplo, un valor para `additionalStoragePools` puede ser similar a `ontapnas_192.168.1.100:aggr1,aggr2;solidfire_192.168.1.101:bronze`.

Estas listas aceptan valores regex para los valores de backend y list. Puede utilizar `tridentctl get backend` para obtener la lista de back-ends y sus pools.

Atributos de Kubernetes

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

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

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

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

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

- Consulte "Expanda los volúmenes"para obtener más información sobre la expansión de volumen.
- El paquete de instalación de Trident proporciona varias definiciones de clase de almacenamiento de ejemplo para su uso con Trident en `sample-input/storage-class-*.yaml`. Al eliminar una clase de almacenamiento Kubernetes, también se elimina el tipo de almacenamiento Trident correspondiente.

`VolumeSnapshotClass`Objetos de Kubernetes

Los objetos de Kubernetes `VolumeSnapshotClass` son análogos a `StorageClasses`. Ayudan a definir varias clases de almacenamiento y las instantáneas de volumen hacen referencia a ellas para asociar la snapshot a la clase de snapshot necesaria. Cada copia de Snapshot de volumen se asocia con una sola clase de copia de Snapshot de volumen.

Un administrador debe definir a `VolumeSnapshotClass` para crear instantáneas. Una clase de snapshot de volumen se crea con la siguiente definición:

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: csi-snapclass
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

El `driver` especifica a Kubernetes que Trident gestiona las solicitudes de instantáneas de volumen de `csi-snapclass` la clase. El `deletionPolicy` especifica la acción que se debe realizar cuando se debe eliminar una instantánea. `deletionPolicy`Cuando se establece en `Delete`, los objetos Snapshot del volumen, así como la snapshot subyacente en el clúster de almacenamiento, se eliminan cuando se elimina una snapshot. Como alternativa, si se configura en `Retain`, `VolumeSnapshotContent` se conservan la instantánea física y la física.

`VolumeSnapshot`Objetos de Kubernetes

Un objeto de Kubernetes `VolumeSnapshot` es una solicitud para crear una snapshot de un volumen. Del mismo modo que la RVP representa una solicitud al usuario para un volumen, un snapshot de volumen es una solicitud al que hace un usuario para crear una copia Snapshot de una RVP existente.

Cuando se recibe una solicitud de copia de Snapshot de volumen, Trident gestiona automáticamente la creación de la copia de Snapshot para el volumen en el back-end y expone la copia de Snapshot mediante la creación de un objeto único
`VolumeSnapshotContent`. Puede crear instantáneas a partir de EVs existentes y utilizar las instantáneas como DataSource al crear nuevas CVP.

> El ciclo de vida de un VolumeSnapshot es independiente del PVC de origen: Una instantánea persiste incluso después de eliminar el PVC de origen. Cuando se elimina un PVC que tiene instantáneas asociadas, Trident Marca el volumen de respaldo de este PVC con el estado **Eliminación**, pero no lo elimina por completo. El volumen se elimina cuando se eliminan todas las Snapshot asociadas.

`VolumeSnapshotContent`Objetos de Kubernetes

Un objeto de Kubernetes `VolumeSnapshotContent` representa una snapshot tomada de un volumen ya aprovisionado. Es análogo a A `PersistentVolume` y significa una snapshot aprovisionada en el clúster de almacenamiento. Al igual que `PersistentVolumeClaim` y `PersistentVolume` objetos, cuando se crea una snapshot, `VolumeSnapshotContent` el objeto mantiene una asignación uno a uno con el `VolumeSnapshot` objeto, que había solicitado la creación de la snapshot.

El `VolumeSnapshotContent` objeto contiene detalles que identifican de forma exclusiva la instantánea, como el `snapshotHandle`. Esta `snapshotHandle` es una combinación única del nombre del VP y el nombre del `VolumeSnapshotContent` objeto.

Cuando llega una solicitud de Snapshot, Trident crea la snapshot en el back-end. Después de crear la copia Snapshot, Trident configura un `VolumeSnapshotContent` objeto y, por lo tanto, la copia Snapshot se expone a la API de Kubernetes.

> Por lo general, no es necesario administrar el `VolumeSnapshotContent` objeto. Una excepción a esto es cuando se desea "importe una copia de snapshot de volumen"crear fuera de Astra Trident.

