Documentación de Trident 26.06

Trident

NetApp
July 01, 2026

This PDF was generated from https://docs.netapp.com/es-es/trident/index.html on July 01, 2026. Always check docs.netapp.com for the latest.

Tabla de contenidos

Documentación de Trident 26.06 . . . 1
Notas de la versión . . . 2
Lo nuevo . . . 2
Novedades en 26.06 . . . 2
Novedades de 26.02.1 . . . 5
Novedades en 26.02 . . . 6
Novedades en 25.10 . . . 9
Cambios en 25.06.2 . . . 11
Cambios en 25.06.1 . . . 11
Cambios en 25.06 . . . 12
Cambios en 25.02.1 . . . 14
Cambios en 25,02 . . . 14
Cambios en 24.10.1 . . . 16
Cambios en 24,10 . . . 16
Cambios en 24,06 . . . 18
Cambios en 24,02 . . . 19
Cambios en 23,10 . . . 19
Cambios en 23.07.1 . . . 20
Cambios en 23,07 . . . 20
Cambios en 23,04 . . . 21
Cambios en 23.01.1 . . . 22
Cambios en 23.01 . . . 23
Cambios en 22.10 . . . 24
Cambios en 22.07 . . . 25
Cambios en 22.04 . . . 26
Cambios en 22.01.1 . . . 26
Cambios en 22.01.0 . . . 27
Cambios en 21.10.1 . . . 27
Cambios en 21.10.0 . . . 28
Problemas conocidos . . . 29
Obtenga más información . . . 30
Versiones anteriores de la documentación . . . 30
NetApp soporte Trident para sistemas de almacenamiento ONTAP ASA r2 . . . 30
Operaciones compatibles . . . 31
Operaciones no admitidas . . . 31
Problemas conocidos . . . 31
VolumeSnapshots no alcanza el estado ReadyToUse . . . 31
La restauración de copias de seguridad de archivos grandes de Restic puede fallar . . . 32
Manos a la obra . . . 33
Obtenga más información sobre Trident . . . 33
Obtenga más información sobre Trident . . . 33
Arquitectura de Trident . . . 34
Conceptos . . . 37

Inicio rápido para Trident . . . 41
El futuro . . . 42
Requisitos . . . 43
Información crítica sobre Trident . . . 43
Front-ends compatibles (orquestadores) . . . 43
Back-ends compatibles (almacenamiento) . . . 44
Compatibilidad de Trident con KubeVirt y OpenShift Virtualization . . . 44
Compatibilidad de Trident con Kubernetes en Kubernetes y con los planos de control alojados en OpenShift . . . 44
Requisitos de funciones . . . 45
Se probaron sistemas operativos host . . . 46
Configuración de hosts . . . 46
Configuración del sistema de almacenamiento . . . 46
Puertos Trident . . . 47
Imágenes de contenedor y las versiones de Kubernetes correspondientes . . . 47
Instale Trident . . . 48
Obtenga información acerca de la instalación de Trident . . . 48
Información crítica sobre Trident 26.02 . . . 48
Antes de empezar . . . 48
Elija el método de instalación . . . 48
Elija el modo de instalación . . . 50
Seleccione el proceso según el método y el modo . . . 51
Moverse entre los métodos de instalación . . . 52
Otras opciones de configuración conocidas . . . 52
Realice la instalación mediante el operador Trident . . . 52
Implemente manualmente el operador de Trident (modo estándar) . . . 52
Implemente manualmente el operador Trident (modo sin conexión). . . . 58
Puesta en marcha del operador de Trident con Helm (modo estándar) . . . 64
Implementar el operador de Trident con Helm (modo sin conexión) . . . 72
Personalice la instalación del operador de Trident . . . 80
Instale utilizando trimentctl . . . 92
Instale utilizando trimentctl . . . 92
Personalice la instalación trimentctl . . . 96
Realice la instalación mediante el operador certificado de OpenShift . . . 97
Instale Trident mediante OpenShift OperatorHub . . . 97
Cambia al operador Trident certificado OpenShift . . . 100
Utilizar Trident . . . 104
Prepare el nodo de trabajo . . . 104
Seleccionar las herramientas adecuadas . . . 104
Detección del servicio de nodos . . . 104
Volúmenes de NFS . . . 105
Volúmenes iSCSI . . . 105
Volúmenes NVMe/TCP . . . 109
Volúmenes SCSI sobre FC . . . 110
Prepárese para aprovisionar los volúmenes de SMB . . . 113

Configurar y gestionar back-ends . . . 114
Configurar los back-ends . . . 114
Azure NetApp Files . . . 115
NetApp Volumes para Google Cloud . . . 135
Configure un back-end de NetApp HCI o SolidFire . . . 168
Unidades SAN de ONTAP . . . 173
Unidades NAS de ONTAP . . . 203
Amazon FSX para ONTAP de NetApp . . . 242
Cree back-ends con kubectl . . . 279
Gestionar back-ends . . . 286
Crear y gestionar clases de almacenamiento . . . 296
Cree una clase de almacenamiento . . . 296
Gestione las clases de almacenamiento . . . 299
Aprovisione y gestione volúmenes . . . 301
Aprovisione un volumen . . . 301
Mover un volumen . . . 305
Expanda los volúmenes . . . 308
Entiende los límites del subsistema RWX NVMe . . . 320
Escalabilidad del controlador . . . 322
Ampliación automática de volúmenes . . . 327
Gestiona las políticas de Autogrow . . . 334
Importar volúmenes . . . 343
Personalizar nombres y etiquetas de volúmenes . . . 356
Comparta un volumen NFS en espacios de nombres . . . 359
Clone volúmenes en espacios de nombres . . . 363
Replicar volúmenes mediante SnapMirror . . . 366
Utilice Topología CSI . . . 372
Trabajar con instantáneas . . . 380
Trabajar con instantáneas de grupos de volúmenes . . . 388
Gestione y supervise Trident . . . 393
Actualice Trident . . . 393
Actualice Trident . . . 393
Actualizar con el operador . . . 394
Actualice con trimentctl . . . 399
Gestione Trident con tridentctl . . . 400
Comandos e indicadores globales . . . 400
Opciones de comando y indicadores . . . 402
Compatibilidad con complementos . . . 408
Supervisar Trident . . . 408
Descripción general . . . 408
Paso 1: Definir un objetivo Prometheus . . . 408
Paso 2: Cree un Prometheus ServiceMonitor . . . 409
Paso 3: Consulte las métricas de Trident con PromQL . . . 411
Más información sobre la telemetría de Trident AutoSupport . . . 412
Deshabilitar las métricas de Trident . . . 413

Desinstale Trident . . . 413
Determine el método de instalación original . . . 413
Desinstale una instalación del operador Trident . . . 413
Desinstale un `tridentctl` instalación . . . 414
Trident para Docker . . . 415
Requisitos previos para la implementación . . . 415
Compruebe los requisitos . . . 415
Herramientas de NVMe . . . 417
Herramientas de FC . . . 418
Ponga en marcha Trident . . . 420
Método de complemento gestionado por Docker (versión 1.13/17.03 y posteriores) . . . 420
Método tradicional (versión 1.12 o anterior) . . . 422
Inicie Trident al iniciar el sistema . . . 423
Actualice o desinstale Trident . . . 424
Renovar . . . 425
Desinstalar . . . 426
Trabaje con volúmenes . . . 426
Cree un volumen . . . 426
Quitar un volumen . . . 427
Clonar un volumen . . . 427
Acceso a volúmenes creados externamente . . . 429
Opciones de volumen específicas del controlador . . . 429
Recopilar registros . . . 434
Recopile registros para solucionar problemas . . . 434
Sugerencias generales para la solución de problemas . . . 435
Gestione múltiples instancias de Trident . . . 435
Pasos para el complemento gestionado de Docker (versión 1.13/17.03 o posterior) . . . 435
Pasos para tradicional (versión 1.12 o anterior) . . . 436
Opciones de configuración de almacenamiento . . . 436
Opciones de configuración global . . . 437
Configuración de ONTAP . . . 437
Configuración del software Element . . . 446
Problemas y limitaciones conocidos . . . 448
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. . . . 448
Los nombres de volumen deben tener una longitud mínima de 2 caracteres. . . . 449
Docker Swarm tiene ciertos comportamientos que impiden que Trident ofrezca compatibilidad con cada combinación de almacenamiento y controladores. . . . 449
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. . . . 449
Prácticas recomendadas y recomendaciones . . . 450
Puesta en marcha . . . 450
Póngalo en marcha a un espacio de nombres dedicado . . . 450
Utilice cuotas y límites de rango para controlar el consumo de almacenamiento . . . 450
Configuración del almacenamiento . . . 450

Descripción general de la plataforma . . . 450
Prácticas recomendadas para ONTAP y Cloud Volumes ONTAP . . . 450
Mejores prácticas para SolidFire . . . 455
¿Dónde encontrar más información? . . . 457
Integre Trident . . . 458
Selección y despliegue del conductor . . . 458
Diseño de clase de almacenamiento . . . 460
Diseño de pool virtual . . . 462
Operaciones de volumen . . . 463
Servicio de métricas . . . 466
Protección de datos y recuperación ante desastres . . . 467
Replicación y recuperación de Trident . . . 468
Replicación y recuperación de SVM . . . 468
Replicación y recuperación de volúmenes . . . 469
Protección de datos Snapshot . . . 470
Automatización de la conmutación por error de aplicaciones con estado con Trident . . . 470
Detalles acerca de forzar separación . . . 470
Detalles sobre la conmutación por error automatizada . . . 471
Seguridad . . . 476
Seguridad . . . 476
Configuración de clave unificada de Linux (LUKS) . . . 477
Cifrado en tránsito de Kerberos . . . 484
Proteja las aplicaciones con Trident Protect . . . 492
Obtenga más información sobre Trident Protect . . . 492
El futuro . . . 492
Instalar Trident Protect . . . 492
Requisitos de Trident Protect . . . 492
Instalar y configurar Trident Protect . . . 496
Instalar el complemento CLI de Trident Protect . . . 500
Personalizar la instalación de Trident Protect . . . 504
Administrar Trident Protect . . . 509
Administrar la autorización y el control de acceso de Trident Protect . . . 509
Supervisar los recursos de Trident Protect . . . 516
Generar un paquete de soporte de Trident Protect . . . 521
Actualizar Trident Protect . . . 523
Gestione y proteja aplicaciones . . . 525
Utilice objetos Trident Protect AppVault para administrar depósitos . . . 525
Definir una aplicación para la gestión con Trident Protect . . . 539
Proteja las aplicaciones con Trident Protect . . . 544
Restaure las aplicaciones . . . 557
Replicar aplicaciones utilizando NetApp SnapMirror y Trident Protect . . . 582
Migrar aplicaciones usando Trident Protect . . . 600
Protege las máquinas virtuales de KubeVirt con Trident Protect . . . 604
Administrar los ganchos de ejecución de Trident Protect . . . 616
Desinstalar Trident Protect . . . 628

Blogs de Trident y Trident Protect . . . 629
Blogs de Trident . . . 629
Blogs de Trident Protect . . . 629
Conocimiento y apoyo . . . 631
Preguntas frecuentes . . . 631
Preguntas generales . . . 631
Instale y utilice Trident en un clúster de Kubernetes . . . 631
Solución de problemas y soporte técnico . . . 632
Actualice Trident . . . 633
Gestione back-ends y volúmenes . . . 634
Resolución de problemas . . . 638
Resolución de problemas generales . . . 638
Implementación incorrecta de Trident con el operador . . . 640
Puesta en marcha de Trident incorrecta mediante `tridentctl` . . . 642
Eliminar completamente Trident y CRD . . . 642
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 . . . 643
Los clientes NFSv4.2 informan "argumento no válido" después de actualizar ONTAP cuando esperaban que "v4.2-xattrs" estuviera habilitado . . . 644
Soporte técnico . . . 644
Ciclo de vida del soporte de Trident . . . 644
Autosoporte . . . 645
Soporte de la comunidad . . . 645
Soporte técnico de NetApp . . . 645
Si quiere más información . . . 645
Referencia . . . 646
Puertos Trident . . . 646
Descripción general . . . 646
API de REST de Trident . . . 648
Cuándo utilizar la API DE REST . . . 648
Uso de la API DE REST . . . 648
Opciones de línea de comandos . . . 649
Registro . . . 649
Kubernetes . . . 649
Docker . . . 650
DESCANSO . . . 650
Objetos de Kubernetes y Trident . . . 650
¿Cómo interactúan los objetos entre sí? . . . 650
Kubernetes `PersistentVolumeClaim` objetos . . . 651
Kubernetes `PersistentVolume` objetos . . . 653
Kubernetes `StorageClass` objetos . . . 653
Kubernetes `VolumeSnapshotClass` objetos . . . 657
Kubernetes `VolumeSnapshot` objetos . . . 658
Kubernetes `VolumeSnapshotContent` objetos . . . 658

`VolumeGroupSnapshotClass`Objetos de Kubernetes . . . 658
`VolumeGroupSnapshot`Objetos de Kubernetes . . . 659
`VolumeGroupSnapshotContent`Objetos de Kubernetes . . . 659
Kubernetes `CustomResourceDefinition` objetos . . . 660
Objetos deTrident `StorageClass` . . . 660
Objetos de back-end de Trident . . . 660
Objetos deTrident `StoragePool` . . . 661
Objetos deTrident `Volume` . . . 661
Objetos deTrident `Snapshot` . . . 662
ObjetoTrident `ResourceQuota` . . . 663
Pod Security Standards (PSS) y las restricciones de contexto de seguridad (SCC) . . . 664
Contexto de Kubernetes Security y campos relacionados necesarios . . . 665
Estándares de seguridad para POD (PSS) . . . 665
Directivas de seguridad de POD (PSP) . . . 666
Restricciones de contexto de seguridad (SCC) . . . 668
Avisos legales . . . 670
Derechos de autor . . . 670
Marcas comerciales . . . 670
Estadounidenses . . . 670
Política de privacidad . . . 670
Código abierto . . . 670

Documentación de Trident 26.06

Notas de la versión

Lo nuevo

Las notas de la versión proporcionan información sobre nuevas características, mejoras y correcciones de errores en la última versión de NetApp Trident.

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

Novedades en 26.06

Descubre qué hay de nuevo en NetApp Trident y Trident Protect, incluyendo mejoras, correcciones y elementos obsoletos.

Trident

Mejoras

- **Kubernetes**:
  - Se añadió compatibilidad con la disponibilidad general (GA) para la concurrencia en los controladores ONTAP-NAS-Economy y ONTAP-SAN-Economy, graduando esta funcionalidad desde Tech Preview, y se añadió compatibilidad GA para la concurrencia en los controladores de Azure NetApp Files (ANF) y SolidFire.
  - Se añadió soporte para Kubernetes 1.36.
  - Se añadió compatibilidad con la disponibilidad general (GA) para el crecimiento automático de volúmenes con los controladores NAS y SAN de Google Cloud NetApp Volumes (GCNV).
  - Se ha añadido la función de distribución equilibrada de Unified ONTAP para una selección óptima de los agregados al aprovisionar nuevos volúmenes ("Problema #64").
  - Se añadió compatibilidad con ONTAP AFX para el controlador ONTAP-NAS-Flexgroup.
  - Se añadió la integración con el NetApp Shift Toolkit para simplificar las migraciones de máquinas virtuales cuando usas el Red Hat OpenShift Migration Toolkit for Virtualization.
  - Se añadió compatibilidad con Cloud Identity para el controlador SAN GCNV.
  - Se añadió compatibilidad con volúmenes subordinados para los flujos de trabajo de propagación y crecimiento automático de volúmenes de Trident Volume Publication (TVP).
  - Se añadió compatibilidad con las actualizaciones de volúmenes y las instantáneas de grupo de volúmenes al ejecutar Trident con la concurrencia activada.
  - Se añadió la limitación dinámica de tasa de gRPC para GCNV.
  - Se añadió una prueba de ejecución de disponibilidad del controlador y se ajustaron los valores predeterminados de QPS y ráfagas de la API de Kubernetes para el controlador y los sidecars.
  - Se mejoró la secuencia de operaciones de los nodos al desacoplar los pasos de conexión, configuración del cifrado y formateo del sistema de archivos.
  - Se montaron los sistemas de archivos raíz de los contenedores Trident como solo lectura para mejorar la seguridad.

- Se ha añadido la eliminación de instantáneas best-effort para las instantáneas de volúmenes clonados en los backends de ONTAP cuando se elimina la instantánea de origen de un volumen clonado.
- Se ha añadido el reconocimiento de clones de PVC de copia de seguridad para evitar que los PVC temporales creados por Veeam Kasten, Trilio y Cohesity queden atascados en la cola de recuperación de volúmenes de ONTAP.
- Mejora de la gestión de ARN de AWS para admitir regiones especiales.
- Evitar que los FlexVol se reduzcan durante la operación de cambio de tamaño para el controlador ONTAP-SAN-ECONOMY.

Mejoras experimentales

> No apto para uso en entornos de producción.

- **[Vista previa técnica]**: Se añadió compatibilidad con la importación de qtree de ONTAP-NAS-Economy.
- **[Vista previa técnica]**: Se añadió compatibilidad con la concurrencia para los controladores ASA-R2 (SAN y NVMe).
- **[Vista previa técnica]**: Se añadió el transporte HTTP en modo ONTAP de GCNV para los controladores ONTAP-SAN y ONTAP-NAS.

Soluciones

- **Kubernetes**:
  - Se corrigieron las condiciones de carrera y la gestión de estados para las operaciones simultáneas de publicación/retirada, clonación, almacenamiento en caché y actualización del backend.
  - Se corrigieron las condiciones de carrera en la política de exportación y el manejo concurrente de publicar/retirar la publicación para volúmenes subordinados y clones de solo lectura en los controladores ONTAP-NAS y ONTAP-NAS-Economy.
  - Se corrigió la detección de ONTAP-NAS-Economy FlexVol después de una conmutación por error de MetroCluster cuando las políticas de snapshot difieren ("Problema #1082").
  - Se corrigieron los reintentos de eliminación de volúmenes en ONTAP-NAS-Economy que interrumpían las eliminaciones de qtree de larga duración ("Problema #1121").
  - Se corrigió el comportamiento de importación y cambio de tamaño en ONTAP-SAN y ONTAP-SAN-Economy, incluyendo la validación de fsType, la gestión de metadatos de volúmenes y el comportamiento del modo de crecimiento automático de volúmenes.
  - Se corrigió una condición de carrera en el espacio de nombres NVMe/TCP durante la creación simultánea de volúmenes ("Problema #1089").
  - Se corrigieron los errores de redimensionamiento de iSCSI que ocurrían cuando las rutas y los portales disponibles no coincidían, y se mejoró el comportamiento al realizar redimensionamientos repetidos.
  - Se corrigieron los errores al eliminar volúmenes ONTAP-SAN causados por condiciones de volumen ocupado.
  - Se corrigió la gestión de errores REST de ONTAP y se mejoró el comportamiento del tiempo de espera del estado de SVM.
  - Se ha corregido la gestión de los grupos zonales de SAN y los grupos de hosts de GCNV, la conversión de identificadores de volumen normalizados y las solicitudes de creación normalizadas por debajo del tamaño mínimo.

- Se corrigió la selección del grupo de capacidad nativo de GCNV para excluir los grupos en modo ONTAP.
- Se corrigió el failover de KubeVirt para las máquinas virtuales que usan Trident Automated Workload Failover y así acelerar el failover de las VMs.
- Se solucionaron problemas de RBAC en instalaciones gestionadas por el operador, incluyendo la falta de ClusterRoles en los nodos y la conciliación incorrecta de ClusterRole y ClusterRoleBinding durante las actualizaciones. Esto aplica a instalaciones basadas en Operator y Helm que usan el chart de Trident Operator.
- Se corrigieron las diferencias persistentes de GitOps en TridentConfigurator y los CRD relacionados, omitiendo los valores de prioridad predeterminados y corrigiendo la sangría de la plantilla YAML ("Problema #915", "Problema #1120").
- Se corrigió el parcheo de los CRD de TridentVolumePublication durante una actualización de Helm ("Problema #1151").
- Se corrigió el orden de inicio de CSI iniciando el servicio gRPC de CSI antes de los reintentos de registro de nodos.
- Se corrigieron las superposiciones de configuración de clones SMB seguros durante operaciones simultáneas.
- Se corrigieron las anotaciones que faltaban en el pod del controlador Trident después de una actualización ("Problema #1004").
- Mejorada la provisión lenta de qtree en SVM grandes ("Problema #1142").
- Se corrigió ControllerPublish para que use el tipo de sistema de archivos de la configuración del volumen en lugar de un tipo de sistema de archivos predeterminado incorrecto.
- Se solucionó un problema donde los dispositivos dm huérfanos podían causar fallos repetidos de CSI NodeUnstageVolume.
- Mejorada la limpieza de nodos obsoletos y de multivía al eliminar artefactos fantasma de multivía y recursos obsoletos de TridentNode después de limpiar nodos eliminados de forma temporal.

Trident Protect

Mejoras

- Se añadió la integración de OpenShift Service CA Operator para automatizar la gestión de certificados TLS para el webhook de admisión de Trident Protect. Actívalo configurando `useOpenshiftServiceCA` en `true` en tus valores de Helm. Para saber más, consulta "Personalizar la instalación de Trident Protect".
- Se añadió compatibilidad para hacer copias de seguridad y restaurar máquinas virtuales de KubeVirt en OpenShift Virtualization. Protege todas las máquinas virtuales en un espacio de nombres o selecciona máquinas virtuales específicas por nombre, y restaura de forma selectiva máquinas virtuales individuales desde una copia de seguridad o una instantánea. Para saber más, consulta "Protege las máquinas virtuales de KubeVirt con Trident Protect".
- Se ha añadido compatibilidad con las transformaciones de recursos, lo que te permite añadir, eliminar y modificar los atributos de los recursos de Kubernetes durante el proceso de restauración. Para obtener más información, consulta "Modifica los recursos a medida que los restauras usando recursos personalizados".
- Se añadió compatibilidad con `storageClassMapping` en AppMirrorRelationship (AMR), lo que permite la asignación de clases de almacenamiento para PVC individuales dentro de una única definición de AMR. Para obtener más información, consulta "Replicar aplicaciones utilizando NetApp SnapMirror y Trident Protect".

Soluciones

- Se solucionó el problema que hacía que las restauraciones de máquinas virtuales entre clústeres fallaran con un error de webhook de admisión cuando el espacio de nombres de la aplicación de origen no existía en el clúster de destino.
- Se ha solucionado el error que provocaba que las restauraciones selectivas sin movimiento de máquinas virtuales fallaran en el paso de creación de la aplicación debido a una discrepancia en las especificaciones con un recurso personalizado de una aplicación ya existente; ahora se reutiliza la aplicación existente.
- Se ha corregido el problema por el que las tareas para restaurar sin movimiento las máquinas virtuales se quedaban bloqueadas en un estado `Running` cuando se eliminaban tanto la máquina virtual como su PersistentVolumeClaim; ahora, la ausencia de PersistentVolumeClaim se considera un error grave.
- Se corrigió un problema donde al editar una programación de protección existente se activaba incorrectamente una copia de seguridad o una instantánea de inmediato.
- Se corrigió el problema por el que el finalizador de instantáneas intentaba obtener el recurso personalizado de la aplicación después de que se eliminaba; ahora el finalizador omite este paso cuando no encuentra la aplicación.
- Se ha corregido el aumento significativo de la capacidad de almacenamiento protegida tras restaurar sin movimiento una máquina virtual mediante el uso de una copia dispersa durante el paso de copia PersistentVolumeClaim.
- Se corrigió un error en Trident Protect que eliminaba el objeto de instantánea de Kubernetes antes de confirmar la eliminación de la instantánea subyacente en ONTAP; ahora el finalizador espera la eliminación de `VolumeSnapshot`.
- Se ha corregido el problema por el que las instantáneas no se realizaban cuando fallaba la operación de congelación de una máquina virtual; ahora, las operaciones de congelación y descongelación continúan aunque se produzca un error.
- Se corrigió un problema por el que las aplicaciones de máquina virtual KubeVirt recreaban incorrectamente PersistentVolumeClaims durante las operaciones de reversa, resinc inversa y resinc de AppMirrorRelationship.

Novedades de 26.02.1

Descubre qué hay de nuevo en NetApp Trident y Trident Protect, incluyendo mejoras, correcciones y elementos obsoletos.

Trident

Soluciones

- **Kubernetes**:
- Se ha corregido el comportamiento de SAN de Google Cloud NetApp Volumes (GCNV) para admitir grupos Flex Unified ZONAL para cargas de trabajo de bloque (iSCSI). Ahora se admiten tanto las configuraciones zonales como las regionales.
- Se ha corregido la gestión de grupos NAS UNIFIED de Google Cloud NetApp Volumes (GCNV) mediante la mejora de la creación de volúmenes, validación, nomenclatura y operaciones de búsqueda de volúmenes de reserva cuando `InternalID` está ausente.
- Se ha solucionado un problema por el que las operaciones de cambio de tamaño de dispositivos iSCSI podían fallar cuando las rutas y los portales disponibles no coincidían.
- Se ha solucionado un problema intermitente por el que `blkid` identificaba incorrectamente el tipo de sistema de archivos para dispositivos LUKS, lo que impedía la creación de nuevos volúmenes.

- Se ha solucionado un problema por el que las operaciones de publicación y puesta en escena de LUKS a escala podían bloquear el controlador Trident. Ahora se espera que los usuarios realicen un seguimiento de las contraseñas LUKS asociadas a los volúmenes.
- Se solucionó un problema por el que las particiones iSCSI multivía podían aparecer como dispositivos fantasma, lo que impedía que las operaciones de CSI `NodeUnstage` tuvieran éxito.
- Se ha corregido una condición de carrera del espacio de nombres NVMe/TCP durante la creación simultánea de volúmenes (Issue #1089).
- Se han corregido las operaciones simultáneas de clonación de instantáneas cuando se ejecutaban en el mismo segundo.
- Se solucionó un problema que impedía que las máquinas virtuales de KubeVirt que utilizaban `dataVolumes` realizaran failover con Trident Automated Workload Failover.
- Se ha corregido un error que se producía al eliminar un recurso personalizado no válido `TridentMirrorRelationship` mientras se utilizaba el núcleo concurrente de Trident.
- Corrección de la sangría en la definición YAML de `tridentactionmirrorupdates` Kubernetes CRD (issue #1120).
- Se ha corregido la llamada a la API REST de eliminación asíncrona de qtree (Issue #1121).
- Corregido `ControllerPublish` para usar el tipo de sistema de archivos del archivo de configuración de volumen en vez de un tipo de sistema de archivos predeterminado incorrecto.
- Se ha corregido el registro de inicio del controlador para que informe correctamente el estado del controlador concurrente.
- Corregido el manejo de backends suspendidos en Trident concurrente.
- Mejora de la gestión de ARN de AWS para admitir regiones especiales.

Trident Protect

Soluciones

Actualizadas las dependencias de Trident Protect para resolver vulnerabilidades de seguridad:

- gRPC-Go: "CVE-2026-33186"
- Go x/crypto SSH: "CVE-2024-45337"
- OpenTelemetry-Go: "CVE-2026-24051"
- Runtime de Go: "CVE-2025-22871"
- Dependencias adicionales: "CVE-2025-23109"

Novedades en 26.02

Descubre qué hay de nuevo en NetApp Trident y Trident Protect, incluyendo mejoras, correcciones y elementos obsoletos.

Trident

Mejoras

- **Kubernetes**:
  - Se añadió compatibilidad de disponibilidad general (GA) para la concurrencia en los drivers de

ONTAP-NAS (solo NFS), ONTAP-SAN (iSCSI, FCP, NVMe) y Google Cloud NetApp Volumes (GCNV), pasando esta funcionalidad de Tech Preview a GA. Consulta la "Documentación sobre la escalabilidad de controladores" para más información.

- Se añadió soporte para el crecimiento automático de volúmenes en Trident basado en políticas personalizadas de Trident AutoGrow definidas por el usuario. Consulta "Documentación sobre la expansión automática de volúmenes" para más información.
- Aumentamos la concurrencia de nodos Trident para mejorar la escalabilidad de las operaciones de nodo para volúmenes NVMe. Consulta el "Documentación de volúmenes NVMe" para más información.
- Se añadió compatibilidad para el auto-tiering de volúmenes NAS de GCNV a través de anotaciones PVC (`tieringPolicy`, `tieringMinimumCoolingDays`), incluyendo la selección de pool y la herencia de clones. Consulta el "Configura el auto-tiering para Google Cloud NetApp Volumes documentation" para más información.
- Se añadió compatibilidad para volúmenes de bloque SAN GCNV (iSCSI) usando el `google-cloud-netapp-volumes-san` driver, incluyendo aprovisionamiento, mapeo de grupos de hosts por nodo para acceso a LUN y operaciones de clonación desde volumen. Consulta el "Documentación de configuración de bloques de Google Cloud NetApp Volumes" para más información.
- Se añadió soporte para la configuración automática del backend para Amazon FSx for NetApp ONTAP. Cuando creas un StorageClass que incluye los parámetros requeridos, Trident crea automáticamente el backend correspondiente y VolumeSnapshotClass, si es necesario. Consulta el "Documentación de configuración del backend de Amazon FSx for NetApp ONTAP" para más información.
- Se añadió compatibilidad con diferentes clouds de Microsoft Azure, como Azure Government y Azure China, y configuración de cloud personalizada para los backends de Azure NetApp Files. Consulta el "Documentación de configuración del backend de Azure NetApp Files" para más información.
- Se añadió soporte para Kubernetes 1.35. Consulta el "Documentación de requisitos" para más información.

Mejoras experimentales

No apto para uso en entornos de producción.

- **[Avance técnico]**: Se añadió soporte para concurrencia para los drivers ONTAP-NAS-Economy y ONTAP-SAN-Economy.

Problemas conocidos

Existe un problema conocido con el `csi-snapshotter` sidecar. En todas las versiones de Kubernetes, VolumeGroupSnapshots `v1beta1` impide que VolumeSnapshots llegue al estado `ReadyToUse`.

Hay dos soluciones alternativas:

1. Elimina los CRD de VolumeGroupSnapshots para desactivar VolumeGroupSnapshots y luego vuelve a instalar Trident.
2. Instala VolumeGroupSnapshots `v1beta2` y `snapshot-controller` versión 8.4.0 o posterior, y luego vuelve a instalar Trident. VolumeGroupSnapshots no funcionan en versiones de Kubernetes anteriores a v1.34.

Soluciones

- **Kubernetes**:
  - Se solucionó un problema donde al despublicar un clon de solo lectura se eliminaban las reglas de la política de exportación del volumen de origen en los drivers ONTAP-NAS, ONTAP-NAS-Economy y Google Cloud NetApp Volumes "Problema #1086".
  - Se cambiaron las imágenes de kubectl a variantes ligeras basadas en Alpine para evitar fallos de pull tras la deprecación pública de imágenes de Bitnami "Problema #1080".
  - Corregido el mantenimiento de la anotación de la implementación existente durante la actualización de Trident "Problema #1004".
  - Permite la clonación entre diferentes clases de almacenamiento si ambas clases de almacenamiento apuntan al mismo backend "Problema #1104".
  - Corregidos los fallos de preparación de nodos causados por tiempos de espera en entornos cloud con latencias de red. Aumentados los valores de tiempo de espera para instalaciones basadas en cloud.
  - Se corrigió un problema en la creación de LUN que hacía que el atributo de tipo de sistema de archivos quedara sin establecer cuando el proceso entraba en un estado de reintento.
  - Se corrigió la búsqueda de volúmenes en la API de REST para que ignore el estado del volumen, evitando falsos negativos durante las consultas de volumen.
  - Mejora de la eficiencia del controlador Trident para el driver ontap-nas-economy cuando se usa a escala.
  - Establece internalID durante la importación de LUN en el controlador ontap-san-economy.
  - Se aumentaron los límites de consulta de Azure Resource Graph para manejar más subredes.
  - Mejorados los tiempos de espera de división de clones de CSI y ONTAP para evitar condiciones de carrera con algunas aplicaciones de backup "Problema #1098", "Problema #1100".
  - Corregida la supresión de mensajes de error LUKS "Problema #1069".
  - Corregida la gestión de mapeadores LUKS obsoletos para los protocolos iSCSI y NVMe. La lógica de limpieza mejorada evita fallos de montaje por mapeadores de dispositivos huérfanos.
- Corregidas las limitaciones de escala para los volúmenes RWX NVMe.
- Se actualizó el paquete OpenTelemetry-Go para corregir "CVE-2026-24051".

Trident Protect

Mejoras

- Trident Protect ahora desactiva automáticamente los programas de protección y cancela las operaciones en curso antes de restaurar sin movimiento, y los vuelve a activar después de que se complete la restauración. Para saber más, consulta "Restaurar aplicaciones usando Trident Protect".
- Se añadió el campo `runImmediately` al schedule CR y la bandera `--run-immediately` de la CLI para activar una copia de seguridad o snapshot inmediata al crear la programación. Para saber más, consulta "Cree un programa de protección de datos".
- Se añadió compatibilidad para especificar un nombre personalizado para la aplicación restaurada usando el campo `destinationApplicationName` en el restore CR o la bandera `--destination-app-name` en la CLI. Para saber más, consulta "Restaurar aplicaciones usando Trident Protect".

Soluciones

- Se corrigieron los fallos de restauración causados por la creación de pods antes de que sus cuentas de servicio requeridas estuvieran disponibles.
- Corregido que `Roles` y `RoleBindings` se omitieran durante la restauración de aplicaciones.
- Se corrigió que el nombre del clúster de origen no se muestre en la salida de `tridentctl-protect get appvaultcontent` a pesar de estar correctamente configurado.
- Se corrigieron los errores de restauración de Kopia que se ignoraban por la falta de manejo de `pipefail`.
- Se corrigieron los fallos de instantáneas y copias de seguridad causados por filtros de recursos que excluían los volúmenes persistentes.
- Se corrigió la restauración incorrecta de PVC en aplicaciones de varios espacios de nombres con PVC con nombres idénticos en diferentes espacios de nombres, lo que podía provocar pérdida de datos.

Novedades en 25.10

Conozca las novedades de Trident y Trident Protect, incluidas mejoras, correcciones y obsolescencias.

Trident

Mejoras

- **Kubernetes**:
  - Se agregó compatibilidad con instantáneas de grupos de volúmenes CSI con las API de Kubernetes de instantáneas de grupos de volúmenes v1beta1 para los controladores ONTAP-NAS NFS y ONTAP-SAN-Economy, además de ONTAP-SAN (iSCSI y FC). Ver"Trabajar con instantáneas de grupos de volúmenes" .
  - Se agregó soporte para la conmutación por error automatizada de cargas de trabajo con desconexión forzada de volumen para ONTAP-NAS y ONTAP-NAS-Economy (excluyendo SMB en ambos controladores NAS), y los controladores ONTAP-SAN y ONTAP-SAN-Economy. Ver"Automatización de la conmutación por error de aplicaciones con estado con Trident" .
  - Concurrencia de nodos Trident mejorada para una mayor escalabilidad en las operaciones de nodos para volúmenes FCP.
  - Se ha añadido compatibilidad con ONTAP AFX para el controlador ONTAP NAS. Ver"Opciones y ejemplos de configuración NAS de ONTAP" .
  - Se ha añadido compatibilidad para configurar las solicitudes y los límites de recursos de CPU y memoria para los contenedores Trident mediante TridentOrchestrator CR y los valores del gráfico Helm. ("Número 1000" ,"Número 927" ,"Número 853" ,"Número 592" ,"Número 110").
  - Se ha añadido compatibilidad con FC para la personalidad ASAr2. Ver"Opciones y ejemplos de configuración DE SAN ONTAP" .
  - Se ha añadido una opción para servir las métricas de Prometheus con HTTPS, en lugar de HTTP. Ver"Supervisar Trident" .
  - Se agregó una opción `--no-rename` Al importar un volumen, se debe conservar el nombre original pero dejar que Trident gestione el ciclo de vida del volumen. Ver"Importar volúmenes" .
  - El despliegue de Trident ahora se ejecuta en la clase de prioridad crítica del sistema y del clúster.
- Se agregó una opción para que el controlador Trident utilice la red del host a través de helm, operator y tridentctl ("Número 858").

- Se agregó soporte manual de QoS al controlador ANF, haciéndolo listo para producción en Trident 25.10; esta mejora experimental se introdujo en Trident 25.06.

Mejoras experimentales

> No apto para uso en entornos de producción.

- **[Vista previa técnica]**: Se agregó compatibilidad con la concurrencia para ONTAP-NAS (solo NFS) y ONTAP-SAN (NVMe para ONTAP 9 unificado), además de la vista previa técnica existente para el controlador ONTAP-SAN (protocolos iSCSI y FCP en ONTAP 9 unificado).

Soluciones

- **Kubernetes**:
  - Se corrigió la inconsistencia en el nombre del contenedor node-driver-registrar de CSI estandarizando Linux DaemonSet a node-driver-registrar para que coincida con la nomenclatura de Windows DaemonSet y de las imágenes de contenedor.
  - Se solucionó un problema por el cual las políticas de exportación para qtrees heredados no se actualizaban correctamente.
- **Openshift**:
  - Se ha corregido un error que impedía el inicio del pod del nodo Trident en nodos Windows en Openshift debido a que SCC tenía allowHostDirVolumePlugin configurado como falso ("Número 950").
- Se ha corregido el problema que impedía configurar las QPS de la API de Kubernetes mediante Helm ("Número 975").
- Se corrigió la imposibilidad de montar una reclamación de volumen persistente (PVC) basada en una instantánea de una PVC de sistema de archivos XFS basada en NVMe en el mismo nodo de Kubernetes.
- Se solucionó el problema del cambio de UUID después del reinicio del host/Docker en modo NDVP agregando nombres de subsistema únicos/compartidos por backend (por ejemplo, netappdvp_subsystem).
- Se corrigieron los errores de montaje de volúmenes iSCSI durante la actualización de Trident desde versiones anteriores a la 23.10 a la 24.10 y superiores, resolviendo el problema de "SANType no válido".
- Se solucionó un problema por el cual el estado del backend de Trident no pasaba a estar en línea/fuera de línea sin reiniciar el controlador de Trident .
- Se corrigió una condición de carrera intermitente que provocaba un cambio de tamaño lento del PVC.
- Se ha corregido un error que impedía la eliminación de instantáneas en caso de fallos en la clonación de volúmenes.
- Se corrigió el fallo que impedía descomprimir el volumen cuando el kernel cambiaba su ruta de dispositivo.
- Se ha corregido el fallo al liberar el volumen debido a que el dispositivo LUKS ya estaba cerrado.
- Se solucionó un problema por el cual las operaciones de almacenamiento lentas provocaban errores de ContextDeadline.
- El operador de Trident esperará el tiempo de espera configurable de k8s-timeout para comprobar la versión de Trident .

Trident Protect

NetApp Trident Protect proporciona capacidades avanzadas de gestión de datos de aplicaciones que mejoran la funcionalidad y la disponibilidad de las aplicaciones Kubernetes con estado respaldadas por los sistemas de almacenamiento NetApp ONTAP y el aprovisionador de almacenamiento NetApp Trident CSI.

Mejoras

- Se agregaron anotaciones para controlar los tiempos de espera de las instantáneas de CR para CR programadas y de copia de seguridad:
  - `protect.trident.netapp.io/snapshot-completion-timeout`
  - `protect.trident.netapp.io/volume-snapshots-ready-to-use-timeout`
  - `protect.trident.netapp.io/volume-snapshots-created-timeout`

  Ver"Anotaciones de copia de seguridad y programación compatibles" .
- Se agregó una anotación al CR de programación para configurar el tiempo de espera de enlace de PVC, que será utilizado por el CR de respaldo: `protect.trident.netapp.io/pvc-bind-timeout-sec` . Ver"Anotaciones de copia de seguridad y programación compatibles" .
- Mejorado `tridentctl-protect` Listados de copias de seguridad e instantáneas con un nuevo campo para indicar fallos en el gancho de ejecución.

Cambios en 25.06.2

Trident

Soluciones

- **Kubernetes**: Se solucionó un problema crítico en el que se detectaban dispositivos iSCSI incorrectos al desconectar volúmenes de los nodos de Kubernetes.

Cambios en 25.06.1

Trident

> Para los clientes que usan SolidFire, no actualices a 25.06.1 debido a un problema conocido al despublicar volúmenes. 25.06.2 se lanzará pronto para solucionar este problema.

Soluciones

- **Kubernetes**:
  - Se solucionó un problema en el cual los NQN no se verificaban antes de desasignarlos de los subsistemas.
  - Se solucionó un problema en el que varios intentos de cerrar un dispositivo LUKS provocaban fallas en la separación de volúmenes.
  - Se solucionó el problema del volumen iSCSI sin preparación cuando la ruta del dispositivo cambió desde su creación.
  - Clonación de bloques de volúmenes en distintas clases de almacenamiento.
- **OpenShift**: se solucionó un problema en el que la preparación del nodo iSCSI fallaba con OCP 4.19.
- Se aumentó el tiempo de espera al clonar un volumen mediante backends de SolidFire ("Problema n.o 1008").

Cambios en 25.06

Trident

Mejoras

- **Kubernetes**:
  - Se agregó soporte para instantáneas de grupos de volúmenes CSI con `v1beta1` API de Kubernetes de instantáneas de grupo de volúmenes para el controlador iSCSI de ONTAP-SAN. Consulte "Trabajar con instantáneas de grupos de volúmenes".

    > VolumeGroupSnapshot es una función beta de Kubernetes con API beta. Kubernetes 1.32 es la versión mínima requerida para VolumeGroupSnapshot.

  - Se ha añadido compatibilidad con ONTAP ASA r2 para NVMe/TCP además de iSCSI. Ver"Opciones y ejemplos de configuración DE SAN ONTAP" .
  - Se agregó compatibilidad con SMB seguro para volúmenes ONTAP-NAS y ONTAP-NAS-Economy. Los usuarios y grupos de Active Directory ahora pueden usarse con volúmenes SMB para mayor seguridad. Consulte "Habilitar SMB seguro".
  - Concurrencia de nodos Trident mejorada para una mayor escalabilidad en las operaciones de nodos para volúmenes iSCSI.
  - Agregado `--allow-discards` al abrir volúmenes LUKS para permitir comandos discard/TRIM para recuperar espacio.
  - Rendimiento mejorado al formatear volúmenes cifrados con LUKS.
  - Limpieza de LUKS mejorada para dispositivos LUKS fallidos o parcialmente formateados.
  - Idempotencia del nodo Trident mejorada para conectar y desconectar volúmenes NVMe.
  - Agregado `internalID` campo para la configuración de volumen Trident para el controlador ONTAP-SAN-Economy.
  - Se agregó soporte para replicación de volumen con SnapMirror para backends NVMe. Consulte "Replicar volúmenes mediante SnapMirror".

Mejoras experimentales

> No apto para uso en entornos de producción.

- [Vista previa técnica] Se habilitaron operaciones simultáneas del controlador Trident a través de `--enable-concurrency` Bandera de función. Esto permite que las operaciones del controlador se ejecuten en paralelo, lo que mejora el rendimiento en entornos con mucha actividad o de gran tamaño.

  > Esta función es experimental y actualmente admite flujos de trabajo paralelos limitados con el controlador ONTAP-SAN (protocolos iSCSI y FCP).

- [Vista previa técnica] Se agregó soporte QOS manual con el controlador ANF.

Soluciones

- **Kubernetes**:
  - Se solucionó un problema con CSI NodeExpandVolume donde los dispositivos de múltiples rutas

podían quedar con tamaños incongruentes cuando los discos SCSI subyacentes no estaban disponibles.
    - Se solucionó el error al limpiar políticas de exportación duplicadas para los controladores ONTAP-NAS y ONTAP-NAS-Economy.
    - Se ha corregido que los volúmenes GCNV usen por defecto NFSv3 cuando `nfsMountOptions` no está configurado; ahora se admiten tanto los protocolos NFSv3 como NFSv4. Si `nfsMountOptions` no se proporciona, se usa la versión de NFS por defecto del host (NFSv3 o NFSv4).
    - Se solucionó el problema de implementación al instalar Trident usando Kustomize ("Problema n.o 831").
    - Se corrigieron las políticas de exportación faltantes para las PVC creadas a partir de instantáneas ("Problema n.o 1016").
    - Se solucionó el problema en el cual los tamaños de volumen ANF no se alineaban automáticamente con incrementos de 1 GiB.
    - Problema solucionado al usar NFSv3 con Bottlerocket.
- Se solucionó el problema con los volúmenes ONTAP-NAS-Economy que se expandían hasta 300 TB a pesar de las fallas de cambio de tamaño.
- Se solucionó el problema en el que las operaciones de división de clones se realizaban de manera sincrónica al usar la API REST de ONTAP.

Desusos:

- **Kubernetes**: Se actualizó el mínimo compatible de Kubernetes a v1.27.

Trident Protect

NetApp Trident Protect proporciona capacidades avanzadas de gestión de datos de aplicaciones que mejoran la funcionalidad y la disponibilidad de las aplicaciones Kubernetes con estado respaldadas por los sistemas de almacenamiento NetApp ONTAP y el aprovisionador de almacenamiento NetApp Trident CSI.

Mejoras

- Tiempos de restauración mejorados, brindando la opción de realizar copias de seguridad completas más frecuentes.
- Granularidad mejorada de la definición de aplicaciones y restauración selectiva con filtrado de grupo-versión-tipo (GVK).
- Resincronización eficiente y replicación inversa al usar AppMirrorRelationship (AMR) con NetApp SnapMirror, para evitar la replicación de PVC completa.
- Se agregó la capacidad de usar la identidad del pod EKS para crear depósitos de AppVault, lo que elimina la necesidad de especificar un secreto con las credenciales del depósito para los clústeres EKS.
- Se agregó la capacidad de omitir la restauración de etiquetas y anotaciones en el espacio de nombres de restauración, si es necesario.
- AppMirrorRelationship (AMR) ahora verificará la expansión de PVC de origen y realizará la expansión adecuada en el PVC de destino según sea necesario.

Soluciones

- Se corrigió un error que provocaba que los valores de anotación de instantáneas anteriores se aplicaran a las nuevas. Ahora todas las anotaciones de instantáneas se aplican correctamente.

- Se define un secreto para el cifrado del transportador de datos (Kopia/Restic) de manera predeterminada, si no se define.
- Se agregaron mensajes de error y validación mejorados para la creación de AppVault S3.
- AppMirrorRelationship (AMR) ahora solo replica PV en el estado enlazado, para evitar intentos fallidos.
- Se solucionó el problema en el que se mostraban errores al obtener AppVaultContent en un AppVault con una gran cantidad de copias de seguridad.
- Las instantáneas de VMS de KubeVirt se excluyen de las operaciones de restauración y conmutación por error para evitar fallas.
- Se solucionó el problema con Kopia donde las instantáneas se eliminaban prematuramente debido a que el programa de retención predeterminado de Kopia anulaba lo que había configurado el usuario en el programa.

Cambios en 25.02.1

Trident

Soluciones

- **Kubernetes**:
  - Se ha solucionado un problema en el operador Trident por el que los nombres y las versiones de las imágenes de sidecar se rellenaban incorrectamente al utilizar un registro de imágenes no predeterminado ("Problema n.o 983").
  - Se ha solucionado el problema por el que las sesiones multivía no se podían recuperar durante un retorno al nodo de respaldo de ONTAP ("Problema n.o 961").

Cambios en 25,02

A partir de Trident 25.02, el resumen de novedades proporciona detalles sobre mejoras, correcciones y desuso de las versiones Trident y Trident Protect.

Trident

Mejoras

- **Kubernetes**:
  - Se ha añadido compatibilidad con ONTAP ASA R2 para iSCSI.
  - Se añadió compatibilidad con la desconexión forzada para volúmenes ONTAP-NAS durante los escenarios de apagado de nodos sin gracia. Los nuevos volúmenes ONTAP-NAS ahora utilizarán políticas de exportación por volumen que gestiona Trident. Se proporcionó una ruta de actualización para que los volúmenes existentes transitaran al nuevo modelo de política de exportación en la no publicación sin afectar a las cargas de trabajo activas.
  - Se ha añadido la anotación cloneFromSnapshot.
  - Compatibilidad añadida para la clonación de volúmenes de espacios de nombres entre nombres.
  - Soluciones de análisis de reparación automática de iSCSI mejoradas para iniciar análisis mediante host, canal, destino e ID de LUN exactos.
  - Añadido soporte para Kubernetes 1,32.
- **OpenShift**:

    - Se añadió compatibilidad con la preparación automática de nodos iSCSI para RHCOS en clústeres ROSA.
    - Se ha añadido soporte para los controladores de virtualización de OpenShift para ONTAP.
- Se añadió compatibilidad con Fibre Channel en el controlador ONTAP-SAN.
- Se añadió soporte NVMe LUKS.
- Se ha cambiado a imagen de rasguño para todas las imágenes base.
- Se han agregado la detección y el registro del estado de conexión iSCSI cuando las sesiones iSCSI deben iniciar sesión, pero no ("Problema n.o 961").
- Se añadió compatibilidad con volúmenes SMB con el controlador google-cloud-NetApp-Volumes.
- Se ha agregado compatibilidad para permitir que los volúmenes de ONTAP omitan la cola de recuperación durante la eliminación.
- Se ha añadido soporte para sustituir las imágenes predeterminadas mediante SHAs en lugar de etiquetas.
- Se ha añadido image-pull-secrets flag al instalador de tridentctl.

Soluciones

- **Kubernetes**:
    - Se corrigieron las direcciones IP del nodo que faltaban de las políticas de exportación automáticas ("Problema n.o 965").
    - Se corrigió el cambio prematuro de políticas de exportación automática a políticas por volumen para ONTAP-NAS-Economy.
    - Credenciales de configuración de backend fijas para admitir todas las particiones ARN de AWS disponibles ("Problema n.o 913").
    - Se ha añadido la opción para desactivar la reconciliación del configurador automático en el operador Trident ("Problema n.o 924").
    - Añadido securityContext para csi-resizer container ("Problema n.o 976").

Trident Protect

NetApp Trident Protect proporciona capacidades avanzadas de gestión de datos de aplicaciones que mejoran la funcionalidad y la disponibilidad de las aplicaciones Kubernetes con estado respaldadas por los sistemas de almacenamiento NetApp ONTAP y el aprovisionador de almacenamiento NetApp Trident CSI.

Mejoras

- Se ha añadido compatibilidad con copias de seguridad y restauración para máquinas virtuales de virtualización KubeVirt / OpenShift tanto para almacenamiento volumeMode: File como volumeMode: Block (dispositivo sin formato). Este soporte es compatible con todos los controladores Trident y mejora las funciones de protección existentes al replicar almacenamiento utilizando NetApp SnapMirror con Trident Protect.
- Se ha añadido la capacidad de controlar el comportamiento de congelación a nivel de aplicación para entornos Kubevirt.
- Se ha agregado soporte para configurar conexiones proxy AutoSupport.
- Se agregó la capacidad de definir un secreto para el cifrado de los transmisores de datos (KOPIA / Restic).
- Se ha añadido la capacidad de ejecutar manualmente un gancho de ejecución.

- Se agregó la capacidad de configurar restricciones de contexto de seguridad (SCC) durante la instalación de Trident Protect.
- Se agregó soporte para configurar nodeSelector durante la instalación de Trident Protect.
- Se ha añadido soporte para el proxy de salida HTTP / HTTPS para objetos AppVault.
- Filtro de recursos ampliado para permitir la exclusión de recursos de ámbito de cluster.
- Se ha agregado soporte para el token de sesión de AWS en las credenciales de AppVault de S3.
- Se ha agregado soporte para la recopilación de recursos después de los ganchos de ejecución previos a la instantánea.

Soluciones

- Se mejoró la gestión de volúmenes temporales para omitir la cola de recuperación de volúmenes de ONTAP.
- Las anotaciones SCC ahora se restauran a los valores originales.
- Eficiencia de restauración mejorada con compatibilidad para operaciones en paralelo.
- Soporte mejorado para tiempos de espera de conexión de ejecución para aplicaciones de mayor tamaño.

Cambios en 24.10.1

Mejoras

- **Kubernetes**: Agregado soporte para Kubernetes 1,32.
- Se han agregado la detección y el registro del estado de conexión iSCSI cuando las sesiones iSCSI deben iniciar sesión, pero no ("Problema n.o 961").

Soluciones

- Se corrigieron las direcciones IP del nodo que faltaban de las políticas de exportación automáticas ("Problema n.o 965").
- Se corrigió el cambio prematuro de políticas de exportación automática a políticas por volumen para ONTAP-NAS-Economy.
- Se han actualizado las dependencias de Trident y Trident-ASUP para abordar CVE-2024-45337 y CVE-2024-45310.
- Se han eliminado los cierres de sesión de los portales no CHAP que no están en mal estado durante la reparación automática de iSCSI ("Problema n.o 961").

Cambios en 24,10

Mejoras

- El controlador de volúmenes de Google Cloud NetApp ahora está disponible de forma general para los volúmenes de NFS y es compatible con el aprovisionamiento que tiene en cuenta las zonas.
- La identidad de carga de trabajo de GCP se usará como identidad de cloud para NetApp Volumes de Google Cloud con GKE.
- Se ha añadido `formatOptions` el parámetro de configuración a los controladores ONTAP-SAN y ONTAP-SAN-Economy para permitir a los usuarios especificar las opciones de formato de LUN.

- Se redujo el tamaño mínimo de volumen mínimo de Azure NetApp Files a 50 GiB. Se espera que el nuevo tamaño mínimo de Azure esté disponible de forma general en noviembre.
- Se ha añadido `denyNewVolumePools` el parámetro de configuración para restringir los controladores ONTAP-NAS-Economy y ONTAP-SAN-Economy a pools FlexVol preexistentes.
- Detección añadida para la adición, eliminación o cambio de nombre de agregados desde la SVM en todos los controladores de ONTAP.
- Se agregaron 18 MiB de sobrecarga a los LUN LUKS para garantizar que el tamaño de PVC informado sea utilizable.
- Mejora de la etapa de nodos ONTAP-SAN y ONTAP-SAN-Economy y eliminación de errores en la etapa de almacenamiento para permitir la eliminación de dispositivos después de una etapa fallida.
- Se ha añadido un generador de roles personalizado que permite a los clientes crear un rol minimalista para Trident en ONTAP.
- Se ha añadido un registro adicional para la solución de problemas `lsscsi` ("Problema n.o 792").

Kubernetes

- Se han agregado nuevas funciones Trident para flujos de trabajo nativos de Kubernetes:
  - Protección de datos
  - Migración de datos
  - Recuperación tras siniestros
  - Movilidad de aplicaciones

  "Obtenga más información sobre Trident Protect".
- Se agregó una nueva bandera `--k8s-api-qps` para que los instaladores establezcan el valor QPS utilizado por Trident para comunicarse con el servidor API de Kubernetes.
- Se ha agregado `--node-prep` un indicador a los instaladores para la gestión automática de las dependencias del protocolo de almacenamiento en los nodos del clúster de Kubernetes. Compatibilidad probada y verificada con el protocolo de almacenamiento iSCSI Amazon Linux 2023
- Se ha añadido compatibilidad con la desconexión forzada para volúmenes ONTAP-NAS-Economy durante los escenarios de apagado de nodos sin gracia.
- Los nuevos volúmenes de NFS de ONTAP-NAS-Economy utilizan políticas de exportación por qtree cuando se utiliza `autoExportPolicy` la opción back-end. Qtrees solo se asignarán a políticas de exportación restrictivas de nodos en el momento de la publicación para mejorar el control de acceso y la seguridad. Los qtrees existentes se cambiarán al nuevo modelo de políticas de exportación cuando Trident cancelará el volumen de todos los nodos para hacerlo sin afectar a las cargas de trabajo activas.
- Añadido soporte para Kubernetes 1,31.

Mejoras experimentales

- Se ha agregado una vista previa técnica para el soporte de Fibre Channel en el controlador ONTAP-SAN.

Soluciones

- **Kubernetes**:
  - Webhook de admisión de ranchero fijo que impide las instalaciones de Trident Helm ("Problema n.o 839").

- Clave de afinidad fija en valores del gráfico de timón ("Problema n.o 898").
- Fijo tridentControllerPluginNodeSelector/tridentNodePluginNodeSelector no funcionará con el valor verdadero ("Problema n.o 899").
- Se han eliminado las snapshots efímeras creadas durante la clonación ("Problema n.o 901").

- Se ha añadido soporte para Windows Server 2019.
- Arreglado `go mod tidy`en Trident repo ("Problema n.o 767").

Amortización

- **Kubernetes:**
  - Se actualizó el mínimo admitido de Kubernetes a 1,25.
  - Se ha eliminado el soporte para la política de seguridad de POD.

Cambio de marca de productos

A partir del lanzamiento de la versión 24,10, Astra Trident cambia la marca a Trident (NetApp Trident). Esta nueva marca no afecta a ninguna función, plataforma compatible ni interoperabilidad de Trident.

Cambios en 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 agregado la capacidad de los ajustes de registro configurados por el usuario para que la controladora Trident se propague a los pods de nodos de Trident.
- Se ha añadido compatibilidad en Trident para utilizar REST DE forma predeterminada, en lugar de ONTAPI (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 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 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 Trident `SecurityContextConstraint` a `0` ("Problema n.o 887").
- Los controladores ONTAP ahora aceptan tamaños de volumen inferiores a 20 MiB ("Problema[#885").
- Se corrigió un Trident para evitar que se redujeran los volúmenes de FlexVol 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.

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.
- Se ha añadido soporte para instalar Trident como complemento en el clúster EKS desde la consola EKS.
- Se ha añadido la capacidad para configurar y deshabilitar la reparación automática de iSCSI ("Problema n.o 864").
- Se ha añadido la personalidad de Amazon FSx a los controladores de ONTAP para permitir la integración con IAM y SecretsManager de AWS, y para permitir que Trident elimine volúmenes FSx con backups ("Problema n.o 453").

Kubernetes

- Añadido soporte para Kubernetes 1,29.

Soluciones

- Se corrigieron los mensajes de advertencia ACP, cuando ACP no está habilitado ("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 tamaño nuevo solicitado es menor que el tamaño de volumen total de 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 los 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 Trident en un nodo de Windows cuando se reinicia el nodo.

Mejoras

Kubernetes

Añadido soporte para Kubernetes 1,28.

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 establecido en estado suspendido por el usuario ("Problema n.o 558").

Cambios en 23.07.1

Kubernetes: Eliminación de inicio de datos fija 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").

Trident

- Se han corregido las solicitudes de ONTAPI (ZAPI) para garantizar que se consultan 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.
- Arreglado 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 desde `rawDevicePath` para `devicePath`; se ha añadido lógica para rellenar y recuperar (en algunos casos) `devicePath` 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").

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. La desconexión forzada debe estar habilitada en el momento de la instalación mediante `--enable-force-detach` Indicador del instalador de Trident.

Soluciones

- Se ha corregido el operador Trident para usar IPv6 localhost para la instalación cuando se especifica en SPEC.
- Se corrigieron los permisos de rol de clúster de operador de Trident que estaban sincronizados con los permisos del paquete ("Número 799").
- Se ha solucionado el problema al conectar un volumen de bloques sin configurar en varios nodos en el modo RWX.

- Compatibilidad con clonado de FlexGroup fijo e importación de volúmenes para volúmenes de SMB.
- Se corrigió el problema por el que la controladora Trident no podía apagarse inmediatamente ("Problema n.o 811").
- Se agregó una corrección para mostrar todos los nombres de igroup asociados con un LUN especificado aprovisionado con controladores ontap-san-*.
- Se ha agregado una corrección para permitir que los procesos externos se ejecuten hasta su finalización.
- Corregido error de compilación para la arquitectura s390 ("Problema n.o 537").
- Se solucionó un nivel de registro incorrecto durante las operaciones de montaje de volúmenes ("Problema n.o 781").
- Se ha corregido el error de afirmación de tipo potencial ("Problema n.o 802").

Mejoras

- Kubernetes:
  - Añadido soporte para Kubernetes 1,27.
  - Se ha añadido soporte para importar volúmenes LUKS.
  - Se ha añadido soporte para el modo de acceso de PVC ReadWriteOncePod.
  - Se añadió compatibilidad con la desconexión forzada para volúmenes ONTAP-SAN-* durante los escenarios de apagado de nodos sin gracia.
  - Todos los volúmenes de ONTAP-SAN-* ahora utilizarán iGroups por nodo. Las LUN solo se asignarán a iGroups, mientras que se publicarán de forma activa en esos nodos para mejorar nuestra política de seguridad. Los volúmenes existentes se cambiarán de forma oportunista al nuevo esquema de igroup cuando Trident determina que es seguro hacerlo sin afectar a las cargas de trabajo activas ("Problema n.o 758").
  - Mejora en la seguridad de Trident mediante la limpieza de los iGroups gestionados por Trident sin utilizar de los back-ends ONTAP-SAN-*.
- Se ha añadido soporte para volúmenes SMB con Amazon FSx para la economía de ontap-nas y los controladores de almacenamiento de ontap-nas-flexgroup.
- Se añadió compatibilidad con recursos compartidos SMB con los controladores de almacenamiento ONTAP-nas, ontap-nas y ontap-nas-flexgroup.
- Se ha añadido compatibilidad con los nodos arm64 ("Problema n.o 732").
- Ha mejorado el procedimiento de apagado de Trident desactivando los servidores API en primer lugar ("Problema n.o 811").
- Agregado soporte de compilación multiplataforma para hosts Windows y arm64 a Makefile; consulte BUILD.md.

Amortización

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

Cambios en 23.01.1

Soluciones

- Se ha corregido el operador Trident para usar IPv6 localhost para la instalación cuando se especifica en

SPEC.

- Se han corregido los permisos de rol de clúster del operador de Trident para que estén sincronizados con los permisos del paquete "Número 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 Trident antes de actualizar Kubernetes.

Soluciones

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

Mejoras

Kubernetes

- Se ha añadido la compatibilidad con Kubernetes 1.26.
- Mejora de la utilización general de recursos de RBAC de Trident ("Número 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.

Trident

- Se añadió soporte para volúmenes SMB con Amazon FSx para NetApp ONTAP en el 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 económico de ONTAP-nas al gestionar varios volúmenes de FlexVol
- Actualizaciones de datLIF activadas para todas las controladoras de almacenamiento NAS de ONTAP.
- Se han actualizado la convención de nomenclatura Trident Deployment y DemonSet para reflejar el sistema operativo del nodo del host.

Amortización

- Kubernetes: Se ha actualizado el mínimo admitido de Kubernetes a 1.21.
- Ya no se deben especificar LIF de datos durante la configuración de `ontap-san` los controladores o.

`ontap-san-economy`

Cambios en 22.10

Debe leer la siguiente información crítica antes de actualizar a Trident 22,10.

Información crítica sobre Trident 22.10

- Kubernetes 1,25 ahora es compatible con Trident. Debe actualizar Trident a 22,10 antes de actualizar a Kubernetes 1,25.
- Ahora, Trident aplica estrictamente el uso de la configuración de rutas múltiples en entornos SAN, con un valor recomendado de `find_multipaths: no` en el archivo multipath.conf.

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

Soluciones

- Se ha solucionado un problema específico del back-end de ONTAP creado mediante `credentials` el campo no se puede conectar durante la actualización 22.07.0 ("Número 759").
- **Docker:** se ha solucionado un problema que provocaba que el complemento para volúmenes de Docker no empezara en algunos entornos ("Número 548" y.. "Número 760").
- Se solucionó el problema de SLM específico de los back-ends de SAN de ONTAP para garantizar que solo se publica un subconjunto de LIF de datos que pertenecen a los 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 dentro del flujo de trabajo de iSCSI de Trident para producir fallos 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 Trident a 22,10 antes de actualizar a Kubernetes 1,25.
  - Se ha agregado una cuenta de servicio, ClusterRole y ClusterRoleBinding aparte para la implementación de Trident y DemonSet para permitir futuras mejoras de permisos.
  - Se ha agregado compatibilidad con "uso compartido de volúmenes entre espacios de nombres".
- Todos los Trident `ontap-*` Los controladores de almacenamiento ahora funcionan con la API DE REST de ONTAP.
- Se ha añadido un nuevo operador yaml (`bundle_post_1_25.yaml`) sin a. `PodSecurityPolicy` Para admitir Kubernetes 1.25.
- Añadido "Compatibilidad con volúmenes cifrados LUKS" para `ontap-san` y.. `ontap-san-economy` impulsores del almacenamiento.

- Se ha agregado compatibilidad con nodos de Windows Server 2019.
- Añadido "Compatibilidad con volúmenes SMB en nodos de Windows" a través de la `azure-netapp-files` controlador de almacenamiento.
- La detección de conmutación automática de MetroCluster para controladores ONTAP está disponible por lo general.

Amortización

- **Kubernetes:** Actualizado el mínimo admitido de Kubernetes a 1.20.
- Se ha eliminado el controlador Astra Data Store (ADS).
- Se ha quitado el soporte de `yes` y.. `smart` opciones para `find_multipaths` Al configurar accesos múltiples de nodos de trabajo para iSCSI.

Cambios en 22.07

Soluciones

Kubernetes

- Se ha solucionado el problema para manejar los valores booleanos and Number para el selector de nodos cuando se configura Trident con Helm o el operador de Trident. ("GitHub número 700")
- Se ha solucionado el problema al gestionar errores de ruta no CHAP, de modo que kubelet lo volverá a intentar si falla. "GitHub número 736")

Mejoras

- Pasar de k8s.gcr.io a registry.k8s.io como registro predeterminado para las imágenes CSI
- Los volúmenes de ONTAP-SAN ahora utilizan iGroups por nodo y solo asignan LUN a iGroups, mientras se publican de forma activa en esos nodos para mejorar nuestra política de seguridad. Los volúmenes existentes se conmutarán de manera oportunista al nuevo esquema de igroup, cuando Trident determina 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:** Actualizado el mínimo admitido de Kubernetes a 1.19.
- La configuración de back-end ya no permite múltiples tipos de autenticación en una única configuración.

Absorciones

- Se ha eliminado el controlador CVS de AWS (obsoleto desde 22.04).
- Kubernetes
  - Se eliminó la capacidad SYS_ADMIN innecesaria de los POD de nodos.

- Reduce 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 agregado una nueva "Estándares de seguridad de POD"sección (PSS) que detalla los permisos habilitados por Trident en la instalación.

Cambios en 22.04

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

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

Soluciones

- Análisis mejorado de nombres de iniciadores iSCSI. ("GitHub número 681")
- Se ha solucionado un problema en el que no se permitían los parámetros de clase de almacenamiento CSI. ("GitHub número 598")
- Se ha corregido la declaración de clave duplicada en Trident CRD. ("GitHub número 671")
- Se han corregido registros de instantánea CSI imprecisos. ("GitHub número 629"))
- Se ha solucionado el problema con la anulación de la publicación de volúmenes en nodos eliminados. ("GitHub número 691")
- Se ha añadido el tratamiento de incoherencias del sistema de archivos en dispositivos de bloque. ("GitHub número 656")
- Se ha solucionado el problema al extraer imágenes de soporte automático al configurar el `imageRegistry` indicador durante la instalación. ("GitHub número 715")
- Se 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.
- Las LIF de datos de SAN IPv6 de ONTAP funcionan ahora si se especifica con paréntesis.
- Un problema fijo en el que intentar importar un volumen ya importado devuelve EOF dejando PVC en estado pendiente. ("GitHub número 489")
- Se solucionó el problema cuando el rendimiento de Trident se ralentiza al crear más de 32 snapshots en un volumen de 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:
  - Se ha añadido la compatibilidad con 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 el problema por el que Trident registraba un error fatal al cerrar (GitHub problema 597).
- Correcciones relacionadas con Kubernetes:
  - Devuelva el espacio usado de un volumen como el tamaño mínimo de restoreSize a la hora de crear snapshots con `ontap-nas` y.. `ontap-nas-flexgroup` Controladores (GitHub, número 645).
  - Se ha solucionado el problema `Failed to expand filesystem` Se registró el error después de cambiar el tamaño del volumen (problema 560 de GitHub).
  - Se ha solucionado un problema por el que se podría atascar un pod `Terminating` estado (GitHub número 572).
  - Se ha fijado la caja donde un `ontap-san-economy` Es posible que FlexVol esté lleno de LUN de snapshot (GitHub, número 533).
  - Se ha solucionado el problema del instalador de YAML personalizado con una imagen diferente (GitHub, número 613).
  - Se ha corregido el cálculo del tamaño de la instantánea (GitHub, número 611).
  - Se solucionó un problema por el que todos los instaladores de Trident podían identificar Kubernetes sin formato como OpenShift (GitHub número 639).
  - Se ha solucionado el operador Trident para detener la reconciliación si no se puede acceder al servidor API de Kubernetes (GitHub, número 599).

Mejoras

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

Mejoras experimentales

- Se añadió la compatibilidad con la replicación de volúmenes en `ontap-san` controlador.

- Se ha añadido el soporte DE DESCANSO **vista previa tecnológica** para el `ontap-nas-flexgroup`, `ontap-san`, y. `ontap-nas-economy` de windows

Problemas conocidos

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

- Cuando actualice un clúster de Kubernetes de 1,24 a 1,25 o una versión posterior que tiene Trident instalado, debe actualizar los valores.yaml para establecer `excludePodSecurityPolicy` `true` o agregar `--set excludePodSecurityPolicy=true` al `helm upgrade` comando antes de poder actualizar el clúster.
- Trident ahora aplica un espacio en blanco `fsType` (`fsType=""`) para volúmenes que no tienen 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 backend en varias instancias de Trident, cada archivo de configuración de backend debe tener un valor diferente `storagePrefix` para los back-ends de ONTAP o utilizar otro `TenantName` para los back-ends de SolidFire. Trident no puede detectar volúmenes que han creado otras instancias de Trident. Si se intenta crear un volumen existente en back-ends ONTAP o SolidFire se completa correctamente, porque 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.
- Al instalar Trident (utilizando `tridentctl` o el Operador Trident) y utilizar `tridentctl` para administrar Trident, debe asegurarse de que la `KUBECONFIG` variable de entorno está definida. 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 es cierto para las instancias de RHEL/Red Hat Enterprise Linux CoreOS (RHCOS), que requieren el `discard` "opción de montaje"; Asegúrese de que la opción Descartar mountOption está incluida en el[`StorageClass`^] para admitir el descarte de bloques en línea.
- Si hay más de una instancia de Trident por clúster de Kubernetes, Trident no puede comunicarse con otras instancias y no puede detectar otros volúmenes que hayan creado, lo que provoca un comportamiento inesperado e incorrecto si más de una instancia se ejecuta en un clúster. Solo debería haber una instancia de Trident por clúster de Kubernetes.
- Si los objetos basados en Trident `StorageClass` se eliminan de Kubernetes mientras Trident está desconectado, Trident no elimina las clases de almacenamiento correspondientes de su base de datos cuando vuelve a estar online. Debe eliminar estas clases de almacenamiento mediante `tridentctl` o la API de REST.
- Si un usuario elimina un volumen persistente aprovisionado por Trident antes de eliminar la RVP correspondiente, Trident no elimina automáticamente el volumen de 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 Trident a través de IPv6, debe especificar `managementLIF` y `dataLIF` en la definición de back-end dentro de corchetes. Por ejemplo, `[fd20:8b1e:b258:2000:f816:3eff:feec:0]`.

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

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

Obtenga más información

- "Trident GitHub"
- "Blogs de Trident"

Versiones anteriores de la documentación

Si no estás ejecutando Trident 26.02, la documentación de las versiones anteriores está disponible según el "Ciclo de vida del soporte de Trident".

- "Trident 25.10"
- "Trident 25.06"
- "Trident 25,02"
- "Trident 24,10"
- "Trident 24,06"
- "Trident 24,02"
- "Trident 23,10"
- "Trident 23,07"
- "Trident 23,04"
- "Trident 23,01"

NetApp soporte Trident para sistemas de almacenamiento ONTAP ASA r2

NetApp Trident 25.02 y versiones posteriores son compatibles con los sistemas NetApp ASA r2 como backends de almacenamiento. Consulta "Sistemas ASA r2" para más información.

Los sistemas ASA r2 requieren el `ontap-san` driver. Trident no admite el `ontap-san-economy` driver para los sistemas ASA r2.

Cuando especificas `ontap-san` como `storageDriverName` en la configuración del backend, Trident detecta automáticamente el sistema de almacenamiento ASA r2.

Trident proporciona protección de datos limitada para los sistemas ASA r2 con Trident protect.

Los protocolos SAN compatibles dependen de tu versión de Trident:

- Trident 25.02 y versiones posteriores son compatibles con iSCSI.

- Trident 25.06 y versiones posteriores admiten NVMe/TCP además de iSCSI.
- Trident 25.10 y posteriores admiten FC además de iSCSI y NVMe/TCP.

Debes asignar al menos un agregado a la máquina virtual de almacenamiento (SVM) para el almacenamiento backend de ONTAP. Consulta "Asignar agregados a SVM en sistemas ASA r2" para instrucciones.

Operaciones compatibles

- Aprovisionamiento de volúmenes persistentes (PVs)
- Aprovisionamiento dinámico de volúmenes
- Creación y eliminación de volúmenes
- Clonación de volúmenes
- Expandir volúmenes
- Gestión de clases de almacenamiento

Operaciones no admitidas

- Cifrado LUKS
- SnapMirror replicación de volumen
- Limitar el uso de agregados
- Modo de reserva de espacio
- Snapshot
- Organización en niveles

Para más información, consulta "Opciones y ejemplos de configuración DE SAN ONTAP".

Problemas conocidos

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

Los siguientes problemas conocidos afectan a la versión actual:

VolumeSnapshots no alcanza el estado ReadyToUse

Existe un problema conocido con el `csi-snapshotter` sidecar. En todas las versiones de Kubernetes, VolumeGroupSnapshots `v1beta1` impide que VolumeSnapshots llegue al estado `ReadyToUse`.

Hay dos soluciones alternativas:

1. Elimina los CRD de VolumeGroupSnapshots para desactivar VolumeGroupSnapshots y luego vuelve a instalar Trident.
2. Instala VolumeGroupSnapshots `v1beta2` y `snapshot-controller` versión 8.4.0 o posterior, y luego vuelve a instalar Trident. VolumeGroupSnapshots no funcionan en versiones de Kubernetes anteriores a v1.34.

La restauración de copias de seguridad de archivos grandes de Restic puede fallar

Al restaurar archivos de 30 GB o más grandes desde una copia de seguridad de Amazon S3 realizada con Restic, la operación de restauración puede fallar. Como solución alternativa, realice una copia de seguridad de los datos utilizando Kopia como gestor de copias de seguridad (Kopia es el gestor de copias de seguridad predeterminado). Referirse a "Proteja las aplicaciones con Trident Protect" para obtener instrucciones.

Manos a la obra

Obtenga más información sobre Trident

Obtenga más información sobre Trident

Trident es un proyecto de código abierto totalmente compatible y mantenido por NetApp. 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 Trident?

Netapp Trident permite el consumo y la gestión de recursos de almacenamiento en todas las plataformas de almacenamiento populares de NetApp, en la nube pública o en las instalaciones, incluidos los clústeres ONTAP locales (AFF, FAS y ASA), ONTAP Select, Cloud Volumes ONTAP, Element software (NetApp HCI, SolidFire), Azure NetApp Files, Amazon FSx for NetApp ONTAP y Google Cloud NetApp Volumes.

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". Trident se ejecuta como un pod de controladora más un pod de nodo en cada nodo trabajador del clúster. Consulte "Arquitectura de Trident" para obtener más información.

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 Trident para Docker" para obtener más información.

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

Plataformas Kubernetes compatibles

Trident es compatible con una amplia gama de distribuciones y plataformas de Kubernetes.

Entre las plataformas compatibles se incluyen: * Upstream Kubernetes * Red Hat OpenShift * SUSE Harvester 1.7.0 (ONTAP iSCSI)

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 con la tecnología de NetApp. Puede ejecutar sus cargas de trabajo basadas en archivos más exigentes de forma nativa en Azure, con el rendimiento y la gestión de datos enriquecidos que espera de NetApp.

Cloud Volumes 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.

NetApp Volumes para Google Cloud

"NetApp Volumes para Google Cloud" Es un servicio de almacenamiento de archivos completamente gestionado en Google Cloud que ofrece un almacenamiento de archivos de alto rendimiento y clase empresarial.

Software Element

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

NetApp HCI

"NetApp HCI" simplifica la gestión y el escalado del centro de datos mediante la automatización de las tareas rutinarias y permite que los administradores de la infraestructura se centren en funciones más importantes.

Trident 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 únicamente HDD y ofrecen distintos modelos de puesta en marcha: Clústeres ASA, FAS y AFA en las instalaciones, ONTAP Select y Cloud Volumes ONTAP. Trident admite estos modelos de puesta en marcha de ONTAP.

Arquitectura de Trident

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

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

Trident se pone en marcha como Pod de controladora de Tridentuno o más 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 Trident, se pueden configurar selectores de nodos y toleraciones para pods de nodos y controladoras.

- 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.

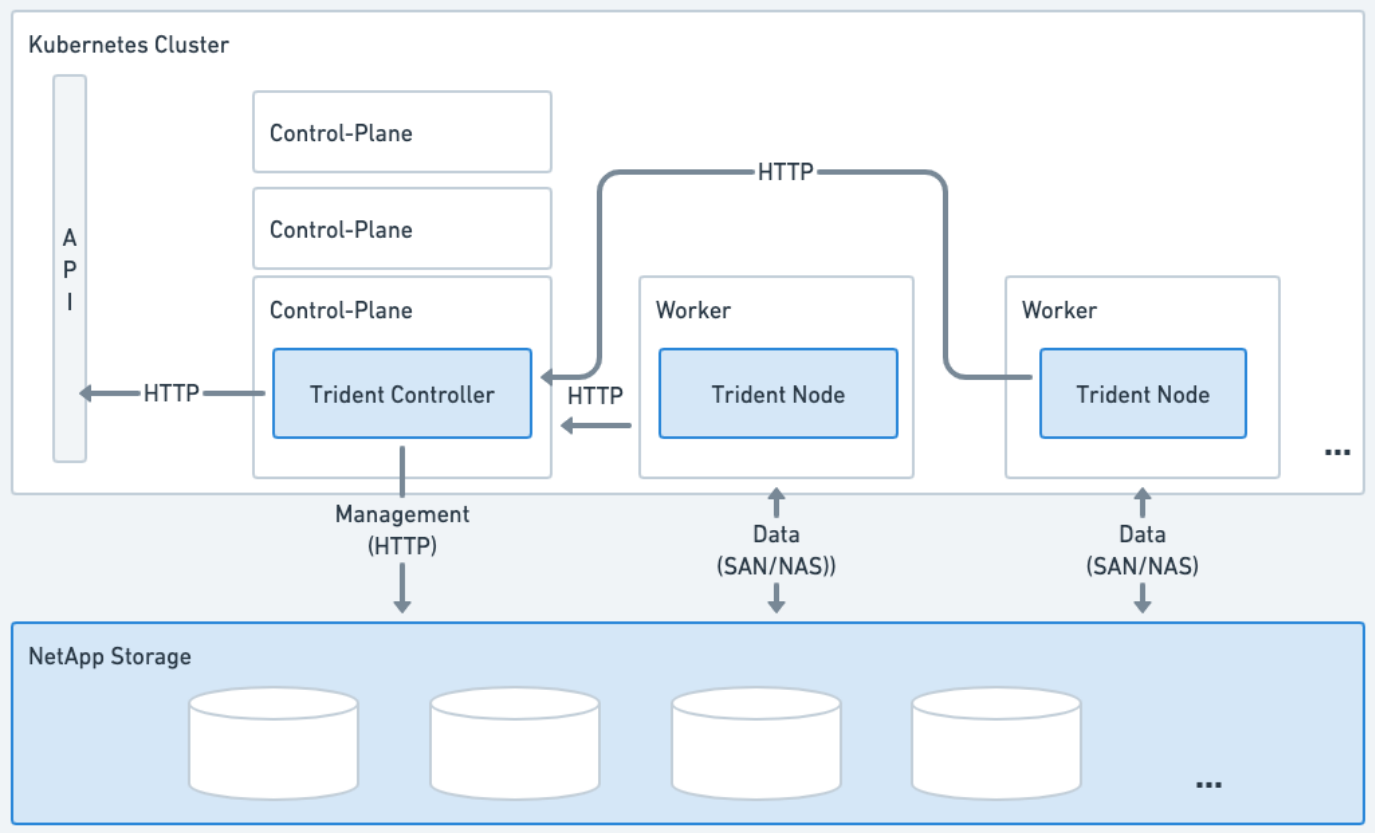


Figura 1. Trident puesto en marcha en el clúster de Kubernetes

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.

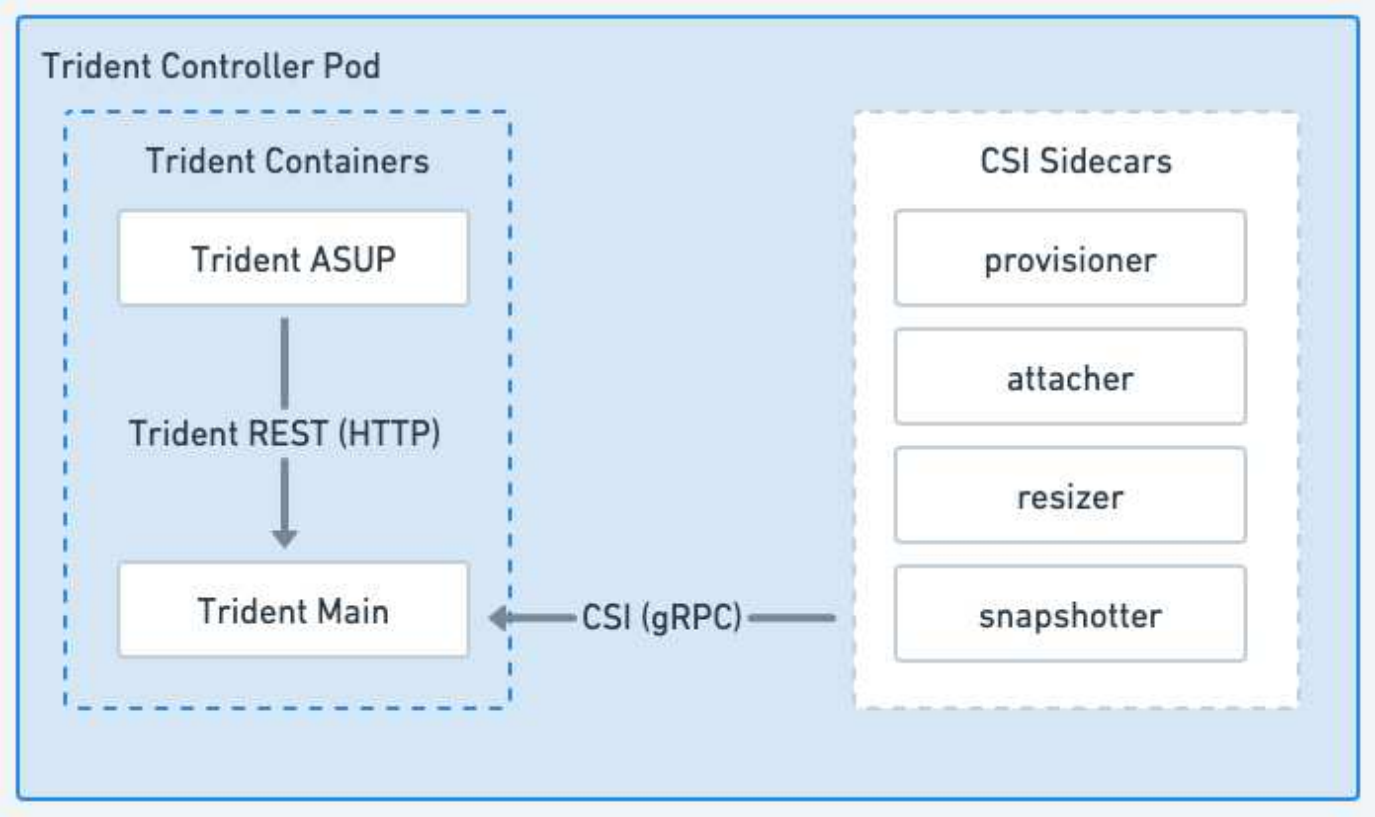


Figura 2. Diagrama de Trident Controller Pod

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

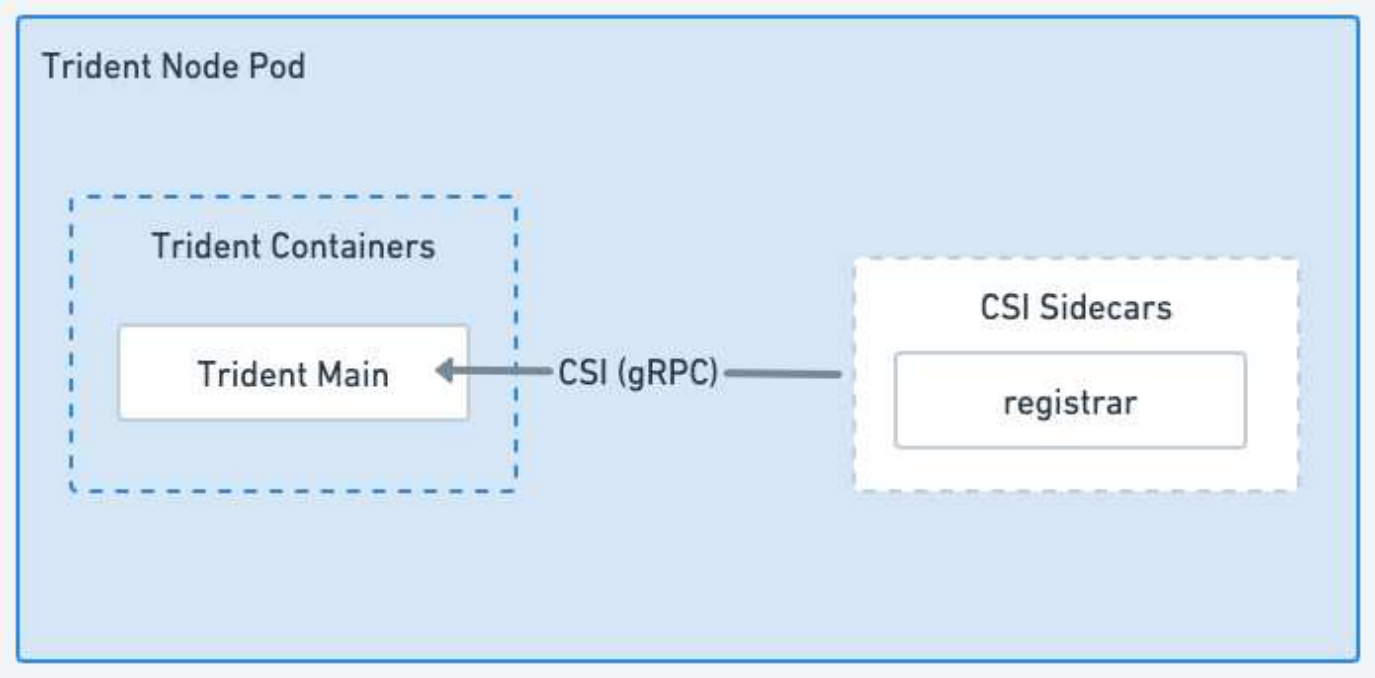


Figura 3. Diagrama de Trident Node Pod

Arquitecturas de clúster de Kubernetes compatibles

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 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 nombres de pools solicitados, Trident agrega ese pool de almacenamiento al conjunto de pools de almacenamiento adecuados para esa clase de almacenamiento. Además, Trident añade a ese conjunto todos los pools de almacenamiento mostrados en `additionalStoragePools` la lista, aunque sus atributos no cumplan todos o ninguno 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. Trident realiza un proceso similar cada vez que agrega un nuevo back-end, comprobando si sus pools de almacenamiento satisfacen las de las clases de almacenamiento existentes y eliminando las que se hayan marcado como excluidas.

Creación del volumen

A continuación, Trident utiliza las asociaciones entre las clases de almacenamiento y los pools de almacenamiento para determinar dónde aprovisionar volúmenes. Al crear un volumen, Trident obtiene primero el conjunto de pools de almacenamiento para la clase de almacenamiento de ese volumen y, si especifica un protocolo para el volumen, Trident quita los pools de almacenamiento que no pueden proporcionar el protocolo solicitado (por ejemplo, un back-end NetApp HCI/SolidFire no puede proporcionar un volumen basado en archivos, mientras que un back-end NAS de ONTAP no puede ofrecer un volumen basado en bloques). Trident aleatoriza el orden de este conjunto resultante para facilitar una distribución uniforme de los volúmenes y, luego, repite a través del mismo, intentando aprovisionar el volumen en cada pool de almacenamiento. Si se produce correctamente en una, vuelve con éxito y registra los fallos encontrados en el proceso. Trident devuelve un fallo **solo si** no proporciona en **all** los pools de almacenamiento disponibles para la clase de almacenamiento y el protocolo solicitados.

Copias de Snapshot de volumen

Obtenga más información sobre cómo Trident maneja la creación de snapshots de volúmenes para sus controladores.

Obtenga información acerca de la creación de snapshots de volúmenes

- Para el `ontap-nas` , `ontap-san` , y `azure-netapp-files` En cada controlador, cada volumen persistente (PV) se asigna a un FlexVol volume . Como resultado, las instantáneas de volumen se crean como instantáneas de NetApp . La tecnología de instantáneas de NetApp ofrece mayor estabilidad, escalabilidad, capacidad de recuperación y rendimiento que las tecnologías de instantáneas de la competencia. Estas copias instantáneas son extremadamente eficientes tanto en el tiempo necesario para crearlas como en el espacio de almacenamiento.
- Para la `ontap-nas-flexgroup` Cada controlador, cada volumen persistente (PV) se asigna a una FlexGroup. Como resultado, las copias de Snapshot de volumen se crean como copias de Snapshot de FlexGroup de NetApp. La tecnología Snapshot de NetApp ofrece una mayor estabilidad, escalabilidad, capacidad de recuperación y rendimiento que 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 volúmenes FlexVol compartidos. Las Snapshots de los VP se logran realizando 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 la `solidfire-san` Cada controlador, cada VP asigna una LUN creada en el software NetApp Element/clúster HCI de NetApp. Las copias Snapshot de volumen están representadas por copias

Snapshot de Element de la LUN subyacente. Estas copias Snapshot son copias puntuales y solo ocupan una pequeña cantidad de recursos y espacio del sistema.

- 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 snapshot de volumen creada mediante Kubernetes, Trident elimina la snapshot de ONTAP asociada. También se pueden eliminar las copias de Snapshot de ONTAP que no se crearon con Kubernetes.

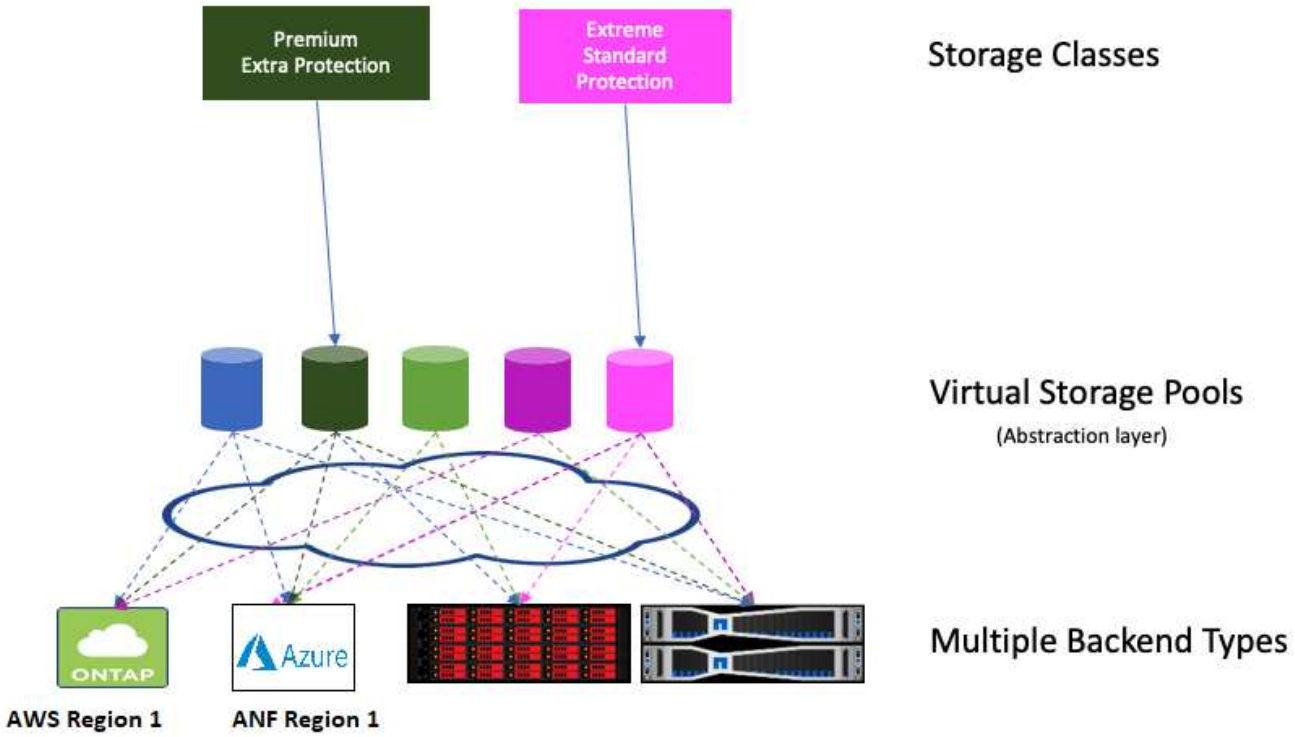
Con Trident, puedes usar VolumeSnapshots para crear nuevos PV a partir de ellos. La creación de PV a partir de estas instantáneas se realiza utilizando la tecnología FlexClone para backends ONTAP compatibles. Al crear un PV a partir de una instantánea, el volumen de respaldo es un FlexClone del volumen principal de la instantánea. El `solidfire-san` El controlador utiliza clones de volumen de software Element para crear PV a partir de instantáneas. Aquí se crea un clon a partir de la instantánea del elemento.

Pools virtuales

Los pools virtuales proporcionan una capa de abstracción entre los back-ends de almacenamiento de 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 Trident en un archivo de definición JSON o YAML.



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

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

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

Las etiquetas del pool virtual se pueden desafiar usando estos caracteres:

- letras mayúsculas `A-Z`
- letras minúsculas `a-z`
- números `0-9`
- guiones bajos _
- guiones -

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

Operador	Ejemplo	El valor de etiqueta de un pool debe:
`=`	rendimiento=premium	Coincidencia
`!=`	rendimiento!=extremo	No coincide
`in`	ubicación en (este, oeste)	Esté en el conjunto de valores
`notin`	rendimiento de la muesca (plata, bronce)	No esté en el conjunto de valores
`<key>`	protección	Existe con cualquier valor
`!<key>`	!protección	No existe

Los grupos de acceso de volúmenes

Obtenga más información sobre cómo utiliza 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 utiliza Trident en modo CSI, puede ignorar esta sección. Trident utiliza CHAP cuando se instala como un aprovisionador de CSI mejorado.

Obtenga información acerca de los grupos de acceso de volúmenes

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.

Mientras que 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 Trident y deben contener los IQN iSCSI de cada nodo en el 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 desde una que esté utilizando el valor predeterminado `trident` Incluya el ID de para uno que utilice también otros `trident` grupo de acceso de la lista.

Inicio rápido para Trident

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

Para Docker, consulte "Trident para Docker".

1 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"

2 Instale Trident

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

"Instale Trident"

3 Cree un back-end

Un back-end define la relación entre Trident y un sistema de almacenamiento. Indica a 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

4 Cree una clase de almacenamiento de Kubernetes

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

"Cree una clase de almacenamiento"

5 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 Trident, debe revisar estos requisitos generales del sistema. Es posible que los back-ends específicos tengan requisitos adicionales.

Información crítica sobre Trident

Debe leer la siguiente información crítica sobre Trident.

**Información crítica sobre Trident **

- Kubernetes 1.36 ya es compatible con Trident. Actualiza Trident antes de actualizar Kubernetes.
- Trident aplica estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en archivo multipath.conf.

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

Front-ends compatibles (orquestadores)

Trident admite varios motores de contenedor y orquestadores, entre los que se incluyen los siguientes:

- Anthos on-premises (VMware) y Anthos en 1,16 básico
- Kubernetes 1.27 - 1.36
- OpenShift 4.12, 4.14 - 4.22 (si planeas usar la preparación de nodos iSCSI con OpenShift 4.19, la versión mínima de Trident compatible es 25.06.1.)