`CustomResourceDefinition`Objetos de Kubernetes

Los recursos personalizados de Kubernetes son extremos en la API de Kubernetes que define el administrador y que se usan para agrupar objetos similares. Kubernetes admite la creación de recursos personalizados para almacenar un conjunto de objetos. Puede obtener estas definiciones de recursos ejecutando `kubectl get crds`.

Kubernetes almacena en su almacén de metadatos las definiciones de recursos personalizadas (CRD) y los metadatos de objetos asociados. De este modo, no es necesario disponer de un almacén aparte para Trident.

Astra Trident usa `CustomResourceDefinition` objetos para conservar la identidad de los 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 sencillo, cuando se crea un backend con `tridentctl`, se crea un objeto CRD correspondiente `tridentbackends` para su consumo por Kubernetes.

A continuación se indican algunos puntos que hay que tener en cuenta sobre los CRD de Trident:

- Cuando se instala Trident, se crea un conjunto de CRD que se puede utilizar como cualquier otro tipo de recurso.
- Al desinstalar Trident mediante el `tridentctl uninstall` comando, los pods de Trident se eliminan pero los CRD creados no se limpian. Consulte "Desinstale Trident"para comprender cómo Trident se puede eliminar por completo y volver a configurar desde cero.

Objetos de Astra Trident `StorageClass`

Trident crea clases de almacenamiento coincidentes para los objetos de Kubernetes `StorageClass` que se especifican `csi.trident.netapp.io` en su campo aprovisionador. El nombre de la clase de almacenamiento coincide con el del objeto de Kubernetes `StorageClass` que representa.

Con Kubernetes, estos objetos se crean automáticamente cuando se registra un Kubernetes `StorageClass` que utiliza Trident como aprovisionador.

Las clases de almacenamiento comprenden un conjunto de requisitos para los volúmenes. Trident enlaza estos requisitos con los atributos presentes en cada pool de almacenamiento; si coinciden, ese pool de almacenamiento es un objetivo válido para aprovisionar volúmenes que utilizan esa clase de almacenamiento.

Puede crear configuraciones de clase de almacenamiento para definir clases de almacenamiento directamente mediante la API DE REST. Sin embargo, para implementaciones de Kubernetes, esperamos que se creen al registrar nuevos objetos de Kubernetes `StorageClass`.

Objetos back-end de Astra Trident

Los back-ends representan a los proveedores de almacenamiento, además de los cuales Trident aprovisiona volúmenes; una única instancia de Trident puede gestionar cualquier número de back-ends.

Éste es uno de los dos tipos de objeto que se crean y administran a sí mismo. El otro es el objeto de Kubernetes `StorageClass`.

Para obtener más información sobre cómo construir estos objetos, consulte "configuración de los back-ends".

Objetos de Astra Trident `StoragePool`

Los pools de almacenamiento representan las distintas ubicaciones disponibles para aprovisionar en cada back-end. Para ONTAP, corresponden a los agregados en las SVM. Para HCI/SolidFire de NetApp, corresponden a las bandas de calidad de servicio especificadas por el administrador. Para Cloud Volumes Service, se corresponden con las regiones de proveedores de cloud. Cada pool de almacenamiento tiene un conjunto de atributos de almacenamiento distintos que definen sus características de rendimiento y sus características de protección de datos.

Al contrario de lo que ocurre con otros objetos aquí, los candidatos de pools de almacenamiento siempre se detectan y gestionan automáticamente.

Objetos de Astra Trident `Volume`

Los volúmenes son la unidad básica de aprovisionamiento y constan de extremos back-end, como recursos compartidos de NFS y LUN iSCSI. En Kubernetes, estos corresponden directamente a `PersistentVolumes`. Cuando crea un volumen, asegúrese de que tiene una clase de almacenamiento, que determina dónde se puede aprovisionar ese volumen junto con un tamaño.

- En Kubernetes, estos objetos se gestionan automáticamente. Es posible verlos para ver qué ha aprovisionado Trident.
- Al eliminar un VP con instantáneas asociadas, el volumen Trident correspondiente se actualiza a un estado **Eliminación**. Para que se elimine el volumen de Trident, es necesario quitar las snapshots del volumen.

Una configuración de volumen define las propiedades que debe tener un volumen aprovisionado.