> Trident continúa brindando soporte para versiones anteriores de OpenShift en consonancia con la"Ciclo de vida de lanzamiento de Red Hat Extended Update Support (EUS)" , incluso si dependen de versiones de Kubernetes que ya no reciben soporte oficial. Al instalar Trident en estos casos, puede ignorar con seguridad cualquier mensaje de advertencia sobre la versión de Kubernetes.

- Rancher Kubernetes Engine 2 (RKE2) v1.28.x - 1.36.x

Trident también funciona con una gran cantidad de 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) y la cartera de VMware Tanzu.

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

> Antes de actualizar un clúster de Kubernetes de la versión 1,25 a la versión 1,26 o una versión posterior que tenga instalado Trident, consulte "Actualizar una instalación Helm".

Back-ends compatibles (almacenamiento)

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

- Amazon FSX para ONTAP de NetApp
- Azure NetApp Files
- Cloud Volumes ONTAP
- NetApp Volumes para Google Cloud
- Cabina All SAN de NetApp (ASA)
- FAS, AFF o ASA r2 locales (iSCSI, NVMe/TCP y FC) que ejecutan versiones de ONTAP con soporte total o limitado de NetApp . Ver "Compatibilidad con versiones de software".
- Software HCI/Element de NetApp 11 o posterior

Compatibilidad de Trident con KubeVirt y OpenShift Virtualization

Controladores de almacenamiento compatibles:

Trident admite los siguientes controladores de ONTAP para KubeVirt y OpenShift Virtualization:

- ontap-nas
- ONTAP-san (iSCSI, FCP, NVMe sobre TCP)
- Economía ONTAP-san (solo iSCSI)

Puntos que se deben tener en cuenta:

- Actualice la clase de almacenamiento para tener el `fsType` parámetro (por ejemplo `fsType: "ext4"`:) En el entorno de virtualización OpenShift. Si es necesario, configure el modo de volumen para bloquear explícitamente mediante el `volumeMode=Block` parámetro de `dataVolumeTemplates` para notificar a CDI que cree volúmenes de datos de bloque.
- *RWX modo de acceso para controladores de almacenamiento en bloque*: Los controladores ONTAP-san (iSCSI, NVMe/TCP, FC) y ONTAP-san-economy (iSCSI) solo se admiten con «volumeMode: Block» (dispositivo sin procesar). Para estos controladores, no se puede utilizar el `fstype` parámetro porque los volúmenes se proporcionan en modo de dispositivo raw.
- Para flujos de trabajo de migración en tiempo real en los que se requiere el modo de acceso RWX, se admiten estas combinaciones:
  - NFS + `volumeMode=Filesystem`
  - ISCSI + `volumeMode=Block` (dispositivo sin formato)
  - NVMe/TCP + `volumeMode=Block` (dispositivo sin configurar)
  - FC `volumeMode=Block` + (dispositivo bruto)

Compatibilidad de Trident con Kubernetes en Kubernetes y con los planos de control alojados en OpenShift

A partir de la versión 26.06, Trident califica completamente la topología de Kubernetes on Kubernetes. Clústeres de Kubernetes cuyos nodos se ejecutan como máquinas virtuales KubeVirt en un clúster de infraestructura (host). Esta topología respalda ofertas de planos de control alojados como los hosted control planes (HCP) de OpenShift en OpenShift Virtualization.

Esta configuración funcionaba en versiones anteriores, pero la certificación completa comienza con la 26.06.

El clúster invitado (inquilino) utiliza el controlador CSI KubeVirt (`kubevirt-csi` como proveedor de almacenamiento. Cuando una carga de trabajo en el clúster invitado solicita un `PersistentVolumeClaim`, el controlador CSI KubeVirt crea un volumen correspondiente en el clúster de infraestructura. Trident aprovisiona el volumen desde ONTAP y el volumen se conecta en caliente a la VM del nodo invitado. Trident solo se ejecuta en el clúster de infraestructura. El clúster invitado consume almacenamiento ONTAP de forma indirecta a través del controlador CSI KubeVirt.

Requisitos:

- Trident se instala en el clúster de infraestructura con un backend ONTAP compatible.
- En el clúster de infraestructura existe una clase de almacenamiento Trident para que el controlador CSI de KubeVirt pueda asignar los volúmenes del clúster de invitados.
- KubeVirt o OpenShift Virtualization se implementa en el clúster de infraestructura, y el controlador CSI KubeVirt se instala en cada clúster invitado.

Componente	Versión
Trident	26,06
Kubernetes (infraestructura y clúster invitado)	1,36
Virt de KubeVirt	1.9.0-beta.0
Importador de datos en contenedores (CDI)	1,65

Puntos que se deben tener en cuenta:

- En el clúster invitado, los volúmenes aparecen como dispositivos de bloque conectados en caliente en las máquinas virtuales del nodo invitado.
- El flujo de trabajo del controlador CSI de KubeVirt no requiere Trident en el clúster invitado. Configuras el backend de ONTAP y la clase de almacenamiento en el clúster de infraestructura.

Instala Trident directamente en el clúster invitado:

También puedes instalar Trident directamente en el clúster invitado (inquilino). Instala Trident en el clúster invitado cuando necesites el conjunto completo de funciones de Trident, como instantáneas, clones, cambio de tamaño de volúmenes e importación, que `kubevirt-csi` no ofrece por sí solo.

En el clúster de invitados, utiliza protocolos basados en IP: NFS, iSCSI y NVMe/TCP. El clúster de invitados no es compatible con Fibre Channel (FC) porque los nodos de invitados son máquinas virtuales sin conectividad FC directa.

Requisitos de funciones

En la tabla siguiente se resumen las funciones disponibles con este lanzamiento de Trident y las versiones de Kubernetes compatibles.

Función	La versión de Kubernetes	¿Se requieren puertas de funciones?
Trident	1.27 - 1.36	No

Función	La versión de Kubernetes	¿Se requieren puertas de funciones?
Snapshots de volumen	1.27 - 1.36	No
RVP desde snapshots de volumen	1.27 - 1.36	No
Cambio de tamaño del VP de iSCSI	1.27 - 1.36	No
CHAP bidireccional de ONTAP	1.27 - 1.36	No
Políticas de exportación dinámicas	1.27 - 1.36	No
Operador de Trident	1.27 - 1.36	No
Topología CSI	1.27 - 1.36	No

Se probaron sistemas operativos host

Aunque Trident no admite oficialmente sistemas operativos específicos, se sabe que los siguientes funcionan:

- Versiones de Red Hat Enterprise Linux CoreOS (RHCOS) compatibles con OpenShift Container Platform en AMD64 y ARM64
- Red Hat Enterprise Linux (RHEL) 8 o posterior en AMD64 y ARM64

  NVMe/TCP requiere RHEL 9 o posterior.

- Ubuntu 22.04 LTS o posterior en AMD64 y ARM64
- Windows Server 2022
- SUSE Linux Enterprise Server (SLES) 15 o posterior

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

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

Configuración de hosts

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

"Prepare el nodo de trabajo"

Configuración del sistema de almacenamiento

Es posible que Trident requiera cambios en un sistema de almacenamiento antes de que la configuración de back-end pueda utilizarlo.

"Configurar los back-ends"

Puertos Trident

Trident requiere acceso a puertos específicos para la comunicación.

"Puertos Trident"

Imágenes de contenedor y las versiones de Kubernetes correspondientes

En el caso de instalaciones de aire acondicionado, la siguiente lista es una referencia de las imágenes de contenedores necesarias para instalar Trident. Utilice `tridentctl images` el comando para verificar la lista de imágenes de contenedor necesarias.

Imágenes de contenedor requeridas para Trident 26.02

Versiones de Kubernetes	Imagen de contenedor
v1.27.0, v1.28.0, v1.29.0, v1.30.0, v1.31.0, v1.32.0, v1.33.0, v1.34.0, v1.35.0, v1.36.0	• docker.io/netapp/trident:26.02.0 • docker.io/netapp/trident-autosupport:26.02 • registry.k8s.io/sig-storage/csi-provisioner:v6.1.0 • registro.k8s.io/sig-storage/csi-attacher:v4.10.0 • registry.k8s.io/sig-storage/csi-resizer:v2.0.0 • registry.k8s.io/sig-storage/csi-snapshotter:v8.5.0 • registro.k8s.io/sig-storage/csi-node-driver-registrar:v2.15.0 • docker.io/netapp/trident-operator:26.02.0 (opcional)

Instale Trident

Obtenga información acerca de la instalación de Trident

Para garantizar que Trident se puede instalar en una amplia variedad de entornos y organizaciones, NetApp ofrece varias opciones de instalación. Puede instalar 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 crítica sobre Trident 26.02

Debe leer la siguiente información crítica sobre Trident.

**Información crítica sobre Trident **

- Kubernetes ya es compatible con Trident. Actualiza Trident antes de actualizar Kubernetes.
- Trident aplica estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en archivo multipath.conf.

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

Antes de empezar

Independientemente de la ruta de instalación, debe tener:

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

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

Elija el método de instalación

Selecciona el método de instalación que sea mejor para ti. También deberías revisar las consideraciones para

"moverse entre métodos" antes de tomar tu decisión.

Utilice el operador Trident

Ya sea que implementes manualmente o usando Helm, el operador Trident es una gran manera de simplificar la instalación y administrar dinámicamente los recursos de Trident. Incluso puedes "Personalice la implementación del operador de Trident" usando los atributos en el `TridentOrchestrator` recurso personalizado (CR).

Algunas de las ventajas de usar el operador Trident son:

Creación de objetos de Trident.

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 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 Trident y toma medidas activas para solucionar problemas, como cuando se elimina el despliegue o se modifica accidentalmente. Se crea un `trident-operator-<generated-id>` pod que asocia un `TridentOrchestrator` CR a una instalación de Trident. Esto garantiza que solo haya una instancia de Trident en el clúster y controla su configuración, asegurándose de que la instalación sea idempotente. Cuando se realizan cambios en la instalación (como eliminar el despliegue o el conjunto de nodos), el operador los identifica y los corrige individualmente.

® actualizaciones en la existente

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

Por ejemplo, considere un escenario en el que debe activar 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 `TridentOrchestrator` se actualiza, el operador procesa las actualizaciones y parches de 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 Trident y reinstalar fácilmente.

**Mejora a de Kubernetes a mano **

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

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

Uso `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 poner en marcha Trident.

Puede generar los manifiestos para los recursos de Trident. Esto incluye la puesta en marcha, el inicio de datos, la cuenta de servicio y la función de clúster que crea 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 Trident. Con esta versión, 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 fácil de instalar Trident y funciona en la mayoría de 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 utilice el modo estándar, el instalador de Trident:

- Obtiene las imágenes del contenedor por Internet
- Crea un inicio de nodo o una implementación, lo que aumenta la velocidad de giro de los pods de 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:

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

Seleccione el proceso según el método y el modo

Después de tomar sus decisiones, seleccione el proceso apropiado.

Método	Modo de instalación
Operador de Trident (manualmente)	"Instalación estándar" "Instalación sin conexión"
Operador Trident (Helm)	"Instalación estándar" "Instalación sin conexión"
`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 Trident. Si ha desplegado con `tridentctl`, debe utilizar la versión apropiada del `tridentctl` binario para desinstalar Trident. Del mismo modo, si está desplegando con el operador, debe editar el `TridentOrchestrator` CR y configurar `spec.uninstall=true` la desinstalación de Trident.
- Si tiene un despliegue basado en el operador que desea eliminar y utilizar en su lugar `tridentctl` para implementar Trident, primero debe editar `TridentOrchestrator` y establecer `spec.uninstall=true` para desinstalar Trident. A continuación, suprima `TridentOrchestrator` y despliegue del operador. A continuación, puede instalar utilizando `tridentctl`.
- Si tiene una implementación manual basada en operador y desea utilizar la implementación del operador Trident basada en Helm, primero debe desinstalar manualmente el operador y luego realizar la instalación de Helm. Esto permite a Helm implementar el operador Trident con las etiquetas y anotaciones necesarias. Si no lo hace, su implementación del operador Trident basada en Helm fallará con un error de validación de etiqueta y un error de validación de anotación.
- Si tienes una `tridentctl` Implementación basada en Helm: puede realizar una implementación basada en operador sin desinstalar Trident.

Otras opciones de configuración conocidas

Al instalar Trident en productos de la cartera de VMware Tanzu:

- La `--kubelet-dir` el indicador se debe establecer en la ubicación del directorio kubelet. De forma predeterminada, esta es `/var/vcap/data/kubelet`.

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

Realice la instalación mediante el operador Trident

Implemente manualmente el operador de Trident (modo estándar)

Puede implementar manualmente el operador Trident para instalar Trident. Este proceso se aplica a instalaciones en las que las imágenes de contenedor requeridas por 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 crítica sobre Trident 26.02

Debe leer la siguiente información crítica sobre Trident.

**Información crítica sobre Trident **

- Kubernetes 1.35 ya es compatible con Trident. Actualiza Trident antes de actualizar Kubernetes.
- Trident aplica estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en archivo multipath.conf.

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

Implemente manualmente el operador de Trident e instale Trident

Revisa "descripción general de la instalación" para asegurarte de que cumpliste los requisitos previos de instalación y seleccionaste la opción de instalación correcta para tu entorno.

Antes de empezar

Antes de comenzar la instalación, inicia sesión en el host Linux y verifica que está gestionando un servidor en funcionamiento, "Clúster de Kubernetes compatible" y que tienes los privilegios necesarios.

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

1. Compruebe su versión de Kubernetes:

```
kubectl version
```

2. Comprobar los privilegios de administrador de clúster:

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

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

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

Paso 1: Descargue el paquete de instalación de Trident

El paquete de instalación de Trident contiene todo lo necesario para implementar el operador Trident e instalar 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/v26.02.0/trident-
installer-26.02.0.tar.gz
tar -xf trident-installer-26.02.0.tar.gz
cd trident-installer
```

Paso 2: Cree la `TridentOrchestrator` CRD

Crea el `TridentOrchestrator` Definición de recurso personalizado (CRD). Crearás un `TridentOrchestrator` Recurso personalizado más adelante. Utilice la versión CRD YAML apropiada en `deploy/crds` Para crear el `TridentOrchestrator` CRD.

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

Paso 3: Ponga en marcha el operador de Trident

El instalador de Trident proporciona un archivo de paquete que se puede utilizar para instalar el operador y crear objetos asociados. El archivo bundle es una manera fácil de implementar el operador e instalar Trident usando 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 implementa el operador en la `trident` espacio de nombres. Si la `trident` el espacio de nombres no existe, créelo con:

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

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

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

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

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

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

Pasos

1. Crear los recursos e implementar el operador:

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

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

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

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

Paso 4: Cree el `TridentOrchestrator` E instale Trident

Ahora puede crear `TridentOrchestrator` e instalar 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
  nodePrep:
  - iscsi
Status:
  Current Installation Params:
    IPv6:                      false
    Autosupport Hostname:
    Autosupport Image:         netapp/trident-autosupport:26.02
    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:26.02.0
  Message:                  Trident installed  Namespace:
trident
  Status:                   Installed
  Version:                  v26.02.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

Compruebe la instalación

Existen varias formas de verificar su instalación.

Uso `TridentOrchestrator` estado

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

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

Uso del estado de creación de pod

Puede confirmar si la instalación de Trident ha finalizado revisando el estado de los pods creados:

```
kubectl get pods -n trident

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

Uso `tridentctl`

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

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 26.02.0        | 26.02.0        |
+----------------+----------------+
```

Implemente manualmente el operador Trident (modo sin conexión).

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

Información crítica sobre Trident

Debe leer la siguiente información crítica sobre Trident.

**Información crítica sobre Trident **

- Kubernetes 1.36 ya es compatible con Trident. Actualiza Trident antes de actualizar Kubernetes.
- Trident aplica estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en archivo multipath.conf.

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

Implemente manualmente el operador de Trident e instale Trident

Revisa "descripción general de la instalación" para asegurarte de que cumpliste los requisitos previos de instalación y seleccionaste la opción de instalación correcta para tu entorno.

Antes de empezar

Inicia sesión en el host Linux y verifica que está gestionando un servidor que funciona y "Clúster de Kubernetes compatible" y que tienes los privilegios necesarios.

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

1. Compruebe su versión de Kubernetes:

```
kubectl version
```

2. Comprobar los privilegios de administrador de clúster:

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

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

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

Paso 1: Descargue el paquete de instalación de Trident

El paquete de instalación de Trident contiene todo lo necesario para implementar el operador Trident e instalar 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/v6.0/trident-
installer-26.02.0.tar.gz
tar -xf trident-installer-26.02.0.tar.gz
cd trident-installer
```

Paso 2: Cree la `TridentOrchestrator` CRD

Crea el `TridentOrchestrator` Definición de recurso personalizado (CRD). Crearás un `TridentOrchestrator` Recursos personalizados más adelante. Utilice la versión CRD YAML apropiada 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`, actualiza `image: docker.io/netapp/trident-operator:26.02.0` para reflejar la ubicación de tu registro de imágenes. Tu "Imágenes Trident y CSI" puede estar en un registro o en registros diferentes, pero todas las imágenes CSI deben estar en el mismo registro. Por ejemplo:

- `image: <your-registry>/trident-operator:26.02.0` si tus imágenes están todas ubicadas en un registro.
- `image: <your-registry>/netapp/trident-operator:26.02.0` si tu imagen de Trident está en un registro diferente al de tus imágenes CSI.

Paso 4: Ponga en marcha el operador de Trident

El instalador de Trident proporciona un archivo de paquete que se puede utilizar para instalar el operador y crear objetos asociados. El archivo bundle es una manera fácil de implementar el operador e instalar Trident usando 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 implementa el operador en la `trident` espacio de nombres. Si la `trident` el espacio de nombres no existe, créelo con:

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

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

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

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

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

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

Pasos

1. Crear los recursos e implementar el operador:

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

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

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

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

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

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

Imágenes en un registro

```
imageRegistry: "<your-registry>"
autosupportImage: "<your-registry>/trident-autosupport:26.02"
tridentImage: "<your-registry>/trident:26.02.0"
```

Imágenes en diferentes registros

```
imageRegistry: "<your-registry>"
autosupportImage: "<your-registry>/trident-autosupport:26.02"
tridentImage: "<your-registry>/trident:26.02.0"
```

Paso 6: Cree el `TridentOrchestrator` E instale Trident

Ahora puede crear `TridentOrchestrator` e instalar 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>/trident-autosupport:26.02
  Debug:              true
  Image Registry:     <your-registry>
  Namespace:          trident
  Trident Image:      <your-registry>/trident:26.02.0
Status:
  Current Installation Params:
    IPv6:                       false
    Autosupport Hostname:
    Autosupport Image:          <your-registry>/trident-autosupport:26.02
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                      true
    Http Request Timeout:       90s
    Image Pull Secrets:
    Image Registry:       <your-registry>
    k8sTimeout:           30
    Kubelet Dir:          /var/lib/kubelet
    Log Format:           text
    Probe Port:           17546
    Silence Autosupport:  false
    Trident Image:        <your-registry>/trident:26.02.0
  Message:                Trident installed
  Namespace:              trident
  Status:                 Installed
  Version:                v26.02.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

Compruebe la instalación

Existen varias formas de verificar su instalación.

Uso `TridentOrchestrator` estado

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

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

Uso del estado de creación de pod

Puede confirmar si la instalación de Trident ha finalizado revisando el estado de los pods creados:

```
kubectl get pods -n trident

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

Uso `tridentctl`

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

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 26.02.0        | 26.02.0        |
+----------------+----------------+
```

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

Puede desplegar el operador Trident e instalar Trident con Helm. Este proceso se aplica a instalaciones en las que las imágenes de contenedor requeridas por 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 crítica sobre Trident 25.10

Debe leer la siguiente información crítica sobre Trident.

Información crítica sobre Trident

- Kubernetes 1.35 ya es compatible con Trident. Actualiza Trident antes de actualizar Kubernetes.
- Trident aplica estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en archivo multipath.conf.

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

Despliegue el operador Trident e instale Trident usando Helm

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

Revisa "descripción general de la instalación" para asegurarte de que cumpliste los requisitos previos de instalación y seleccionaste la opción de instalación correcta para tu entorno.

Antes de empezar

Además de "requisitos previos a la implementación" tú necesitas "Versión timón 3".

Pasos

1. Añada el repositorio Helm de Trident:

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

2. Usa `helm install` y especifica un nombre para tu despliegue como en el siguiente ejemplo donde `100..0` es la versión de Trident que estás instalando.

```
helm install <name> netapp-trident/trident-operator --version 100.2602.0
--create-namespace --namespace <trident-namespace>
```

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

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

Pasar los datos de configuración durante la instalación

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

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

Por ejemplo, para cambiar el valor por defecto de `debug`, ejecuta el siguiente comando donde `100.2602.0` es la versión de Trident que estás instalando:

```
helm install <name> netapp-trident/trident-operator --version 100.2602.0
--create-namespace --namespace trident --set tridentDebug=true
```

Opciones de configuración

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

Opción	Descripción	Predeterminado
`nodeSelector`	Etiquetas de nodo para la asignación de pod	
`podAnnotations`	Anotaciones del pod	

Opción	Descripción	Predeterminado
`deploymentAnnotations`	Anotaciones de despliegue	
`tolerations`	Toleraciones para la asignación de POD	

`affinity`	Afinidad para la asignación de pod	`affinity:`
		`  nodeAffinity:`
		
		`requiredDuringSchedulingIgnoredDuringExecution:`
		`  nodeSelectorTerms:`
		`  - matchExpressions:`
		`  - key: kubernetes.io/arch`
		`  operator: In`
		`  values:`
		`  - arm64`
		`  - amd64`
		`  - key: kubernetes.io/os`
		`  operator: In`
		`  values:`
		`  - linux`
		
		No elimine la afinidad predeterminada del archivo values.yaml. Cuando desee proporcionar una afinidad personalizada, amplíe la afinidad predeterminada.

`tridentControllerPluginNodeSelector`	Selectores de nodos adicionales para POD. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`tridentControllerPluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`tridentNodePluginNodeSelector`	Selectores de nodos adicionales para POD. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	

Opción	Descripción	Predeterminado
`tridentNodePluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte Descripción de los pods de la controladora y los pods de nodo para obtener más detalles.	
`imageRegistry`	Identifica el registro de `trident-operator` , `trident` y otras imágenes. Déjelo vacío para aceptar el valor predeterminado. IMPORTANTE: Al instalar Trident en un repositorio privado, si está utilizando el `imageRegistry` conmutador para especificar la ubicación del repositorio, no lo utilice `/netapp/` en la ruta de acceso del repositorio.	`""`
`imagePullPolicy`	Establece la política de extracción de imágenes para el `trident-operator`.	`IfNotPresent`
`imagePullSecrets`	Establece los secretos de extracción de imágenes para el `trident-operator`, `trident`, y otras imágenes.	
`kubeletDir`	Permite anular la ubicación del host del estado interno de kubelet.	`"/var/lib/kubelet"`
`operatorLogLevel`	Permite establecer el nivel de registro del operador Trident en: `trace`, `debug`, `info`, `warn`, `error`, o. `fatal`.	`"info"`
`operatorDebug`	Permite configurar en debug el nivel de registro del operador Trident.	`true`
`operatorImage`	Permite la sustitución completa de la imagen durante `trident-operator`.	`""`
`operatorImageTag`	Permite sobrescribir la etiqueta del `trident-operator` imagen.	`""`
`tridentIPv6`	Permite permitir que 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`

Opción	Descripción	Predeterminado
`tridentHttpRequestTimeout`	Sustituye el timeout por defecto de 90 segundos para las solicitudes HTTP, con `0s` ser una duración infinita para el timeout. No se permiten valores negativos.	`"90s"`
`tridentSilenceAutosupport`	Permite deshabilitar la generación de informes periódicos de AutoSupport de Trident.	`false`
`tridentAutosupportImageTag`	Permite sobrescribir la etiqueta de la imagen para el contenedor Trident AutoSupport.	`<version>`
`tridentAutosupportProxy`	Permite al contenedor Trident AutoSupport llamar a casa a través de un proxy HTTP.	`""`
`tridentLogFormat`	Define el formato de registro de Trident (`text`o `json`).	`"text"`
`tridentDisableAuditLog`	Deshabilita el registrador de auditorías de Trident.	`true`
`tridentLogLevel`	Permite definir el nivel de log de Trident en: `trace`, , , , , `debug`, `info` `warn` `error` O `fatal`.	`"info"`
`tridentDebug`	Permite configurar el nivel de registro de Trident a `debug` . Puede automatizar el proceso de desconexión forzada mediante la integración con el operador de comprobación del estado del nodo (NHC). Para obtener más información, consulte"Automatización de la conmutación por error de aplicaciones con estado con Trident" .	`false`
`tridentLogWorkflows`	Permite habilitar flujos de trabajo de Trident específicos para el registro de seguimiento o la supresión de registros.	`""`
`tridentLogLayers`	Permite activar capas de Trident específicas para el registro de rastreo o la supresión de registros.	`""`
`tridentImage`	Permite la sustitución completa de la imagen para Trident.	`""`
`tridentImageTag`	Permite sobrescribir la etiqueta de la imagen para Trident.	`""`

Opción	Descripción	Predeterminado
`tridentProbePort`	Permite sobrescribir el puerto predeterminado utilizado para las sondas de vida/preparación de Kubernetes.	`""`
`windows`	Permite instalar 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`	Establezca en `"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/Trident-ROLE') cuando utilice la identidad de cloud en un clúster de EKS.	`""`
`iscsiSelfHealingInterval`	El intervalo en el que se invoca la reparación automática de iSCSI.	`5m0s`
`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`
`nodePrep`	Permite a Trident preparar los nodos del clúster Kubernetes para administrar volúmenes utilizando el protocolo de almacenamiento de datos especificado. **Actualmente, `iscsi` Es el único valor admitido.** NOTA: A partir de OpenShift 4.19, la versión mínima de Trident compatible con esta función es 25.06.1.	

Opción	Descripción	Predeterminado
`enableConcurrency`	Permite operaciones simultáneas del controlador Trident para mejorar el rendimiento. **Vista previa técnica**: Esta función es experimental y actualmente admite flujos de trabajo paralelos limitados con los controladores ONTAP-NAS (solo NFS) y ONTAP-SAN (NVMe para unified ONTAP 9), además de la vista previa técnica existente para el controlador ONTAP-SAN (protocolos iSCSI y FCP en unified ONTAP 9).	falso
`k8sAPIQPS`	El límite de consultas por segundo (QPS) utilizado por el controlador mientras se comunica con el servidor de API de Kubernetes. El valor de ráfaga se establece automáticamente en función del valor QPS.	`100`; opcional

Opción	Descripción	Predeterminado

`resources`	Establece los límites de recursos y las solicitudes de Kubernetes para los pods del controlador, nodo y operador de Trident . Puedes configurar la CPU y la memoria para cada contenedor y sidecar para gestionar la asignación de recursos en Kubernetes.
	
	Para obtener más información sobre la configuración de solicitudes y límites de recursos, consulte"Gestión de recursos para pods y contenedores" .
	
	• NO cambie los nombres de ningún contenedor ni campo.
	• NO cambie la sangría: la sangría de YAML es fundamental para un análisis correcto.
		`resources:`
		`  controller:`
		`  trident-main:`
		`  requests:`
		`  cpu: 10m`
		`  memory: 80Mi`
		`  limits:`
		`  cpu:`
		`  memory:`
		`  csi-provisioner:`
		`  requests:`
		`  cpu: 2m`
		`  memory: 20Mi`
		`  limits:`
		`  cpu:`
		`  memory:`
		`  csi-attacher:`
		`  requests:`
		`  cpu: 2m`
		`  memory: 20Mi`
		`  limits:`
		`  cpu:`
		`  memory:`
		`  csi-resizer:`
		`  requests:`
		`  cpu: 3m`
		`  memory: 20Mi`
		`  limits:`
		`  cpu:`
		`  memory:`
		`  csi-snapshotter:`
		`  requests:`
		`  cpu: 2m`
		`  memory: 20Mi`
		`  limits:`
		`  cpu:`
		`  memory:`
		`  trident-autosupport:`
		`  requests:`
		`  cpu: 1m`
		`  memory: 30Mi`
		`  limits:`
		`  cpu:`
		`  memory:`
		`  node:`
		`  linux:`
		`  trident-main:`

Opción	Descripción	Predeterminado
`httpsMetrics`	Habilita HTTPS para el punto de conexión de métricas de Prometheus.	falso
`hostNetwork`	Habilita la conexión en red del host para el controlador Trident . Esto resulta útil cuando se desea separar el tráfico del frontend del tráfico del backend en una red multihome.	falso

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

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

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

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

- Los nombres de los contenedores se enumeran tal como aparecen en las especificaciones del pod.
- Los sidecares se enumeran debajo de cada contenedor principal.
- Consulta el TORC `status.currentInstallationParams` campo para ver `tridentctl` Los valores aplicados actualmente.

```
        memory: 10Mi
      limits:
        cpu:
        memory:
    windows:
      trident-main:
      requests:
        cpu: 6m
        memory: 40Mi
      limits:
        cpu:
        memory:
      node-driver-registrar:
      requests:
        cpu: 6m
        memory: 40Mi
      limits:
        cpu:
        memory:
      liveness-probe:
        requests:
          cpu: 2m
          memory: 40Mi
        limits:
          cpu:
```

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

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

Información crítica sobre Trident 26.02

Debe leer la siguiente información crítica sobre Trident.

**Información crítica sobre Trident **

- Kubernetes 1.35 ya es compatible con Trident. Actualiza Trident antes de actualizar Kubernetes.
- Trident aplica estrictamente el uso de la configuración de múltiples rutas en entornos SAN, con un valor recomendado de `find_multipaths: no` en archivo multipath.conf.

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

Despliegue el operador Trident e instale Trident usando Helm

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

Revisa "descripción general de la instalación" para asegurarte de que cumpliste los requisitos previos de instalación y seleccionaste la opción de instalación correcta para tu entorno.

Antes de empezar

Además de "requisitos previos a la implementación" tú necesitas "Versión timón 3".

> Al instalar Trident en un repositorio privado, si está utilizando el `imageRegistry` conmutador para especificar la ubicación del repositorio, no lo utilice `/netapp/` en la ruta de acceso del repositorio.

Pasos

1. Añada el repositorio Helm de Trident:

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

2. Usa `helm install` y especifica un nombre para tu deployment y la ubicación del image registry. Tu "Imágenes Trident y CSI" puede estar en un registro o en registros diferentes, pero todas las imágenes CSI deben estar en el mismo registro. En los ejemplos, `100.2602.0` es la versión de Trident que estás instalando.

Imágenes en un registro

```
helm install <name> netapp-trident/trident-operator --version
100.2602.0 --set imageRegistry=<your-registry> --create-namespace
--namespace <trident-namespace> --set nodePrep={iscsi}
```

Imágenes en diferentes registros

```
helm install <name> netapp-trident/trident-operator --version
100.2602.0 --set imageRegistry=<your-registry> --set operatorImage
=<your-registry>/trident-operator:26.02.0 --set
tridentAutosupportImage=<your-registry>/trident-autosupport:26.02
--set tridentImage=<your-registry>/trident:26.02.0 --create
-namespace --namespace <trident-namespace> --set nodePrep={iscsi}
```

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

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

Pasar los datos de configuración durante la instalación

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

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

Por ejemplo, para cambiar el valor por defecto de `debug`, ejecuta el siguiente comando donde `100.2602.0` es la versión de Trident que estás instalando:

```
helm install <name> netapp-trident/trident-operator --version 100.2602.0
--create-namespace --namespace trident --set tridentDebug=true
```

Para agregar el valor nodePrep, ejecute el siguiente comando:

```
helm install <name> netapp-trident/trident-operator --version 100.2602.0
--create-namespace --namespace trident --set nodePrep={iscsi}
```

Opciones de configuración

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

> No elimine la afinidad predeterminada del archivo values.yaml. Cuando desee proporcionar una afinidad personalizada, amplíe la afinidad predeterminada.

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	`affinity:`
		`  nodeAffinity:`
		
		`requiredDuringSchedul`
		`ingIgnoredDuringExecu`
		`tion:`
		
		`nodeSelectorTerms:`
		`  -`
		`matchExpressions:`
		`  - key:`
		`kubernetes.io/arch`
		
		`operator: In`
		`  values:`
		`  - arm64`
		`  - amd64`
		`  - key:`
		`kubernetes.io/os`
		
		`operator: In`
		`  values:`
		`  - linux`
		
		No elimine la afinidad predeterminada del archivo values.yaml. Cuando desee proporcionar una afinidad personalizada, amplíe la afinidad predeterminada.

`tridentControllerPluginNodeSelector`	Selectores de nodos adicionales para POD. Consulte "Descripción de los pods de la controladora y los pods de nodo" para obtener más detalles.	
`tridentControllerPluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte "Descripción de los pods de la controladora y los pods de nodo" para obtener más detalles.	

Opción	Descripción	Predeterminado
`tridentNodePluginNodeSelector`	Selectores de nodos adicionales para POD. Consulte "Descripción de los pods de la controladora y los pods de nodo" para obtener más detalles.	
`tridentNodePluginTolerations`	Anula la toleración de Kubernetes en pods. Consulte "Descripción de los pods de la controladora y los pods de nodo" para obtener más detalles.	
`imageRegistry`	Identifica el registro de `trident-operator` , `trident` y otras imágenes. Déjelo vacío para aceptar el valor predeterminado. IMPORTANTE: Al instalar Trident en un repositorio privado, si está utilizando el `imageRegistry` conmutador para especificar la ubicación del repositorio, no lo utilice `/netapp/` en la ruta de acceso del repositorio.	""
`imagePullPolicy`	Establece la política de extracción de imágenes para el `trident-operator`.	`IfNotPresent`
`imagePullSecrets`	Establece los secretos de extracción de imágenes para el `trident-operator`, `trident`, y otras imágenes.	
`kubeletDir`	Permite anular la ubicación del host del estado interno de kubelet.	`"/var/lib/kubelet"`
`operatorLogLevel`	Permite establecer el nivel de registro del operador Trident en: `trace`, `debug`, `info`, `warn`, `error`, o. `fatal`.	`"info"`
`operatorDebug`	Permite configurar en debug el nivel de registro del operador Trident.	`true`
`operatorImage`	Permite la sustitución completa de la imagen durante `trident-operator`.	""
`operatorImageTag`	Permite sobrescribir la etiqueta del `trident-operator` imagen.	""
`tridentIPv6`	Permite permitir que Trident funcione en clústeres de IPv6.	`false`

Opción	Descripción	Predeterminado
`tridentK8sTimeout`	Anula el tiempo de espera predeterminado de 180 segundos para la mayoría de las operaciones de API de Kubernetes (si es distinto de cero, en segundos). El `tridentK8sTimeout` El parámetro solo se aplica a la instalación de Trident .	`180`
`tridentHttpRequestTimeout`	Sustituye el timeout por defecto de 90 segundos para las solicitudes HTTP, con `0s` ser una duración infinita para el timeout. No se permiten valores negativos.	`"90s"`
`tridentSilenceAutosupport`	Permite deshabilitar la generación de informes periódicos de AutoSupport de Trident.	`false`
`tridentAutosupportImageTag`	Permite sobrescribir la etiqueta de la imagen para el contenedor Trident AutoSupport.	`<version>`
`tridentAutosupportProxy`	Permite al contenedor Trident AutoSupport llamar a casa a través de un proxy HTTP.	""
`tridentLogFormat`	Define el formato de registro de Trident (`text`o `json`).	`"text"`
`tridentDisableAuditLog`	Deshabilita el registrador de auditorías de Trident.	`true`
`tridentLogLevel`	Permite definir el nivel de log de Trident en: `trace`, , , , , `debug`, `info` `warn` `error` O `fatal`.	`"info"`
`tridentDebug`	Permite establecer el nivel de registro de Trident en `debug`.	`false`
`tridentLogWorkflows`	Permite habilitar flujos de trabajo de Trident específicos para el registro de seguimiento o la supresión de registros.	""
`tridentLogLayers`	Permite activar capas de Trident específicas para el registro de rastreo o la supresión de registros.	""
`tridentImage`	Permite la sustitución completa de la imagen para Trident.	""

Opción	Descripción	Predeterminado
`tridentImageTag`	Permite sobrescribir la etiqueta de la imagen para Trident.	""
`tridentProbePort`	Permite sobrescribir el puerto predeterminado utilizado para las sondas de vida/preparación de Kubernetes.	""
`windows`	Permite instalar Trident en el nodo de trabajo de Windows.	`false`
`enableForceDetach`	Permite activar la función de desconexión forzada. Puede automatizar el proceso de desconexión forzada mediante la integración con el operador de comprobación del estado del nodo (NHC). Para obtener más información, consulte"Automatización de la conmutación por error de aplicaciones con estado con Trident" .	`false`
`excludePodSecurityPolicy`	Excluye la política de seguridad del pod del operador de la creación.	`false`
`nodePrep`	Permite a Trident preparar los nodos del clúster de Kubernetes para gestionar volúmenes mediante el protocolo de almacenamiento de datos especificado. **Actualmente, `iscsi` es el único valor soportado.** A partir de OpenShift 4.19, la versión mínima de Trident compatible con esta función es 25.06.1.	

Opción	Descripción	Predeterminado

`resources`	Establece los límites de recursos y las solicitudes de Kubernetes para los pods del controlador, nodo y operador de Trident . Puedes configurar la CPU y la memoria para cada contenedor y sidecar para gestionar la asignación de recursos en Kubernetes.
	
	Para obtener más información sobre la configuración de solicitudes y límites de recursos, consulte"Gestión de recursos para pods y contenedores" .
	
	• NO cambie los nombres de ningún contenedor ni campo.
	• NO cambie la sangría: la sangría de YAML es fundamental para un análisis correcto.
		`resources:`
		`  controller:`
		`  trident-main:`
		`  requests:`
		`  cpu: 10m`
		`  memory: 80Mi`
		`  limits:`
		`  cpu:`
		`  memory:`
		`  csi-provisioner:`
		`  requests:`
		`  cpu: 2m`
		`  memory: 20Mi`
		`  limits:`
		`  cpu:`
		`  memory:`
		`  csi-attacher:`
		`  requests:`
		`  cpu: 2m`
		`  memory: 20Mi`
		`  limits:`
		`  cpu:`
		`  memory:`
		`  csi-resizer:`
		`  requests:`
		`  cpu: 3m`
		`  memory: 20Mi`
		`  limits:`
		`  cpu:`
		`  memory:`
		`  csi-snapshotter:`
		`  requests:`
		`  cpu: 2m`
		`  memory: 20Mi`
		`  limits:`
		`  cpu:`
		`  memory:`
		`  trident-autosupport:`
		`  requests:`
		`  cpu: 1m`
		`  memory: 30Mi`
		`  limits:`
		`  cpu:`
		`  memory:`
		`  node:`
		`  linux:`

Personalice la instalación del operador de Trident

El operador Trident le permite personalizar la instalación de Trident utilizando 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

Trident se ejecuta como un único pod de controlador y un pod de nodo en cada nodo de trabajo del clúster. El pod del nodo debe estar ejecutándose en cualquier host donde se desee potencialmente montar un volumen Trident .

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

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

Opciones de configuración

`spec.namespace` Se especifica en `TridentOrchestrator` para indicar el espacio de nombres donde se instala Trident. Este parámetro **no se puede actualizar después de instalar Trident**. Si lo intenta, el `TridentOrchestrator` estado cambia a `Failed`. La intención de Trident no es migrar entre espacios de nombres.

Esta tabla detalla `TridentOrchestrator` atributos.

Parámetro	Descripción	Predeterminado
`namespace`	Espacio de nombres para instalar Trident en	`"default"`
`debug`	Active la depuración para Trident	`false`
`enableForceDetach`	`ontap-san`, `ontap-san-economy` , `ontap-nas` , y `ontap-nas-economy` solo. Funciona con Kubernetes Non-Graceful Node Shutdown (NGNS) para brindar a los administradores del clúster la capacidad de migrar de forma segura las cargas de trabajo con volúmenes montados a nuevos nodos en caso de que un nodo deje de funcionar correctamente. Para obtener más información, consulte"Automatización de la conmutación por error de aplicaciones con estado con Trident" .	`false`
`windows`	Ajuste a. `true` Permite la instalación en nodos de trabajo de Windows.	`false`

- Para detectar se aplican límites; solo las solicitudes tienen valores predeterminados .
- Los nombres de los contenedores se enumeran tal como aparecen en las especificaciones del pod.
- Los sidecares se enumeran debajo de cada contenedor principal.
- Consulta el TORC `status.CurrentInstallationParams` campo para ver los valores aplicados actualmente.

```
    trident-main:
      requests:
        cpu: 10m
        memory: 60Mi
      limits:
        cpu:
        memory:
    autosupport:
    registrar:
      requests:
        cpu: 1m
        memory: 10Mi
      limits:
        cpu:
        memory:
  windows:
    trident-main:
      requests:
        cpu: 6m
        memory: 40Mi
      limits:
        cpu:
        memory:
    node-driver-
        memory:
  operator:
    requests:
      cpu: 10m
      memory: 40Mi
    limits:
```

Parámetro	Descripción	Predeterminado
`cloudProvider`	Establecer como `"Azure"` cuando se utilizan identidades gestionadas o una identidad de nube en un clúster de AKS. Establecer como `"AWS"` cuando se utiliza una identidad de nube en un clúster de EKS. Establecer como `"GCP"` cuando se utiliza una identidad de nube en un clúster de GKE.	`""`
`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/Trident-ROLE') cuando utilice la identidad de cloud en un clúster de EKS. Se establece en la identidad de la nube («iam.gke.io/gcp-service-account: xxxx@mygcpproject.iam.gserviceaccount.com») cuando se utiliza la identidad de la nube en un clúster de GKE.	`""`
`IPv6`	Instale Trident sobre IPv6	falso
`k8sTimeout`	Tiempo de espera para las operaciones de Kubernetes. El `k8sTimeout` El parámetro solo se aplica a la instalación de Trident .	`180sec`
`silenceAutosupport`	No envíe paquetes AutoSupport a NetApp automáticamente	`false`
`autosupportImage`	La imagen contenedora para telemetría AutoSupport	`"netapp/trident-autosupport10"`
`autosupportProxy`	La dirección/puerto de un proxy para enviar AutoSupport Telemetría	`"http://proxy.example.com:8888"`
`uninstall`	Indicador utilizado para desinstalar Trident	`false`
`logFormat`	Formato de registro Trident que se va a utilizar [text,json]	`"text"`
`tridentImage`	Imagen Trident para instalar	`"netapp/trident:26.02"`
`imageRegistry`	Ruta de acceso al registro interno, del formato `<registry FQDN>[:port][/subpath]`	`"registry.k8s.io"`
`kubeletDir`	Ruta al directorio kubelet del host	`"/var/lib/kubelet"`
`wipeout`	Una lista de recursos que se deben eliminar para realizar una eliminación completa de Trident	

Parámetro	Descripción	Predeterminado
`imagePullSecrets`	Secretos para extraer imágenes de un registro interno	
`imagePullPolicy`	Establece la política de extracción de imágenes para el operador Trident. Valores válidos: `Always` para tirar siempre de la imagen. `IfNotPresent` para extraer la imagen solo si aún no existe en el nodo. `Never` para no tirar nunca de la imagen.	`IfNotPresent`
`controllerPluginNodeSelector`	Selectores de nodos adicionales para POD. Sigue el mismo formato que `pod.spec.nodeSelector`.	Sin valores predeterminados; opcional
`controllerPluginTolerations`	Anula la toleración de Kubernetes en pods. Sigue el mismo formato que `pod.spec.Tolerations`.	Sin valores predeterminados; opcional
`nodePluginNodeSelector`	Selectores de nodos adicionales para POD. Sigue el mismo formato que `pod.spec.nodeSelector`.	Sin valores predeterminados; opcional
`nodePluginTolerations`	Anula la toleración de Kubernetes en pods. Sigue el mismo formato que `pod.spec.Tolerations`.	Sin valores predeterminados; opcional
`nodePrep`	Permite a Trident preparar los nodos del clúster de Kubernetes para gestionar volúmenes mediante el protocolo de almacenamiento de datos especificado. **Actualmente, `iscsi` es el único valor soportado.** A partir de OpenShift 4.19, la versión mínima de Trident compatible con esta función es 25.06.1.	
`k8sAPIQPS`	El límite de consultas por segundo (QPS) utilizado por el controlador mientras se comunica con el servidor de API de Kubernetes. El valor de ráfaga se establece automáticamente en función del valor QPS.	`100`; opcional
`enableConcurrency`	Permite operaciones simultáneas del controlador Trident para mejorar el rendimiento. **Vista previa técnica**: Esta función es experimental y actualmente admite flujos de trabajo paralelos limitados con los controladores ONTAP-NAS (solo NFS) y ONTAP-SAN (NVMe para unified ONTAP 9), además de la vista previa técnica existente para el controlador ONTAP-SAN (protocolos iSCSI y FCP en unified ONTAP 9).	falso

Parámetro	Descripción	Predeterminado

`resources`	Establece los límites de recursos y las solicitudes de Kubernetes para el controlador Trident y los pods de nodos. Puedes configurar la CPU y la memoria para cada contenedor y sidecar para gestionar la asignación de recursos en Kubernetes.
	
	Para obtener más información sobre la configuración de solicitudes y límites de recursos, consulte"Gestión de recursos para pods y contenedores" .
	
	• NO cambie los nombres de ningún contenedor ni campo.
	• NO cambie la sangría: la sangría de YAML es fundamental para un análisis correcto.
	
	• Por defecto no se aplican límites; solo las solicitudes tienen valores predeterminados y se aplican automáticamente si no se especifican.
	• Los nombres de los contenedores se enumeran tal como aparecen en las especificaciones del pod.
	• Los sidecares se enumeran debajo de cada contenedor principal.
	• Consulta el TORC `status.CurrentInstallationParams` campo para ver los valores aplicados actualmente.
		<pre>resources:
		  controller:
		    trident-main:
		      requests:
		        cpu: 10m
		        memory: 80Mi
		      limits:
		        cpu:
		        memory:
		    csi-provisioner:
		      requests:
		        cpu: 2m
		        memory: 20Mi
		      limits:
		        cpu:
		        memory:
		    csi-attacher:
		      requests:
		        cpu: 2m
		        memory: 20Mi
		      limits:
		        cpu:
		        memory:
		    csi-resizer:
		      requests:
		        cpu: 3m
		        memory: 20Mi
		      limits:
		        cpu:
		        memory:
		    csi-snapshotter:
		      requests:
		        cpu: 2m
		        memory: 20Mi
		      limits:</pre>

Parámetro	Descripción	Predeterminado
`httpsMetrics`	Habilita HTTPS para el punto de conexión de métricas de Prometheus.	falso
`hostNetwork`	Habilita la conexión en red del host para el controlador Trident . Esto resulta útil cuando se desea separar el tráfico del frontend del tráfico del backend en una red multihome.	falso

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

Configuraciones de ejemplo

Puedes usar los atributos enOpciones de configuración al definir `TridentOrchestrator` para personalizar su instalación.

Configuración personalizada básica

Este ejemplo, creado después de ejecutar el `cat deploy/crds/tridentorchestrator_cr_imagepullsecrets.yaml` Este comando representa una instalación personalizada básica:

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

selectores de nodos

Este ejemplo instala 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
```

memory:
liveness-

nodos de trabajo de Windows

Este ejemplo, creado después de ejecutar el `cat deploy/crds/tridentorchestrator_cr.yaml` Este comando instala Trident en un nodo de trabajo de Windows.

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

Identidades administradas en un clúster de AKS

Este ejemplo instala Trident para habilitar identidades administradas en un clúster de AKS.

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

Identidad en la nube en un clúster AKS

Este ejemplo instala Trident para su uso con una identidad en la nube 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 en la nube en un clúster EKS

Este ejemplo instala Trident para su uso con una identidad en la nube 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/trident-role'"
```

Identidad de nube para GKE

Este ejemplo instala Trident para su uso con una identidad en la nube en un clúster de GKE.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcp-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: '012345678901'
  network: gcnv-network
  location: us-west2
  serviceLevel: Premium
  storagePool: pool-premium1
```

Configuración de solicitudes y límites de recursos de Kubernetes para el controlador Trident y los pods de nodos Linux de Trident

Este ejemplo configura las solicitudes y los límites de recursos de Kubernetes para el controlador Trident y los pods de nodos Linux de Trident .

> **Aviso legal**: Los valores de solicitud y límite proporcionados en este ejemplo son solo para fines demostrativos. Ajuste estos valores según las necesidades de su entorno y carga de trabajo.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullSecrets:
  - thisisasecret
  resources:
    controller:
      trident-main:
        requests:
          cpu: 10m
          memory: 80Mi
        limits:
          cpu: 200m
          memory: 256Mi
      # sidecars
      csi-provisioner:
        requests:
          cpu: 2m
          memory: 20Mi
        limits:
          cpu: 100m
          memory: 64Mi
      csi-attacher:
        requests:
          cpu: 2m
          memory: 20Mi
        limits:
          cpu: 100m
          memory: 64Mi
      csi-resizer:
        requests:
          cpu: 3m
```

```
          memory: 20Mi
        limits:
          cpu: 100m
          memory: 64Mi
      csi-snapshotter:
        requests:
          cpu: 2m
          memory: 20Mi
        limits:
          cpu: 100m
          memory: 64Mi
      trident-autosupport:
        requests:
          cpu: 1m
          memory: 30Mi
        limits:
          cpu: 50m
          memory: 128Mi
    node:
      linux:
        trident-main:
          requests:
            cpu: 10m
            memory: 60Mi
          limits:
            cpu: 200m
            memory: 256Mi
        # sidecars
        node-driver-registrar:
          requests:
            cpu: 1m
            memory: 10Mi
          limits:
            cpu: 50m
            memory: 32Mi
```

Configuración de solicitudes y límites de recursos de Kubernetes para el controlador Trident y los pods de nodos Trident de Windows y Linux

Este ejemplo configura las solicitudes y los límites de recursos de Kubernetes para el controlador Trident y los pods de nodos Trident de Windows y Linux.

> **Aviso legal**: Los valores de solicitud y límite proporcionados en este ejemplo son solo para fines demostrativos. Ajuste estos valores según las necesidades de su entorno y carga de trabajo.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullSecrets:
  - thisisasecret
  windows: true
  resources:
    controller:
      trident-main:
        requests:
          cpu: 10m
          memory: 80Mi
        limits:
          cpu: 200m
          memory: 256Mi
        # sidecars
      csi-provisioner:
        requests:
          cpu: 2m
          memory: 20Mi
        limits:
          cpu: 100m
          memory: 64Mi
      csi-attacher:
        requests:
          cpu: 2m
          memory: 20Mi
        limits:
          cpu: 100m
          memory: 64Mi
      csi-resizer:
        requests:
```

```
          cpu: 3m
          memory: 20Mi
        limits:
          cpu: 100m
          memory: 64Mi
      csi-snapshotter:
        requests:
          cpu: 2m
          memory: 20Mi
        limits:
          cpu: 100m
          memory: 64Mi
      trident-autosupport:
        requests:
          cpu: 1m
          memory: 30Mi
        limits:
          cpu: 50m
          memory: 128Mi
    node:
      linux:
        trident-main:
          requests:
            cpu: 10m
            memory: 60Mi
          limits:
            cpu: 200m
            memory: 256Mi
        # sidecars
        node-driver-registrar:
          requests:
            cpu: 1m
            memory: 10Mi
          limits:
            cpu: 50m
            memory: 32Mi
      windows:
        trident-main:
          requests:
            cpu: 6m
            memory: 40Mi
          limits:
            cpu: 200m
            memory: 128Mi
        # sidecars
        node-driver-registrar:
```

```
          requests:
            cpu: 6m
            memory: 40Mi
          limits:
            cpu: 100m
            memory: 128Mi
        liveness-probe:
          requests:
            cpu: 2m
            memory: 40Mi
          limits:
            cpu: 50m
            memory: 64Mi
```

Instale utilizando trimentctl

Instale utilizando trimentctl

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

Información crítica sobre Trident10

Debe leer la siguiente información crítica sobre Trident.

Información crítica sobre Trident

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

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

Instalar Trident mediante `tridentctl`

Revisa "descripción general de la instalación" para asegurarte de que cumpliste los requisitos previos de instalación y seleccionaste la opción de instalación correcta para tu entorno.

Antes de empezar

Antes de comenzar la instalación, inicia sesión en el host Linux y verifica que está gestionando un servidor en funcionamiento, "Clúster de Kubernetes compatible" y que tienes los privilegios necesarios.

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

1. Compruebe su versión de Kubernetes:

```
kubectl version
```

2. Comprobar los privilegios de administrador de clúster:

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

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

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

Paso 1: Descargue el paquete de instalación de Trident

El paquete del instalador de Trident crea un pod Trident, configura los objetos CRD que se utilizan para mantener su estado e inicializa los sidecars CSI para realizar acciones como el aprovisionamiento y la conexión de 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 de Trident seleccionada.

```
wget https://github.com/NetApp/trident/releases/download/v26.02.0/trident-
installer-26.02.0.tar.gz
tar -xf trident-installer-26.02.0.tar.gz
cd trident-installer
```

Paso 2: Instale Trident

Instale 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:26.02 --trident
-image <your-registry>/trident:26.02.0
```

Imágenes en diferentes registros

```
./tridentctl install -n trident --image-registry <your-registry>
--autosupport-image <your-registry>/trident-autosupport:26.02 --trident
-image <your-registry>/trident:26.02.0
```

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=26.10.0
INFO Trident installation succeeded.
....
```

Compruebe la instalación

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

Uso del estado de creación de pod

Puede confirmar si la instalación de Trident ha finalizado revisando 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. Usa `-d` para "activa el modo de depuración" y solucionar el problema.

Uso `tridentctl`

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

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 26.02.0        | 26.02.0        |
+----------------+----------------+
```

Configuraciones de ejemplo

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

Nodos de Windows

Para habilitar Trident para que se ejecute en los nodos de Windows:

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

Forzar separación

Para obtener más información, consulte"Automatización de la conmutación por error de aplicaciones con estado con Trident" .

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

Habilitar operaciones simultáneas del controlador Trident

Para habilitar operaciones simultáneas del controlador Trident para un mejor rendimiento, agregue el `--enable-concurrency` opción durante la instalación como se muestra en este ejemplo.

> **Vista previa técnica**: Esta función es experimental y actualmente admite flujos de trabajo paralelos limitados con los controladores ONTAP-NAS (solo NFS) y ONTAP-SAN (NVMe para unified ONTAP 9), además de la vista previa técnica existente para el controlador ONTAP-SAN (protocolos iSCSI y FCP en unified ONTAP 9).

```
tridentctl install --enable-concurrency -n trident
```

Personalice la instalación trimentctl

Puede usar el instalador de Trident para personalizar la instalación.

Obtenga más información sobre el instalador

El instalador de 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]`.

> Al instalar Trident en un repositorio privado, si está utilizando el `--image-registry` conmutador para especificar la ubicación del repositorio, no lo utilice `/netapp/` en la ruta de acceso del repositorio. Por ejemplo: `./tridentctl install --image-registry <image-registry> -n <namespace>`

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

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

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

- `trident-resourcequota.yaml*`

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

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

Realice la instalación mediante el operador certificado de OpenShift

Instale Trident mediante OpenShift OperatorHub

Si utiliza Red Hat OpenShift, puede instalar NetApp Trident con el operador certificado de Red Hat. Utilice este procedimiento para instalar Trident desde Red Hat OpenShift Container Platform.

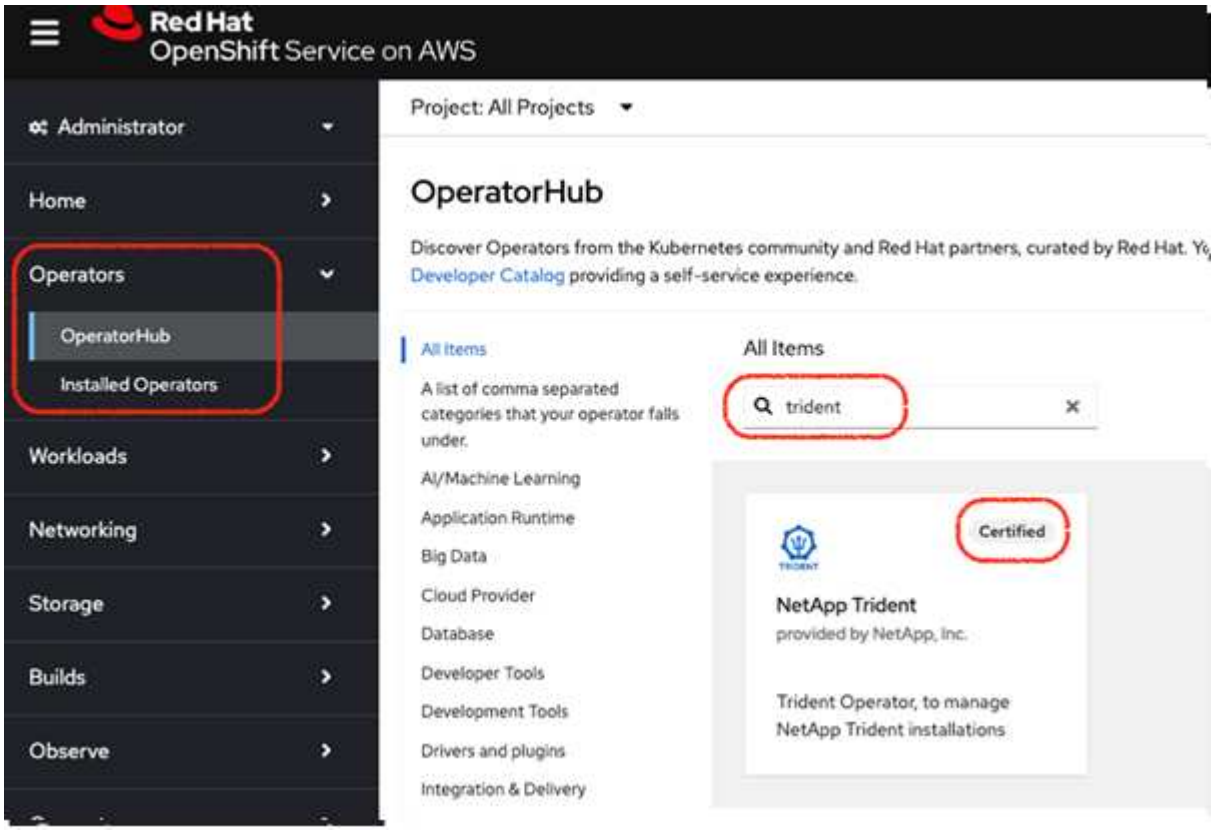
Antes de empezar

Antes de comenzar la instalación, "Preparar el entorno para la instalación de Trident".

Busque e instale el operador Trident

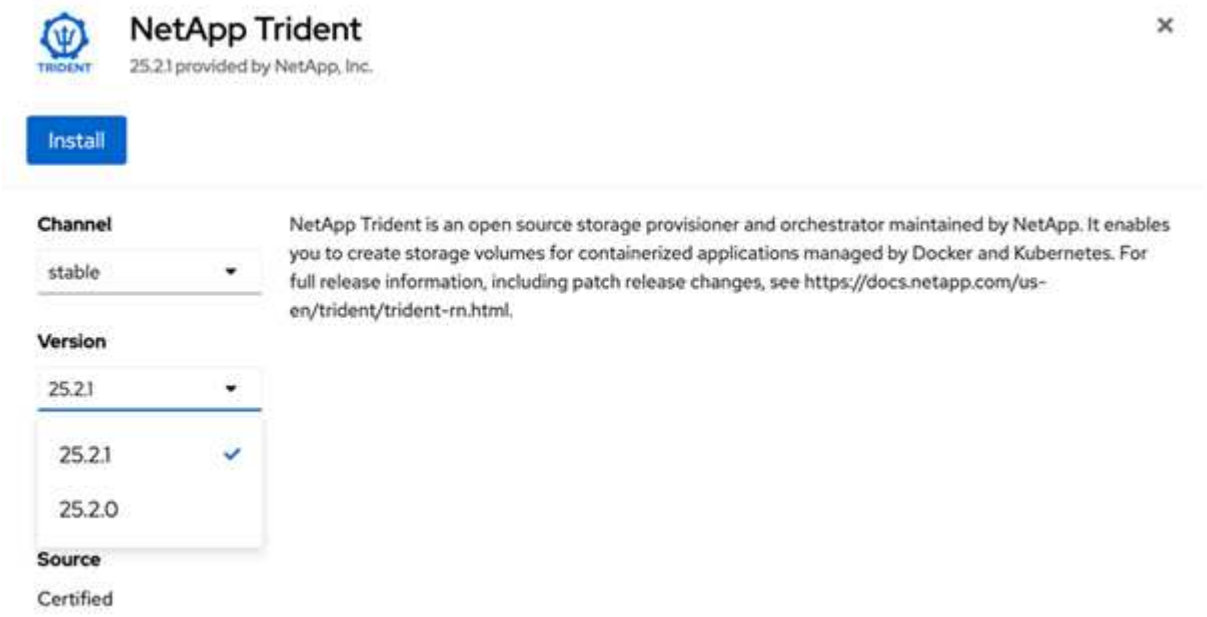
Pasos

1. Navegue hasta OpenShift OperatorHub y busque NetApp Trident.



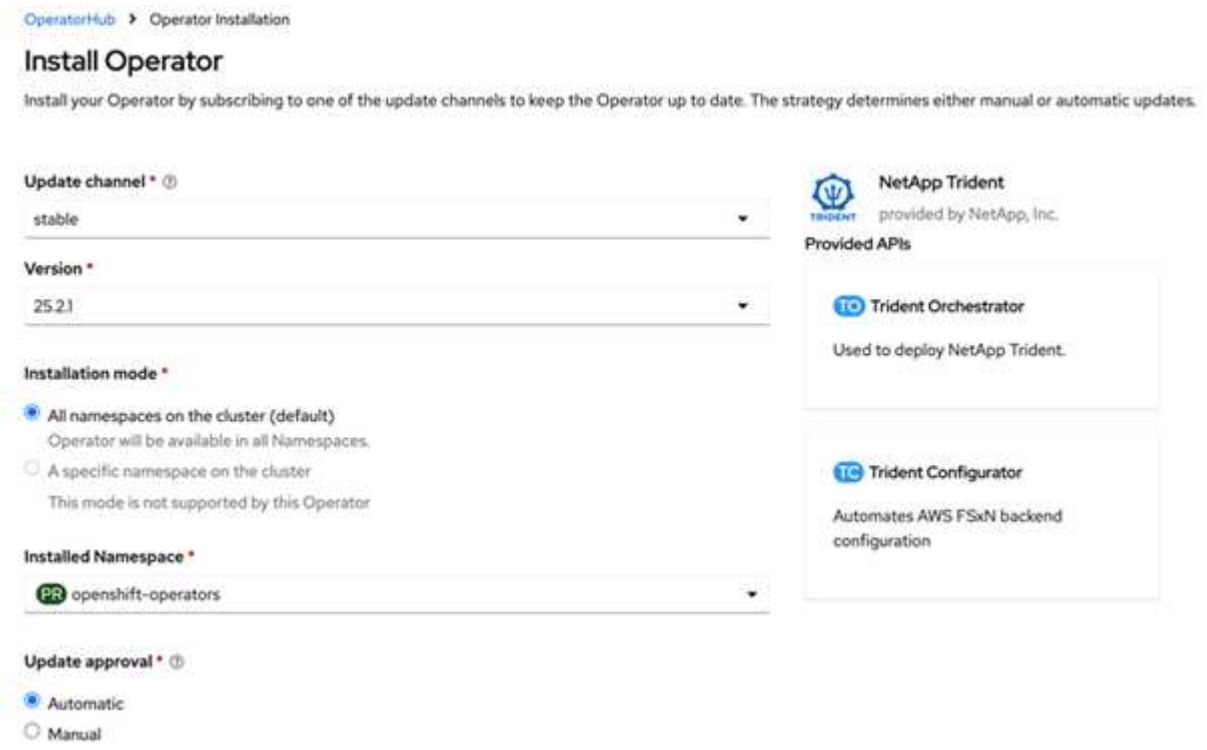
2. Haga clic en * NetApp Trident* para abrir la configuración de instalación.

3. Seleccione las opciones requeridas y haga clic en **Instalar** para abrir la configuración del Operador.

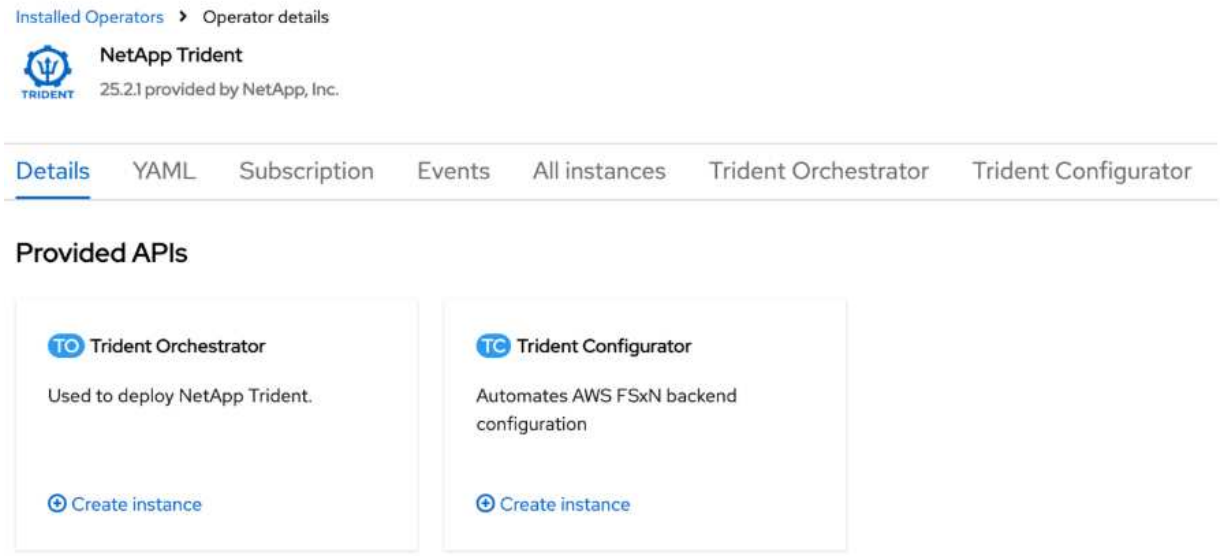


Asegúrese de seleccionar la versión más reciente del operador.

4. Mantenga todos los parámetros como están y haga clic en **Instalar**.



5. Haga clic en **Ver operador** para ver los detalles del operador.



6. Haga clic en **Vista YAML** y pegue lo siguiente en el formulario:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
  namespace: openshift-operators
spec:
  IPv6: false
  debug: false
  nodePrep:
  - iscsi
  imageRegistry: ''
  k8sTimeout: 180s
  namespace: trident
  silenceAutosupport: false
```

La interfaz de usuario proporciona un ejemplo por defecto. Puedes editarlo directamente en lugar de copiar una configuración completa.

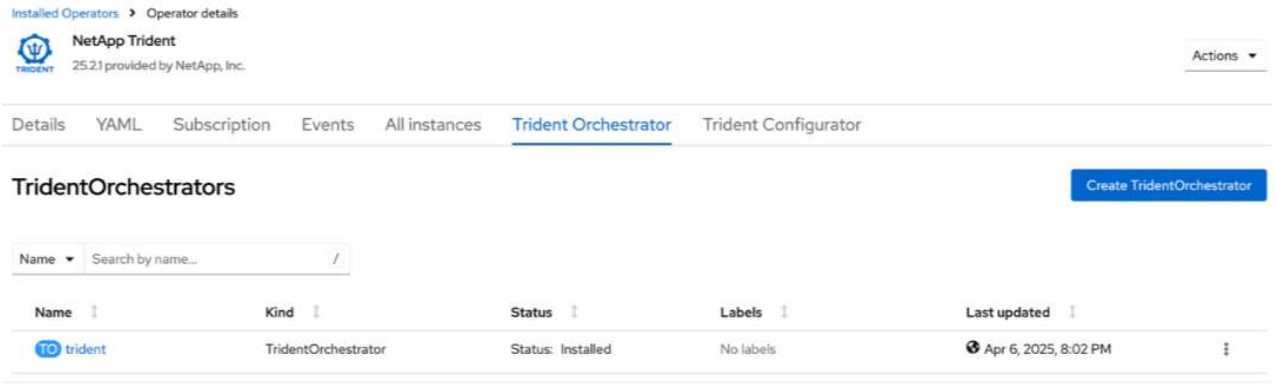
Opcional: activar la concurrencia

Para activar la concurrencia, agrega el siguiente campo a la especificación:

```
enableConcurrency: true
```

- Red Hat Enterprise Linux CoreOS (RHCOS) no tiene iSCSI habilitado ni configurado.
- Puedes agregar el `nodePrep` parámetro para configurar y habilitar los servicios iSCSI y Multipath en todos los nodos de trabajo de OpenShift.
- A partir de OpenShift 4.19, la versión mínima de Trident compatible con esta función es 25.06.1.

1. Haga clic en **Crear**; el Trident Orchestrator se instalará completamente.



Desinstalar el operador Trident

Pasos

1. Seleccione el operador Trident de la lista de operadores instalados.
2. Seleccione esta opción si desea suprimir toda la instancia de operando del operador.

Si no selecciona la casilla de verificación **Eliminar todas las instancias de operando de este operador**, Trident no se desinstalará.

3. Haga clic en **Desinstalar**.

Cambia al operador Trident certificado OpenShift

Puedes cambiar al operador Trident certificado de Red Hat OpenShift desde el operador de la comunidad, una instalación basada en Helm o un operador desplegado manualmente. El proceso para cada método implica desinstalar el operador existente y luego instalar el operador certificado usando el OperatorHub.

Antes de empezar

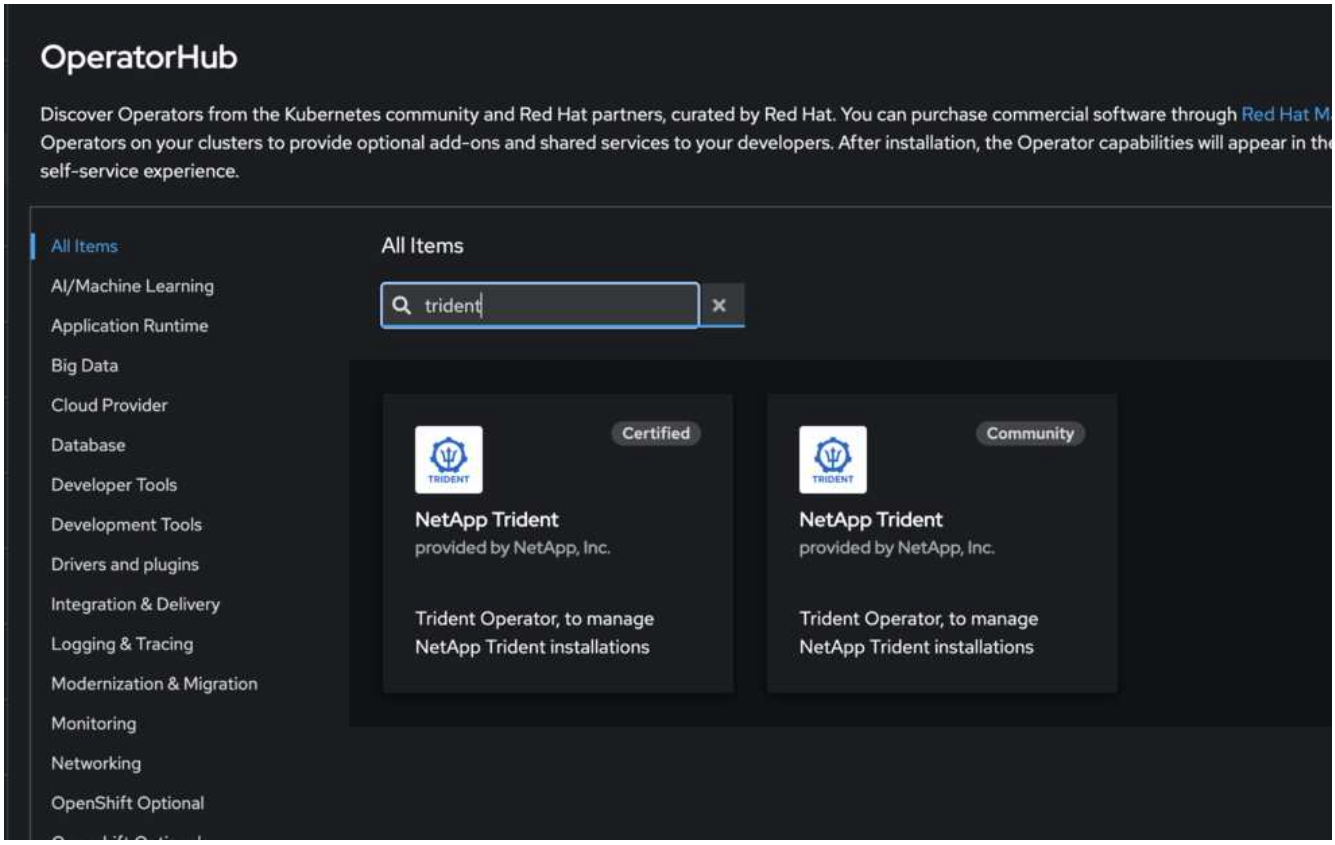
Antes de comenzar la instalación, "Preparar el entorno para la instalación de Trident".

No elimines el `TridentOrchestrator` recurso personalizado (CR) durante el proceso de desinstalación. El `TridentOrchestrator` CR conserva tu configuración de backend y clase de almacenamiento, que se necesita después de instalar el operador certificado.

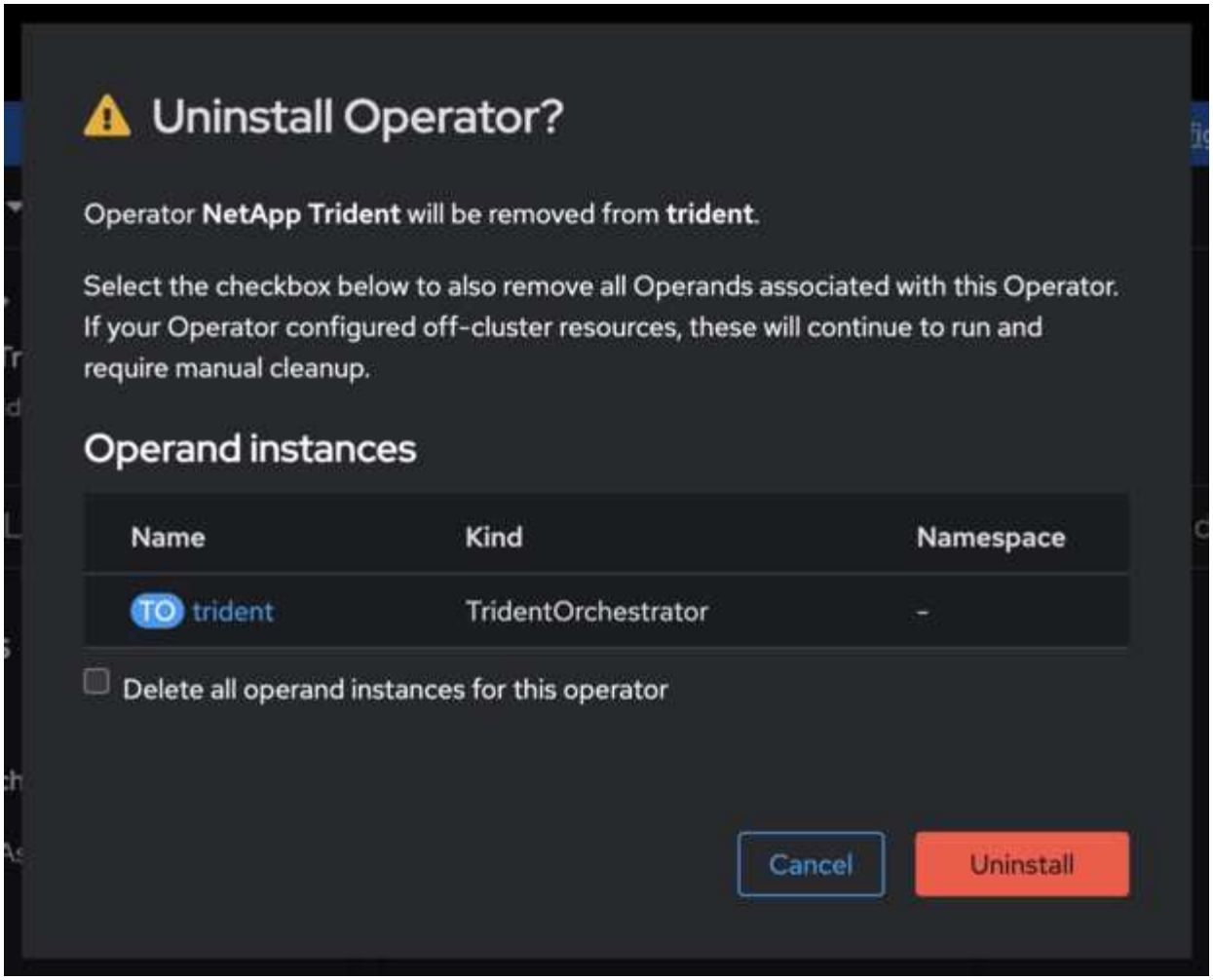
Cambia desde el operador comunitario

Pasos

1. Utilice la consola de OpenShift para navegar hasta OperatorHub.



2. Busque al operador de la comunidad NetApp Trident.



No seleccione **Eliminar todas las instancias de operando de este operador**.

3. Haga clic en **Desinstalar**.
4. Después de que termine la desinstalación, sigue con Instala el operador certificado OpenShift.

Cambia desde una instalación de operador basada en Helm

Pasos

1. Muestra la versión de Helm para tu instalación de Trident:

```
helm ls -n trident
```

2. Desinstala la versión de Helm:

```
helm uninstall <release-name> -n trident
```

3. Después de que termine la desinstalación, sigue con Instala el operador certificado OpenShift.

Cambiar desde un operador desplegado manualmente

Si instalaste Trident desplegando manualmente el operador usando un `bundle.yaml` del paquete de instalación, elimínalo borrando el mismo manifiesto.

Pasos

1. Elimina el despliegue del operador usando el manifiesto del bundle:

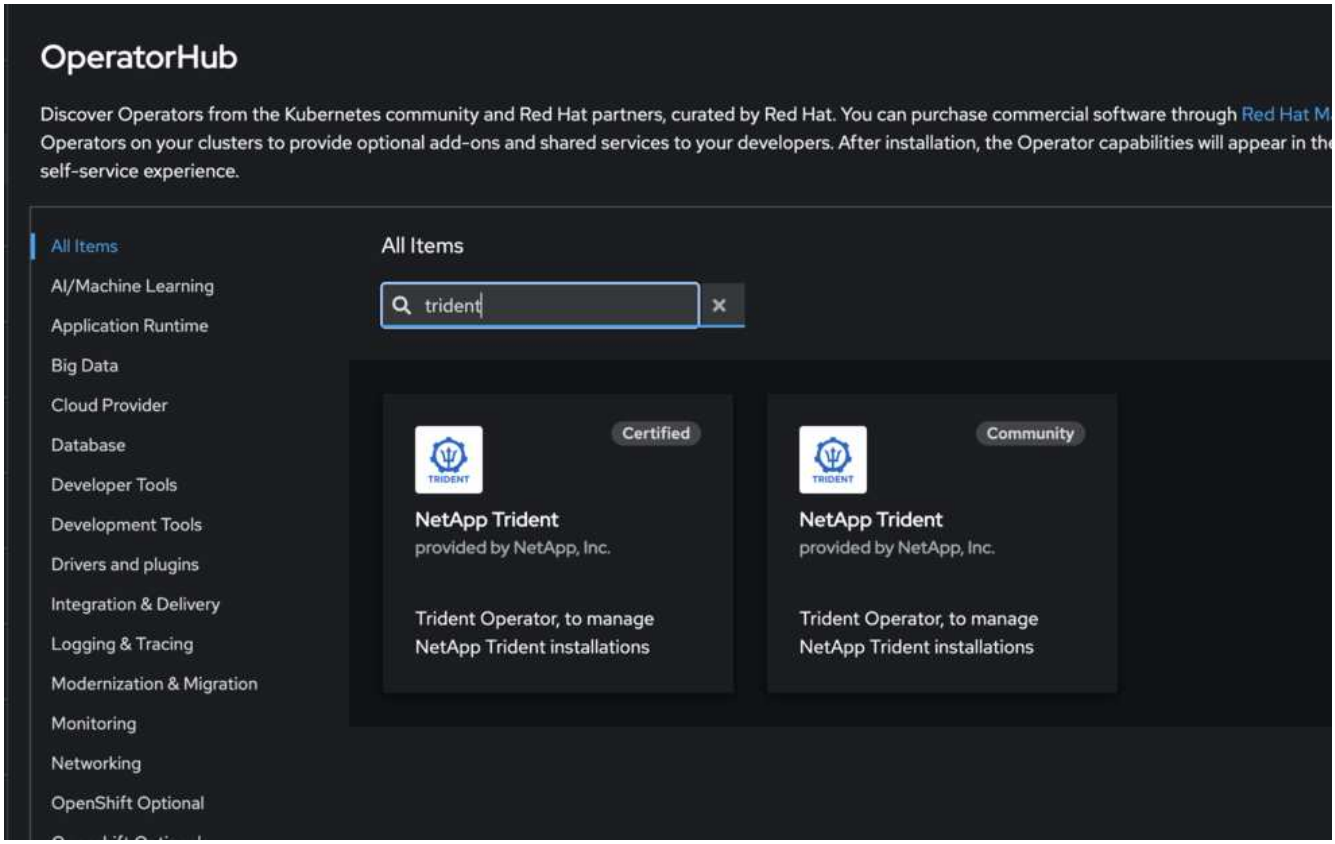
```
kubectl delete -f deploy/bundle.yaml -n trident
```

2. Después de que termine la desinstalación, sigue con Instala el operador certificado OpenShift.

Instala el operador certificado OpenShift

Pasos

1. Vaya a Red Hat OperatorHub.
2. Busque y seleccione el operador de NetApp Trident.



3. Siga las instrucciones que aparecen en pantalla para instalar el operador.

Verificación

- Compruebe el OperatorHub en la consola para asegurarse de que el nuevo operador certificado se ha instalado correctamente.

Utilizar 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 herramientas NFS, iSCSI, NVMe/TCP o FC 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 Red Hat Enterprise Linux CoreOS (RHCOS) tienen las herramientas instaladas de forma predeterminada.

Herramientas de NFS

"Instale las herramientas NFS"Si estás usando: `ontap-nas` , `ontap-nas-economy` , `ontap-nas-flexgroup` , o `azure-netapp-files` .

Herramientas iSCSI

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

Herramientas de NVMe

"Instale las herramientas NVMe" si está utilizando `ontap-san` Para el protocolo de memoria no volátil rápida (NVMe) sobre TCP (NVMe/TCP).

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

Herramientas de SCSI sobre FC

Consulte "Formas de configurar hosts de SAN FC FC-NVMe"para obtener más información sobre cómo configurar los hosts SAN FC y FC-NVMe.

"Instale las herramientas FC" Si utiliza `ontap-san` con sanType `fcp` (SCSI sobre FC).

Puntos a tener en cuenta: * SCSI sobre FC es compatible con los entornos OpenShift y KubeVirt. * SCSI sobre FC no es compatible con Docker. * La reparación automática de iSCSI no es aplicable a SCSI a través de FC.

Herramientas SMB

"Prepárese para aprovisionar los volúmenes de SMB" Si estás usando: `ontap-nas` para aprovisionar volúmenes SMB.

Detección del servicio de nodos

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

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

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

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

Volúmenes de NFS

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

RHEL 8 O POSTERIOR

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

Ubuntu

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

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

Volúmenes iSCSI

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

Funcionalidades de reparación automática de iSCSI

En el caso de los sistemas ONTAP, 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. Trident determina las prioridades de reparación y cuándo se deben adelantar 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 haberse establecido `debug` en «true» durante la instalación de Trident.

Las funcionalidades de reparación automática de iSCSI de Trident pueden ayudar a evitar lo siguiente:

- Sesiones iSCSI obsoletas o poco saludables que podrían producirse después de un problema de conectividad de red. En el caso de una sesión obsoleta, 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 Linux compatible con RHEL, con `solidfire-san` Controlador y Element OS 12.5 o anterior, asegúrese de que el algoritmo de autenticación CHAP esté establecido en MD5 in `/etc/iscsi/iscsid.conf`. Los algoritmos CHAP SHA1, SHA-256 y SHA3-256 compatibles con FIPS están disponibles con Element 12.7.

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

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

RHEL 8 O POSTERIOR

1. Instale los siguientes paquetes del sistema:

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

2. Compruebe que la versión de iscsi-initiator-utils sea 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 así sea `iscsid` y.. `multipathd` están en ejecución:

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

6. Activar e iniciar `iscsi`:

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

Ubuntu

1. Instale los siguientes paquetes del sistema:

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

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

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

3. Configure el escaneo en manual:

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

4. Activar accesos múltiples:

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

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

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

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

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

Configure o deshabilite la reparación automática de iSCSI

Es posible configurar los siguientes ajustes de reparación automática de iSCSI de Trident para corregir las 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 el `iscsiSelfHealingInterval` y.. `iscsiSelfHealingWaitTime` parámetros durante la instalación del timón o 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-100.2602.0.tgz --set
iscsiSelfHealingInterval=3m0s --set iscsiSelfHealingWaitTime=6m0s -n
trident
```

tridentctl

Para configurar o cambiar los ajustes de reparación automática de iSCSI, pase el `iscsi-self-healing-interval` y.. `iscsi-self-healing-wait-time` parámetros 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
```

Trident modifica `ctrl_device_tmo` el valor para garantizar que NVMe no se rinde en la ruta si deja de funcionar. No cambie esta configuración.

Volúmenes SCSI sobre FC

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

Requisitos previos

Configure los ajustes de nodo y red necesarios para FC.

Ajustes de red

1. Obtenga el WWPN de las interfaces de destino. Consulte "se muestra la interfaz de red" si desea obtener más información.
2. Obtenga el WWPN de las interfaces del iniciador (host).

   Consulte las utilidades del sistema operativo host correspondientes.

3. Configure la división en zonas en el switch de FC mediante WWPN del host y el destino.

   Consulte la documentación nueva del proveedor de switches para obtener más información.

   Consulte la siguiente documentación de ONTAP para obtener más detalles:

   - "Información general sobre la división en zonas de Fibre Channel y FCoE"

- "Formas de configurar hosts de SAN FC FC-NVMe"

Instale las herramientas FC

Instale las herramientas de FC mediante los comandos del sistema operativo.

- Cuando se utilicen nodos de trabajo que ejecuten RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) con VP FC, especifique `discard` mountOption en StorageClass para realizar la recuperación de espacio inline. Consulte "Documentación de Red Hat".

RHEL 8 O POSTERIOR

1. Instale los siguientes paquetes del sistema:

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

2. Activar accesos múltiples:

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

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

3. Asegúrese de que `multipathd` se está ejecutando:

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

Ubuntu

1. Instale los siguientes paquetes del sistema:

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

2. 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`.

3. Asegúrese de que `multipath-tools` está activado y en ejecución:

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

Prepárese para aprovisionar los volúmenes de SMB

Puede aprovisionar volúmenes SMB mediante `ontap-nas` conductores.

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

> `autoExportPolicy` No es compatible con los volúmenes 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. Trident admite volúmenes de SMB montados en pods que se ejecutan solo en nodos de Windows.
- Al menos un secreto 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 nodos Kubernetes que se ejecutan en Windows.

Pasos

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

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

   Puede crear recursos compartidos de administrador de SMB de una de dos formas mediante el "Consola de administración de Microsoft" Complemento carpetas compartidas o uso de la CLI de ONTAP. Para crear los recursos compartidos de SMB mediante la CLI de ONTAP:

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

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

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

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

c. Compruebe que se ha creado el recurso compartido:

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

Consulte "Cree un recurso compartido de SMB" para obtener todos los detalles.

2. Al crear el back-end, debe configurar lo siguiente para especificar volúmenes de SMB. Para obtener información sobre todas las opciones de configuración del entorno de administración de ONTAP, consulte "Opciones y ejemplos de configuración de FSX para ONTAP".

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

Configurar y gestionar back-ends

Configurar los back-ends

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

Trident ofrece automáticamente pools de almacenamiento de back-ends que coincidan con 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"
- "Configura un back-end de Google Cloud NetApp Volumes"
- "Configure un back-end de NetApp HCI o SolidFire"

- "Configure un back-end con controladores NAS ONTAP o Cloud Volumes ONTAP"
- "Configurar un back-end con controladores SAN ONTAP o Cloud Volumes ONTAP"
- "Utiliza Trident con Amazon FSx para NetApp ONTAP"

Azure NetApp Files

Configure un back-end de Azure NetApp Files

Usa Azure NetApp Files como backend para Trident. Este backend admite volúmenes NFS y SMB. Trident admite identidades gestionadas e identidad de carga de trabajo para clústeres de Azure Kubernetes Service (AKS).

Entornos cloud de Azure compatibles

Trident es compatible con los backends de Azure NetApp Files en varios entornos cloud de Azure.

Las nubes de Azure compatibles incluyen:

- Azure Commercial
- Azure Government (Azure Government / MAG)

Cuando despliegues Trident o configures un backend de Azure NetApp Files, asegúrate de que Azure Resource Manager y los endpoints de autenticación coincidan con tu entorno de cloud de Azure.

Revisa el soporte del controlador de Azure NetApp Files

Trident proporciona el siguiente controlador de almacenamiento Azure NetApp Files.

Los modos de acceso admitidos incluyen *ReadWriteOnce* (RWO), *ReadOnlyMany* (ROX), *ReadWriteMany* (RWX) y *ReadWriteOncePod* (RWOP).

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

Revisa las consideraciones

- Azure NetApp Files no admite volúmenes menores de 50 GiB. Trident crea un volumen de 50 GiB cuando se solicita un volumen menor.
- Trident admite volúmenes de SMB montados en pods que se ejecutan solo en nodos de Windows.
- Las implementaciones de Azure NetApp Files en nubes Azure no comerciales requieren puntos finales de Azure Resource Manager y de autenticación específicos para cada nube. Asegúrate de que Trident y cualquier configuración de backend usen los puntos finales apropiados para tu entorno de nube Azure.

Usa identidades gestionadas para AKS

Trident es compatible con "identidades administradas" para clústeres AKS.

Si usas `tridentctl` para crear o gestionar backends de Azure NetApp Files, asegúrate de que esté

configurado para el entorno de cloud de Azure correcto.

Para usar identidades gestionadas, debes tener:

- Un clúster de Kubernetes puesto en marcha mediante AKS
- Identidades gestionadas configuradas en el clúster de Kubernetes de AKS
- Trident instalado con `cloudProvider` configurado en `"Azure"`

Operador de Trident

Edita `tridentorchestrator_cr.yaml` y configura `cloudProvider` en `"Azure"`.

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

Timón

El siguiente ejemplo instala Trident y configura `cloudProvider` usando la variable de entorno `$CP`:

```
helm install trident trident-operator-100.2602.0.tgz --create-namespace
--namespace <trident-namespace> --set cloudProvider=$CP
```

<code>tridentctl</code>

El siguiente ejemplo instala Trident y establece la `cloud-provider` bandera en `Azure`:

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

Usa la identidad de carga de trabajo para AKS

La identidad de carga de trabajo permite que los pods de Kubernetes accedan a los recursos de Azure autenticándose como una identidad de carga de trabajo.

Si usas `tridentctl` para crear o gestionar backends de Azure NetApp Files, asegúrate de que esté configurado para el entorno de cloud de Azure correcto.

Para usar la identidad de carga de trabajo, 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
- Trident instalado con `cloudProvider` configurado en `"Azure"` y `cloudIdentity` configurado en el valor de identidad de la carga de trabajo

Operador de Trident

Edita `tridentorchestrator_cr.yaml` y establece `cloudProvider` en `"Azure"`. Establece `cloudIdentity` en `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx`.

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

Timón

Establece los valores para las opciones **cloud-provider (CP)** y **cloud-identity (CI)** usando las siguientes variables de entorno:

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

El siguiente ejemplo instala Trident y establece `cloudProvider` usando `$CP` y establece `cloudIdentity` usando `$CI`:

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

<code>tridentctl</code>

Establece los valores para las banderas **cloud provider** y **cloud identity** usando las siguientes variables de entorno:

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

El siguiente ejemplo instala Trident y establece `cloud-provider` 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.

Entornos cloud de Azure compatibles

Trident es compatible con los backends de Azure NetApp Files en varios entornos cloud de Azure.

Las nubes de Azure compatibles incluyen:

- Azure Commercial
- Azure Government (Azure Government / MAG)

Cuando prepares tu entorno, asegúrate de que tu suscripción de Azure, la configuración de identidad y los recursos de Azure NetApp Files se creen en el entorno de cloud de Azure adecuado.

Requisitos previos para volúmenes NFS y SMB

Si estás usando Azure NetApp Files por primera vez o en una nueva ubicación, necesitas hacer una configuración inicial para preparar Azure NetApp Files y crear un volumen NFS. Consulta "Azure: Configure Azure NetApp Files y cree un volumen NFS".

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

> - `subscriptionID`, `tenantID`, `clientID`, `location`, y. `clientSecret` Son opcionales cuando se utilizan identidades administradas en un clúster AKS.
> - `tenantID`, `clientID`, y. `clientSecret` Son opcionales cuando se utiliza una identidad de nube en un clúster de AKS.
> - Las implementaciones de Azure NetApp Files en nubes Azure no comerciales requieren puntos finales de Azure Resource Manager y de autenticación específicos para cada nube. Asegúrate de que Trident y cualquier configuración de backend usen los puntos finales apropiados para tu entorno de nube Azure.

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

- Azure `location` que contiene al menos uno "subred delegada". A partir de Trident 22.01, la `location` parámetro es un campo obligatorio en el nivel superior del archivo de configuración del back-end. Los valores de ubicación especificados en los pools virtuales se ignoran.
- Para usar `Cloud Identity`, obtén el `client ID` desde 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. Trident admite volúmenes de SMB montados en pods que se ejecutan solo en nodos de Windows.
- Al menos un secreto 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 nodos 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

Trident utiliza tu configuración de backend (subred, red virtual, nivel de servicio y ubicación) para crear volúmenes de Azure NetApp Files en pools de capacidad que están disponibles en la ubicación solicitada y coinciden con el nivel de servicio y la subred solicitados.

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

Parámetro	Descripción	Predeterminado
`version`	Versión de configuración del backend.	Siempre 1
`storageDriverName`	Nombre del controlador de almacenamiento	"azure-netapp-files"
`backendName`	Nombre personalizado para el backend de almacenamiento	Nombre del controlador + "_" + caracteres aleatorios
`subscriptionID`	El ID de suscripción de su suscripción de Azure Opcional cuando se activan identidades gestionadas en un clúster de AKS.	
`tenantID`	El ID de inquilino de un registro de aplicación Opcional cuando se utilizan identidades gestionadas o identidad de nube en un clúster de AKS.	
`clientID`	El ID de cliente de un registro de aplicación 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 activan 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 a. `Microsoft.Netapp/volumes`	""
`networkFeatures`	Conjunto de funciones de VNet para un volumen, puede ser `Basic` o `Standard`. Network Features no está disponible en todas las regiones y podría tener que habilitarse en una suscripción. Especificar `networkFeatures` cuando la funcionalidad no está habilitada hace que falle el aprovisionamiento del volumen.	""
`nfsMountOptions`	Control detallado de las opciones de montaje de NFS. Ignorado para volúmenes SMB. Para montar volúmenes usando la versión de NFS 4.1, incluye `nfsvers=4` en la lista de opciones de montaje separadas por comas para elegir NFS v4.1. Las opciones de montaje establecidas en una definición de clase de almacenamiento reemplazan las opciones de montaje establecidas en la configuración de backend.	"nfsvers=3"
`limitVolumeSize`	No se puede aprovisionar si el tamaño del volumen solicitado es superior a este valor	"" (no se aplica de forma predeterminada)

Parámetro	Descripción	Predeterminado
`debugTraceFlags`	Indicadores de depuración que se deben usar para la solución de problemas. Ejemplo: `\{"api": false, "method": true, "discovery": true}`. No lo utilice a menos que esté solucionando problemas y necesite un volcado de registro detallado.	nulo
`nasType`	Configure la creación de volúmenes NFS o SMB. Las opciones son `nfs`, `smb` o nulo. El valor predeterminado es nulo en volúmenes de NFS.	`nfs`
`supportedTopologies`	Representa una lista de regiones y zonas soportadas por este backend. Para obtener más información, consulte "Utilice Topología CSI".	
`qosType`	Representa el tipo de QoS: Automático o Manual.	Auto
`maxThroughput`	Establece el rendimiento máximo permitido en MiB/seg. Compatible solo con grupos de capacidad de QoS manuales.	`4 MiB/sec`

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

Considera los entornos cloud de Azure (26.02)

A partir de la versión 26.02, Trident admite la creación y gestión de backends de Azure NetApp Files en varios entornos cloud de Azure.

Las nubes de Azure compatibles incluyen:

- Azure Commercial
- Azure Government (Azure Government / MAG)

Cuando despliegas Trident o creas un backend de Azure NetApp Files, asegúrate de que Azure Resource Manager y los endpoints de autenticación coincidan con tu entorno de Azure cloud. Si los endpoints no coinciden, `tridentctl` no puede autenticarse y la creación del backend falla.

Permisos y recursos necesarios

Si recibes el error "No capacity pools found" al crear un PVC, es probable que el registro de tu aplicación no tenga los permisos y recursos necesarios (subred, red virtual, capacity pool) asociados. Si la depuración está activada, Trident registra los recursos de Azure descubiertos cuando se crea el backend. Verifica que se esté usando un rol adecuado.

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

> Si la red virtual (vNet) se encuentra en un grupo de recursos diferente al de la cuenta de almacenamiento de Azure NetApp Files (ANF), especifique el grupo de recursos para la red virtual al configurar la lista de grupos de recursos para el back-end.

Los `resourceGroups`, `netappAccounts`, y `capacityPools` valores son filtros que restringen el conjunto de recursos detectados a los disponibles para este backend de almacenamiento y pueden especificarse 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

Puedes controlar el aprovisionamiento de volúmenes predeterminados especificando las siguientes opciones en una sección especial del archivo de configuración. Consulta Configuraciones de ejemplo para más detalles.

Parámetro	Descripción	Predeterminado
`exportRule`	Reglas de exportación de volúmenes nuevos. `exportRule` Debe ser una lista separada por comas con cualquier combinación de direcciones IPv4 o subredes IPv4 en notación CIDR. Ignorada para volúmenes de SMB.	"0.0.0.0/0"
`snapshotDir`	Acceso a la `.snapshot` directorio	`true`, `false` (establecer explícitamente).
`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 siguientes ejemplos muestran configuraciones básicas que dejan la mayoría de los parámetros en sus valores predeterminados. Esta es la forma más fácil de definir un backend.

Configuración mínima

Esta es la configuración mínima absoluta del backend. Con esta configuración, Trident detecta todas tus cuentas NetApp, grupos de capacidad y subredes delegadas a Azure NetApp Files en la ubicación configurada, y coloca los nuevos volúmenes en uno de esos grupos y subredes aleatoriamente. Porque `nasType` se omite, la `nfs` configuración predeterminada se aplica y el backend aprovisionará volúmenes 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 cuando se utilizan 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:
    - resource-group-1/netapp-account-1/ultra-pool
  resourceGroups:
    - resource-group-1
  netappAccounts:
    - resource-group-1/netapp-account-1
  virtualNetwork: resource-group-1/eastus-prod-vnet
  subnet: resource-group-1/eastus-prod-vnet/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 volúmenes en la `eastus` ubicación de Azure en un `Ultra` grupo de capacidad. Trident detecta automáticamente todas las subredes delegadas a Azure NetApp Files en esa ubicación y coloca un nuevo volumen en una de ellas aleatoriamente.

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

Ejemplo de backend con grupos de capacidad de QoS manuales

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

```
---
version: 1
storageDriverName: azure-netapp-files
backendName: anf1
location: eastus
labels:
  clusterName: test-cluster-1
  cloud: anf
  nasType: nfs
defaults:
  qosType: Manual
storage:
  - serviceLevel: Ultra
    labels:
      performance: gold
    defaults:
      maxThroughput: 10
  - serviceLevel: Premium
    labels:
      performance: silver
    defaults:
      maxThroughput: 5
  - serviceLevel: Standard
    labels:
      performance: bronze
    defaults:
      maxThroughput: 3
```

Configuración avanzada

Esta configuración de 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: application-group-1/eastus-prod-vnet
subnet: application-group-1/eastus-prod-vnet/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 de backend define varios grupos de almacenamiento en un solo archivo. Esto es útil cuando tienes varios grupos de capacidad que admiten diferentes niveles de servicio y quieres crear clases de almacenamiento en Kubernetes que los representen. Se usaron etiquetas de grupo virtual para diferenciar los grupos según `performance`.

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

Configuración de topologías admitidas

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 usa 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 en cada nodo del clúster de Kubernetes. Estas regiones y zonas representan la lista de valores permitidos que se pueden proporcionar en una clase de almacenamiento. Para las clases de almacenamiento que contienen un subconjunto de las regiones y zonas proporcionadas en un backend, Trident crea volúmenes en la región y zona mencionadas. Para más información, consulta "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

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

Definiciones de ejemplo mediante `parameter.selector` campo

Usando `parameter.selector` puedes especificar para cada `StorageClass`el pool virtual que se usa para alojar un volumen. El volumen tendrá los aspectos definidos en el pool elegido.

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

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

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

Definiciones de ejemplo de volúmenes SMB

Usando `nasType`, `node-stage-secret-name` y `node-stage-secret-namespace`, puedes 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 grupos que admiten volúmenes SMB. `nasType: nfs` o `nasType: null`filtros para grupos 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 usas una nube Azure no comercial, asegúrate de que `tridentctl` esté configurado para usar los endpoints de Azure Resource Manager y autenticación para tu entorno de nube Azure. Si falla la creación del backend, revisa tu configuración del backend y mira los registros para ver cuál es la causa:

```
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 Google Cloud NetApp Volumes

Puedes configurar Google Cloud NetApp Volumes como backend para que Trident aprovisione almacenamiento para cargas de trabajo de Kubernetes.