Atributo	Tipo	Obligatorio	Descripción
versión	cadena	no	Versión de la API de Trident ("1")
nombre	cadena	sí	Nombre del volumen que se va a crear
Clase de almacenamiento	cadena	sí	Clase de almacenamiento que se utilizará al aprovisionar el volumen
tamaño	cadena	sí	El tamaño del volumen que se va a aprovisionar en bytes
protocolo	cadena	no	Tipo de protocolo que se va a utilizar; "archivo" o "bloque"
InternalName	cadena	no	Nombre del objeto en el sistema de almacenamiento, generado por Trident
ClonSourceVolume	cadena	no	ONTAP (nas, san) y SolidFire-*: Nombre del volumen desde el que se va a clonar
SplitOnClone	cadena	no	ONTAP (nas, san): Divida el clon entre su primario
Política de copias Snapshot	cadena	no	ONTAP-*: Política de instantánea a utilizar
Reserva de copias Snapshot	cadena	no	ONTAP-*: Porcentaje del volumen reservado para instantáneas

Atributo	Tipo	Obligatorio	Descripción
Política de exportoPolicy	cadena	no	ontap-nas*: Política de exportación que se va a utilizar
Snapshot shotDirectory	bool	no	ontap-nas*: Si el directorio de instantáneas está visible
Permisos univalados	cadena	no	ontap-nas*: Permisos iniciales de UNIX
Tamaño del bloque	cadena	no	SolidFire-*: Tamaño de bloque/sector
Sistema de archivos	cadena	no	Tipo de sistema de archivos

Trident genera `internalName` al crear el volumen. Esto consta de dos pasos. En primer lugar, antepone el prefijo de almacenamiento (ya sea el predeterminado `trident` o el prefijo en la configuración de backend) al nombre del volumen, lo que da como resultado un nombre del formulario `<prefix>-<volume-name>`. A continuación, procede a desinfectar el nombre y a reemplazar los caracteres no permitidos en el backend. En el caso de los back-ends de ONTAP, reemplaza guiones con guiones bajos (por lo tanto, el nombre interno se convierte `<prefix>_<volume-name>`en). En los back-ends de Element, reemplaza guiones bajos por guiones.

Puedes utilizar las configuraciones de volúmenes para aprovisionar volúmenes directamente mediante la API de REST, pero en las implementaciones de Kubernetes esperamos que la mayoría de los usuarios usen el método Kubernetes estándar `PersistentVolumeClaim`. Trident crea este objeto de volumen automáticamente como parte del proceso de aprovisionamiento.

Objetos de Astra Trident `Snapshot`

Las Snapshot son una copia de un momento específico de los volúmenes, que se pueden usar para aprovisionar nuevos volúmenes o restaurar el estado. En Kubernetes, estos corresponden directamente `VolumeSnapshotContent` a objetos. Cada copia de Snapshot se asocia con un volumen, que es el origen de los datos de la copia de Snapshot.

Cada `Snapshot` objeto incluye las propiedades enumeradas a continuación:

Atributo	Tipo	Obligatorio	Descripción
versión	Cadena	Sí	Versión de la API de Trident ("1")
nombre	Cadena	Sí	Nombre del objeto Snapshot de Trident
InternalName	Cadena	Sí	Nombre del objeto Snapshot de Trident en el sistema de almacenamiento

Atributo	Tipo	Obligatorio	Descripción
Nombre de volumen	Cadena	Sí	Nombre del volumen persistente para el que se crea la snapshot
VolumeInternalName	Cadena	Sí	Nombre del objeto de volumen de Trident asociado en el sistema de almacenamiento

En Kubernetes, estos objetos se gestionan automáticamente. Es posible verlos para ver qué ha aprovisionado Trident.

Cuando se crea una solicitud de objetos de Kubernetes `VolumeSnapshot`, Trident funciona creando un objeto Snapshot en el sistema de almacenamiento de respaldo. Para `internalName` este objeto Snapshot se genera combinando el prefijo `snapshot-` con el `UID` `VolumeSnapshot` objeto (por ejemplo, `snapshot-e8d8a0ca-9826-11e9-9807-525400f3f660`). `volumeName` y `volumeInternalName` se completan mediante la obtención de los detalles del volumen de respaldo.

Objeto Astra Trident `ResourceQuota`

El desamonset de Trident consume una `system-node-critical` clase de prioridad, la clase de prioridad más alta disponible en Kubernetes, para garantizar que Astra Trident pueda identificar y limpiar volúmenes durante el apagado de nodo correcto y permitir que los pods de inicio de datos de Trident se adelanten a las cargas de trabajo con una prioridad más baja en los clústeres donde hay una alta presión de recursos.