Descripción general

Trident es compatible con Google Cloud NetApp Volumes tanto para cargas de trabajo NAS (NFS y SMB) como en bloque (iSCSI).

- Las cargas de trabajo NAS utilizan el `google-cloud-netapp-volumes` backend
- Las cargas de trabajo en bloque (iSCSI) utilizan el `google-cloud-netapp-volumes-san` backend

Los volúmenes NAS proporcionan almacenamiento basado en archivos y se accede a ellos mediante protocolos NFS o SMB. Estos volúmenes admiten el acceso compartido a través de varios pods o nodos.

Los volúmenes en bloque proporcionan almacenamiento en bloque sin procesar y se accede a ellos como dispositivos iSCSI conectados a nodos Kubernetes. Estos volúmenes se utilizan cuando las aplicaciones requieren acceso a nivel de bloque.

Esto se aplica a los siguientes entornos:

- Trident 26.02 y posteriores
- Google Kubernetes Engine (GKE) o Red Hat OpenShift
- Pools de almacenamiento de Google Cloud NetApp Volumes

Para configurar el almacenamiento en bloque (iSCSI), consulta "Configura el almacenamiento en bloque (iSCSI)".

Prepárese para la configuración

La identidad en la nube permite que las cargas de trabajo de Kubernetes accedan a los recursos de Google Cloud autenticándose como una identidad de carga de trabajo en vez de usar credenciales estáticas.

Para usar la identidad en la nube con Google Cloud NetApp Volumes, debes tener:

- Un clúster de Kubernetes desplegado usando Google Kubernetes Engine (GKE)
- Identidad de carga de trabajo habilitada en el clúster GKE y el servidor de metadatos habilitado en los grupos de nodos
- Una cuenta de servicio de Google Cloud con la función de Google Cloud NetApp Volumes Admin (`roles/netapp.admin`) o una función personalizada equivalente
- Trident instalado con el cloud provider establecido en `GCP` y la cloud identity annotation configurada

Operador de Trident

Para instalar Trident utilizando el operador Trident, edita `tridentorchestrator_cr.yaml`:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  namespace: trident
  cloudProvider: "GCP"
  cloudIdentity: "iam.gke.io/gcp-service-account: cloudvolumes-admin-
sa@mygcpproject.iam.gserviceaccount.com"
```

Timón

Establece el proveedor de la nube y la identidad de la nube al instalar Trident con Helm:

```
helm install trident trident-operator-100.6.0.tgz \
  --set cloudProvider=GCP \
  --set cloudIdentity="iam.gke.io/gcp-service-account: cloudvolumes-
admin-sa@mygcpproject.iam.gserviceaccount.com"
```

tridentctl

Instala Trident especificando el proveedor de la nube y la identidad de la nube:

```
tridentctl install \
  --cloud-provider=GCP \
  --cloud-identity="iam.gke.io/gcp-service-account: cloudvolumes-admin-
sa@mygcpproject.iam.gserviceaccount.com" \
  -n trident
```

Configurar el almacenamiento NAS

> Para los grupos de almacenamiento UNIFIED de Google Cloud NetApp Volumes, Trident aplica reglas de nomenclatura y validación específicas de UNIFIED durante las operaciones de volumen.

Al localizar un volumen, Trident puede evaluar múltiples variantes de nombre de volumen compatibles (por ejemplo, formatos de guion y guion bajo) para mejorar la fiabilidad de la importación y el descubrimiento.

Detalles del controlador

Trident proporciona el `google-cloud-netapp-volumes` controlador para aprovisionar almacenamiento NAS desde Google Cloud NetApp Volumes.

El controlador admite los siguientes modos de acceso:

- ReadWriteOnce (RWO)
- ReadOnlyMany (ROX)
- ReadWriteMany (RWX)
- ReadWriteOncePod (RWOP)

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

Configura un backend NAS Trident

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: gcnv-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "<project-number>"
  location: "<region>"
  sdkTimeout: "600"
  storage:
  - labels:
      cloud: gcp
    network: "<vpc-network>"
```

Aprovisionar volúmenes NAS

Los volúmenes NAS se aprovisionan usando el `google-cloud-netapp-volumes` backend y son compatibles con los protocolos NFS y SMB.

StorageClass para volúmenes NFS

Para aprovisionar volúmenes NFS, establece `nasType` en `nfs`.

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

StorageClass para volúmenes SMB

Para aprovisionar volúmenes SMB, establece `nasType` en `smb` y proporciona las credenciales.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
allowVolumeExpansion: true
```

Ejemplo de PersistentVolumeClaim (RWX)

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

Ejemplo de PersistentVolumeClaim (RWO)

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: gcnv-nas-rwo
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Gi
  storageClassName: gcnv-nfs
```

Los volúmenes NAS usan `volumeMode: Filesystem`.

Configura Google Cloud NetApp Volumes para cargas de trabajo SAN

Puedes configurar Trident para aprovisionar volúmenes de almacenamiento en bloque usando el protocolo iSCSI desde Google Cloud NetApp Volumes. Los volúmenes SAN se aprovisionan desde grupos de almacenamiento Flex Unified usando el controlador de almacenamiento `google-cloud-netapp-volumes-san`.

Este controlador está dedicado a cargas de trabajo en bloque y no es compatible con protocolos NAS.

El `google-cloud-netapp-volumes-san` backend es necesario para aprovisionar volúmenes de bloques iSCSI. El `google-cloud-netapp-volumes` backend admite solo protocolos NAS y no puede usarse para cargas de trabajo SAN.

Descripción general

Trident es compatible con las cargas de trabajo SAN (iSCSI) de Google Cloud NetApp Volumes mediante el controlador `google-cloud-netapp-volumes-san`.

Los volúmenes SAN se aprovisionan desde grupos de almacenamiento Flex Unified y se presentan a los nodos de Kubernetes como dispositivos de bloques iSCSI.

Esto se aplica a los siguientes entornos:

- Trident 26.02 y posteriores
- Google Kubernetes Engine (GKE) o Red Hat OpenShift
- Pools de almacenamiento unificado de Google Cloud NetApp Volumes Flex
- cargas de trabajo basadas en iSCSI

Pools de almacenamiento unificado Flex

Los grupos de almacenamiento de Flex Unified proporcionan almacenamiento en bloque mediante el protocolo iSCSI y son necesarios para el aprovisionamiento de SAN:

- Se admiten los pools REGIONALES de Flex Unified.
- Las agrupaciones de Flex Unified ZONAL son compatibles a partir de Trident 26.02.1.
- Solo se admite el nivel de servicio **Flex** para cargas de trabajo SAN.

Configura un backend Trident SAN

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: gcnv-san
  namespace: trident
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes-san
  projectNumber: "<project-number>"
  location: "<region>"
  sdkTimeout: "600"
  storage:
  - labels:
      cloud: gcp
      performance: flex
    network: "<vpc-network>"
    serviceLevel: Flex
```

Crea un StorageClass

Después de configurar el backend SAN, crea un StorageClass que haga referencia al `google-cloud-netapp-volumes-san` driver.

El tipo de sistema de archivos se define en el StorageClass, no en el backend.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-san
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes-san"
  fsType: "ext4"
allowVolumeExpansion: true
```

Tipos de sistemas de archivos compatibles:

- `ext4` (predeterminado)
- `ext3`

- `xfs`

> El controlador SAN solo es compatible con el nivel de servicio Flex y no utiliza parámetros de backend específicos de NAS como `exportRule`, `unixPermissions`, `nasType`, `snapshotDir`, `nfsMountOptions` o configuraciones relacionadas con el tiering.

Provisiona volúmenes en bloque

ReadWriteOnce (RWO)

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: gcnv-san-rwo
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Gi
  storageClassName: gcnv-san
```

ReadWriteOncePod (RWOP)

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: gcnv-san-rwop
spec:
  accessModes:
    - ReadWriteOncePod
  resources:
    requests:
      storage: 100Gi
  storageClassName: gcnv-san
```

ReadOnlyMany (ROX)

Un patrón común para ROX es clonar un volumen ReadWriteOnce existente y montar el clon como de solo lectura.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: gcnv-san-rox
spec:
  accessModes:
    - ReadOnlyMany
  resources:
    requests:
      storage: 100Gi
  storageClassName: gcnv-san
  dataSource:
    kind: PersistentVolumeClaim
    name: gcnv-san-rwo
```

ReadWriteMany (RWX) — solo bloque en bruto

ReadWriteMany solo se admite cuando `volumeMode: Block`.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: gcnv-san-raw-rwx
spec:
  accessModes:
    - ReadWriteMany
  volumeMode: Block
  resources:
    requests:
      storage: 100Gi
  storageClassName: gcnv-san
```

Comportamiento del volumen de bloques

Los volúmenes de bloques se aprovisionan como LUN iSCSI y se presentan a los nodos de Kubernetes como dispositivos de bloque.

Volúmenes en bloque:

- Usa el protocolo iSCSI
- Soporte para sistema de archivos y presentación de bloques sin procesar
- Están adjuntos y gestionados por Trident
- Soporta múltiples modos de acceso a Kubernetes

Modos de acceso

Los volúmenes en bloque aprovisionados por Trident admiten los siguientes modos de acceso:

- `ReadWriteOnce` (RWO)
- `ReadOnlyMany` (ROX)
- `ReadWriteOncePod` (RWOP)
- `ReadWriteMany` (RWX), compatible solo cuando `volumeMode: Block`

volumeMode comportamiento

El `volumeMode` campo controla cómo se expone un volumen de bloque:

- `Filesystem` Trident formatea y monta el volumen.
- `Block` Trident conecta el dispositivo y lo expone como un dispositivo de bloque sin procesar.

Operaciones compatibles

Los volúmenes en bloque aprovisionados usando el controlador `google-cloud-netapp-volumes-san` admiten:

- Crear
- Borrar
- Clonar
- Snapshot
- Cambie el tamaño
- Importar

Comportamiento de la sobreaprovisionamiento de GiB extra

Google Cloud NetApp Volumes los volúmenes en bloque incluyen una sobrecarga interna de metadatos. Esta sobrecarga reduce el tamaño del dispositivo visible desde el kernel en comparación con la capacidad provisionada.

Las pruebas muestran:

- Aproximadamente 300 KiB de sobrecarga en la creación inicial
- Hasta aproximadamente 107 MiB de sobrecarga después de un redimensionamiento

Porque Google Cloud NetApp Volumes acepta solo asignaciones de GiB enteros, Trident se asegura de que el tamaño utilizable del dispositivo siempre cumpla o supere la solicitud de PVC:

- Redondeando el tamaño solicitado al siguiente GiB entero
- Agregando un búfer adicional de 1 GiB

Ejemplo:

- Solicitud de PVC: 100 GiB
- Tamaño aprovisionado en Google Cloud NetApp Volumes: 101 GiB

- Espacio útil visible para la aplicación: al menos 100 GiB

Ejemplos de pods

Volumen de bloque montado en el sistema de archivos (RWO)

```
apiVersion: v1
kind: Pod
metadata:
  name: app-rwo
spec:
  containers:
  - name: app
    image: ubuntu:22.04
    command: ["sleep", "infinity"]
    volumeMounts:
    - name: data
      mountPath: /mnt/data
  volumes:
  - name: data
    persistentVolumeClaim:
      claimName: gcnv-san-rwo
```

Dispositivo de bloque raw (RWX)

```
apiVersion: v1
kind: Pod
metadata:
  name: app-raw-rwx
spec:
  containers:
  - name: app
    image: ubuntu:22.04
    command: ["sleep", "infinity"]
    volumeDevices:
    - name: data
      devicePath: /dev/xda
  volumes:
  - name: data
    persistentVolumeClaim:
      claimName: gcnv-san-raw-rwx
```

Comportamiento de attach y mount

Para volúmenes SAN aprovisionados desde Google Cloud NetApp Volumes:

- Trident crea un Logical Unit Number (LUN) en un grupo de almacenamiento Flex Unified.
- Durante la publicación, Trident asigna el LUN a un grupo de hosts por nodo.
- Durante la puesta en escena del nodo, Trident:
  - Inicia sesión en el target iSCSI
  - Descubre el LUN
  - Configura multivía
- Si `volumeMode: Filesystem`, Trident formatea el dispositivo si es necesario y lo monta.
- Si `volumeMode: Block`, Trident conecta el dispositivo y lo expone directamente al pod sin formatear ni montar.

> Los volúmenes de bloques SAN no proporcionan bloqueo distribuido ni coordinación de escritura. Cuando un volumen de bloques es accedido por varios nodos (ReadWriteMany con `volumeMode: Block`), la aplicación o el sistema de archivos debe gestionar la concurrencia.

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 NFS o SMB

Si estás usando Google Cloud NetApp Volumes por primera vez o en una ubicación nueva, necesitas hacer una configuración inicial para configurar Google Cloud NetApp Volumes y crear un volumen NFS o SMB. Consulta "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 (`roles/netapp.admin`). Consulte "Funciones y permisos de Identity and Access Management".
- Archivo de claves de API para tu cuenta de GCNV. Consulte "Cree una clave de cuenta de servicio"
- 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 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.	
`location`	La ubicación de Google Cloud donde 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 `netapp.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`.	
`labels`	Conjunto de etiquetas con formato JSON arbitrario que se aplica en los volúmenes	""
`network`	La red de Google Cloud utilizada para los volúmenes GCNV.	

Parámetro	Descripción	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
`nasType`	Configure la creación de volúmenes NFS o SMB. Las opciones son `nfs`, `smb` o nulo. El valor predeterminado es nulo en volúmenes de NFS.	`nfs`
`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: asia-east1 topology.kubernetes.io/zone: asia-east1-a`	

Opciones de aprovisionamiento de volúmenes

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

Parámetro	Descripción	Predeterminado
`exportRule`	Las reglas de exportación de nuevos volúmenes. Debe ser una lista separada por comas de cualquier combinación de direcciones IPv4.	"0.0.0.0/0"
`snapshotDir`	Acceso a la `.snapshot` directorio	`true`, `false` (el comportamiento por defecto puede variar. Establece explícitamente) "false" para NFSv3
`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, Trident detecta todos sus pools de almacenamiento delegados a 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 estás empezando a usar Google Cloud NetApp Volumes y probando cosas, pero en la práctica podrías necesitar proporcionar un alcance adicional para los volúmenes que aprovisionas.

Reemplaza `<id_value>` y `<key_value>` con las credenciales de tu cuenta de servicio.

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----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 para volúmenes SMB

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv1
  namespace: trident
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123456789"
  location: asia-east1
  serviceLevel: flex
  nasType: smb
  apiKey:
    type: service_account
    project_id: cloud-native-data
    client_email: trident-sample@cloud-native-
data.iam.gserviceaccount.com
    client_id: "123456789737813416734"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/trident-
sample%40cloud-native-data.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

Configuración con filtro StoragePools

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----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: "<id_value>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <key_value>
    -----END PRIVATE KEY-----

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

```
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
  defaults:
    snapshotReserve: "10"
    exportRule: 10.0.0.0/24
  storage:
    - labels:
        performance: extreme
      serviceLevel: extreme
      defaults:
        snapshotReserve: "5"
        exportRule: 0.0.0.0/0
    - labels:
        performance: premium
      serviceLevel: premium
    - labels:
        performance: standard
      serviceLevel: standard
```

Identidad de nube para GKE

Al instalar Trident con una identidad en la nube de GKE, los backends se autentican como una identidad de carga de trabajo y omiten el bloque `apiKey`. La identidad en la nube se configura de la misma manera para ambos controladores de Google Cloud NetApp Volumes. La única diferencia es el valor de `storageDriverName`: usa `google-cloud-netapp-volumes` para volúmenes NAS (NFS o SMB) y `google-cloud-netapp-volumes-san` para volúmenes SAN (iSCSI block).

> La compatibilidad con identidades en la nube para el `google-cloud-netapp-volumes-san` driver requiere Trident 26.06 o una versión posterior. Los volúmenes SAN se aprovisionan desde los grupos de almacenamiento Flex Unified, así que configura `serviceLevel` en `flex`.

Controlador NAS (<code>google-cloud-netapp-volumes</code>)

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcp-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: '012345678901'
  network: gcnv-network
  location: us-west2
  serviceLevel: Premium
  storagePool: pool-premium1
```

Controlador SAN (<code>google-cloud-netapp-volumes-san</code>)

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcp-gcnv-san
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes-san
  projectNumber: '012345678901'
  network: gcnv-network
  location: us-west2
  serviceLevel: flex
  storagePool: pool-flex1
```

Configuración de topologías admitidas

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 contienen un subconjunto de las regiones y zonas proporcionadas en un backend, Trident crea volúmenes en la región y zona mencionadas. Para obtener más información, consulte "Utilice Topología CSI".

```
---
version: 1
storageDriverName: google-cloud-netapp-volumes
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: asia-east1
serviceLevel: flex
supportedTopologies:
  - topology.kubernetes.io/region: asia-east1
    topology.kubernetes.io/zone: asia-east1-a
  - topology.kubernetes.io/region: asia-east1
    topology.kubernetes.io/zone: asia-east1-b
```

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.

Definiciones de clases 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".

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. Se puede utilizar cualquier usuario/contraseña de Active Directory con permisos o sin permisos para el secreto de etapa de nodos.

Configuración básica en el espacio de nombres predeterminado

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

Uso de diferentes secretos por espacio de nombres

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

Uso de diferentes secretos por volumen

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

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

Ejemplo de definición de PVC

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

Para verificar si 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
```

Configura la jerarquización automática para Google Cloud NetApp Volumes

La clasificación automática se configura a través de los parámetros de backend de Trident y las anotaciones de PersistentVolumeClaim durante el aprovisionamiento de volúmenes. Puedes configurar la clasificación automática para Google Cloud NetApp Volumes usando Trident.

Descripción general

La jerarquización automática permite a Trident aprovisionar volúmenes que mueven automáticamente los datos inactivos de un nivel de rendimiento a un nivel de capacidad. Esto reduce el coste de almacenamiento mientras se preserva el rendimiento para los datos a los que se accede con frecuencia.

Trident aplica la configuración de auto-tiering solo en el momento de la creación del volumen. Los cambios posteriores al aprovisionamiento no son compatibles con Trident 26.02.

Conceptos

Organización automática en niveles

La jerarquización automática mueve los datos a los que se accede con poca frecuencia de un nivel de rendimiento a un nivel de capacidad según los patrones de acceso. El movimiento de datos ocurre de forma asíncrona y no es inmediato.

Política de tiering

La política de jerarquización determina si la jerarquización automática está habilitada para un volumen.

Se admiten las siguientes políticas: * `auto`: permite la jerarquización automática basada en patrones de acceso * `none`: desactiva la jerarquización automática

Días de refrigeración

Los días de enfriamiento especifican el número mínimo de días que un bloque de datos debe permanecer inactivo antes de que sea elegible para la clasificación por niveles. Los días de enfriamiento se aplican solo cuando la política de clasificación por niveles se establece en `auto`.

Modelo de configuración

Ámbitos de configuración

La jerarquización automática puede configurarse en varios ámbitos:

- **Ámbito del storage pool** se aplica a todos los volúmenes aprovisionados desde el pool.
- **Ámbito del volumen** Aplica a un solo volumen a través de las anotaciones de PersistentVolumeClaim.

Trident determina la configuración efectiva según dónde se defina cada ajuste.

Precedencia de la configuración

Cuando se define la misma configuración en varios ámbitos, Trident aplica el siguiente orden de precedencia:

1. Anotaciones de PersistentVolumeClaim
2. Configuración del backend Trident
3. Valores predeterminados del pool de almacenamiento

Los ajustes definidos con mayor precedencia reemplazan los valores de nivel inferior.

Funcionalidad compatible en Trident 26.02

Trident 26.02 es compatible con las siguientes funciones de auto-tiering para Google Cloud NetApp Volumes:

- Habilitar o deshabilitar la clasificación automática durante el aprovisionamiento de volúmenes
- Definir una política de niveles en la configuración del backend de Trident
- Anular la política de niveles y los días de enfriamiento por volumen usando anotaciones de PVC
- Configurar los días de enfriamiento para los volúmenes con la jerarquización automática activada

Funcionalidad no soportada en Trident 26.02

Las siguientes operaciones no son compatibles:

- Modificar la configuración de auto-tiering después de crear el volumen
- Cambio de las políticas de niveles en volúmenes existentes usando actualizaciones de Kubernetes
- Aplicar la configuración de jerarquización automática fuera de los flujos de trabajo de aprovisionamiento gestionados por Trident

Parámetros de configuración del backend

Los siguientes parámetros controlan el comportamiento del auto-tiering cuando se definen en la configuración del backend de Trident:

Parámetro	Obligatorio	Descripción
tieringPolicy	No	Política de tiering para volúmenes (`auto` o `none`)
tieringMinimumCoolingDays	No	Número de días inactivos antes de que los datos se transfieran de nivel (intervalo: 2–183, predeterminado: 31)

Anulaciones a nivel de volumen usando anotaciones de PersistentVolumeClaim

Anotaciones admitidas

Las anotaciones de PersistentVolumeClaim permiten anular por volumen la configuración de auto-tiering.

Anotación	Descripción
`trident.netapp.io/tieringPolicy`	Anula la política de tiering del volumen
`trident.netapp.io/tieringMinimumCoolingDays`	Anula el valor de los días de cooling para el volumen

Ejemplo: PersistentVolumeClaim con anulaciones de auto-tiering

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: auto-tiering-pvc
  annotations:
    trident.netapp.io/tieringPolicy: auto
    trident.netapp.io/tieringMinimumCoolingDays: "45"
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: google-cloud-netapp-volumes-auto-tiering
  resources:
    requests:
      storage: 500Gi
```

Comportamiento y limitaciones

Comportamiento del aprovisionamiento

- Los ajustes de clasificación automática se evalúan y aplican solo en el momento de la creación del volumen.
- Trident no reconcilia la configuración de tiering después del aprovisionamiento.
- Los días de enfriamiento se ignoran cuando la política de niveles se establece en `none`.

Limitaciones de la plataforma

- La jerarquización automática solo es compatible con volúmenes NAS (NFS y SMB).
- Los volúmenes en bloque (iSCSI) no admiten auto-tiering.
- El grupo de almacenamiento de Google Cloud NetApp Volumes debe tener la auto-tiering activada en Google Cloud.

Valores admitidos

- Rango válido para `tieringMinimumCoolingDays`: 2 a 183
- Valor predeterminado: 31

Configura los grupos de almacenamiento en modo ONTAP de Google Cloud NetApp Volumes

A partir de Trident 26.06, puedes configurar los backends existentes `ontap-san` y `ontap-nas` para aprovisionar almacenamiento en los grupos de almacenamiento en modo ONTAP de Google Cloud NetApp Volumes (GCNV). Esta función es una Tech Preview.

NOTA	Esta función es una Tech Preview en Trident 26.06. Las funciones Tech Preview no están soportadas para uso en producción. La funcionalidad, los campos de configuración y la matriz compatible pueden cambiar antes de la disponibilidad general (GA).

Cómo funciona

Trident reutiliza la lógica existente de los controladores `ontap-san` y `ontap-nas` y redirige las llamadas REST de ONTAP a través de un endpoint proxy de GCNV. Este enfoque te permite aprovisionar y gestionar el almacenamiento en clústeres ONTAP alojados en GCNV mientras mantienes los mismos flujos de trabajo operativos que usas para un backend ONTAP directo. ONTAP-mode no introduce un nuevo controlador de almacenamiento. Lo habilitas por backend añadiendo un bloque de configuración `gcnv` a un backend `ontap-san` o `ontap-nas`.

La ruta de aprovisionamiento es la siguiente:

`PersistentVolumeClaim` → Trident (`ontap-san` o `ontap-nas`) → cliente REST de ONTAP → proxy GCNV → clúster de ONTAP en el pool GCNV

Ámbito compatible y no compatible

Esta versión preliminar (Tech Preview) admite lo siguiente:

- Controladores: `ontap-san` (iSCSI) y `ontap-nas` (NFS o SMB).
- Operaciones del ciclo de vida a través del proxy: inicialización del backend, creación y eliminación de volúmenes y flujos REST de ONTAP relacionados.
- Modelos de autenticación: Workload Identity Pool (WIP), clave de cuenta de servicio y Application Default Credentials (ADC) como recurso de reserva.

Lo siguiente queda fuera del alcance de esta versión preliminar técnica:

- Los `ontap-san-economy`, `ontap-nas-economy` y `ontap-nas-flexgroup` controladores.
- La trayectoria de personalidad ASA r2.
- Solución alternativa de ZAPI. El modo ONTAP usa solo ONTAP REST.

Requisitos previos

Antes de configurar un backend en modo ONTAP, asegúrate de que tienes lo siguiente:

- Trident 26.06 o posterior.
- Un grupo de almacenamiento GCNV en modo ONTAP en tu proyecto y ubicación de destino.
- Permisos de GCP IAM para operaciones de proxy, adaptados adecuadamente a tu entorno.
- Una configuración de backend que incluye `proxyURL`, `projectNumber`, `location` y `poolID`.
- Para los flujos de trabajo de SAN iSCSI, los requisitos previos de iSCSI y multivía del lado del nodo para tu plataforma. Consulta "Prepare el nodo de trabajo".

Configura el backend

Un `TridentBackendConfig` requiere `credentials`. Para el modo ONTAP, ese secreto contiene las credenciales del proxy de GCP, no las de ONTAP `managementLIF` / `username` / `password`. `svm` es opcional: cada pool de GCNV en modo ONTAP actualmente tiene un SVM, y Trident lo obtiene del pool cuando se omite.

Utiliza el mismo patrón de cuenta de servicio que el GCNV nativo: los campos no confidenciales en `gcnv.apiKey`, `private_key` y `private_key_id` en el secreto. Consulta "Ejemplos de configuración de Google Cloud NetApp Volumes".

backend de ontap-san

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: gcnv-ontap-san
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-san
  backendName: gcnv-ontap-san
  credentials:
    name: gcnv-sa-secret
    type: secret
  gcnv:
    proxyURL: "https://netapp.googleapis.com"
    projectNumber: "<project-number>"
    location: "<region-or-zone>"
    poolID: "<pool-id>"
  svm: "<svm-name>"
```

backend de ONTAP-NAS

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: gcnv-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  backendName: gcnv-ontap-nas
  credentials:
    name: gcnv-sa-secret
    type: secret
  gcnv:
    proxyURL: "https://netapp.googleapis.com"
    projectNumber: "<project-number>"
    location: "<region-or-zone>"
    poolID: "<pool-id>"
    apiKey:
      type: service_account
      project_id: "<project-id>"
      client_email: "<service-account-email>"
      client_id: "<client-id>"
      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: "<client-x509-cert-url>"
```

Secreto

```
apiVersion: v1
kind: Secret
metadata:
  name: gcnv-sa-secret
  namespace: trident
type: Opaque
stringData:
  private_key_id: "<private-key-id>"
  private_key: |
    -----BEGIN PRIVATE KEY-----
    <private-key>
    -----END PRIVATE KEY-----
```

NOTA No incluyas `private_key` ni `private_key_id` en la `TridentBackendConfig` spec; el CRD los rechaza. Utiliza `ontap-san` con el mismo `gcnv` y patrón de credenciales para el almacenamiento en bloque.

```
svm: "<svm-name>"
```

```
== Authentication

Trident resolves credentials for proxy access in the following order:

. Workload Identity Pool (WIP)
. Service account key
. Application Default Credentials (ADC)

For this Tech Preview, use the following practices:

* Prefer Workload Identity Pool where it is available.
* Do not embed raw private keys in version-controlled backend files. Store
credentials in a Kubernetes secret.
* Apply least-privilege IAM scoping to the service account.

== Validation and initialization behavior

Trident validates an ONTAP-mode backend during initialization and fails
fast rather than partially initializing.

Backend initialization fails if a required `gcnv` field is missing, the
driver is unsupported for ONTAP-mode, or credential resolution fails.

== Known limitations

* This feature is a Tech Preview. Behavior and the supported matrix can
change before GA.
* ZAPI fallback is not used in ONTAP-mode.
* Protocol and node-readiness requirements still apply. For example, iSCSI
workflows require the node-side prerequisites described in
xref:{relative_path}../trident-use/worker-node-prep.html[Prepare the
worker node].
* Existing ONTAP and GCNV environment constraints still apply.

== Upgrade and compatibility

ONTAP-mode is opt-in per backend through the `gcnv` configuration block:

* Existing non-GCNV ONTAP backends are unaffected.
```

```
* Mixed deployments that combine direct ONTAP backends and ONTAP-mode
proxy backends are supported through backend-level configuration. Each
backend is configured independently.
* To stop using ONTAP-mode, remove or replace the affected backends. No
global switch is required.

== What's next?

Apply secret before TBC; use `kubectl apply` and `-n trident` to match
examples.
```

Configure un back-end de NetApp HCI o SolidFire

Aprende a crear y usar un back-end de Element con tu instalación de Trident.

Detalles del controlador de elementos

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 el `Filesystem` volumeMode, 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	«SolidFire_» + dirección IP de 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. Trident utiliza CHAP.	verdadero
`AccessGroups`	Lista de ID de grupos de acceso que se van a usar	Busca el ID de un grupo de acceso llamado 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

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

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

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

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

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

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

Las etiquetas de copias de Trident están presentes en un pool de almacenamiento en el LUN de almacenamiento back-end en el momento del aprovisionamiento. Para mayor comodidad, los administradores de almacenamiento pueden definir etiquetas por pool virtual y agrupar volúmenes por etiqueta.

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

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

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

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

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

El primer 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. Trident decidirá qué pool virtual se selecciona y garantiza que se cumpla el requisito 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"

Unidades SAN de ONTAP

Información general del controlador de SAN de ONTAP

Obtenga información sobre la configuración de un back-end de ONTAP con controladores SAN de ONTAP y Cloud Volumes ONTAP.

Información sobre el controlador de SAN de ONTAP

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).

Controlador	Protocolo	VolumeMode	Modos de acceso compatibles	Sistemas de archivos compatibles
`ontap-san`	SCSI iSCSI sobre FC	Bloque	RWO, ROX, RWX, RWOP	Sin sistema de archivos; dispositivo de bloque sin procesar
`ontap-san`	SCSI iSCSI sobre FC	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

Controlador	Protocolo	VolumeMode	Modos de acceso compatibles	Sistemas de archivos compatibles
`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`
`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`

ADVERTENCIA

- Uso `ontap-san-economy` solo si se espera que el número de uso de volúmenes persistentes sea superior a "Límites de volumen ONTAP compatibles".
- Uso `ontap-nas-economy` solo si se espera que el número de uso de volúmenes persistentes sea superior a "Límites de volumen ONTAP compatibles" y la `ontap-san-economy` no se puede utilizar el conductor.
- No utilizar `ontap-nas-economy` si prevé la necesidad de protección de datos, recuperación ante desastres o movilidad.
- NetApp no recomienda el uso de crecimiento automático de FlexVol en todos los controladores de ONTAP, excepto ONTAP-san. Como solución alternativa, Trident admite el uso de la reserva Snapshot y escala los volúmenes de FlexVol en consecuencia.

Permisos de usuario

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

NOTA

Si se usa `limitAggregateUsage` el parámetro, se requieren permisos de administrador del clúster. Cuando se usa Amazon FSx para NetApp ONTAP con Trident, el `limitAggregateUsage` 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

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 Trident para que su ciclo de vida se pueda gestionar mediante Trident
- Multivía nativa de NVMe
- Cierre correcto o sin complicaciones de los K8s nodos (24,06)

Trident no admite:

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

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

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

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

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

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

Autentique el backend de ONTAP

Trident ofrece dos modos de autenticación de un backend ONTAP.

- Basado en credenciales: El nombre de usuario y la contraseña de un usuario ONTAP con los permisos requeridos. Se recomienda utilizar un rol de inicio de sesión de seguridad predefinido, como `admin` o. `vsadmin` Garantizar la máxima compatibilidad con versiones de ONTAP.
- Basado en certificado: 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.

ADVERTENCIA	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

Trident requiere que las credenciales se comuniquen con un administrador de SVM o con el ámbito del clúster para que se comunique con el back-end de ONTAP. Se recomienda hacer uso de roles estándar, predefinidos como `admin` o `vsadmin`. Esto garantiza la compatibilidad con futuras versiones de ONTAP que podrían exponer API de funciones que podrán utilizarse en futuras versiones de Trident. Puede crearse y utilizarse un rol de inicio de sesión de seguridad personalizado con Trident, pero no se recomienda.

Una definición de backend de ejemplo tendrá este aspecto:

YAML

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

JSON

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

Tenga en cuenta que la definición 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.

Habilitar 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
```

NOTA Después de ejecutar este comando, ONTAP solicita la introducción del certificado. Pega el contenido del archivo `k8senv.pem` generado en el paso 1, luego ingresa `END` para completar la instalación.

4. Confirme los compatibilidad con el rol de inicio de sesión de seguridad ONTAP `cert` método de

autenticación.

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

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

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

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

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

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

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

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

Actualice los métodos de autenticación o gire las credenciales

Puede actualizar un back-end existente para utilizar un método de autenticación diferente o para rotar sus credenciales. Esto funciona de las dos maneras: Los back-ends que utilizan nombre de usuario/contraseña se pueden actualizar para usar certificados. Los back-ends que utilizan certificados pueden actualizarse a nombre de usuario/contraseña. Para ello, debe eliminar el método de autenticación existente y agregar el nuevo método de autenticación. A continuación, utilice el archivo backend.json actualizado que contiene los parámetros necesarios para ejecutarse `tridentctl backend update`.

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

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

NOTA	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 Trident puede comunicarse con el back-end de ONTAP y manejar operaciones de volumen futuras.

Crear rol de ONTAP personalizado para Trident

Puede crear un rol de clúster de ONTAP con un Privileges mínimo de modo que no tenga que utilizar el rol de administrador de ONTAP para realizar operaciones en Trident. Cuando incluye el nombre de usuario en una configuración de back-end de Trident, Trident utiliza el rol de clúster de ONTAP que creó para realizar las operaciones.

Consulte "Generador de roles personalizados de Trident"para obtener más información sobre la creación de roles personalizados de Trident.

Con la CLI de ONTAP

1. Cree un rol nuevo mediante el siguiente comando:

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

2. Cree un nombre de usuario para el usuario de Trident:

```
security login create -username <user_name\> -application ontapi
-authmethod <password\> -role <name_of_role_in_step_1\> -vserver
<svm_name\> -comment "user_description"
```

3. Asignar el rol al usuario:

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

Mediante System Manager

Realice los pasos siguientes en ONTAP System Manager:

1. **Crear un rol personalizado**:
   a. Para crear un rol personalizado a nivel de clúster, seleccione **Cluster > Settings**.

      (O) Para crear un rol personalizado en el nivel de SVM, seleccione **Almacenamiento > Storage VMs > `required SVM`> Settings > Users and Roles**.

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

Consulte las siguientes páginas si quiere más información:

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

Autenticar conexiones con CHAP bidireccional

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`, Trident configura la seguridad del iniciador predeterminado de la SVM 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
```

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

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

Cómo funciona

Al configurarse `useCHAP` en true, el administrador de almacenamiento le ordena a 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, Trident establecerá el tipo de seguridad predeterminado en `CHAP` y continuará configurando el iniciador CHAP y el nombre de usuario y los secretos de destino.
  - Si la SVM contiene LUN, 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 backend, Trident crea un CRD correspondiente `tridentbackend` y almacena los secretos CHAP y nombres de usuario como secretos de Kubernetes. Todos los VP que crea Trident en este back-end se montarán y conectarán mediante CHAP.

Rotar credenciales y actualizar backends

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

ADVERTENCIA	Al actualizar los secretos CHAP para un backend, debe utilizar `tridentctl` para actualizar el backend. No actualice las credenciales en el clúster de almacenamiento mediante la interfaz de línea de comandos de ONTAP o ONTAP System Manager, ya que 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 las credenciales se actualizan mediante Trident en la SVM. Las nuevas conexiones utilizan las credenciales actualizadas y las conexiones existentes siguen activas. Al desconectar y volver a conectar los VP antiguos, se utilizarán las credenciales actualizadas.

Opciones y ejemplos de configuración DE SAN ONTAP

Aprende cómo crear y usar controladores ONTAP SAN con tu instalación de Trident. Esta sección ofrece ejemplos de configuración de backends y detalles para asignar backends a StorageClasses. "Sistemas ASA r2" se diferencian de otros sistemas ONTAP (ASA, AFF y FAS) en la implementación de su capa de almacenamiento. Estas variaciones afectan el uso de ciertos parámetros como se indica. "Obtenga más información sobre las diferencias entre los sistemas ASA r2 y otros sistemas ONTAP". En la configuración del backend de Trident, no necesitas especificar que tu sistema es ASA r2. Cuando

seleccionas `ontap-san` como `storageDriverName`, Trident detecta automáticamente el ASA r2 u otros sistemas ONTAP. Algunos parámetros de configuración del backend no aplican a los sistemas ASA r2, como se indica en la tabla de abajo.

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

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-san` o. `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). Se puede configurar para utilizar direcciones IPv6 si Trident se instaló con el indicador 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 Ejemplo de MetroCluster. **NOTA** Si utiliza las credenciales «vsadmin», `managementLIF` debe ser la de la SVM; si utiliza credenciales «admin», `managementLIF` debe ser la del clúster.	«10,0.0,1», «[2001:1234:abcd::fefe]»
`dataLIF`	Dirección IP de LIF de protocolo. Se puede configurar para utilizar direcciones IPv6 si Trident se instaló con el indicador IPv6. Las direcciones IPv6 deben definirse entre corchetes, `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` como . **No especifique para iSCSI.** Trident utiliza "Asignación de LUN selectiva de ONTAP" para detectar las LIF iSCSI necesarias para establecer una sesión de rutas múltiples. Se genera una advertencia si `dataLIF` se define explícitamente. **Omitir para MetroCluster.** Consulte la Ejemplo de MetroCluster.	Derivado del SVM

Parámetro	Descripción	Predeterminado
`svm`	Máquina virtual de almacenamiento que usar **Omitir para MetroCluster.** Ver Ejemplo de MetroCluster.	Derivado si una SVM `managementLIF` está especificado
`useCHAP`	Use CHAP para autenticar iSCSI para los controladores SAN de ONTAP [Boolean]. Establezca como `true` para que Trident configure y utilice CHAP bidireccional como la autenticación predeterminada para la SVM especificada en el back-end. Consulte "Prepárese para configurar el back-end con los controladores SAN de ONTAP" para obtener más información. **No compatible con FCP o NVMe/TCP.**	`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`	""
`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 necesario para comunicarse con el clúster ONTAP . Se utiliza para la autenticación basada en credenciales. Para la autenticación de Active Directory, consulte "Autenticar Trident en un SVM backend mediante credenciales de Active Directory".	""
`password`	Contraseña necesaria para comunicarse con el clúster ONTAP . Se utiliza para la autenticación basada en credenciales. Para la autenticación de Active Directory, consulte "Autenticar Trident en un SVM backend mediante credenciales de Active Directory".	""
`svm`	Máquina virtual de almacenamiento que usar	Derivado si una SVM `managementLIF` está especificado

Parámetro	Descripción	Predeterminado
`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`
`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. Si no se asigna, cualquiera de los agregados disponibles puede usarse para aprovisionar un volumen FlexGroup. **NOTA** Cuando el agregado se actualiza en SVM, se actualiza automáticamente en Trident sondeando SVM sin tener que reiniciar la controladora Trident. Cuando se haya configurado un agregado específico en Trident para aprovisionar volúmenes, si se cambia el nombre de este agregado o se saca de la SVM, el back-end se moverá al estado Failed en Trident mientras se sondea el agregado de SVM. Debe cambiar el agregado por uno presente en la SVM o quitarlo por completo para que el back-end vuelva a estar en línea. **No especificar para sistemas ASA r2**.	""
`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 Trident. **No especificar para sistemas ASA r2**.	"" (no se aplica de forma predeterminada)
`limitVolumeSize`	Error en el aprovisionamiento si el tamaño del volumen solicitado es superior a este valor. Además, restringe el tamaño máximo de los volúmenes que gestiona para las LUN.	" (no se aplica por defecto)
`lunsPerFlexvol`	El número máximo de LUN por FlexVol debe estar comprendido entre [50 y 200]	`100`
`debugTraceFlags`	Indicadores de depuración que se deben usar para la solución de problemas. Ejemplo, {«api»:false, «method»:true} No lo utilice a menos que esté solucionando problemas y necesite un volcado de log detallado.	`null`

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

Parámetro	Descripción	Predeterminado
`formatOptions`	Puede `formatOptions` usarse para especificar argumentos de línea de comandos para `mkfs` el comando, que se aplicará cada vez que se formatee un volumen. Esto permite formatear el volumen según sus preferencias. Asegúrese de especificar las opciones formatOptions similares a las de los comandos mkfs, excluyendo la ruta del dispositivo. Ejemplo: «-E nodiscard» **Compatible con `ontap-san` y `ontap-san-economy` Controladores con protocolo iSCSI. Además, compatible con sistemas ASA r2 cuando se utilizan los protocolos iSCSI y NVMe/TCP.**	
`limitVolumePoolSize`	Tamaño máximo de FlexVol solicitable al usar LUN en back-end económico de ONTAP-san.	"" (no se aplica de forma predeterminada)
`denyNewVolumePools`	Restringe `ontap-san-economy` los back-ends para que no creen nuevos volúmenes de FlexVol para contener sus LUN. Solo se utilizan los FlexVols preexistentes para aprovisionar nuevos VP.	

Recomendaciones para utilizar formatOptions

Trident recomienda las siguientes opciones para agilizar el proceso de formateo:

- **-E nodiscard (ext3, ext4):** No intente descartar bloques en el momento de mkfs (descartar bloques inicialmente es útil en dispositivos de estado sólido y almacenamiento disperso/de aprovisionamiento ligero). Esto reemplaza la opción obsoleta "-K" y es aplicable a los sistemas de archivos ext3 y ext4.
- **-K (xfs):** No intente descartar bloques en el momento de mkfs. Esta opción es aplicable al sistema de archivos xfs.