Para lograrlo, Astra Trident emplea un `ResourceQuota` objeto para garantizar que se satisfaga una clase de prioridad «crítica sistema-nodo» en el inicio de datos de Trident. Antes de la implementación y la creación de desastres, Astra Trident busca `ResourceQuota` el objeto y, si no se detecta, lo aplica.

Si necesita más control sobre la cuota de recursos predeterminada y la clase de prioridad, puede generar `custom.yaml` o configurar el `ResourceQuota` objeto mediante el gráfico Helm.

A continuación se muestra un ejemplo de un objeto "ResourceQuota'object que da prioridad al demonset de Trident.

```
apiVersion: <version>
kind: ResourceQuota
metadata:
  name: trident-csi
  labels:
    app: node.csi.trident.netapp.io
spec:
  scopeSelector:
     matchExpressions:
       - operator : In
         scopeName: PriorityClass
         values: ["system-node-critical"]
```

Para obtener más información sobre las cuotas de recursos, consulte "Kubernetes: Cuotas de recursos".

Limpie `ResourceQuota` si la instalación falla

En el caso raro de que la instalación falle después de `ResourceQuota` crear el objeto, primero intente "desinstalando" y luego vuelva a instalarlo.

Si eso no funciona, elimine manualmente el `ResourceQuota` objeto.

Quitar `ResourceQuota`

Si prefieres controlar tu propia asignación de recursos, puedes eliminar el objeto Astra Trident `ResourceQuota` mediante el comando:

```
kubectl delete quota trident-csi -n trident
```

Pod Security Standards (PSS) y las restricciones de contexto de seguridad (SCC)

Los estándares de seguridad de Kubernetes Pod (PSS) y las políticas de seguridad de Pod (PSP) definen los niveles de permisos y restringen el comportamiento de los POD. OpenShift Security Context restriction (SCC) define de forma similar la restricción de POD específica para OpenShift Kubernetes Engine. Para proporcionar esta personalización, 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 montajes iSCSI y utiliza redes de host para comunicarse con el daemon iSCSI.
IPC del host	NFS utiliza la comunicación entre procesos (IPC) para comunicarse con NFSD.

Permiso	Descripción
PID del host	Necesario para iniciar `rpc-statd` para NFS. Astra Trident consulta a los procesos del host para determinar `rpc-statd` si se están ejecutando antes de montar los volúmenes de NFS.

Funcionalidades	La `SYS_ADMIN` capacidad se proporciona como parte de las capacidades predeterminadas para contenedores con privilegios. Por ejemplo, Docker establece estas capacidades para contenedores con privilegios:
	`CapPrm: 0000003fffffffff`
	`CapEff: 0000003fffffffff`

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 el `spc_t` dominio («Super Privileged Container») y los contenedores sin privilegios se ejecutan en el `container_t` dominio. Activado `containerd`, con `container-selinux` instalado, todos los contenedores se ejecutan en el `spc_t` dominio, lo que desactiva efectivamente SELinux. Por 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 se ha cambiado la etiqueta de espacio de nombres para `privileged`. En ese caso, vuelva a instalar Trident.

Directivas de seguridad de POD (PSP)

Campo	Descripción	Predeterminado
`allowPrivilegeEscalation`	Los contenedores con privilegios deben permitir la escala de privilegios.	`true`
`allowedCSIDrivers`	Trident no utiliza volúmenes efímeros de CSI en línea.	Vacío
`allowedCapabilities`	Los contenedores Trident no con privilegios no requieren más funcionalidades de las que se establece de forma predeterminada y se conceden todas las funcionalidades posibles a los contenedores con privilegios.	Vacío
`allowedFlexVolumes`	Trident no utiliza un "Controlador FlexVolume", por lo tanto, no se incluyen en la lista de volúmenes permitidos.	Vacío
`allowedHostPaths`	El pod del nodo Trident monta el sistema de archivos raíz del nodo, por lo que no hay ninguna ventaja para configurar esta lista.	Vacío
`allowedProcMountTypes`	Trident no utiliza ninguna `ProcMountTypes`.	Vacío
`allowedUnsafeSysctls`	Trident no requiere ningún tipo inseguro `sysctls`.	Vacío
`defaultAddCapabilities`	No es necesario añadir capacidades a contenedores con privilegios.	Vacío
`defaultAllowPrivilegeEscalation`	En cada POD de Trident, se permite el escalado de privilegios.	`false`
`forbiddenSysctls`	No `sysctls` se permiten.	Vacío
`fsGroup`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`hostIPC`	El montaje de los volúmenes de NFS requiere que el IPC del host se comunique con el `nfsd`	`true`
`hostNetwork`	Iscsiadm requiere que la red del host se comunique con el demonio iSCSI.	`true`
`hostPID`	El PID del host es necesario para comprobar si `rpc-statd` se está ejecutando en el nodo.	`true`
`hostPorts`	Trident no utiliza puertos de host.	Vacío