Autenticar Trident en un SVM backend mediante credenciales de Active Directory

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

pasos

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

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

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

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

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

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

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

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

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

Parámetro	Descripción	Predeterminado
`spaceAllocation`	Asignación de espacio para las LUN	"verdadero" **Si se especifica, configúrelo en `true` para sistemas ASA r2**.
`spaceReserve`	Modo de reserva de espacio; «ninguno» (fino) o «volumen» (grueso). **Empezar a `none` para sistemas ASA r2**.	ninguno
`snapshotPolicy`	Política de Snapshot para utilizar. **Empezar a `none` para sistemas ASA r2**.	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. Usar grupos de políticas de QoS con Trident requiere ONTAP 9 Intersight 8 o posterior. Debe usar 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 aplica el techo máximo para el rendimiento total de todas las cargas de trabajo.	""
`adaptiveQosPolicy`	Grupo de políticas de calidad de servicio adaptativo que permite asignar los volúmenes creados. Elija uno de qosPolicy o adaptiveQosPolicy por pool/back-end de almacenamiento	""
`snapshotReserve`	Porcentaje de volumen reservado para snapshots. **No especificar para sistemas ASA r2**.	«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 Trident será habilitado NAE. Para obtener más información, consulte: "Cómo funciona Trident con NVE y NAE".	"falso" **Si se especifica, configúrelo en `true` para sistemas ASA r2**.

Parámetro	Descripción	Predeterminado
`luksEncryption`	Active el cifrado LUKS. Consulte "Usar la configuración de clave unificada de Linux (LUKS)".	"" **Establecer en `false` para sistemas ASA r2**.
`tieringPolicy`	Política de niveles para utilizar "ninguno" **No especificar para sistemas ASA r2**.	
`nameTemplate`	Plantilla para crear nombres de volúmenes personalizados.	""

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'
```

NOTA Para todos los volúmenes creados con `ontap-san` el controlador, 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. Trident agrega un 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`, Trident calcula el tamaño de los volúmenes de la siguiente manera:

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

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

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.

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

Ejemplo de SAN ONTAP

Se trata de una configuración básica que utiliza el `ontap-san` controlador.

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

Ejemplo de MetroCluster

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

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

```
version: 1
storageDriverName: ontap-san
managementLIF: 192.168.1.66
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>
```

Ejemplo de autenticación basada en certificados

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

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

Ejemplos de CHAP bidireccional

Estos ejemplos crean un backend con `useCHAP` establezca en `true`.

Ejemplo de CHAP de SAN de ONTAP

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

Ejemplo de CHAP de economía de SAN ONTAP

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

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 SCSI sobre FC (FCP)

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

```
---
version: 1
backendName: fcp-backend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_fc
username: vsadmin
password: password
sanType: fcp
useREST: true
```

Ejemplo de configuración de backend con 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}}_{{.vo\
    lume.RequestName}}"
labels:
  cluster: ClusterA
  PVC: "{{.volume.Namespace}}_{{.volume.RequestName}}"
```

Ejemplo de formatOptions para el controlador ONTAP-san-economy

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: ""
svm: svm1
username: ""
password: "!"
storagePrefix: whelk_
debugTraceFlags:
  method: true
  api: true
defaults:
  formatOptions: -E nodiscard
```

Ejemplos de back-ends con pools virtuales

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

Trident establece las etiquetas de aprovisionamiento en el campo de comentarios. En las copias FlexVol volume Trident se establecen comentarios Todas las etiquetas presentes en un pool virtual para el volumen de almacenamiento durante el aprovisionamiento. Para mayor comodidad, los administradores de almacenamiento pueden definir etiquetas por pool virtual y agrupar volúmenes por etiqueta.

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

Ejemplo de SAN ONTAP

```yaml
---
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. Con el `parameters.selector` Cada StorageClass llama la atención sobre qué pools virtuales pueden usarse para alojar un volumen. El volumen tendrá los aspectos definidos en el pool virtual elegido.

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

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

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

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

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

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

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

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

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

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

- La `my-test-app-sc` StorageClass se asignará al `testAPP` pool virtual en el `ontap-san` conductor con `sanType: nvme`. Esta es la única oferta de pool `testApp`.

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

Trident decidirá qué pool virtual se selecciona y garantiza que se cumpla el requisito de almacenamiento.

Unidades NAS de ONTAP

Información general del controlador NAS de ONTAP

Obtenga información sobre la configuración de un back-end de ONTAP con controladores NAS de ONTAP y Cloud Volumes ONTAP.

Información del controlador NAS de ONTAP

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).

Controlador	Protocolo	VolumeMode	Modos de acceso compatibles	Sistemas de archivos compatibles
`ontap-nas`	NFS SMB	Sistema de archivos	RWO, ROX, RWX, RWOP	« », `nfs`, `smb`
`ontap-nas-economy`	NFS SMB	Sistema de archivos	RWO, ROX, RWX, RWOP	« », `nfs`, `smb`

Controlador	Protocolo	VolumeMode	Modos de acceso compatibles	Sistemas de archivos compatibles
`ontap-nas-flexgroup`	NFS SMB	Sistema de archivos	RWO, ROX, RWX, RWOP	« », `nfs`, `smb`

ADVERTENCIA

- Uso `ontap-san-economy` solo si se espera que el número de uso de volúmenes persistentes sea superior a "Límites de volumen ONTAP compatibles".
- Uso `ontap-nas-economy` solo si se espera que el número de uso de volúmenes persistentes sea superior a "Límites de volumen ONTAP compatibles" y la `ontap-san-economy` no se puede utilizar el conductor.
- No utilizar `ontap-nas-economy` si prevé la necesidad de protección de datos, recuperación ante desastres o movilidad.
- NetApp no recomienda el uso de crecimiento automático de FlexVol en todos los controladores de ONTAP, excepto ONTAP-san. Como solución alternativa, Trident admite el uso de la reserva Snapshot y escala los volúmenes de FlexVol en consecuencia.

Permisos de usuario

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

Para puestas en marcha de Amazon FSx para NetApp ONTAP, Trident espera ejecutarse como administrador de ONTAP o SVM, utilizando el usuario del clúster `fsxadmin` o un `vsadmin` 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.

NOTA

Si se usa `limitAggregateUsage` el parámetro, se requieren permisos de administrador del clúster. Cuando se usa Amazon FSx para NetApp ONTAP con Trident, el `limitAggregateUsage` 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

Entiende los requisitos, las opciones de autenticación y las políticas de exportación para configurar un backend ONTAP con drivers ONTAP NAS. A partir de la versión 25.10, NetApp Trident es compatible con "Sistema de almacenamiento NetApp AFX". NetApp AFX storage systems difieren de otros sistemas ONTAP (ASA, AFF y FAS) en la implementación de su capa de almacenamiento. En la configuración del backend de Trident, no necesitas especificar que tu sistema es AFX. Cuando seleccionas `ontap-nas` como `storageDriverName`, Trident detecta automáticamente los sistemas AFX.

NOTA	Solo el `ontap-nas` El controlador (con protocolo NFS) es compatible con los sistemas AFX; el protocolo SMB no es compatible.

Requisitos

- Para todos los backends de ONTAP, Trident requiere que se asigne al menos un agregado al SVM.
- Puede ejecutar más de un controlador y crear clases de almacenamiento que apunten a uno u otro. Por ejemplo, puede configurar una clase Gold que utilice `ontap-nas` Controlador y clase Bronze que utiliza `ontap-nas-economy` uno.
- Todos sus nodos de trabajo de Kubernetes deben tener instaladas las herramientas NFS adecuadas. Consulte "aquí" para obtener más detalles.
- 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

Trident ofrece dos modos de autenticación de un backend ONTAP.

- Basado en Credenciales: Este modo requiere permisos suficientes para el backend de ONTAP. Se recomienda utilizar una cuenta asociada con un rol de inicio de sesión de seguridad predefinido, como `admin` o. `vsadmin` Garantizar la máxima compatibilidad con versiones de ONTAP.
- Basado en certificado: Este modo requiere un certificado instalado en el back-end para que 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.

ADVERTENCIA	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

Trident requiere que las credenciales se comuniquen con un administrador de SVM o con el ámbito del clúster para que se comunique con el back-end de ONTAP. Se recomienda hacer uso de roles estándar, predefinidos como `admin` o `vsadmin`. Esto garantiza la compatibilidad con futuras versiones de ONTAP que podrían exponer API de funciones que podrán utilizarse en futuras versiones de Trident. Puede crearse y utilizarse un rol de inicio de sesión de seguridad personalizado con Trident, pero no se recomienda.

Una definición de backend de ejemplo tendrá este aspecto:

YAML

```yaml
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
credentials:
  name: secret-backend-creds
```

JSON

```json
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "credentials": {
        "name": "secret-backend-creds"
    }
}
```

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

Habilite la autenticación basada en certificados

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

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

Un flujo de trabajo típico implica los pasos siguientes.

Pasos

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

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

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

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

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

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

4. Confirme los compatibilidad con el rol de inicio de sesión de seguridad ONTAP `cert` método de autenticación.

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

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

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

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

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

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

```
cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "NasBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"clientCertificate": "Faaaakkkkeeee...Vaaalllluuuueeee",
"clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
"storagePrefix": "myPrefix_"
}

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

Actualice los métodos de autenticación o gire las credenciales

Puede actualizar un back-end existente para utilizar un método de autenticación diferente o para rotar sus credenciales. Esto funciona de las dos maneras: Los back-ends que utilizan nombre de usuario/contraseña se pueden actualizar para usar certificados. Los back-ends que utilizan certificados pueden actualizarse a nombre de usuario/contraseña. Para ello, debe eliminar el método de autenticación existente y agregar el nuevo método de autenticación. A continuación, utilice el archivo backend.json actualizado que contiene los parámetros necesarios para ejecutarse `tridentctl update backend`.

```
cat cert-backend-updated.json
```

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

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

NOTA 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 Trident puede comunicarse con el back-end de ONTAP y manejar operaciones de volumen futuras.

Crear rol de ONTAP personalizado para Trident

Puede crear un rol de clúster de ONTAP con un Privileges mínimo de modo que no tenga que utilizar el rol de administrador de ONTAP para realizar operaciones en Trident. Cuando incluye el nombre de usuario en una configuración de back-end de Trident, Trident utiliza el rol de clúster de ONTAP que creó para realizar las operaciones.

Consulte "Generador de roles personalizados de Trident"para obtener más información sobre la creación de roles personalizados de Trident.

Con la CLI de ONTAP

1. Cree un rol nuevo mediante el siguiente comando:

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

2. Cree un nombre de usuario para el usuario de Trident:

```
security login create -username <user_name\> -application ontapi
-authmethod <password\> -role <name_of_role_in_step_1\> –vserver
<svm_name\> -comment "user_description"
```

3. Asignar el rol al usuario:

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

Mediante System Manager

Realice los pasos siguientes en ONTAP System Manager:

1. **Crear un rol personalizado**:
   a. Para crear un rol personalizado a nivel de clúster, seleccione **Cluster > Settings**.

      (O) Para crear un rol personalizado en el nivel de SVM, seleccione **Almacenamiento > Storage VMs > `required SVM`> Settings > Users and Roles**.

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

Consulte las siguientes páginas si quiere más información:

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

Gestione las políticas de exportación de NFS

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

Trident proporciona dos opciones al trabajar con políticas de exportación:

- Trident puede gestionar de manera dinámica la política de exportación; en este modo de funcionamiento,

el administrador de almacenamiento especifica una lista de bloques CIDR que representan direcciones IP admisibles. Trident agrega IP de nodo aplicables que se encuentran en estos rangos a la política de exportación de forma automática en el momento de la publicación. Como alternativa, cuando no se especifica ningún CIDR, todas las IP de unidifusión de ámbito global que se encuentran en el nodo en el que se publica el volumen se agregarán a la política de exportación.

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

Gestione de forma dinámica políticas de exportación

Trident proporciona la capacidad de gestionar dinámicamente políticas de exportación para 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 los nodos de trabajador que se montan volúmenes y que tienen IP en el rango especificado, lo que permite una gestión automatizada y precisa.

NOTA 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-economy
backendName: ontap_nas_auto_export
managementLIF: 192.168.0.135
svm: svm1
username: vsadmin
password: password
autoExportCIDRs:
  - 192.168.0.0/24
autoExportPolicy: true
```

NOTA 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 por NetApp para dedicar una SVM para Trident.

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

- `autoExportPolicy` se establece en `true`. Esto indica que Trident crea una política de exportación para cada volumen aprovisionado con este back-end para `svm1` la SVM y administra la adición y eliminación de

reglas mediante `autoexportCIDRs` bloques de direcciones. Hasta que un volumen se conecta a un nodo, el volumen usa una política de exportación vacía sin reglas para evitar el acceso no deseado a ese volumen. Cuando se publica un volumen en un nodo, Trident crea una política de exportación con el mismo nombre que el qtree subyacente que contiene la IP de nodo en el bloque CIDR especificado. Estas IP también se agregarán a la política de exportación utilizada por la FlexVol volume principal

  - Por ejemplo:
    - UUID de backend 403b5326-8482-40dB-96d0-d83fb3f4daec
    - `autoExportPolicy` establezca en `true`
    - prefijo de almacenamiento `trident`
    - PVC UUID a79bcf5f-7b6d-4a40-9876-e2551f159c1c
    - el qtree denominado Trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c crea una política de exportación para la FlexVol llamada `trident-403b5326-8482-40db96d0-d83fb3f4daec`, una política de exportación para el qtree llamado `trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c`y una política de exportación vacía denominada `trident_empty` en la SVM. Las reglas de la política de exportación de FlexVol serán un superconjunto de reglas contenidas en las políticas de exportación de qtree. La política de exportación vacía será reutilizada por cualquier volumen que no esté asociado.

- `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, Trident agrega todas las direcciones unicast de ámbito global que se encuentran en los nodos de trabajo con publicaciones.

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 se encuentran dentro de este rango de direcciones con publicaciones se agregarán a la política de exportación que crea Trident. Cuando Trident registra un nodo en el que se ejecuta, recupera las direcciones IP del nodo y las comprueba con respecto a los bloques de direcciones proporcionados en `autoExportCIDRs`. En el momento de la publicación, después de filtrar las IP, Trident crea las reglas de política de exportación para las IP del cliente para el nodo en el que está publicando.

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

Después de que Trident cree o actualice un backend, puede comprobar el backend utilizando `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
```

Cuando se elimina un nodo, Trident comprueba todas las políticas de exportación para eliminar las reglas de acceso correspondientes al nodo. Al eliminar esta IP de nodo de las políticas de exportación de los back-ends gestionados, Trident evita los montajes no autorizados, a menos que un nuevo nodo del clúster reutilice esta IP.

Para los back-ends existentes anteriormente, al actualizar el backend con `tridentctl update backend` se garantiza que Trident administre las políticas de exportación automáticamente. Esto crea dos nuevas políticas de exportación llamadas después del UUID del back-end y el nombre de qtree cuando son necesarias. Los volúmenes presentes en el back-end utilizarán las políticas de exportación recién creadas después de desmontarlas y montarlas de nuevo.

NOTA 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 de Trident en el nodo. A continuación, Trident actualizará la política de exportación de los back-ends que gestiona para reflejar este cambio de IP.

Prepárese para aprovisionar los volúmenes de SMB

Con un poco de preparación adicional, puede aprovisionar volúmenes SMB con `ontap-nas` de windows

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

NOTA `autoExportPolicy` No es compatible con los volúmenes 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. Trident admite volúmenes de SMB montados en pods que se ejecutan solo en nodos de Windows.
- Al menos un secreto 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 nodos Kubernetes que se ejecutan en Windows.

Pasos

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

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

   Puede crear recursos compartidos de administrador de SMB de una de dos formas mediante el "Consola de administración de Microsoft" Complemento carpetas compartidas o uso de la CLI de ONTAP. Para crear los recursos compartidos de SMB mediante la CLI de ONTAP:

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

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

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

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

   c. Compruebe que se ha creado el recurso compartido:

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

   NOTA Consulte "Cree un recurso compartido de SMB" para obtener todos los detalles.

2. Al crear el back-end, debe configurar lo siguiente para especificar volúmenes de SMB. Para obtener información sobre todas las opciones de configuración del entorno de administración de ONTAP, consulte

"Opciones y ejemplos de configuración de FSX para ONTAP".

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

Habilitar SMB seguro

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

Puntos que debe recordar

- Importador `ontap-nas-economy` Los volúmenes no son compatibles.
- Solo se admiten clones de solo lectura. `ontap-nas-economy` volúmenes.
- Si SMB seguro está habilitado, Trident ignorará el recurso compartido SMB mencionado en el backend.
- La actualización de la anotación de PVC, la anotación de la clase de almacenamiento y el campo backend no actualiza la ACL del recurso compartido SMB.
- La ACL de recurso compartido SMB especificada en la anotación del PVC clonado tendrá prioridad sobre las del PVC de origen.
- Asegúrese de proporcionar usuarios de AD válidos al habilitar SMB seguro. Los usuarios no válidos no se añadirán a la ACL.
- Si proporciona el mismo usuario de AD en el backend, la clase de almacenamiento y PVC con diferentes permisos, la prioridad de permisos será: PVC, clase de almacenamiento y luego backend.
- SMB seguro es compatible con `ontap-nas` importaciones de volumen gestionadas y no aplicable a importaciones de volumen no gestionadas.

Pasos

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

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.193.176.x
  svm: svm0
  useREST: true
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

2. Añade la anotación en la clase de almacenamiento.

   Añade el `trident.netapp.io/smbShareAdUser` Anotación a la clase de almacenamiento para habilitar SMB seguro sin fallos. El valor de usuario especificado para la anotación `trident.netapp.io/smbShareAdUser` debe ser el mismo que el nombre de usuario especificado en el `smbcreds` secreto. es `full_control` .

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

1. Cree una PVC.

   El siguiente ejemplo crea una PVC:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/snapshotDirectory: "true"
    trident.netapp.io/smbShareAccessControl: |
      read:
        - tridentADtest
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

Opciones y ejemplos de configuración NAS de ONTAP

Aprende a crear y usar controladores NAS de ONTAP con tu instalación de Trident. Esta sección ofrece ejemplos de configuración de backends y detalles para asignar backends a StorageClasses. A partir de la versión 25.10, NetApp Trident es compatible con "Sistemas de almacenamiento NetApp AFX". Los sistemas de almacenamiento NetApp AFX se diferencian de otros sistemas basados en ONTAP (ASA, AFF y FAS) en la implementación de su capa de almacenamiento.

NOTA	Solo el `ontap-nas` El controlador (con protocolo NFS) es compatible con los sistemas NetApp AFX; el protocolo SMB no es compatible.

Opciones de configuración del back-end

En la configuración del backend de Trident, no necesitas especificar que tu sistema es un sistema de almacenamiento AFX de NetApp. Cuando seleccionas `ontap-nas` como `storageDriverName`, Trident detecta automáticamente el sistema de almacenamiento AFX. Algunos parámetros de configuración del backend no aplican a los sistemas de almacenamiento AFX.

La siguiente tabla muestra las opciones de configuración del backend:

Parámetro	Descripción	Predeterminado
`version`		Siempre 1

Parámetro	Descripción	Predeterminado
`storageDriverName`	Nombre del controlador de almacenamiento **NOTA** Para sistemas NetApp AFX, solo `ontap-nas` está soportado.	`ontap-nas`, , `ontap-nas-economy` o. `ontap-nas-flexgroup`
`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). Se puede configurar para utilizar direcciones IPv6 si Trident se instaló con el indicador 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 Ejemplo de MetroCluster.	«10,0.0,1», «[2001:1234:abcd::fefe]»
`dataLIF`	Dirección IP de LIF de protocolo. NetApp recomienda especificar `dataLIF`. Si no se proporciona, Trident recupera las LIF de datos de la SVM. Puede especificar un nombre de dominio completo (FQDN) que se utilice para las operaciones de montaje de NFS, lo que permite crear un DNS por turnos para equilibrar la carga en varias LIF de datos. Se puede cambiar después del ajuste inicial. Consulte . Se puede configurar para utilizar direcciones IPv6 si Trident se instaló con el indicador IPv6. Las direcciones IPv6 deben definirse entre corchetes, `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` como . **Omitir para MetroCluster.** Consulte la Ejemplo de MetroCluster.	Dirección especificada o derivada de la SVM, si no se especifica (no recomendada)
`svm`	Máquina virtual de almacenamiento que usar **Omitir para MetroCluster.** Ver Ejemplo de MetroCluster.	Derivado si una SVM `managementLIF` está especificado
`autoExportPolicy`	Habilite la creación y actualización automática de la política de exportación [Boolean]. Mediante las `autoExportPolicy` opciones y `autoExportCIDRs`, 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`, 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	""

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`	Nombre de usuario para conectarse al clúster/SVM. Se utiliza para la autenticación basada en credenciales. Para la autenticación de Active Directory, consulte "Autenticar Trident en un SVM backend mediante credenciales de Active Directory".	
`password`	Contraseña para conectarse al cluster/SVM. Se utiliza para la autenticación basada en credenciales. Para la autenticación de Active Directory, consulte "Autenticar Trident en un SVM backend mediante credenciales de Active Directory".	
`storagePrefix`	El prefijo que se utiliza cuando se aprovisionan volúmenes nuevos en la SVM. No se puede actualizar después de configurarlo **NOTA** Al utilizar ONTAP-nas-economy y un prefijo de almacenamiento con 24 caracteres o más, los qtrees no tendrán el prefijo de almacenamiento incrustado, pero estarán en el nombre del volumen.	«trident»

Parámetro	Descripción	Predeterminado
`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. Si no se asigna, cualquiera de los agregados disponibles puede usarse para aprovisionar un volumen FlexGroup. **NOTA** Cuando el agregado se actualiza en SVM, se actualiza automáticamente en Trident sondeando SVM sin tener que reiniciar la controladora Trident. Cuando se haya configurado un agregado específico en Trident para aprovisionar volúmenes, si se cambia el nombre de este agregado o se saca de la SVM, el back-end se moverá al estado Failed en Trident mientras se sondea el agregado de SVM. Debe cambiar el agregado por uno presente en la SVM o quitarlo por completo para que el back-end vuelva a estar en línea. **No especificar para sistemas de almacenamiento AFX.**	""
`limitAggregateUsage`	Fallará el aprovisionamiento si el uso supera este porcentaje. **No se aplica a Amazon FSx para ONTAP**. **No especificar para sistemas de almacenamiento AFX.**	"" (no se aplica de forma predeterminada)

Parámetro	Descripción	Predeterminado
Lista de Agregados de Flexgroup	Lista de agregados para el aprovisionamiento (opcional; si se ha definido, debe asignarse a la SVM). Todos los agregados asignados a la SVM se usan para aprovisionar un volumen FlexGroup. Compatible con el controlador de almacenamiento **ONTAP-nas-FlexGroup**. **NOTA** Cuando la lista de agregados se actualiza en SVM, la lista se actualiza automáticamente en Trident sondeando la SVM sin tener que reiniciar la controladora Trident. Cuando se configuró una lista de agregado específica en Trident para aprovisionar volúmenes, si se cambia el nombre de la lista de agregados o se sale de SVM, el back-end se moverá al estado Failed en Trident mientras se sondea el agregado de SVM. Debe cambiar la lista de agregados por una que esté presente en la SVM o quitarla por completo para que el back-end vuelva a estar en línea.	""
`limitVolumeSize`	Fallará el aprovisionamiento si el tamaño de volumen solicitado supera este valor.	" (no se aplica por defecto)
`debugTraceFlags`	Indicadores de depuración que se deben usar para la solución de problemas. Ejemplo, {«api»:false, «method»:true} No utilizar `debugTraceFlags` a menos que esté solucionando problemas y necesite un volcado de registro detallado.	nulo
`nasType`	Configure la creación de volúmenes NFS o SMB. Las opciones son `nfs` , `smb` o nulo. Si se establece en nulo, se utilizarán volúmenes NFS por defecto. **Si se especifica, siempre se establecerá en `nfs` para sistemas de almacenamiento AFX**.	`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 las clases de almacenamiento, pero si no se especifican opciones de montaje en una clase de almacenamiento, Trident volverá a utilizar 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 almacenamiento o el archivo de configuración, Trident no definirá 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 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 utilizar las API REST de ONTAP . `useREST`Cuando se configura para `true` Trident utiliza las API REST de ONTAP para comunicarse con el backend; cuando se configura en `false` Trident utiliza llamadas ONTAPI (ZAPI) para comunicarse con el backend. Esta función requiere ONTAP 9.11.1 y versiones posteriores. Además, el rol de inicio de sesión de ONTAP utilizado debe tener acceso a `ontapi` solicitud. Esto se satisface mediante lo predefinido. `vsadmin` y `cluster-admin` roles. A partir de la versión Trident 24.06 y ONTAP 9.15.1 o posterior, `useREST` está configurado para `true` por defecto; cambiar `useREST` a `false` para utilizar llamadas ONTAPI (ZAPI). **Si se especifica, siempre se establecerá en `true` para sistemas de almacenamiento AFX**.	`true` Para ONTAP 9.15.1 o posterior, de lo contrario `false`.
`limitVolumePoolSize`	Tamaño máximo de FlexVol que se puede solicitar cuando se utilizan qtrees en el back-end económico de ONTAP-nas.	"" (no se aplica de forma predeterminada)
`denyNewVolumePools`	Restringe `ontap-nas-economy` los back-ends de la creación de nuevos volúmenes de FlexVol para contener sus Qtrees. Solo se utilizan los FlexVols preexistentes para aprovisionar nuevos VP.	

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

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

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

Parámetro	Descripción	Predeterminado
`spaceAllocation`	Asignación de espacio para Qtrees	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 Trident será habilitado NAE. Para obtener más información, consulte: "Cómo funciona Trident con NVE y NAE".	"falso"
`tieringPolicy`	Política de organización en niveles para utilizar ninguna	
`unixPermissions`	Modo para volúmenes nuevos	«777» para volúmenes NFS; vacío (no aplicable) para volúmenes SMB
`snapshotDir`	Controla el acceso al `.snapshot` directorio	`true`, `false` (establecer explícitamente).

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

NOTA Usar grupos de políticas de QoS con Trident requiere ONTAP 9 Intersight 8 o posterior. Debe usar 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 aplica el techo máximo para el rendimiento total de todas las cargas de trabajo.

Ejemplos de aprovisionamiento de volúmenes

Aquí hay un ejemplo con los valores predeterminados definidos:

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

Para `ontap-nas` y `ontap-nas-flexgroups` Trident ahora utiliza un nuevo cálculo para garantizar que el

FlexVol tenga el tamaño correcto con el porcentaje de snapshotReserve y el PVC. Cuando el usuario solicita un PVC, Trident crea el FlexVol original con más espacio mediante el nuevo cálculo. Este cálculo garantiza que el usuario reciba el espacio de escritura que solicitó en el PVC, y no menos espacio del que solicitó. Antes de la versión v21.07, cuando el usuario solicitaba un PVC (por ejemplo, 5 GiB), con el snapshotReserve al 50 %, obtenía solo 2,5 GiB de espacio de escritura. Esto se debe a que lo que el usuario solicitó fue el volumen completo y `snapshotReserve` es un porcentaje de eso. Con Trident 21.07, lo que el usuario solicita es el espacio de escritura y Trident define el `snapshotReserve` número como porcentaje del volumen total. Esto no se aplica a `ontap-nas-economy` . Vea el siguiente ejemplo para ver cómo funciona

El cálculo es el siguiente:

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

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

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

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

Ejemplos de configuración mínima

Los 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.

NOTA	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 NAS de ONTAP

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

Ejemplo de 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 del switchover y el switchover durante "Replicación y recuperación de SVM".

Para obtener una conmutación de sitios y una conmutación de estado sin problemas, especifique la SVM con `managementLIF` y omita la `dataLIF` y.. `svm` parámetros. 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 mínima. `clientCertificate`, `clientPrivateKey`, y. `trustedCACertificate` (Opcional, si se utiliza una CA de confianza) se completan en `backend.json` Y tome los valores codificados base64 del certificado de cliente, la clave privada y el certificado de CA de confianza, respectivamente.

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

Ejemplo de política de exportación automática

En este ejemplo, se muestra cómo puede indicar a Trident que utilice políticas de exportación dinámicas para crear y gestionar la política de exportación automáticamente. 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 muestra `managementLIF` Uso de una dirección IPv6.

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

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

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

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

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}}_{{.vo\
    lume.RequestName}}"
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` en ninguno, `spaceAllocation` en falso, y. `encryption` en falso. Los pools virtuales se definen en la sección de almacenamiento.

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

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

Ejemplo de NAS de ONTAP

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

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

Ejemplo de FlexGroup NAS de ONTAP

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

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

Ejemplo de economía NAS de ONTAP

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

```
      encryption: "false"
      unixPermissions: "0775"
```

Asigne los back-ends a StorageClass

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

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

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

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

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

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

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

- El `protection-silver-creditpoints-20k` StorageClass se asignará al tercer pool virtual en el `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"
```

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

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

Trident decidirá qué pool virtual se selecciona y garantiza que se cumpla el requisito de almacenamiento.

Actualizar `dataLIF` tras la configuración inicial

Puede cambiar la LIF de datos después de la configuración inicial. Para ello, ejecute el siguiente comando para proporcionar el nuevo archivo JSON de back-end con dataLIF actualizado.

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

NOTA Si los RVP están conectados a uno o varios POD, debe desactivar todos los POD correspondientes y a continuación volver a eliminarlos para que el nuevo LIF de datos entre en vigor.

Ejemplos de PYMES seguras

Configuración del backend con el controlador ontap-nas

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Configuración del backend con el controlador ontap-nas-economy

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas-economy
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Configuración de backend con pool de almacenamiento

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm0
  useREST: false
  storage:
  - labels:
      app: msoffice
    defaults:
      adAdminUser: tridentADuser
  nasType: smb
  credentials:
    name: backend-tbc-ontap-invest-secret
```

Ejemplo de clase de almacenamiento con controlador ontap-nas

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADtest
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

NOTA Asegúrese de agregar `annotations` Para habilitar SMB seguro. El SMB seguro no funciona sin las anotaciones, independientemente de la configuración establecida en el backend o PVC.

Ejemplo de clase de almacenamiento con controlador ontap-nas-economy

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser3
parameters:
  backendType: ontap-nas-economy
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

Ejemplo de PVC con un solo usuario de AD

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      change:
        - tridentADtest
      read:
        - tridentADuser
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

Ejemplo de PVC con múltiples usuarios de AD

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-test-pvc
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      full_control:
        - tridentTestuser
        - tridentuser
        - tridentTestuser1
        - tridentuser1
      change:
        - tridentADuser
        - tridentADuser1
        - tridentADuser4
        - tridentTestuser2
      read:
        - tridentTestuser2
        - tridentTestuser3
        - tridentADuser2
        - tridentADuser3
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

Amazon FSX para ONTAP de NetApp

Utiliza Trident con Amazon FSx para NetApp ONTAP

"Amazon FSX para ONTAP de NetApp" es un servicio de AWS totalmente administrado que ejecuta sistemas de archivos impulsados por el sistema operativo de almacenamiento NetApp ONTAP. Proporciona características, rendimiento y administración de ONTAP con la escalabilidad y la simplicidad operativa de AWS. Un sistema de archivos es el recurso principal en Amazon FSx y es análogo a un clúster ONTAP local. Cada sistema de archivos contiene una o varias máquinas virtuales de almacenamiento (SVM), y cada SVM contiene uno o varios volúmenes que almacenan archivos y directorios. Esta integración permite a los clústeres de Kubernetes que se ejecutan en Amazon Elastic Kubernetes Service (EKS) aprovisionar volúmenes persistentes respaldados por ONTAP para cargas de trabajo de bloques y archivos.

Requisitos

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

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

> **NOTA** Asegúrese de seguir los pasos de preparación de nodos necesarios para Amazon Linux y Ubuntu "Imágenes de máquina de Amazon" (AMI) en función del tipo de IAM EKS.

Consideraciones

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

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

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

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

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

Autenticación

Trident ofrece dos modos de autenticación.

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

ADVERTENCIA	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 Trident.

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

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

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

Imágenes de máquina de Amazon probadas (AMI)

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

IAM	NAS	Economía NAS	ISCSI	Economía iSCSI
AL2023_x86_64_STANDARD	Sí	Sí	Sí	Sí
AL2_x86_64	Sí	Sí	Sí*	Sí*
BOTTLEROCKET_x86_64	Sí**	Sí	NO DISPONIBLE	NO DISPONIBLE
AL2023_ARM_64_STANDARD	Sí	Sí	Sí	Sí
AL2_ARM_64	Sí	Sí	Sí*	Sí*
BOTTLEROCKET_ARM_64	Sí**	Sí	NO DISPONIBLE	NO DISPONIBLE

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

NOTA	Si su AMI deseado no aparece aquí, no significa que no sea compatible; simplemente significa que no se ha probado. Esta lista sirve como guía para las AMI que se sabe que funcionan.

Pruebas realizadas con:

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

Pruebas realizadas:

- Crear: Clase de almacenamiento, pvc, pod
- Eliminar: Pod, pvc (normal, qtree/lun: Economía, NAS con backup de AWS)

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.

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

Crear secreto de AWS Secrets Manager

Como Trident emitirá API con un Vserver FSx para gestionar el almacenamiento por usted, necesitará credenciales para hacerlo. La forma segura de pasar esas credenciales es a través de un secreto de AWS Secrets Manager. Por lo tanto, si aún no tiene uno, deberá crear un secreto de AWS Secrets Manager que contenga las credenciales de la cuenta vsadmin.

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

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

Crear política de IAM

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

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
Secrets manager"
```

Ejemplo de Política 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"
}
```

Crear una identidad de pod o un rol de IAM para la asociación de cuentas de servicio (IRSA)

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

Identidad de pod

Las asociaciones de identidad de pod de Amazon EKS brindan la capacidad de administrar credenciales para sus aplicaciones, de manera similar a la forma en que los perfiles de instancia de Amazon EC2 proporcionan credenciales a las instancias de Amazon EC2.

Instalar Pod Identity en tu clúster EKS:

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

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

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

Crear trust-relationship.json:

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

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

archivo trust-relationship.json:

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

Adjuntar la política de rol al rol de IAM:

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

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

Crear una asociación de identidad de pod:

Cree una asociación de identidad de pod entre el rol de IAM y la cuenta de servicio Trident (trident-controller)

```
aws eks create-pod-identity-association \
  --cluster-name <EKS_CLUSTER_NAME> \
  --role-arn arn:aws:iam::111122223333:role/fsxn-csi-role \
  --namespace trident --service-account trident-controller
```

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

Usando la AWS CLI:

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

archivo trust-relationship.json:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::<account_id>:oidc-
provider/<oidc_provider>"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "<oidc_provider>:aud": "sts.amazonaws.com",
          "<oidc_provider>:sub":
"system:serviceaccount:trident:trident-controller"
        }
      }
    }
  ]
}
```

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

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

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

Adjuntar el rol de IAM con la política de IAM:

Una vez creado el rol, adjunte la política (que se creó en el paso anterior) al rol mediante este comando:

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

Verificar que el proveedor de OICD está asociado:

Verifique que su proveedor de OIDC está asociado al clúster. Puede verificarlo con este comando:

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

Si la salida está vacía, utilice el siguiente comando para asociar IAM OIDC al cluster:

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

Si está usando eksctl, utilice el siguiente ejemplo para crear una función de IAM para una cuenta de servicio en EKS:

```
eksctl create iamserviceaccount --name trident-controller --namespace
trident \
  --cluster <my-cluster> --role-name AmazonEKS_FSxN_CSI_DriverRole
--role-only \
  --attach-policy-arn <IAM-Policy ARN> --approve
```

Instale Trident

Trident agiliza la administración del almacenamiento de Amazon FSx para NetApp ONTAP en Kubernetes para que tus desarrolladores y administradores se enfoquen en la puesta en marcha de aplicaciones. Puedes instalar Trident usando uno de los siguientes métodos:

- Timón
- Complemento EKS

Si desea utilizar la funcionalidad Snapshot, instale el complemento del controlador de instantáneas CSI. Consulte "Habilite la funcionalidad Snapshot para volúmenes CSI" si desea obtener más información.

Instale Trident a través del timón

Identidad de pod

1. Añada el repositorio Helm de Trident:

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

2. Instale Trident utilizando el siguiente ejemplo:

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

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-
100.2502.0    25.02.0
```

Asociación de cuentas de servicio (IRSA)

1. Añada el repositorio Helm de Trident:

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

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

```
helm install trident-operator netapp-trident/trident-operator
--version 100.2502.1  \
--set cloudProvider="AWS" \
--set cloudIdentity="'eks.amazonaws.com/role-arn:
arn:aws:iam::<accountID>:role/<AmazonEKS_FSxN_CSI_DriverRole>'" \
--namespace trident \
--create-namespace
```

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-
100.2510.0    25.10.0
```

NOTA

Si planea usar iSCSI, asegúrese de que esté habilitado en su equipo cliente. Si usa el sistema operativo de nodo de trabajo AL2023, puede automatizar la instalación del cliente iSCSI añadiendo el parámetro node prep en la instalación de Helm:

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

Instale Trident a través del complemento EKS

El complemento Trident EKS incluye los parches de seguridad más recientes, correcciones de errores y está validado por AWS para funcionar 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

Asegúrese de tener lo siguiente antes de configurar el complemento 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

Habilite el complemento Trident para AWS

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, seleccione **Clusters**.
3. Seleccione el nombre del cluster para el que desea configurar el complemento CSI de NetApp Trident.
4. Selecciona **Add-ons** y luego selecciona **Get more add-ons**.
5. Siga estos pasos para seleccionar el complemento:
   a. Desplácese hacia abajo hasta la sección **Complementos de AWS Marketplace** y escriba **"Trident"** en el cuadro de búsqueda.
   b. Seleccione la casilla de verificación en la esquina superior derecha del cuadro Trident by NetApp.
   c. Seleccione **Siguiente**.
6. En la página de configuración **Configure Selected add-ons**, haga lo siguiente:

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

   a. Seleccione la **Versión** que desea usar.
   b. Si utiliza la autenticación IRSA, asegúrese de establecer los valores de configuración disponibles en la configuración opcional:
      - Seleccione la **Versión** que desea usar.
      - Siga el **Esquema de configuración del complemento** y configure 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):

```
{

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

}
```

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

Para la identidad del pod, utilice el siguiente formato:

NOTA Usa la

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

Para la autenticación de IRSA, utilice el siguiente formato:

```
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v25.6.0-eksbuild.1",
  "serviceAccountRoleArn": "<role ARN>",
  "configurationValues": {
    "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'",
    "cloudProvider": "AWS"
  }
}
```

NOTA Reemplace `<role ARN>` por el ARN del rol que se creó en el paso anterior.

2. Instalar el complemento Trident EKS.

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

eksctl

El siguiente comando de ejemplo instala el complemento Trident EKS:

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

Actualice el complemento Trident EKS

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, seleccione **Clusters**.
3. Seleccione el nombre del cluster para el que desea actualizar el complemento CSI de NetApp Trident.
4. Seleccione la pestaña **Add-ons**.
5. Selecciona **Trident by NetApp** y luego selecciona **Editar**.
6. En la página **Configure Trident by NetApp**, haga lo siguiente:
   a. Seleccione la **Versión** que desea usar.
   b. Expanda la **Configuración opcional** y modifique según sea necesario.
   c. Seleccione **Guardar cambios**.

CLI DE AWS

El siguiente ejemplo actualiza el complemento EKS:

```
aws eks update-addon --cluster-name <eks_cluster_name> --addon-name
netapp_trident-operator --addon-version v25.6.0-eksbuild.1 \
  --service-account-role-arn <role-ARN> --resolve-conflict preserve \
  --configuration-values "{\"cloudIdentity\":
\"'eks.amazonaws.com/role-arn: <role ARN>'\"}"
```

eksctl

- 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    v25.6.0-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
v25.6.0-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".

Desinstale/elimine el complemento 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 *: NetApp recomienda eliminar 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.

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

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, seleccione **Clusters**.
3. Seleccione el nombre del cluster para el que desea quitar el complemento CSI de NetApp Trident.
4. Selecciona la pestaña **Complementos** y luego selecciona **Trident by NetApp**.*
5. Seleccione **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. Seleccione **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
```

eksctl

El siguiente comando desinstala el complemento Trident EKS:

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

Configurar una clase de almacenamiento

El "Objeto de Kubernetes StorageClass" identifica a un proveedor y le indica cómo aprovisionar volúmenes. Esta sección te muestra cómo configurar un objeto de Kubernetes StorageClass que especifica a Trident como el proveedor.

Crear un objeto StorageClass

Cuando creas un StorageClass para FSx for ONTAP, Trident creará automáticamente la configuración del backend.

NOTA	Si quieres configurar manualmente el backend de almacenamiento, consulta la sección [create-a-kubernetes-storageclass-without-automatic-backend-configuration] para saber cómo crear el backend Trident y la clase de almacenamiento por separado.

Especifica los parámetros requeridos StorageClass

Los siguientes tres parámetros deben definirse al crear un StorageClass:

Parámetro	Obligatorio	Tipo	Descripción
`fsxFilesystemID`	Sí	cadena	FSx para NetApp ONTAP filesystem ID
`storageDriverName`	Sí	cadena	Controlador de almacenamiento Trident (por ejemplo, `ontap-nas` o `ontap-san`)
`credentialsName`	Sí	cadena	Nombre del secreto de Kubernetes que contiene FSx for ONTAP credenciales

Especifica parámetros opcionales

Puedes pasar parámetros opcionales de backend a través de StorageClass. Define todos los valores opcionales como cadenas en la sección StorageClass `parameters`. Para obtener una lista completa de los parámetros de backend, consulta: "Configuración del backend de FSx para NetApp ONTAP".

Ejemplo de archivos de configuración de StorageClass.

El siguiente ejemplo muestra un StorageClass que activa la configuración automática del backend.

YAML

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-fsx-demo
  annotations:
    description: "Demo StorageClass for FSx for NetApp ONTAP"
provisioner: csi.trident.netapp.io
parameters:
  fsxFilesystemID: "fs-0abc123"
  storageDriverName: "ontap-nas"
  credentialsName: trident-fsx-credentials
allowVolumeExpansion: true
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

JSON

```json
{
  "apiVersion": "storage.k8s.io/v1",
  "kind": "StorageClass",
  "metadata": {
    "name": "ontap-fsx-demo",
    "annotations": {
      "description": "Demo StorageClass for FSx for NetApp ONTAP"
    }
  },
  "provisioner": "csi.trident.netapp.io",
  "parameters": {
    "fsxFilesystemID": "fs-0abc123",
    "storageDriverName": "ontap-nas",
    "credentialsName": "trident-fsx-credentials"
  },
  "allowVolumeExpansion": true,
  "reclaimPolicy": "Delete",
  "volumeBindingMode": "Immediate"
}
```

Crear el StorageClass

Una vez que hayas creado tu archivo de configuración, ejecuta el siguiente comando para crear la clase de almacenamiento.

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

Ahora deberías ver una clase de almacenamiento **basic-csi** tanto en Kubernetes como en Trident, y Trident debería haber descubierto los pools en el back-end.

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

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

Después de aplicar StorageClass, Trident crea el backend automáticamente. Luego, puedes crear PersistentVolumeClaims que hacen referencia a este StorageClass.

Verifica el estado de la configuración del backend

Trident registra el resultado de la creación del backend en las anotaciones de StorageClass.

Anotación	Descripción
`trident.netapp.io/configuratorStatus`	Resultado de la configuración (`Success` o `Failure`)
`trident.netapp.io/configuratorMessage`	Estado detallado o mensaje de error
`trident.netapp.io/configuratorName`	Nombre del recurso interno del configurador
`trident.netapp.io/managed`	Indica que el StorageClass está gestionado por Trident
`trident.netapp.io/additionalStoragePools`	Grupos de almacenamiento creados para este backend

Para verificar el estado, ejecuta:

```
kubectl get storageclass ontap-fsx-demo -o yaml
```

Confirma que `trident.netapp.io/configuratorStatus` está configurado en `Success`. Si el valor es `Failure`, revisa `trident.netapp.io/configuratorMessage` para ver el error.

Añade sistemas de archivos FSxN adicionales

Si necesitas capacidad de almacenamiento adicional mientras sigues usando el mismo StorageClass, añade ID de sistemas de archivos FSxN adicionales.

Edita el StorageClass y añade la siguiente anotación:

```
metadata:
  annotations:
    trident.netapp.io/additionalFsxNFileSystemID: '["fs-
xxxxxxxxxxxxxxxxx"]'
```

Después de aplicar el cambio, Trident actualiza la configuración del backend y actualiza las anotaciones de StorageClass.

Consideraciones operativas y limitaciones

- Eliminar un StorageClass que tiene la configuración automática de backend normalmente elimina el backend Trident asociado. Esto puede interrumpir la conectividad de almacenamiento y afectar las cargas de trabajo en ejecución. Valida el impacto antes de eliminar un StorageClass administrado.
- La configuración automática del backend solo es compatible con AWS FSx for NetApp ONTAP.

Crear un Kubernetes StorageClass sin configuración automática del backend

Si quieres crear el backend de Trident y StorageClass por separado, entonces sigue estos pasos.

Entiende cómo funciona la configuración automática del backend

Trident obtiene la configuración del backend a partir de la definición de StorageClass. Cuando aplicas la StorageClass, Trident valida los parámetros necesarios, crea el backend y anota la StorageClass con el estado.

Trident crea el VolumeSnapshotClass solo una vez. Trident reutiliza el mismo VolumeSnapshotClass para los siguientes StorageClasses.

Crear el backend Trident

Para crear un backend Trident, necesitas crear un archivo de configuración en formato JSON o YAML. El archivo debe especificar el tipo de almacenamiento que quieres (NAS o SAN), el sistema de archivos, la SVM de la que lo vas a obtener y cómo autenticarte con ella. El siguiente ejemplo muestra cómo definir almacenamiento basado en NAS y usar un secreto de AWS para guardar las credenciales de la SVM que quieres usar:

YAML

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

JSON

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

FSX para ONTAP detalles del controlador

Puedes integrar Trident con Amazon FSx for NetApp ONTAP mediante los siguientes controladores:

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

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

Cree el back-end

Después de crear el archivo de configuración, ejecuta los siguientes comandos para crear y validar la configuración del backend de Trident (TBC):

- Cree la configuración de backend de Trident (TBC) desde el archivo yaml y ejecute el siguiente comando:

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

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

- Validar que la configuración de backend de Trident (TBC) se ha creado correctamente:

```
Kubectl get tbc -n trident
```

```
NAME                          BACKEND NAME          BACKEND UUID
PHASE   STATUS

backend-tbc-ontap-nas         tbc-ontap-nas         933e0071-66ce-4324-
b9ff-f96d916ac5e9   Bound   Success
```

Para más información sobre otras opciones de configuración, consulta la sección [Backend-advanced-configuration-and-examples] a continuación.

Configura una clase de almacenamiento sin configuración automática del backend

A continuación se muestran ejemplos de configuraciones de Storage Class para su uso con Trident y FSx for ONTAP.

Clase de almacenamiento para NFS

Puedes usar este ejemplo para configurar StorageClass para volúmenes que usan NFS (consulta la sección de atributos de Trident a continuación para la lista completa de atributos):

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

Storage Calss para iSCSI

Usa este ejemplo para configurar StorageClass para volúmenes que usan iSCSI:

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

Clase de almacenamiento con NFSv3 y AWS Bottlerocket

Para aprovisionar volúmenes de NFSv3 TB en AWS Bottlerocket, agregue el necesario `mountOptions` a la clase de almacenamiento:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  media: "ssd"
  provisioningType: "thin"
  snapshots: "true"
mountOptions:
  - nfsvers=3
  - nolock
```

Atributos de Trident StorageClass

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, 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
clones	bool	verdadero, falso	Pool admite el clonado de volúmenes	Volumen con clones habilitados	ontap-nas, ontap-san, solidfire-san

Atributo	Tipo	Valores	Oferta	Solicitud	Admitido por
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

[1]: No soportado por ONTAP Select o FSx for ONTAP systems

Consulte el "Objetos de Kubernetes y Trident" para obtener más detalles sobre cómo interactúan las clases de almacenamiento con los `PersistentVolumeClaim` parámetros y para controlar la forma en que Trident aprovisiona los volúmenes.

Crear la clase de almacenamiento

Una vez que hayas configurado StorageClass, puedes crearlo en Kubernetes.

Pasos

1. Este es un objeto de Kubernetes, así que use `kubectl` Para crear en Kubernetes.

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

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

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

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

aprovisionamiento de volúmenes SMB

Puedes aprovisionar volúmenes SMB usando el controlador `ontap-nas`. Sin embargo, para hacerlo debes completar estos pasos: "Prepárese para aprovisionar los volúmenes de SMB".