Campo	Descripción	Predeterminado
`privileged`	Los pods de nodo Trident deben ejecutar un contenedor privilegiado para montar volúmenes.	`true`
`readOnlyRootFilesystem`	Los contenedores de nodos Trident deben escribir en el sistema de archivos del nodo.	`false`
`requiredDropCapabilities`	Los pods de nodo de Trident ejecutan un contenedor privilegiado y no pueden soltar las funcionalidades.	`none`
`runAsGroup`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`runAsUser`	Los contenedores Trident se ejecutan como raíz.	`runAsAny`
`runtimeClass`	Trident no utiliza `RuntimeClasses`.	Vacío
`seLinux`	Trident no se establece `seLinuxOptions` porque actualmente hay diferencias en cómo los tiempos de ejecución de los contenedores y las distribuciones de Kubernetes gestionan SELinux.	Vacío
`supplementalGroups`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`volumes`	Los pods de Trident requieren estos complementos de volumen.	`hostPath, projected, emptyDir`

Restricciones de contexto de seguridad (SCC)

Etiquetas	Descripción	Predeterminado
`allowHostDirVolumePlugin`	Los contenedores de nodos Trident montan el sistema de archivos raíz del nodo.	`true`
`allowHostIPC`	El montaje de volúmenes NFS requiere que el IPC del host se comunique con `nfsd`el .	`true`
`allowHostNetwork`	Iscsiadm requiere que la red del host se comunique con el demonio iSCSI.	`true`
`allowHostPID`	El PID del host es necesario para comprobar si `rpc-statd` se está ejecutando en el nodo.	`true`
`allowHostPorts`	Trident no utiliza puertos de host.	`false`

Etiquetas	Descripción	Predeterminado
`allowPrivilegeEscalation`	Los contenedores con privilegios deben permitir la escala de privilegios.	`true`
`allowPrivilegedContainer`	Los pods de nodo Trident deben ejecutar un contenedor privilegiado para montar volúmenes.	`true`
`allowedUnsafeSysctls`	Trident no requiere ningún tipo inseguro `sysctls`.	`none`
`allowedCapabilities`	Los contenedores Trident no con privilegios no requieren más funcionalidades de las que se establece de forma predeterminada y se conceden todas las funcionalidades posibles a los contenedores con privilegios.	Vacío
`defaultAddCapabilities`	No es necesario añadir capacidades a contenedores con privilegios.	Vacío
`fsGroup`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`groups`	Este SCC es específico de Trident y está vinculado a su usuario.	Vacío
`readOnlyRootFilesystem`	Los contenedores de nodos Trident deben escribir en el sistema de archivos del nodo.	`false`
`requiredDropCapabilities`	Los pods de nodo de Trident ejecutan un contenedor privilegiado y no pueden soltar las funcionalidades.	`none`
`runAsUser`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`seLinuxContext`	Trident no se establece `seLinuxOptions` porque actualmente hay diferencias en cómo los tiempos de ejecución de los contenedores y las distribuciones de Kubernetes gestionan SELinux.	Vacío
`seccompProfiles`	Los contenedores privilegiados siempre funcionan "sin confinar".	Vacío
`supplementalGroups`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`

Etiquetas	Descripción	Predeterminado
`users`	Se proporciona una entrada para vincular este SCC al usuario Trident en el espacio de nombres Trident.	n.a.
`volumes`	Los pods de Trident requieren estos complementos de volumen.	`hostPath, downwardAPI, projected, emptyDir`

Avisos legales

Los avisos legales proporcionan acceso a las declaraciones de copyright, marcas comerciales, patentes y mucho más.

Copyright

"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 los derechos de autor y las licencias de terceros que se utilizan en el software NetApp para Astra Trident en el archivo de avisos de cada versión en https://github.com/NetApp/trident/.