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

Parámetro	Descripción	Ejemplo
`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). Se puede configurar para utilizar direcciones IPv6 si Trident se instaló con el indicador 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 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]»

Parámetro	Descripción	Ejemplo
`dataLIF`	Dirección IP del protocolo LIF. **ONTAP NAS drivers**: NetApp recomienda especificar dataLIF. Si no se proporciona, Trident obtiene los dataLIF de la SVM. Puedes especificar un nombre de dominio completo (FQDN) para las operaciones de montaje NFS, lo que te permite crear un DNS round-robin para equilibrar la carga entre varios dataLIF. Se puede cambiar después de la configuración inicial. **ONTAP SAN drivers**: no especificar para iSCSI. Trident usa ONTAP Selective LUN Map para descubrir los iSCSI LIF necesarios para establecer una sesión multipath. Se genera una advertencia si dataLIF se define explícitamente. Se puede configurar para usar direcciones IPv6 si Trident se instaló usando el indicador IPv6. Las direcciones IPv6 deben definirse entre corchetes, como [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].	
`autoExportPolicy`	Habilite la creación y actualización automática de la política de exportación [Boolean]. Mediante las `autoExportPolicy` opciones y `autoExportCIDRs`, 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`, Trident puede gestionar automáticamente las políticas de exportación.	«[«0,0.0,0/0»::/0»]»
`labels`	Conjunto de etiquetas con formato JSON arbitrario que se aplica en los volúmenes	""
`clientCertificate`	Valor codificado en base64 del certificado de cliente. Se utiliza para autenticación basada en certificados	""

Parámetro	Descripción	Ejemplo
`clientPrivateKey`	Valor codificado en base64 de la clave privada de cliente. Se utiliza para autenticación basada en certificados	""
`trustedCACertificate`	Valor codificado en base64 del certificado de CA de confianza. Opcional. Se utiliza para autenticación basada en certificados.	""
`username`	El nombre de usuario para conectarse al clúster o SVM. Se utiliza para autenticación basada en credenciales. Por ejemplo, vsadmin.	
`password`	La contraseña para conectarse al clúster o SVM. Se utiliza para autenticación basada en credenciales.	
`svm`	Máquina virtual de almacenamiento que usar	Derivado si se especifica una LIF de gestión de SVM.
`storagePrefix`	El prefijo que se utiliza cuando se aprovisionan volúmenes nuevos en la SVM. No se puede modificar una vez creada. Para actualizar este parámetro, deberá crear un nuevo backend.	`trident`
`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 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 volume	"" (no se aplica de forma predeterminada)
`lunsPerFlexvol`	El número máximo de LUN por FlexVol volume debe estar comprendido entre [50, 200]. Solo SAN.	«100»

Parámetro	Descripción	Ejemplo
`debugTraceFlags`	Indicadores de depuración que se deben usar para la solución de problemas. Ejemplo, {«api»:false, «method»:true} No utilizar `debugTraceFlags` a menos que esté solucionando problemas y necesite un volcado de registro detallado.	nulo
`nfsMountOptions`	Lista de opciones de montaje NFS separadas por comas. Las opciones de montaje para los volúmenes persistentes de Kubernetes se especifican normalmente en las clases de almacenamiento, pero si no se especifican opciones de montaje en una clase de almacenamiento, Trident volverá a utilizar 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 almacenamiento o el archivo de configuración, Trident no definirá ninguna opción de montaje en un volumen persistente asociado.	""
`nasType`	Configure la creación de volúmenes NFS o SMB. Las opciones son `nfs`, `smb`, o nulo. **Debe establecer en `smb` Para volúmenes SMB.** el valor predeterminado es null en volúmenes NFS.	`nfs`
`qtreesPerFlexvol`	El número máximo de qtrees por FlexVol volume debe estar en el intervalo [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 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. Cuando se establece en `true`, Trident utilizará las API REST DE ONTAP para comunicarse con el backend. 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`	Puedes especificar lo siguiente en el archivo de configuración de AWS FSx para ONTAP: - `fsxFilesystemID`: Especifique el ID del sistema de archivos AWS FSx. - `apiRegion`: Nombre de la región de la API de AWS. - `apikey`: AWS API key. - `secretKey`: AWS clave secreta.	"" "" ""
`credentials`	Especifique las credenciales de FSX SVM para almacenarlas en AWS Secrets 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 utilizando estas opciones en la `defaults` sección de la configuración. Para ver un ejemplo, vea los ejemplos de configuración siguientes.

Parámetro	Descripción	Predeterminado
`spaceAllocation`	Asignación de espacio para las LUN	`true`
`spaceReserve`	Modo de reserva de espacio; «ninguno» (fino) o «volumen» (grueso)	`none`
`snapshotPolicy`	Política de Snapshot que se debe usar	`none`

Parámetro	Descripción	Predeterminado
`qosPolicy`	Grupo de políticas de calidad de servicio que se asignará a los volúmenes creados. Elija uno de qosPolicy o adaptiveQosPolicy por pool de almacenamiento o back-end. Usar grupos de políticas de QoS con Trident requiere ONTAP 9 Intersight 8 o posterior. Debe usar 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 aplica el techo máximo para el rendimiento total de todas las cargas de trabajo.	""
`adaptiveQosPolicy`	Grupo de políticas de calidad de servicio adaptativo que permite asignar los volúmenes creados. Elija uno de qosPolicy o adaptiveQosPolicy por pool de almacenamiento o back-end. no admitido por ontap-nas-Economy.	""
`snapshotReserve`	Porcentaje de volumen reservado para las 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 Trident será habilitado NAE. Para obtener más información, consulte: "Cómo funciona Trident con NVE y NAE".	`false`
`luksEncryption`	Activa el cifrado LUKS. Consulta "Usar la configuración de clave unificada de Linux (LUKS)". Solo para SAN.	""
`tieringPolicy`	Política de organización en niveles para utilizar `none`	

Parámetro	Descripción	Predeterminado
`unixPermissions`	Modo para volúmenes nuevos. **Dejar vacío para volúmenes SMB.**	""
`securityStyle`	Estilo de seguridad para nuevos volúmenes. Compatibilidad con NFS `mixed` y.. `unix` estilos de seguridad. SMB admite `mixed` y.. `ntfs` estilos de seguridad.	El valor predeterminado de NFS es `unix`. La opción predeterminada de SMB es `ntfs`.

Configura un PVC

Esta sección incluye instrucciones sobre cómo crear un PersistentVolumeClaim (PVC) que utiliza la StorageClass de Kubernetes configurada para solicitar un PV. Si tiene éxito, luego puedes montar el PV en un pod.

Cree la RVP

Una "*Claim de volumen persistente*" (PVC) es una solicitud de acceso al PersistentVolume en el clúster. El PVC se puede configurar para solicitar almacenamiento de un tamaño o modo de acceso determinados. Usando el StorageClass asociado, el administrador del clúster puede controlar más que solo el tamaño y el modo de acceso de PersistentVolume, como el rendimiento o el nivel de servicio.

Después de crear el backend Trident y StorageClass puedes crear un PVC. Después de crear el PVC, puedes montar el volumen en un pod.

Manifiestos de muestra

Los siguientes ejemplos muestran las opciones básicas de configuración del PVC.

PVC con acceso RWX

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: ontap-gold
```

Ejemplo de PVC usando iSCSI

Este ejemplo muestra un PVC básico para iSCSI con acceso RWO que está asociado con una StorageClass

denominada `protection-gold` .

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

Crear PVC

Pasos

1. Cree la RVP.

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

2. 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 los `PersistentVolumeClaim` parámetros y para controlar la forma en que Trident aprovisiona los volúmenes.

Implementa una aplicación

Cuando se crean la clase de almacenamiento y la RVP, puede montar el PV en un pod. Esta sección enumera el comando de ejemplo y la configuración para adjuntar el PV a un pod.

Implementa una aplicación de ejemplo

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
```

NOTA Puede supervisar el progreso con `kubectl get pod --watch`.

2. Compruebe que el volumen está montado en `/my/mount/path`.

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

```
Filesystem                                                      Size
Used  Avail Use%  Mounted on
192.168.188.78:/trident_pvc_ae45ed05_3ace_4e7c_9080_d2a83ae03d06  1.1G
320K  1.0G   1%   /my/mount/path
```

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

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

Configure el complemento Trident EKS en un clúster EKS

NetApp 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 NetApp Trident EKS incluye los parches de seguridad más recientes, correcciones de errores y está validado por AWS para funcionar 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.

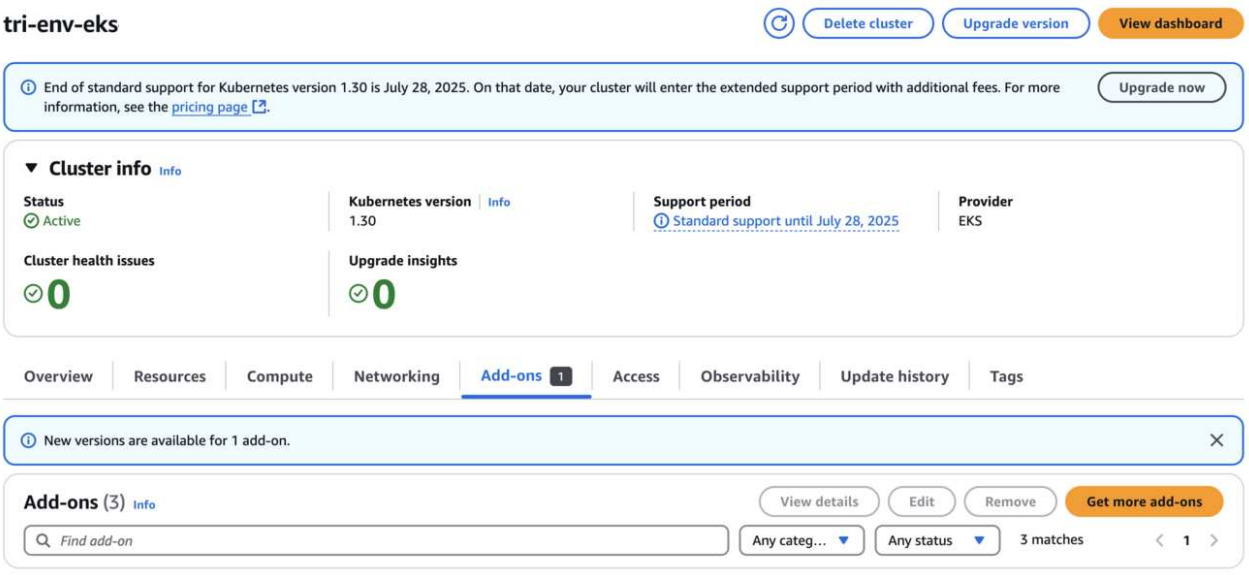
Requisitos previos

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

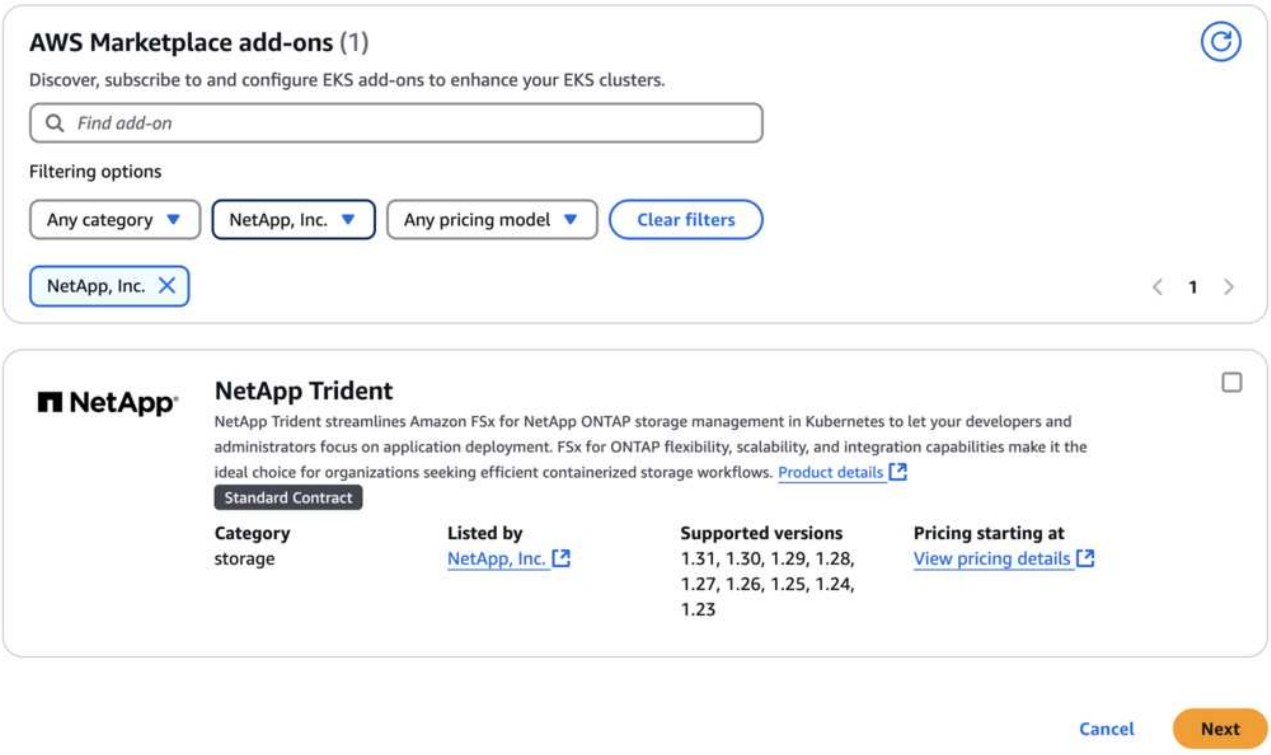
- Una cuenta de clúster de Amazon EKS con permisos para trabajar con complementos. Consulte "Complementos de Amazon EKS".
- Permisos de AWS para AWS Marketplace:
  ```
  "aws-marketplace:ViewSubscriptions",
  "aws-marketplace:Subscribe",
  "aws-marketplace:Unsubscribe
  ```
- Tipo de AMI: Amazon Linux 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

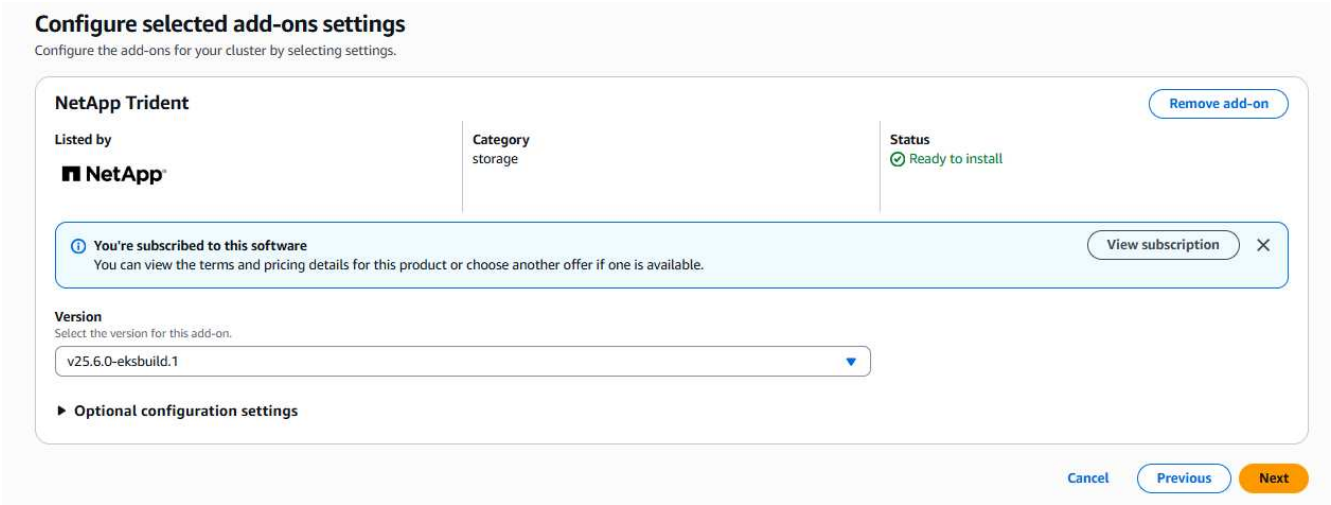
1. Asegúrese de crear el rol de IAM y el secreto de AWS para permitir que los pods de EKS accedan a los recursos de AWS. Para obtener instrucciones, consulte "Cree un rol de IAM y AWS Secret".
2. En tu clúster de Kubernetes de EKS, navega a la pestaña **Add-ons**.



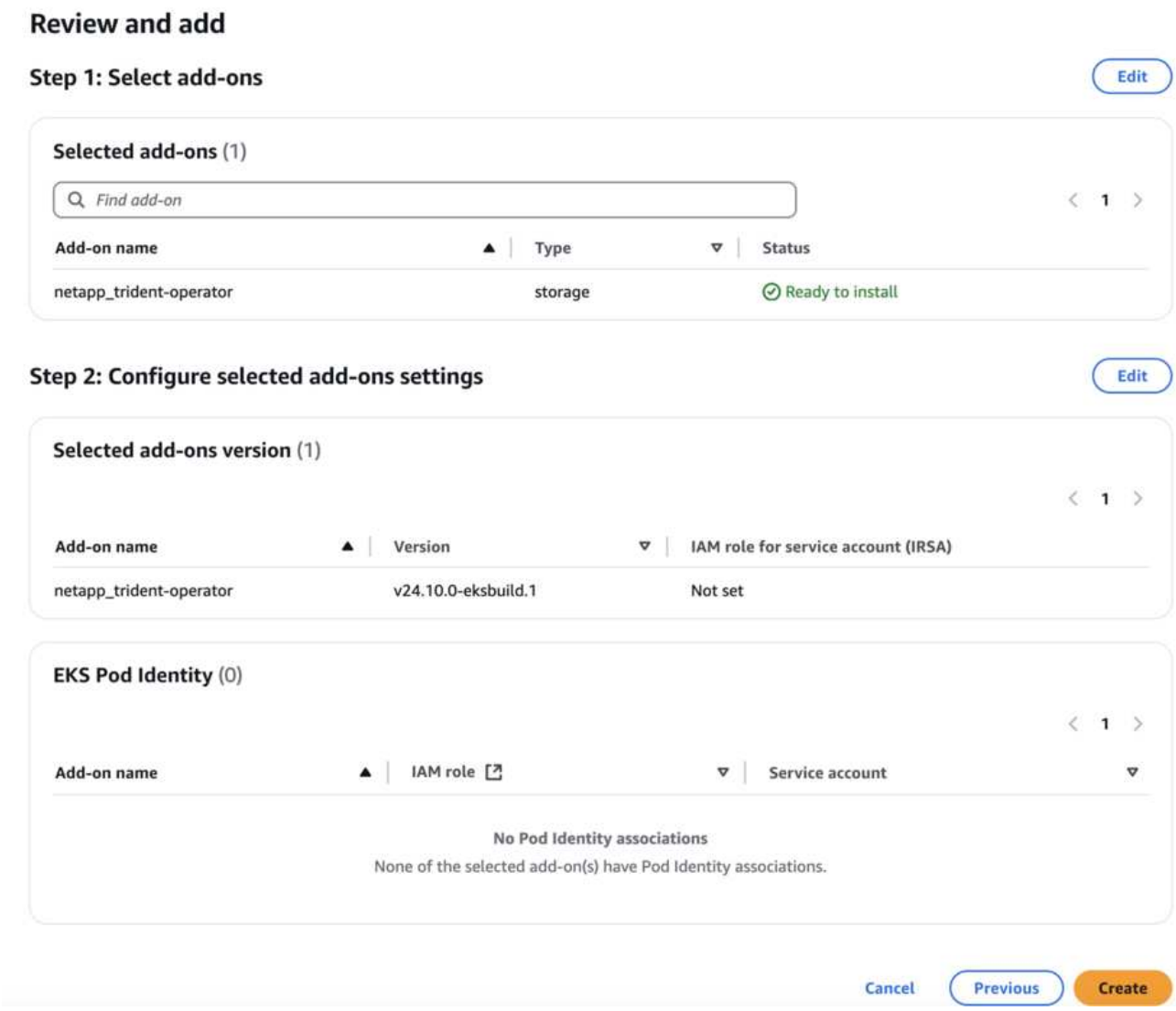
3. Vaya a **AWS Marketplace add-ons** y elija la categoría *storage*.



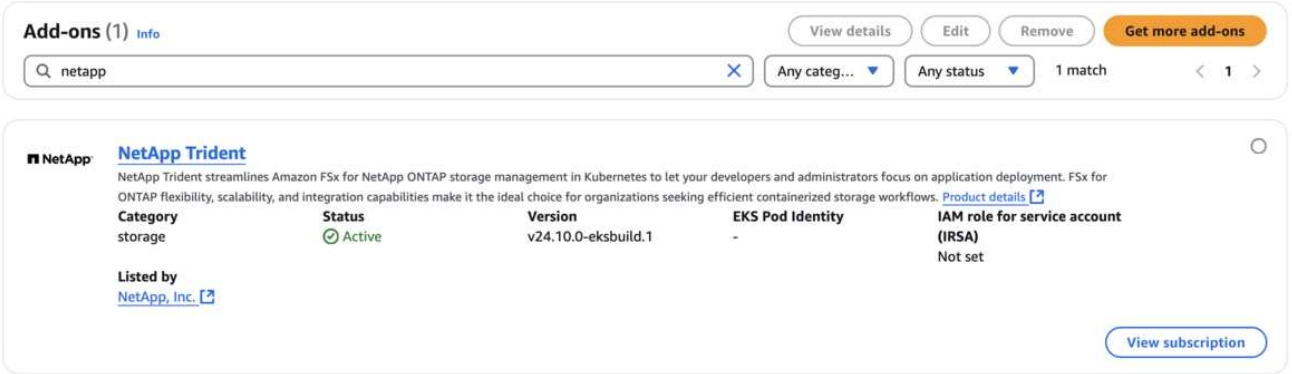
4. Localice **NetApp Trident** y seleccione la casilla de verificación para el complemento Trident, y haga clic en **Siguiente**.
5. Elija la versión deseada del complemento.



6. Configure los ajustes complementarios necesarios.



7. Si está utilizando IRSA (roles de IAM para cuentas de servicio), consulte los pasos de configuración adicionales"aquí" .
8. Seleccione **Crear**.
9. Compruebe que el estado del complemento es *Active*.



10. Ejecute el siguiente comando para comprobar que Trident está correctamente instalado en el clúster:

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

11. Continúa con la configuración y configura el backend de almacenamiento. Para más información, consulta "Configure el backend de almacenamiento".

Instale/desinstale el complemento Trident EKS mediante la interfaz de línea de comandos

Instale el complemento NetApp Trident EKS mediante la interfaz de línea de comandos:

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

El siguiente comando de ejemplo instala el complemento Trident EKS versión 25.6.1: `eksctl create addon --cluster clusterName --name netapp_trident-operator --version v25.6.1-eksbuild.1` (con una versión dedicada)

El siguiente comando de ejemplo instala el complemento Trident EKS versión 25.6.2: `eksctl create addon --cluster clusterName --name netapp_trident-operator --version v25.6.2-eksbuild.1` (con una versión dedicada)

Desinstale el complemento NetApp Trident EKS mediante la interfaz de línea de comandos:

El siguiente comando desinstala el complemento Trident EKS:

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

Cree back-ends con kubectl

Un back-end define la relación entre Trident y un sistema de almacenamiento. Indica a Trident cómo se comunica con ese sistema de almacenamiento y cómo debe aprovisionar volúmenes a partir de él. Después de instalar Trident, el siguiente paso es crear un backend. 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 permite gestionar los backend de Trident mediante `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`).

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

- Trident crea automáticamente un backend basado en la configuración que proporcione. Esto se representa internamente como un `TridentBackend` (`tbe`, `tridentbackend`) CR.
- El `TridentBackendConfig` está vinculado exclusivamente a una `TridentBackend` que fue creada por

Trident.

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

ADVERTENCIA	`TridentBackend` Trident crea los 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 ver el formato del `TridentBackendConfig` CR:

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

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

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

La `spec` la sección también incluye `credentials` y.. `deletionPolicy` campos, que se introducen recientemente en `TridentBackendConfig` CR:

- `credentials`: Este parámetro es un campo obligatorio y contiene las credenciales utilizadas para autenticarse con el sistema/servicio de almacenamiento. Este juego debe ser un secreto de Kubernetes creado por el usuario. Las credenciales no se pueden pasar en texto sin formato y se producirá un error.
- `deletionPolicy`: Este campo define lo que debe suceder cuando `TridentBackendConfig` se ha eliminado. Puede ser necesario uno de los dos valores posibles:
  - `delete`: Esto resulta en la eliminación de ambos `TridentBackendConfig` CR y el back-end asociado. Este es el valor predeterminado.
  - `retain`: Cuando un `TridentBackendConfig` Se elimina la CR, la definición de backend seguirá estando presente y se puede gestionar con `tridentctl`. Establecimiento de la política de eliminación como `retain` permite a los usuarios degradar a una versión anterior (anterior a 21.04) y conservar los back-ends creados. El valor de este campo se puede actualizar después de un

`TridentBackendConfig` se ha creado.

NOTA

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

CONSEJO

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.). Trident enlazará automáticamente los recién creados `TridentBackendConfig` con el backend preexistente.

Descripción general de los pasos

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

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

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

Paso 1: Cree un secreto de Kubernetes

Cree un secreto que contenga las credenciales de acceso para el back-end. Esto es único para cada servicio/plataforma de almacenamiento. Veamos un ejemplo:

```
kubectl -n trident create -f backend-tbc-ontap-san-secret.yaml
```

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-san-secret
type: Opaque
stringData:
  username: cluster-admin
  password: password
```

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

Descripción de campos secretos de la plataforma de almacenamiento	Secreto	Descripción de los campos
Azure NetApp Files	ID del Cliente	El ID de cliente de un registro de aplicación
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 el `spec.credentials` del `TridentBackendConfig` objeto creado en el paso siguiente.

Paso 2: Cree la `TridentBackendConfig` CR

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

```
kubectl -n trident create -f backend-tbc-ontap-san.yaml
```

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

Paso 3: Compruebe el estado del `TridentBackendConfig` CR

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

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

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

La fase puede tomar uno de los siguientes valores:

- `Bound`: La `TridentBackendConfig` CR está asociado con un backend, y ese backend contiene `configRef` establezca en la `TridentBackendConfig` UID de CR.
- `Unbound`: Representado usando `""`. La `TridentBackendConfig` el objeto no está enlazado a un back-end. Creadas recientemente `TridentBackendConfig` CRS se encuentra en esta fase de forma predeterminada. Tras cambiar la fase, no puede volver a «sin límites».
- `Deleting`: La `TridentBackendConfig` CR `deletionPolicy` se ha configurado para eliminar. Cuando la `TridentBackendConfig` La CR se elimina y pasa al estado de supresión.
  - Si no existen reclamaciones de volumen persistentes (RVP) en el back-end, si se elimina el Trident, tanto el `TridentBackendConfig` back-end como la `TridentBackendConfig` CR.

    - Si uno o más EVs están presentes en el backend, pasa a un estado de supresión. La `TridentBackendConfig` Posteriormente, CR también entra en fase de eliminación. El back-end y. `TridentBackendConfig` Se eliminan sólo después de que se hayan eliminado todas las EVs.

- `Lost`: El backend asociado con `TridentBackendConfig` La CR se eliminó accidental o deliberadamente y la `TridentBackendConfig` CR todavía tiene una referencia al backend eliminado. La `TridentBackendConfig` La CR puede ser eliminada independientemente de la `deletionPolicy` valor.
- `Unknown`: Trident no puede determinar el estado o la existencia del backend asociado al `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 se pueden realizar además, como "actualizaciones back-end y eliminaciones backend".

(Opcional) Paso 4: Obtener más detalles

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

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

```
NAME                    BACKEND NAME        BACKEND UUID
PHASE   STATUS    STORAGE DRIVER   DELETION POLICY
backend-tbc-ontap-san   ontap-san-backend   8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8   Bound   Success   ontap-san        delete
```

Además, también puede obtener un volcado YLMA/JSON de `TridentBackendConfig`.

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

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

`backendInfo` Contiene los `backendName` y el `backendUUID` del backend que se creó en respuesta a la `TridentBackendConfig` CR. El `lastOperationStatus` 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 Trident (por ejemplo, durante los reinicios de 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 el `kubectl -n trident describe tbc <tbc-cr-name>` comando para obtener detalles de los registros de eventos.

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

Gestionar back-ends

Realice la gestión del entorno de administración con kubectl

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

Eliminar un back-end

Al suprimir un `TridentBackendConfig`, indica a Trident que suprima/conserve los 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 asegura 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
```

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

Ver los back-ends existentes

Ejecute el siguiente comando:

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

También puede ejecutar `tridentctl get backend -n trident` o. `tridentctl get backend -o yaml -n trident` obtener una lista de todos los back-ends que existen. Esta lista también incluirá los back-ends que se crearon con `tridentctl`.

Actualizar un back-end

Puede haber varias razones para actualizar un back-end:

- Las credenciales del sistema de almacenamiento han cambiado. Para actualizar las credenciales, se debe actualizar el secreto de Kubernetes utilizado en el `TridentBackendConfig` objeto. Trident actualizará automáticamente el backend 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 actualizar `TridentBackendConfig` Objetos directamente a través de Kubernetes usando el siguiente comando:

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

- Como alternativa, puede realizar cambios en los existentes `TridentBackendConfig` CR con el siguiente comando:

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

NOTA

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

Realizar la administración de back-end con trimentctl

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

Cree un back-end

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

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

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

```
tridentctl logs -n trident
```

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

Eliminar un back-end

Para suprimir un backend de 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
```

NOTA Si Trident ha aprovisionado volúmenes y snapshots a partir de este back-end que aún existen, al eliminar el back-end se evita que se aprovisionen nuevos volúmenes. El backend seguirá existiendo en estado de supresión.

Ver los back-ends existentes

Para ver los back-ends que Trident conoce, haga lo siguiente:

- Para obtener un resumen, ejecute el siguiente comando:

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

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

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

Actualizar un back-end

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

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

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

```
tridentctl logs -n trident
```

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

Identifique las clases de almacenamiento que utilizan un back-end

Este es un ejemplo del tipo de preguntas que puede responder con el JSON que `tridentctl` salidas para objetos backend. Utiliza la `jq` utilidad, que debe instalar.

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

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

Pasar entre las opciones de administración del back-end

Obtén información sobre las diferentes formas de administrar back-ends en Trident.

Opciones para gestionar back-ends

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

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

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

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

Esto se aplica a las siguientes situaciones:

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

En ambos escenarios, los back-ends seguirán presentes, con Trident programando volúmenes y operando en ellos. A continuación, los administradores tienen una de estas dos opciones:

- Siga utilizando `tridentctl` para gestionar los back-ends que se crearon con él.
- Enlazar los back-ends creados con `tridentctl` a un nuevo `TridentBackendConfig` objeto. Hacerlo significaría que se gestionarán los back-ends `kubectl` y no `tridentctl`.

Para administrar un back-end preexistente mediante `kubectl`, tendrá que crear un `TridentBackendConfig` que enlaza con el backend existente. A continuación se ofrece una descripción general de cómo funciona:

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

Paso 0: Identificar el back-end

Para crear un `TridentBackendConfig` que se enlaza a un backend existente, necesitará obtener la configuración de backend. En este ejemplo, supongamos que se ha creado un back-end mediante la siguiente definición JSON:

```
tridentctl get backend ontap-nas-backend -n trident
+---------------------+----------------
+--------------------------------------+--------+---------+
|          NAME       | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+---------------------+----------------
+--------------------------------------+--------+---------+
| ontap-nas-backend   | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |      25 |
+---------------------+----------------
+--------------------------------------+--------+---------+
```

```
cat ontap-nas-backend.json
```

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.10.10.1",
  "dataLIF": "10.10.10.2",
  "backendName": "ontap-nas-backend",
  "svm": "trident_svm",
  "username": "cluster-admin",
  "password": "admin-password",
  "defaults": {
    "spaceReserve": "none",
    "encryption": "false"
  },
  "labels": {
    "store": "nas_store"
  },
  "region": "us_east_1",
  "storage": [
    {
      "labels": {
        "app": "msoffice",
        "cost": "100"
      },
      "zone": "us_east_1a",
      "defaults": {
        "spaceReserve": "volume",
        "encryption": "true",
        "unixPermissions": "0755"
      }
    },
    {
      "labels": {
        "app": "mysqldb",
        "cost": "25"
      },
      "zone": "us_east_1d",
      "defaults": {
        "spaceReserve": "volume",
        "encryption": "false",
        "unixPermissions": "0775"
      }
    }
  ]
}
```

Paso 1: Cree un secreto de Kubernetes

Cree un secreto que contenga las credenciales del back-end, como se muestra en este ejemplo:

```
cat tbc-ontap-nas-backend-secret.yaml
```

```
apiVersion: v1
kind: Secret
metadata:
  name: ontap-nas-backend-secret
type: Opaque
stringData:
  username: cluster-admin
  password: admin-password
```

```
kubectl create -f tbc-ontap-nas-backend-secret.yaml -n trident
secret/backend-tbc-ontap-san-secret created
```

Paso 2: Cree un `TridentBackendConfig` CR

El paso siguiente es crear un `TridentBackendConfig` CR que se enlazará automáticamente a la preexistente `ontap-nas-backend` (como en este ejemplo). Asegurarse de que se cumplen los siguientes requisitos:

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

En este caso, el `TridentBackendConfig` tendrá este aspecto:

```
cat backend-tbc-ontap-nas.yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-ontap-nas-backend
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.10.10.1
  dataLIF: 10.10.10.2
  backendName: ontap-nas-backend
  svm: trident_svm
  credentials:
    name: mysecret
  defaults:
    spaceReserve: none
    encryption: 'false'
  labels:
    store: nas_store
  region: us_east_1
  storage:
  - labels:
      app: msoffice
      cost: '100'
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: 'true'
      unixPermissions: '0755'
  - labels:
      app: mysqldb
      cost: '25'
    zone: us_east_1d
    defaults:
      spaceReserve: volume
      encryption: 'false'
      unixPermissions: '0775'
```

```
kubectl create -f backend-tbc-ontap-nas.yaml -n trident
tridentbackendconfig.trident.netapp.io/tbc-ontap-nas-backend created
```

Paso 3: Compruebe el estado del `TridentBackendConfig` CR

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

```
kubectl get tbc tbc-ontap-nas-backend -n trident
NAME                   BACKEND NAME          BACKEND UUID
PHASE   STATUS
tbc-ontap-nas-backend  ontap-nas-backend     52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7   Bound   Success

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

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

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

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

Paso 0: Identificar el back-end

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

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

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

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

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

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

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

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

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

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

Paso 2: Elimine la `TridentBackendConfig` CR

El paso final es eliminar la `TridentBackendConfig` CR. Tras confirmar la `deletionPolicy` se establece en `retain`, puede utilizar Adelante con la eliminación:

```
kubectl delete tbc backend-tbc-ontap-san -n trident
tridentbackendconfig.trident.netapp.io "backend-tbc-ontap-san" deleted

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

Tras la eliminación del `TridentBackendConfig` objeto, Trident simplemente lo elimina sin eliminar realmente el backend.

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 Trident cómo se aprovisionan los volúmenes.

Configurar un objeto de Kubernetes StorageClass

El "Objeto de Kubernetes StorageClass" identifica el Trident como el aprovisionador que se usa para esa clase y le indica a Trident cómo aprovisionar un volumen. Por ejemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
mountOptions:
  - nfsvers=3
  - nolock
parameters:
  backendType: "ontap-nas"
  media: "ssd"
allowVolumeExpansion: true
volumeBindingMode: Immediate
```

Consulte el "Objetos de Kubernetes y Trident" para obtener más detalles sobre cómo interactúan las clases de almacenamiento con los `PersistentVolumeClaim` parámetros y para controlar la forma en que 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. Este es un objeto de Kubernetes, así que use `kubectl` Para crear en Kubernetes.

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

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

Trident ofrece "definiciones simples de clase de almacenamiento para back-ends específicos".

Como alternativa, puede editar `sample-input/storage-class-csi.yaml.templ` archivo que viene con el instalador y reemplácelo *`BACKEND_TYPE`* con el nombre del controlador de almacenamiento.

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

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

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

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 Trident, ejecute el siguiente comando:

```
tridentctl get storageclass
```

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

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

Establecer una clase de almacenamiento predeterminada

Kubernetes 1.6 añadió la capacidad de establecer un tipo de almacenamiento predeterminado. Esta es la clase de almacenamiento que se usará para aprovisionar un volumen persistente si un usuario no especifica una en una solicitud de volumen persistente (PVC).

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

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

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

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

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

NOTA	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 puede responder con el JSON que `tridentctl` genera los objetos de backend 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>` debe sustituirse por su clase de almacenamiento.

Todos los volúmenes persistentes que se hayan creado a través de esta clase de almacenamiento permanecerán intactos, y Trident continuará gestionándolos.

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

Descripción general

Una "*Claim de volumen persistente*" (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 la RVP, puede montar el volumen en un pod.

Cree la RVP

Pasos

1. Cree la RVP.

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

2. Verifique el estado de la RVP.

```
kubectl get pvc
```

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

1. Monte el volumen en un pod.

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

NOTA Puede supervisar el progreso con `kubectl get pod --watch`.

2. Compruebe que el volumen está montado en `/my/mount/path`.

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

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

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

Manifiestos de muestra

Manifiestos de muestra de PersistentVolumeClaim

Estos ejemplos muestran opciones básicas de configuración de PVC.

PVC con acceso RWO

En este ejemplo se muestra una RVP básica con acceso RWO que está asociada con una clase de almacenamiento denominada `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 asociado con una clase de almacenamiento denominada `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: storage
      persistentVolumeClaim:
       claimName: pvc-storage
  containers:
    - name: pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: storage
```

Configuración de NVMe/TCP básica

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-nginx
spec:
  volumes:
    - name: basic-pvc
      persistentVolumeClaim:
        claimName: pvc-san-nvme
  containers:
    - name: task-pv-container
      image: nginx
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: basic-pvc
```

Consulte el "Objetos de Kubernetes y Trident" para obtener más detalles sobre cómo interactúan las clases de almacenamiento con los `PersistentVolumeClaim` parámetros y para controlar la forma en que Trident aprovisiona los volúmenes.

Mover un volumen

A partir de Trident 26.06, un administrador de almacenamiento puede trasladar sin interrupciones un volumen iSCSI a otro nodo ONTAP y a otro agregado dentro del mismo clúster ONTAP, mientras la carga de trabajo que lo utiliza sigue ejecutándose. Solicitas la migración de forma declarativa con un `TridentVolumeMove` recurso personalizado o el `tridentctl` equivalente, y Trident coordina ONTAP y el host para que la E/S nunca se interrumpa.

Requisitos

La función de desplazamiento de volúmenes se aplica solo al controlador `ontap-san` para volúmenes iSCSI. No se admiten los protocolos NVMe ni NAS.

Antes de mover un volumen, asegúrate de que se cumplan los siguientes requisitos:

- El backend usa el controlador `ontap-san` con la API de REST de ONTAP.
- El clúster ONTAP tiene al menos dos nodos.
- Dispones de uno de los siguientes conjuntos de credenciales:
  - ONTAP 9.19.1 o posterior si usas la función personalizada dedicada de ONTAP.
  - Credenciales de Cluster-admin en versiones anteriores de ONTAP.

Cómo funciona

Trident traslada el volumen (LUN) entre dos ubicaciones dentro del mismo clúster de ONTAP. Antes de que se trasladen los datos, Trident añade nuevas rutas iSCSI al destino junto a las rutas existentes, de modo que el dispositivo multivía del host nunca pierda el acceso. Una vez que ONTAP ha completado el traslado, Trident elimina las rutas de origen obsoletas.

Trident gestiona el traslado a través de una secuencia ordenada de estados que se pueden consultar en el recurso `TridentVolumeMove`, de modo que puedas seguir el progreso del traslado.

Mover un volumen

Identifica el volumen y las ubicaciones

1. Busca el nombre del PV que quieres mover. El nombre del PV es también el nombre del recurso de movimiento.
2. Desde el clúster ONTAP, anota el nodo y el agregado actuales (de origen) de ese volumen y elige el nodo y el agregado de destino.

Empieza la mudanza

Utiliza cualquiera de los dos métodos. Ambos crean el mismo `TridentVolumeMove` recurso, cuyo nombre es el del PV.

Usando tridentctl

```
tridentctl create volume-move \
  --volume      pvc-8a814d62-bd58-4253-b0d1-82f2885db671 \
  --source-node ontap-node-01 \
  --source-pool aggr1_node01 \
  --target-node ontap-node-02 \
  --target-pool aggr1_node02 \
  -n trident
```

`tvm` es un alias de `volume-move`. Agrega `--delete-after-success 10m` para eliminar el recurso automáticamente después de que el traslado se complete con éxito.

Uso de un recurso TridentVolumeMove

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeMove
metadata:
  name: pvc-8a814d62-bd58-4253-b0d1-82f2885db671   # the PV to move
  namespace: trident
spec:
  sourceNode: ontap-node-01
  sourcePool: aggr1_node01
  targetNode: ontap-node-02
  targetPool: aggr1_node02
  # deleteAfterSuccess: 10m   # optional
```

Aplica el recurso:

```
kubectl apply -f tridentvolumemove.yaml
```

Campos de especificaciones

`sourceNode` / `targetNode`

Nombres de los nodos ONTAP (no UUIDs).

`sourcePool` / `targetPool`

Nombres de agregados de ONTAP.

`deleteAfterSuccess`

Duración opcional, como `30s` o `10m`. Establece el valor en `0` para eliminar el recurso inmediatamente, u omítelo para conservar el recurso.

Supervisa la mudanza

Sigue el progreso usando el `tvm` nombre abreviado o `tridentvolumemove`:

```
kubectl get tridentvolumemove -n trident
```

Para ver el estado actual y el progreso por nodo, describe el recurso:

```
kubectl describe tridentvolumemove <pv-name> -n trident
```

El volumen puede tener conexiones simultáneas durante el estado `Moving`, porque ONTAP puede tardar un tiempo en completar el traslado de un volumen de gran tamaño. Para garantizar un traslado sin interrupciones, Trident coordina entre el controlador Trident y el nodo Trident. Cada estado realiza los siguientes pasos en orden, terminando en `Succeeded` o `Failed`.

Estado	Qué está pasando
`Pending`	Trident acepta y valida la solicitud de traslado y recopila la información necesaria antes de que se inicien los cambios en el almacenamiento.
`ControllerStaging`	Trident prepara la ubicación de destino. Asigna las interfaces de almacenamiento del nodo de destino al LUN y actualiza Kubernetes para que un futuro reinicio del nodo se vuelva a conectar a la ubicación correcta.
`NodeStaging`	En cada host que tenga el volumen montado, Trident añade las nuevas rutas de almacenamiento del nodo de destino junto a las rutas existentes, de modo que las operaciones de E/S del pod continúen a través de ambos conjuntos de rutas.
`Moving`	ONTAP copia y transfiere los datos del volumen al agregado de destino. Esta es la fase más larga, y ONTAP se encarga de todo mientras Trident realiza sondeos hasta que termina. Durante este estado se permiten nuevas publicaciones.
`ControllerUnstaging`	La transferencia de datos ha finalizado. Trident actualiza Kubernetes para eliminar las rutas de origen del estado deseado y luego desvincula las interfaces de almacenamiento del nodo de origen del LUN.
`NodeUnstaging`	En cada host, Trident elimina las rutas de origen obsoletas. El host solo conserva las rutas de destino y el dispositivo multivía sigue gestionando las operaciones de E/S sin interrupción.
`Succeeded`	El volumen se ha trasladado por completo y ya está disponible en línea en el destino. Trident conserva el `TridentVolumeMove` recurso, o lo elimina automáticamente si configuras `deleteAfterSuccess`.
`Failed`	Un error irrecuperable detuvo la operación de movimiento o impidió que comenzara. Trident restablece el estado del volumen para que las operaciones normales puedan reanudarse y mantiene el recurso con un mensaje de error. Revisa el mensaje, soluciona la causa raíz, elimina el recurso y vuelve a intentarlo.

La `.status.attachments` lista muestra el progreso de cada nodo adjunto en su campo `state`:

Estado de adjunto	Significado
`Pending`	Nodo identificado como conectado; aún no se ha iniciado el trabajo.
`Bridged`	Se han añadido nuevas rutas de destino junto a las rutas existentes.
`Migrated`	El tráfico ahora se sirve por las nuevas rutas.
`Cleaned`	Se eliminaron las rutas de origen obsoletas del nodo.
`Detached`	El nodo se desvinculó a mitad del movimiento y se excluyó de cualquier consideración posterior mientras Trident evalúa si la vinculación está lista para pasar al siguiente paso.
`Failed`	Error de terminal en este nodo. Consulta el mensaje.

Limpia el recurso de traslado

Si configuras `deleteAfterSuccess`, Trident elimina el recurso automáticamente. De lo contrario, puedes eliminarlo manualmente. Antes de solicitar otro traslado en el mismo volumen, asegúrate de que no exista ningún recurso `TridentVolumeMove` para ese volumen en estado `Succeeded` o `Failed`.

NOTA Trident nunca elimina automáticamente las operaciones fallidas. Trident conserva un `Failed` recurso como registro de auditoría. Una vez que hayas solucionado la causa raíz, elimina el recurso manualmente antes de volver a intentarlo.

Expanda los volúmenes

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

Expanda un volumen iSCSI

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

NOTA La ampliación del volumen iSCSI se admite en el `ontap-san`, `ontap-san-economy`, `solidfire-san` Requiere Kubernetes 1.16 o posterior.

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

Edite la definición de StorageClass para establecer el `allowVolumeExpansion` campo a. `true`.

```
cat storageclass-ontapsan.yaml
```

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-san
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
allowVolumeExpansion: True
```

En el caso de un tipo de almacenamiento existente, edítelo para incluir el `allowVolumeExpansion` parámetro.

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

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

```
cat pvc-ontapsan.yaml
```

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

Trident crea un volumen persistente (VP) y lo asocia con esta reclamación de volumen persistente (RVP).

```
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, Trident expande el volumen en el backend de almacenamiento, vuelve a escanear el dispositivo y cambia el tamaño del sistema de archivos.
- Cuando se intenta cambiar el tamaño de un VP no conectado, Trident expande 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`.

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

 kubectl describe pvc san-pvc
Name:          san-pvc
Namespace:     default
StorageClass:  ontap-san
Status:        Bound
Volume:        pvc-8a814d62-bd58-4253-b0d1-82f2885db671
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
               volume.beta.kubernetes.io/storage-provisioner:
csi.trident.netapp.io
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      1Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Mounted By:    ubuntu-pod
```

Paso 4: Expanda el PV

Para cambiar el tamaño del VP que se ha creado de 1GI a 2gi, edite la definición de PVC y actualice el `spec.resources.requests.storage` A 2gi.

```
kubectl edit pvc san-pvc
```

```
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: "2019-10-10T17:32:29Z"
  finalizers:
  - kubernetes.io/pvc-protection
  name: san-pvc
  namespace: default
  resourceVersion: "16609"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/san-pvc
  uid: 8a814d62-bd58-4253-b0d1-82f2885db671
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
 # ...
```

Paso 5: Validar la expansión

Puede validar correctamente la ampliación operativa comprobando el tamaño de la RVP, el VP y el volumen 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 FC

Puede ampliar un volumen persistente de FC (PV) mediante el aprovisionador de CSI.

NOTA El controlador admite la expansión del volumen de FC `ontap-san` y requiere Kubernetes 1,16 y versiones posteriores.

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

Edite la definición de StorageClass para establecer el `allowVolumeExpansion` campo a. `true`.

```
cat storageclass-ontapsan.yaml
```

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-san
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
allowVolumeExpansion: True
```

En el caso de un tipo de almacenamiento existente, edítelo para incluir el `allowVolumeExpansion` parámetro.

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

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

```
cat pvc-ontapsan.yaml
```

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

Trident crea un volumen persistente (VP) y lo asocia con esta reclamación de volumen persistente (RVP).

```
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. Hay dos situaciones al cambiar el tamaño de un VP de FC:

- Si el VP está conectado a un pod, Trident expande el volumen en el backend de almacenamiento, vuelve a escanear el dispositivo y cambia el tamaño del sistema de archivos.
- Cuando se intenta cambiar el tamaño de un VP no conectado, Trident expande 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`.

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

 kubectl describe pvc san-pvc
Name:          san-pvc
Namespace:     default
StorageClass:  ontap-san
Status:        Bound
Volume:        pvc-8a814d62-bd58-4253-b0d1-82f2885db671
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
               volume.beta.kubernetes.io/storage-provisioner:
csi.trident.netapp.io
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      1Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Mounted By:    ubuntu-pod
```

Paso 4: Expanda el PV

Para cambiar el tamaño del VP que se ha creado de 1GI a 2gi, edite la definición de PVC y actualice el `spec.resources.requests.storage` A 2gi.

```
kubectl edit pvc san-pvc
```

```
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: "2019-10-10T17:32:29Z"
  finalizers:
  - kubernetes.io/pvc-protection
  name: san-pvc
  namespace: default
  resourceVersion: "16609"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/san-pvc
  uid: 8a814d62-bd58-4253-b0d1-82f2885db671
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
 # ...
```

Paso 5: Validar la expansión

Puede validar correctamente la ampliación operativa comprobando el tamaño de la RVP, el VP y el volumen 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

Trident admite la expansión de volumen para PV NFS aprovisionados en `ontap-nas` , `ontap-nas-economy` , `ontap-nas-flexgroup` , y `azure-netapp-files` backends.

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

Para cambiar el tamaño de un VP de NFS, el administrador primero tiene que configurar la clase de almacenamiento para permitir la expansión del volumen estableciendo el `allowVolumeExpansion` campo a. `true`:

```
cat storageclass-ontapnas.yaml
```

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontapnas
provisioner: csi.trident.netapp.io
parameters:
  backendType: ontap-nas
allowVolumeExpansion: true
```

Si ya ha creado una clase de almacenamiento sin esta opción, puede simplemente editar la clase de almacenamiento existente mediante `kubectl edit storageclass` para permitir la expansión de volumen.

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

```
cat pvc-ontapnas.yaml
```

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

Trident debería crear un PV NFS de 20 MiB para este PVC:

```
kubectl get pvc
NAME           STATUS   VOLUME
CAPACITY     ACCESS MODES   STORAGECLASS    AGE
ontapnas20mb   Bound    pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   20Mi
RWO            ontapnas        9s

kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM                  STORAGECLASS    REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   20Mi       RWO
Delete           Bound    default/ontapnas20mb   ontapnas
2m42s
```

Paso 3: Expanda el PV

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

```
kubectl edit pvc ontapnas20mb
```

```
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: 2018-08-21T18:26:44Z
  finalizers:
  - kubernetes.io/pvc-protection
  name: ontapnas20mb
  namespace: default
  resourceVersion: "1958015"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/ontapnas20mb
  uid: c1bd7fa5-a56f-11e8-b8d7-fa163e59eaab
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
# ...
```

Paso 4: Validar la expansión

Puede validar el tamaño correctamente trabajado comprobando el tamaño de la RVP, el VP y el volumen 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    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

Entiende los límites del subsistema RWX NVMe

ReadWriteMany (RWX) volúmenes que usan el protocolo NVMe tienen un límite de escalabilidad de 64 nodos por volumen. A continuación se incluyen las limitaciones, se explica la arquitectura del subsistema NVMe involucrada y se describen los pasos de resolución necesarios.

Entiende el límite de 64 nodos

Si planeas usar volúmenes ReadWriteMany (RWX) con el protocolo NVMe, un solo volumen RWX NVMe no puede ser montado por más de 64 nodos en un clúster de Kubernetes.

No programes cargas de trabajo que monten el mismo RWX NVMe PersistentVolumeClaim en más de 64 nodos.

Esta limitación solo se aplica a los volúmenes RWX que usan el protocolo NVMe.

Entiende los modelos de subsistemas NVMe

Modelo de subsistema por volumen (Trident releases earlier than 26.02)

En las versiones de Trident anteriores a 26.02, los volúmenes RWX NVMe se aprovisionan utilizando un modelo de subsistema por volumen. Cada volumen RWX NVMe se asigna a su propio subsistema NVMe dedicado en ONTAP.

Este modelo es sencillo, pero tiene un límite de escalabilidad inferior. En clústeres de Kubernetes grandes, los límites del controlador de subsistema se alcanzan rápidamente porque cada volumen RWX consume un subsistema dedicado.

Modelo de super-subsystem (introducido en Trident 26.02)

A partir de Trident 26.02, los volúmenes RWX NVMe utilizan un modelo de super-subsubsistema compartido. Varios volúmenes RWX NVMe comparten el mismo subsistema NVMe.

Cada super-subsystem admite hasta 1024 namespaces (volúmenes). Este modelo mejora significativamente la escalabilidad para cargas de trabajo RWX y reduce la probabilidad de alcanzar los límites del subsistema ONTAP.

Cada volumen RWX NVMe admite hasta 64 nodos.

Identifica los síntomas de error

Si creas o adjuntas volúmenes RWX NVMe a escala, podrías ver errores similares a los siguientes:

```
Maximum number of controllers reached. No more controllers can be created.
```

Este error indica que se ha alcanzado el límite del controlador del subsistema ONTAP NVMe.

Soluciona errores de límite de subsistema

Para superar las limitaciones de los subsistemas por volumen y aprovechar el modelo de supersubsistema, actualiza a Trident 26.02 o posterior.

Actualiza Trident para aplicar el modelo de super-subsystem

Para aplicar el modelo de super-subsistema para volúmenes RWX NVMe:

1. Actualiza Trident a la versión 26.02 o posterior.
2. Reduce todos los pods que usan volúmenes RWX NVMe a cero réplicas.
3. Verifica que ninguna carga de trabajo esté utilizando activamente volúmenes RWX NVMe.
4. Vuelve a escalar los pods.

Esta secuencia de reinicio garantiza que los volúmenes RWX NVMe se adjunten usando el modelo de super-subsystem.

- Esta limitación solo se aplica a los volúmenes RWX que usan el protocolo NVMe.
- El límite de 64 nodos se aplica por cada volumen RWX NVMe.
- Otros modos de acceso y otros protocolos no están afectados.

Escalabilidad del controlador

Trident mejora la escalabilidad del controlador gracias a una mayor concurrencia entre múltiples controladores de almacenamiento. Al habilitar la escalabilidad del controlador, el controlador de Trident procesa las operaciones de almacenamiento en paralelo en lugar de hacerlo en serie, lo que aumenta el rendimiento en entornos de Kubernetes con numerosas operaciones simultáneas.

Antes de realizar la implementación, comprueba qué controladores de Trident admiten la escalabilidad del controlador en su versión de disponibilidad general y cuáles están disponibles como vista previa técnica en Trident 26.06. Esto te ayuda a tomar decisiones de implementación fundamentadas y a gestionar el riesgo. La escalabilidad del controlador está desactivada de forma predeterminada.

Conceptos clave y definiciones

Escalabilidad del controlador

La escalabilidad del controlador se refiere a la capacidad del controlador Trident para procesar múltiples operaciones de almacenamiento en paralelo, en lugar de serializarlas tras un único bloqueo. Estas operaciones incluyen la creación, eliminación y cambio de tamaño de volúmenes; la creación y eliminación de instantáneas; la publicación y retirada de volúmenes; y la gestión del backend.

Al habilitar la escalabilidad del controlador, las operaciones en diferentes volúmenes y backends se ejecutan de forma simultánea. Esto aumenta el rendimiento y reduce el tiempo de ejecución de principio a fin en entornos con un elevado número de operaciones simultáneas de PersistentVolumeClaim y VolumeSnapshot.

Comportamiento predeterminado (modo serie)

De forma predeterminada, el controlador Trident procesa las operaciones una por una. Cada solicitud de creación, eliminación, cambio de tamaño o instantánea se completa antes de que Trident inicie la siguiente. El modo serie es el valor predeterminado compatible con todas las instalaciones y no requiere configuración alguna.

El modo serie es suficiente para la mayoría de las cargas de trabajo. Activa la escalabilidad del controlador solo cuando el volumen de operaciones genere un retraso en el procesamiento en serie.

Compatibilidad con la escalabilidad del controlador

Trident admite la escalabilidad de los controladores en distintos niveles de madurez para diferentes controladores de almacenamiento.

Disponibilidad general

Los siguientes drivers soportan la escalabilidad del controlador en disponibilidad general en Trident 26.06:

- `ontap-san`
- `ontap-nas`
- `ontap-nas-economy`
- `ontap-san-economy`
- `google-cloud-netapp-volumes`

- `azure-netapp-files`
- `solidfire-san`

NOTA	Los controladores `google-cloud-netapp-volumes` y `google-cloud-netapp-volumes-san` son diferentes. Trident solo es compatible con `google-cloud-netapp-volumes`. No utilices `google-cloud-netapp-volumes-san` en las configuraciones del backend ni en los ejemplos.

Avance técnico

El siguiente controlador admite la escalabilidad del controlador como función de vista previa técnica en Trident 26.06:

- `asa-r2` (SAN y NVMe)

Este controlador presenta las siguientes limitaciones:

- La concurrencia de controladores está disponible únicamente para evaluación y pruebas.
- El comportamiento puede variar de una versión a otra.
- NetApp no recomienda su uso en entornos de producción.

Cómo funciona enableConcurrency

Cuando configuras `enableConcurrency` en `true`, Trident aplica el procesamiento simultáneo en todos los backends que gestiona el controlador. La configuración se aplica a todos los backends a la vez. No puedes habilitarlo para backends individuales ni para drivers individuales.

Cada backend configurado debe utilizar un controlador que figure en la lista de disponibilidad general o de vista previa técnica. Si algún backend utiliza un controlador no compatible, Trident no se inicia y Trident no añade un backend que utilice un controlador no compatible.

Para restablecer el arranque, elimina o reconfigura cualquier backend que utilice un controlador no compatible o desactiva la escalabilidad del controlador.

Antes de activar

IMPORTANTE	Antes de habilitar la escalabilidad del controlador, comprueba que todos los backends configurados utilicen un controlador que figure en la lista de disponibilidad general o de vista previa técnica. Si algún backend utiliza un controlador no compatible, Trident no se inicia después de habilitar la función.

Utiliza la siguiente tabla para decidir si deseas habilitar la escalabilidad del controlador.

Si tu entorno	Entonces
Utiliza únicamente controladores compatibles y presenta un retraso en las operaciones del controlador	Habilita la escalabilidad del controlador.
Utiliza cualquier controlador no compatible	Mantén el modo serie predeterminado. No actives la escalabilidad del controlador.

Si tu entorno	Entonces
Gestiona un volumen de operaciones bajo sin acumular retrasos	Mantén el modo serie predeterminado.

Habilita la escalabilidad del controlador

La opción de configuración `enableConcurrency` controla la escalabilidad del controlador. Debes habilitar esta opción de forma explícita durante la instalación de Trident o cuando actualices una implementación existente.

Despliegue del operador Trident

Para habilitar la escalabilidad del controlador con el operador Trident, configura `enableConcurrency` en `true` en el recurso personalizado (CR) `TridentOrchestrator`.

Nueva instalación

Crea o edita el `TridentOrchestrator` CR y establece `enableConcurrency` en `true`:

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  namespace: trident
  enableConcurrency: true
```

Aplicar el CR:

```
kubectl apply -f tridentorchestrator_cr.yaml
```

Instalación existente

Aplica un parche a la `TridentOrchestrator` CR existente para habilitar la escalabilidad del controlador:

```
kubectl patch torc trident --type=merge -p
'{"spec":{"enableConcurrency":true}}'
```

Comprueba que Trident haya aplicado la configuración:

```
kubectl get torc trident -o
jsonpath='{.status.currentInstallationParams.enableConcurrency}'
```

Despliegue con Helm

Para activar la escalabilidad del controlador con Helm, establece el valor `enableConcurrency` a `true`.

Nueva instalación

```
helm install trident netapp-trident/trident-operator --namespace trident
--create-namespace --set enableConcurrency=true
```

Instalación existente

```
helm upgrade trident netapp-trident/trident-operator --namespace trident
--set enableConcurrency=true
```

Como alternativa, establece `enableConcurrency` en `true` en un archivo `values.yaml` personalizado:

```
# values.yaml
enableConcurrency: true
```

Luego instala o actualiza con el archivo de valores:

```
helm install trident netapp-trident/trident-operator --namespace trident
--create-namespace -f values.yaml
```

implementación de tridentctl

Para habilitar la escalabilidad del controlador con `tridentctl`, pasa la bandera `--enable-concurrency` durante la instalación.

Nueva instalación

```
tridentctl install -n trident --enable-concurrency
```

Instalación existente

Para habilitar la escalabilidad del controlador en una implementación existente `tridentctl`, desinstala Trident y vuelve a instalarlo con el parámetro:

```
tridentctl uninstall -n trident
tridentctl install -n trident --enable-concurrency
```

Comprueba que la escalabilidad del controlador esté activada

Una vez que habilites la escalabilidad del controlador, verifica que el controlador Trident se esté ejecutando con la concurrencia activada. Revisa los registros del pod del controlador:

```
kubectl logs -n trident deploy/trident-controller | grep -i concurrency
```

La salida incluye una entrada en el registro que confirma que la concurrencia está habilitada.

Comportamiento de la concurrencia

Cuando se habilita la escalabilidad del controlador, el controlador Trident aplica el siguiente comportamiento:

- Trident sustituye el bloqueo global único por un sistema de bloqueo más detallado, específico para cada recurso.
- Trident serializa las operaciones que modifican el mismo recurso para mantener la coherencia de los datos.
- Las operaciones que solo leen de un recurso se ejecutan de forma concurrente con otras operaciones de lectura en ese mismo recurso.
- Trident limita a 20 el número de solicitudes simultáneas de la API de ONTAP por cada LIF de gestión para evitar la sobrecarga de los sistemas de almacenamiento backend.
- Si varios backends comparten el mismo LIF de gestión, comparten este límite de 20 solicitudes.

Advertencias y limitaciones

Las siguientes consideraciones se aplican a la escalabilidad de los controladores en Trident 26.06:

- La escalabilidad del controlador solo es compatible con los controladores que figuran en las listas de disponibilidad general y de vista previa técnica. Para obtener más información, consulta Antes de activar.
- El controlador Trident gestiona la concurrencia de forma interna. Esta versión no ofrece límites de concurrencia configurables por el usuario.
- El rendimiento global depende del controlador de almacenamiento en uso, la capacidad de respuesta del backend y el rendimiento del servidor de la API de Kubernetes.
- Una elevada concurrencia puede aumentar la carga en los sistemas de almacenamiento del backend.
- El comportamiento de la escalabilidad del controlador no es idéntico en todos los drivers.
- El controlador de la versión preliminar técnica puede presentar un rendimiento irregular bajo cargas elevadas y su comportamiento puede variar entre versiones.
- La depuración de operaciones concurrentes puede resultar más compleja debido a la ejecución en paralelo. Las métricas y los registros pueden mostrar resultados de operaciones entremezclados.

Recomendaciones

Antes de habilitar la escalabilidad del controlador, completa los siguientes pasos:

1. Comprueba que todos los backends configurados utilicen un controlador que esté en la lista de disponibilidad general o de vista previa técnica.
2. Prueba el cambio en un clúster que no sea de producción antes de aplicarlo en el entorno de producción.

3. Comprueba que la escalabilidad del controlador esté activada después de aplicar el cambio.

Aplica las siguientes recomendaciones generales cuando trabajes con la escalabilidad del controlador:

- Utiliza controladores de disponibilidad general para entornos de producción que requieran alta escalabilidad.
- Prueba el controlador de la versión preliminar técnica únicamente en entornos que no sean de producción.
- Supervisa el rendimiento del backend y de los controladores cuando operas a gran escala.
- No des por sentado el orden de las operaciones en los scripts de automatización.

Ampliación automática de volúmenes

La expansión automática de volúmenes permite que los Volúmenes Persistentes aprovisionados por Trident crezcan automáticamente cuando la capacidad utilizada alcanza un umbral definido. Esta capacidad reduce la sobrecarga operativa y ayuda a evitar la interrupción de la aplicación causada por el agotamiento de la capacidad. La expansión automática de volúmenes se implementa utilizando Autogrow Policies. Una Autogrow Policy define:

- El umbral de utilización que activa la expansión
- La cantidad en que crece el volumen
- El tamaño máximo que puede alcanzar el volumen

NOTA	Los volúmenes aumentan de tamaño automáticamente cuando se supera el umbral de utilización definido. Los volúmenes nunca se reducen automáticamente.

Requisitos

Antes de configurar la expansión automática de volumen, asegúrate de que se cumplen los siguientes requisitos:

- Trident 26.02 o posterior
- Permisos de control de acceso basados en roles para crear `TridentAutogrowPolicy` recursos personalizados
- StorageClasses configurados con `allowVolumeExpansion: true`
- Protocolos ONTAP compatibles:
  - Network File System (NFS)
  - Internet Small Computer Systems Interface (iSCSI)
  - Non-Volatile Memory Express (NVMe)
  - Protocolo de canal de fibra (FCP)

Limitaciones

- Los volúmenes de bloques sin procesar ONTAP Non-Volatile Memory Express anteriores a ONTAP 9.16.1 no admiten la expansión automática.
- Para volúmenes de red de área de almacenamiento, si `growthAmount` es menor o igual a 50 mebibytes,

Trident aumenta automáticamente el valor a 51 mebibytes antes de redimensionar, siempre que el tamaño resultante no supere `maxSize`.

- En entornos brownfield, es posible que la expansión automática no funcione para ciertos volúmenes existentes debido al comportamiento de migración de publicación de volúmenes.
- Cuando un volumen alcanza `maxSize`, no se produce ninguna expansión adicional.
- Protocolos compatibles para la expansión automática del volumen:
  - Network File System (NFS)
  - Internet Small Computer Systems Interface (iSCSI)
  - Non-Volatile Memory Express (NVMe)
  - Protocolo de canal de fibra (FCP)

Provisiona volúmenes con la política de crecimiento automático

La política de Autogrow puede configurarse a dos niveles:

- Nivel de clase de almacenamiento: establece el valor predeterminado para todos los volúmenes (mediante anotación)
- Nivel de PVC: anula el valor por defecto de la clase de almacenamiento (usando anotación)

Crear una política de Autogrow

Las políticas de crecimiento automático permiten la expansión automática de volúmenes cuando los volúmenes alcanzan un umbral de capacidad definido.

Asegúrate de tener:

- Trident 26.02 o posterior instalado
- Permisos de control de acceso basados en roles para crear `TridentAutogrowPolicy` recursos
- Comprender los requisitos de crecimiento de la carga de trabajo

Una política de crecimiento automático define cómo los volúmenes se expanden automáticamente cuando alcanzan un umbral de capacidad definido.

Puedes crear políticas Autogrow en cualquier punto de tu flujo de trabajo:

- Antes de que se creen StorageClasses y volúmenes
- Después de que existan StorageClasses
- Después de que se aprovisionan los volúmenes

Esta flexibilidad te permite introducir la expansión automática sin recrear los recursos existentes.

Especificaciones de la política Autogrow

Las políticas de crecimiento automático son recursos personalizados de Kubernetes definidos como sigue:

Campo	Descripción	Formato	Obligatorio	Ejemplo	Predeterminado
nombre	Identificador único de política	Cadena	Sí	production-db-policy	Ninguno

Campo	Descripción	Formato	Obligatorio	Ejemplo	Predeterminado
usedThreshold	Porcentaje de capacidad que activa la expansión	Cadena de porcentaje	Sí	"80%"	Ninguno
growthAmount	Cantidad a aumentar cuando se alcanza el umbral	Porcentaje o tamaño	No	"10%" o "5Gi"	"10%"
maxSize	Límite máximo de tamaño de volumen	Cantidad de Kubernetes	No	"500Gi"	Ilimitada

Crear una política de Autogrow

Pasos

1. Crea un archivo YAML que defina tu política de crecimiento automático de volúmenes:

```yaml
apiVersion: trident.netapp.io/v1
kind: TridentAutogrowPolicy
metadata:
  name: standard-autogrow
spec:
  usedThreshold: "80%"
  growthAmount: "10%"
  maxSize: "500Gi"
```

2. Aplica la política a tu clúster:

```
kubectl apply -f autogrow-policy.yaml
```

3. Verifica que se haya creado la política:

```
kubectl get tridentautogrowpolicy standard-autogrow
```

Salida esperada

```
NAME                 USED THRESHOLD   GROWTH AMOUNT   STATE
standard-autogrow    80%              10%             Success
```

Política establece

Después de crear una política, Trident valida la especificación y asigna uno de los siguientes estados:

Estado	Descripción	Acción requerida
Éxito	La política está validada y lista para usar.	Ninguno.
Error	Errores de validación detectados.	Revisa y corrige la especificación.
Eliminando	Eliminación en curso.	Espera a que termine.

Asocia una política con un StorageClass

Puedes asociar una política de crecimiento automático con un StorageClass usando la anotación `trident.netapp.io/autogrowPolicy`. Todos los volúmenes aprovisionados desde ese StorageClass heredan la política.

NOTA El StorageClass debe tener `allowVolumeExpansion: true`.

Pasos

1. Crea o modifica un StorageClass con la anotación Autogrow Policy:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
  annotations:
    trident.netapp.io/autogrowPolicy: "production-db-policy"
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
  fsType: "ext4"
allowVolumeExpansion: true
```

2. Aplica el StorageClass:

```
kubectl apply -f storageclass.yaml
```

3. Verifica la anotación:

```
kubectl get storageclass ontap-gold -o
jsonpath='{.metadata.annotations.trident\.netapp\.io/autogrowPolicy}'
```

Salida esperada

```
production-db-policy
```

Precedencia de políticas

Cuando se establecen anotaciones de Autogrow Policy tanto en un StorageClass como en un PVC, Trident aplica las siguientes reglas de precedencia:

1. **La anotación del PVC tiene prioridad.** Si un PVC establece `trident.netapp.io/autogrowPolicy`, ese valor siempre se utiliza.
2. **La anotación StorageClass se aplica solo cuando el PVC no tiene ninguna anotación.**
3. **Si ninguno de los dos tiene la anotación,** no se aplica ninguna Autogrow Policy.

Anotación de StorageClass	Anotación de PVC	Comportamiento eficaz
`trident.netapp.io/autogrowPolicy: standard-agp`	No establecido	Utiliza `standard-agp`.
`trident.netapp.io/autogrowPolicy: standard-agp`	`trident.netapp.io/autogrowPolicy: logs-policy`	Utiliza `logs-policy` (PVC anula StorageClass).
`trident.netapp.io/autogrowPolicy: standard-agp`	`trident.netapp.io/autogrowPolicy: "none"`	No Autogrow Policy (PVC desactiva autogrow).
No establecido	`trident.netapp.io/autogrowPolicy: dev-policy`	Usos `dev-policy`.
No establecido	No establecido	Sin política de Autogrow.

Ejemplos de configuración

Los siguientes ejemplos muestran configuraciones comunes de Autogrow Policy para diferentes casos de uso.

Política conservadora para bases de datos de producción

```
apiVersion: trident.netapp.io/v1
kind: TridentAutogrowPolicy
metadata:
  name: production-db-policy
spec:
  usedThreshold: "75%"
  growthAmount: "20%"
  maxSize: "5Ti"
```

Almacenamiento de logs con incrementos de crecimiento fijos

```
apiVersion: trident.netapp.io/v1
kind: TridentAutogrowPolicy
metadata:
  name: log-storage-policy
spec:
  usedThreshold: "90%"
  growthAmount: "10Gi"
  maxSize: "100Gi"
```

Política mínima con valores predeterminados

Cuando omites `growthAmount` y `maxSize`, Trident usa los valores predeterminados (`10%` crecimiento, tamaño ilimitado):

```
apiVersion: trident.netapp.io/v1
kind: TridentAutogrowPolicy
metadata:
  name: logs-policy
spec:
  usedThreshold: "85%"
```

Política con un maxSize personalizado y growthAmount predeterminado

```
apiVersion: trident.netapp.io/v1
kind: TridentAutogrowPolicy
metadata:
  name: default-ga-policy
spec:
  usedThreshold: "70%"
  maxSize: "100Gi"
```

Crecimiento agresivo con maxSize ilimitado

```
apiVersion: trident.netapp.io/v1
kind: TridentAutogrowPolicy
metadata:
  name: aggressive-growth-policy
spec:
  usedThreshold: "80%"
  growthAmount: "150%"
```

Política con porcentajes fraccionarios

```
apiVersion: trident.netapp.io/v1
kind: TridentAutogrowPolicy
metadata:
  name: precise-policy
spec:
  usedThreshold: "80.28%"
  growthAmount: "10.65%"
  maxSize: "100Gi"
```

NOTA Se admiten porcentajes fraccionarios. Si especificas más de tres decimales, Trident redondea el valor a tres decimales.

NAS StorageClass con Autogrow

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-nas-autogrow
  annotations:
    trident.netapp.io/autogrowPolicy: "standard-autogrow"
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  fsType: "ext4"
allowVolumeExpansion: true
```

SAN StorageClass con Autogrow

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: database-storage
  annotations:
    trident.netapp.io/autogrowPolicy: "production-db-policy"
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
  fsType: "ext4"
allowVolumeExpansion: true
```

Gestiona las políticas de Autogrow

Después de crear las políticas de Autogrow, puedes verlas, actualizarlas y eliminarlas. También puedes supervisar qué volúmenes usan una política determinada.

Ver políticas de Autogrow

Lista todas las políticas

Usa `kubectl` para listar todas las Autogrow Policies en tu cluster:

```
kubectl get tridentautogrowpolicy
```

O también, usa `tridentctl`:

```
tridentctl get autogrowpolicy
```

Ver detalles de la policy

Para ver la especificación completa y el estado de una política:

```
kubectl describe tridentautogrowpolicy production-db-policy
```

Para ver una política con sus volúmenes asociados en formato YAML:

```
tridentctl get autogrowpolicy production-db-policy -o yaml
```

Actualiza una política de Autogrow

Puedes modificar una política existente para cambiar su umbral, cantidad de crecimiento o tamaño máximo. Los cambios surten efecto de inmediato para todos los volúmenes que usan la política.

IMPORTANTE	Los cambios afectan a todos los volúmenes que utilizan la política. Cuando sea posible, prueba primero los cambios en un entorno que no sea de producción.

Pasos

1. Edita la política:

   ```
   kubectl edit tridentautogrowpolicy production-db-policy
   ```

2. Modifica los campos `spec` según lo necesites:

```
spec:
  usedThreshold: "75%"     # Changed from 80%
  growthAmount: "20%"      # Changed from 10%
  maxSize: "1Ti"           # Changed from 500Gi
```

3. Guarda y sal. Los cambios surten efecto inmediatamente.

Consideraciones de actualización

- **Efecto inmediato:** Todos los volúmenes que usan la política adoptan los nuevos parámetros en la próxima evaluación de crecimiento.
- **No es necesario reiniciar el volumen:** los cambios se aplican a la siguiente operación de crecimiento.
- **Prueba primero:** Valida los cambios en un entorno que no sea de producción cuando sea posible.
- **Comunica los cambios:** avisa a los equipos cuando modifiques las políticas compartidas.

Eliminar una política de Autogrow

Las políticas de Autogrow usan la protección del finalizador para evitar el borrado accidental mientras los volúmenes las usan activamente.

Pasos

1. Elimina la política:

```
kubectl delete tridentautogrowpolicy production-db-policy
```

2. Si los volúmenes siguen utilizando la política, la eliminación pasa a un estado `Deleting`. Comprueba qué volúmenes se ven afectados:

```
tridentctl get autogrowpolicy production-db-policy -o yaml
```

3. Elimina la política de cada volumen afectado. Elige una de las siguientes opciones:

   - **Opción A: desactivar explícitamente autogrow** estableciendo la anotación en `"none"`:

```
kubectl annotate pvc <pvc-name> \
  trident.netapp.io/autogrowPolicy="none" \
  --overwrite
```

   - **Opción B: eliminar por completo la anotación:**

```
kubectl annotate pvc <pvc-name> \
  trident.netapp.io/autogrowPolicy-
```

Comportamiento de borrado

Escenario	Comportamiento
Ningún volumen usa la política	La política se elimina inmediatamente.
Los volúmenes utilizan la política	La política entra `Deleting` en estado. Un finalizador bloquea la finalización hasta que se eliminan todos los volúmenes.
Todos los volúmenes se eliminan de la política	Se eliminan los finalizadores y se elimina la política.

Supervisa el uso de la política Autogrow

Comprobar volúmenes usando una política

```
tridentctl get autogrowpolicy production-db-policy -o json | jq '.volumes'
```

Averigua qué política usa un volumen

```
kubectl get pvc database-pvc -o
jsonpath='{.metadata.annotations.trident\.netapp\.io/autogrowPolicy}'
```

Supervisa los eventos de políticas

```
kubectl get events --field-selector
involvedObject.kind=TridentAutogrowPolicy
```

Protocolos compatibles

Autogrow es compatible con los siguientes protocolos de almacenamiento:

- NFS
- ISCSI
- FCP
- NVMe

NOTA Para volúmenes SAN, si el `growthAmount` configurado es de 50 MiB o menos, Trident aumenta automáticamente la cantidad de crecimiento a 51 MB para la operación de redimensionamiento, siempre que el tamaño resultante no supere `maxSize`.

Backends compatibles

Autogrow es compatible con los backends de ONTAP. Autogrow también es compatible con Google Cloud NetApp Volumes (GCNV) para volúmenes NAS y SAN.

Limitaciones conocidas

- **Volúmenes de bloques sin procesar ONTAP NVMe:** Los volúmenes creados con versiones de ONTAP anteriores a 9.16.1 no admiten autogrow.
- **Volúmenes existentes (implementaciones en entornos ya existentes):** Es posible que la función de autogrow no funcione con los volúmenes existentes, incluso si se aplica una Autogrow Policy válida. Esta limitación se debe a una migración en curso de publicaciones de volúmenes. Para confirmar que la migración se ha completado, revisa los registros del controlador Trident en busca de mensajes de `"Migration completed"`.

Preguntas frecuentes

¿Cuándo evalúa Trident el umbral?

Trident supervisa continuamente el uso del volumen. Cuando la capacidad utilizada supera el `usedThreshold`, Trident crea una solicitud interna de redimensionamiento y amplía el volumen por la `growthAmount` configurada.

Por ejemplo, esta política activa la expansión al 80% de la capacidad y aumenta el volumen un 10% cada vez, hasta un máximo de 500 GiB:

```
apiVersion: trident.netapp.io/v1
kind: TridentAutogrowPolicy
metadata:
  name: standard-autogrow
spec:
  usedThreshold: "80%"
  growthAmount: "10%"
  maxSize: "500Gi"
```

¿Puedo aplicar una política después de que los volúmenes ya estén aprovisionados?

Sí. Puedes crear una Autogrow Policy en cualquier momento y aplicarla a los PVC existentes añadiendo o actualizando la anotación `trident.netapp.io/autogrowPolicy`. No necesitas volver a crear el PVC ni el StorageClass.

Para aplicar una política a un PVC existente:

```
kubectl annotate pvc <pvc-name> \
  trident.netapp.io/autogrowPolicy="production-db-policy" \
  --overwrite
```

Para aplicar una política a un StorageClass existente:

```
kubectl annotate storageclass ontap-gold \
  trident.netapp.io/autogrowPolicy="production-db-policy" \
  --overwrite
```

¿Qué pasa si configuro una Autogrow Policy tanto en el StorageClass como en el PVC?

La anotación de PVC siempre tiene prioridad. Si un PVC tiene la `trident.netapp.io/autogrowPolicy` anotación, Trident usa ese valor sin importar lo que especifique StorageClass. Consulta "Precedencia de políticas" para más detalles.

Por ejemplo, dada esta StorageClass:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
  annotations:
    trident.netapp.io/autogrowPolicy: "standard-agp"
provisioner: csi.trident.netapp.io
allowVolumeExpansion: true
```

Y este PVC que anula la política de StorageClass:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: database-pvc
  annotations:
    trident.netapp.io/autogrowPolicy: "logs-policy"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 50Gi
  storageClassName: ontap-gold
```

Trident utiliza `logs-policy` para `database-pvc`, no `standard-agp`.

¿Cómo desactivo el autogrow para un volumen específico?

Configura la anotación de PVC en `"none"`. Esta configuración anula cualquier política a nivel de StorageClass para ese volumen:

```
kubectl annotate pvc <pvc-name> \
  trident.netapp.io/autogrowPolicy="none" \
  --overwrite
```

Puedes comprobar que autogrow está desactivado:

```
kubectl get pvc <pvc-name> -o jsonpath
='{.metadata.annotations.trident\.netapp\.io/autogrowPolicy}'
```

Salida esperada

```
none
```

¿Qué pasa cuando un volumen llega a maxSize?

Trident deja de ampliar el volumen. No genera más solicitudes de redimensionamiento para ese volumen, aunque el uso siga aumentando más allá de `usedThreshold`.

Por ejemplo, con esta política, Trident deja de ampliar el volumen una vez que alcanza los 100 GiB:

```
apiVersion: trident.netapp.io/v1
kind: TridentAutogrowPolicy
metadata:
  name: capped-policy
spec:
  usedThreshold: "90%"
  growthAmount: "10Gi"
  maxSize: "100Gi"
```

Para permitir un crecimiento ilimitado, omite `maxSize` o configúralo en `0`:

```
apiVersion: trident.netapp.io/v1
kind: TridentAutogrowPolicy
metadata:
  name: unlimited-policy
spec:
  usedThreshold: "85%"
  growthAmount: "10%"
```

¿Puedo cambiar una política sin reiniciar los volúmenes?

Sí. Cuando actualizas una política, todos los volúmenes que usan esa política adoptan los nuevos parámetros en la siguiente evaluación de crecimiento. No es necesario reiniciar ningún volumen.

Para actualizar una política en vigor:

```
kubectl edit tridentautogrowpolicy production-db-policy
```

Modifica los campos según lo necesites:

```
spec:
  usedThreshold: "75%"     # Changed from 80%
  growthAmount: "20%"      # Changed from 10%
  maxSize: "1Ti"           # Changed from 500Gi
```

Guarda y sal. Verifica la política actualizada:

```
kubectl get tridentautogrowpolicy production-db-policy
```

Salida esperada

```
NAME                    USED THRESHOLD   GROWTH AMOUNT   STATE
production-db-policy    75%              20%             Success
```

¿Por qué mi policy está en estado Failed?

Un estado `Failed` indica que la especificación de la política contiene errores de validación. Ejecuta el siguiente comando para ver los detalles del error:

```
kubectl describe tridentautogrowpolicy <policy-name>
```

Las causas comunes incluyen un `usedThreshold` inválido (debe ser 1–99%), un `growthAmount` que excede `maxSize` o un formato de cantidad de Kubernetes inválido. Corrige la especificación y vuelve a aplicar:

```
kubectl apply -f autogrow-policy.yaml
```

¿Por qué no puedo eliminar una policy?

Las políticas utilizan la protección de finalizadores. Si los volúmenes siguen utilizando la política, la eliminación pasa a un estado `Deleting` y queda en espera hasta que elimines todos los volúmenes de la política.

Identifica los volúmenes afectados:

```
tridentctl get autogrowpolicy production-db-policy -o yaml
```

Luego elimina la anotación de cada PVC:

```
# Option A: Explicitly disable autogrow
kubectl annotate pvc <pvc-name> \
  trident.netapp.io/autogrowPolicy="none" \
  --overwrite

# Option B: Remove the annotation entirely
kubectl annotate pvc <pvc-name> \
  trident.netapp.io/autogrowPolicy-
```

Después de que elimines todos los volúmenes, Trident libera el finalizador y elimina la política.

¿Qué backends admite la ampliación automática?

Autogrow es compatible con los backends de ONTAP. También es compatible con Google Cloud NetApp Volumes (GCNV) tanto para volúmenes NAS como SAN.

Autogrow es compatible con los protocolos NFS, iSCSI, FCP y NVMe. Los volúmenes de bloques sin procesar NVMe requieren ONTAP 9.16.1 o una versión posterior.

Es posible que los volúmenes existentes en implementaciones en entornos brownfield requieran que se complete la migración de la publicación de volúmenes antes de que el autogrow surta efecto. Para comprobar el estado de la migración, consulta los registros del controlador Trident:

```
kubectl logs -l app=trident-controller -n trident | grep "Migration
completed"
```

Los siguientes ejemplos de StorageClass muestran autogrow configurado para backends NAS y SAN:

Backend NAS

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-nas-autogrow
  annotations:
    trident.netapp.io/autogrowPolicy: "standard-autogrow"
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  fsType: "ext4"
allowVolumeExpansion: true
```

Backend SAN

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: database-storage
  annotations:
    trident.netapp.io/autogrowPolicy: "production-db-policy"
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
  fsType: "ext4"
allowVolumeExpansion: true
```

¿Cuál es la cantidad mínima de crecimiento para los volúmenes SAN?

Para volúmenes SAN, la cantidad mínima de crecimiento efectivo es de 51 MB. Si configuras un `growthAmount` de 50 MiB o menos, Trident aumenta automáticamente el crecimiento a 51 MB para la operación de redimensionamiento.

Por ejemplo, esta política establece un `growthAmount` de `"40Mi"`, pero Trident aplica un crecimiento de 51 MB para cualquier volumen SAN que lo use:

```
apiVersion: trident.netapp.io/v1
kind: TridentAutogrowPolicy
metadata:
  name: san-minimal-policy
spec:
  usedThreshold: "85%"
  growthAmount: "40Mi"
  maxSize: "100Gi"
```

Para evitar este ajuste automático, establece `growthAmount` en un valor mayor a 50 MiB:

```
apiVersion: trident.netapp.io/v1
kind: TridentAutogrowPolicy
metadata:
  name: san-policy
spec:
  usedThreshold: "85%"
  growthAmount: "100Mi"
  maxSize: "500Gi"
```

Importar volúmenes

Puedes importar volúmenes de almacenamiento existentes como un PV de Kubernetes usando `tridentctl import` o creando una Persistent Volume Claim (PVC) con anotaciones de importación de Trident.

Descripción general y consideraciones

Es posible importar un volumen en 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.

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

ADVERTENCIA	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.

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

◦ `--no-rename`* - Si usas la `--no-rename` flag, Trident conserva el nombre del volumen existente al importar volúmenes y gestiona el ciclo de vida de los volúmenes. Esta opción solo es compatible con los controladores `ontap-nas`, `ontap-san` (incluidos los sistemas ASA r2), `ontap-san-economy` y `ontap-nas-economy` (importación de qtree).

CONSEJO	Estas opciones son útiles si desea utilizar Kubernetes para cargas de trabajo en contenedores, pero por lo demás desea administrar 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

Puedes importar un volumen usando `tridentctl import` o creando un PVC con anotaciones de importación de Trident.

NOTA	Si usas anotaciones de PVC, no necesitas descargar ni usar `tridentctl` para importar el volumen.

Usando tridentctl

Pasos

1. Crea un archivo PVC (por ejemplo, `pvc.yaml`) que se usará para crear el PVC. El archivo PVC debe incluir `name`, `namespace`, `accessModes` y `storageClassName`. Opcionalmente, puedes especificar `unixPermissions` en tu definición de PVC.

   A continuación se muestra un ejemplo de una especificación mínima:

   ```
   kind: PersistentVolumeClaim
   apiVersion: v1
   metadata:
     name: my_claim
     namespace: my_namespace
   spec:
     accessModes:
       - ReadWriteOnce
     storageClassName: my_storage_class
   ```

   NOTA Incluye solo los parámetros necesarios. Parámetros adicionales como el nombre de PV o el tamaño del volumen pueden hacer que falle el comando de import.

2. Utilice el `tridentctl import` comando para especificar el nombre del backend de Trident que contiene el volumen y el nombre que identifica de forma única el volumen en el almacenamiento (por ejemplo: ONTAP FlexVol, Element Volume). El `-f` Se requiere un argumento para especificar la ruta al archivo PVC.

   ```
   tridentctl import volume <backendName> <volumeName> -f <path-to-pvc-file>
   ```

Uso de anotaciones de PVC

Pasos

1. Crea un archivo PVC YAML (por ejemplo, `pvc.yaml`) con las anotaciones de importación de Trident necesarias. El archivo PVC debe incluir:

   - `name` y `namespace` en metadatos
   - `accessModes`, `resources.requests.storage` y `storageClassName` en spec
   - Anotaciones:
     - `trident.netapp.io/importOriginalName`: nombre del volumen en el backend
     - `trident.netapp.io/importBackendUUID`: UUID del backend donde existe el volumen
     - `trident.netapp.io/notManaged` (*Opcional*): configúralo en `"true"` para volúmenes no gestionados. El valor predeterminado es `"false"`.

       El siguiente es un ejemplo de especificación para importar un volumen gestionado:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: <pvc-name>
  namespace: <namespace>
  annotations:
    trident.netapp.io/importOriginalName: "<volume-name>"
    trident.netapp.io/importBackendUUID: "<backend-uuid>"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: <size>
  storageClassName: <storage-class-name>
```

2. Aplica el archivo YAML de PVC a tu clúster de Kubernetes:

```
kubectl apply -f <pvc-file>.yaml
```

Trident importará automáticamente el volumen y lo vinculará al PVC.

Ejemplos

Revise los siguientes ejemplos de importación de volúmenes para los controladores compatibles.

NAS de ONTAP y NAS FlexGroup de ONTAP

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

NOTA

- El `ontap-nas-economy` driver gestiona el almacenamiento a nivel de qtree, así que no admite importar un volumen FlexVol por sí solo. Para importar qtrees con el `ontap-nas-economy` driver, consulta Importar qtrees (Tech Preview). Cuando Trident importa un qtree, también importa el FlexVol principal como un bucket que contiene el qtree.
- La `ontap-nas` y.. `ontap-nas-flexgroup` las controladoras no permiten nombres de volúmenes duplicados.

Cada volumen creado con el `ontap-nas` controlador es un FlexVol volume en el clúster de ONTAP. Al importar los volúmenes de FlexVol con `ontap-nas` el controlador, funciona igual. Los volúmenes de FlexVol que ya existen en un clúster de ONTAP pueden importarse como `ontap-nas` una RVP. Del mismo modo, los volúmenes de FlexGroup se pueden importar como `ontap-nas-flexgroup` RVP.

Ejemplos de ONTAP NAS usando tridentctl

Los siguientes ejemplos muestran cómo importar volúmenes gestionados y no gestionados usando

`tridentctl`.

Volumen gestionado

En el ejemplo siguiente se importa un volumen llamado `managed_volume` en un backend llamado `ontap_nas`:

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

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

Volumen no gestionado

Cuando se utiliza `--no-manage` el argumento, Trident no cambia el nombre del volumen.

El siguiente ejemplo importa `unmanaged_volume` en la `ontap_nas` backend:

```
tridentctl import volume nas_blog unmanaged_volume -f <path-to-pvc-
file> --no-manage

+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
| pvc-df07d542-afbc-11e9-8d9f-5254004dfdb7 | 1.0 GiB | standard      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | false   |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
```

Ejemplos de ONTAP NAS usando anotaciones de PVC

Los siguientes ejemplos muestran cómo importar volúmenes gestionados y no gestionados usando anotaciones de PVC.

Volumen gestionado

El siguiente ejemplo importa un volumen de 1GiB `ontap-nas` llamado `ontap_volume1` desde backend `81abcb27-ea63-49bb-b606-0a5315ac5f21` con el modo de acceso RWO establecido usando anotaciones PVC:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: <managed-imported-volume>
  namespace: <namespace>
  annotations:
    trident.netapp.io/importOriginalName: "ontap_volume1"
    trident.netapp.io/importBackendUUID: "81abcb27-ea63-49bb-b606-
0a5315ac5f21"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: <storage-class-name>
```

Volumen no gestionado

El siguiente ejemplo importa 1Gi `ontap-nas` volumen llamado `ontap-volume2` desde backend `34abcb27-ea63-49bb-b606-0a5315ac5f34` con el modo de acceso RWO configurado usando anotaciones PVC:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: <umanaged-imported-volume>
  namespace: <namespace>
  annotations:
    trident.netapp.io/importOriginalName: "ontap-volume2"
    trident.netapp.io/importBackendUUID: "34abcb27-ea63-49bb-b606-
0a5315ac5f34"
    trident.netapp.io/notManaged: "true"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: <storage-class-name>
```

Importar qtrees

NOTA	La importación de qtree es una función en fase de Tech Preview de Trident 26.06. Las funciones en fase de Tech Preview no son compatibles para uso en producción.

A partir de Trident 26.06, es posible importar qtrees de ONTAP ya existentes a Trident usando el controlador `ontap-nas-economy`. Identifica el qtree por su volumen FlexVol y los nombres del qtree en formato `flexvol/qtree`, junto con el UUID del backend que lo contiene. Las consideraciones generales sobre la importación en Descripción general y consideraciones también se aplican a la importación de qtrees.

Importar un qtree usando tridentctl

Especifica el qtree en formato `flexvol/qtree` como nombre del volumen. El siguiente ejemplo importa el qtree `qtree1` desde el volumen FlexVol `flexvol1` en el backend `ontap_nas_economy`:

```
tridentctl import volume ontap_nas_economy flexvol1/qtree1 -f <path-to-
pvc-file>
```

Para importar el qtree sin cambiarle el nombre, añade el `--no-rename` flag. Para importar el qtree sin que Trident gestione su ciclo de vida, añade el `--no-manage` flag.

Importar un qtree usando anotaciones PVC

Crea un PVC con las siguientes anotaciones:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: <pvc-name>
  namespace: <namespace>
  annotations:
    trident.netapp.io/importOriginalName: "flexvol1/qtree1"
    trident.netapp.io/importBackendUUID: "<backend-uuid>"
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: <size>
  storageClassName: <storage-class-name>
```

Para conservar los nombres existentes del volumen FlexVol y del qtree, añade la anotación `trident.netapp.io/importNoRename: "true"`. Para importar el qtree como no gestionado, añade la anotación `trident.netapp.io/notManaged: "true"`.

Nomenclatura y comportamiento de gestión

Importación gestionada (predeterminada)

Trident cambia el nombre del qtree para que coincida con el nombre del PVC y renombra el volumen FlexVol según la `ontap-nas-economy` convención de nomenclatura. La ruta de unión no cambia.

--no-rename

Trident conserva los nombres actuales de los volúmenes FlexVol y de las qtree. Los nombres ya deben seguir las convenciones de nomenclatura de Trident, o Trident no podrá gestionar las operaciones de eliminación o montaje de la qtree.

--no-manage

Trident hace referencia al qtree, pero no gestiona su ciclo de vida. Los nombres deben seguir las convenciones de nomenclatura de Trident, y Trident no utiliza ni gestiona políticas de exportación automática para el qtree.

Comportamiento de la política de exportación

Si el backend utiliza políticas de exportación automática, el qtree importado utiliza el modelo de políticas de exportación por qtree, en el que cada qtree tiene su propia política de exportación. Trident asigna al qtree una política de exportación vacía en el momento de la importación. En el momento de la publicación, Trident actualiza el volumen FlexVol con la política de exportación específica de cada backend si la exportación automática está habilitada y la política de exportación existente FlexVol es `default` o está vacía.

PRECAUCIÓN	Actualizar la política de exportación del volumen FlexVol puede interrumpir otros qtrees en el mismo volumen FlexVol que estén montados actualmente. Importa los qtrees solo cuando los qtrees en el volumen FlexVol estén desmontados.
ADVERTENCIA	No apuntes dos clústeres de Kubernetes al mismo SVM activo, y no importes un qtree que otro clúster siga gestionando. Si ambos clústeres gestionan el mismo volumen, las operaciones conflictivas pueden dañar o destruir los datos.

SAN de ONTAP

Trident admite la importación de volumen mediante el `ontap-san` (iSCSI, NVMe/TCP y FC) y `ontap-san-economy` Conductores.

Trident puede importar volúmenes ONTAP SAN FlexVol que contengan un solo LUN. Esto es coherente con la `ontap-san` controlador, que crea un FlexVol volume para cada PVC y un LUN dentro del FlexVol volume. Trident importa el FlexVol volume y lo asocia con la definición de PVC. Trident puede importar `ontap-san-economy` volúmenes que contienen múltiples LUN.

Los siguientes ejemplos muestran cómo importar volúmenes gestionados y no gestionados:

Volumen gestionado

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

El siguiente ejemplo importa el `ontap-san-managed` FlexVol volume 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 la `ontap_san` backend:

```
tridentctl import volume -n trident san_blog unmanaged_example_volume
-f pvc-import.yaml --no-manage
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
| pvc-1fc999c9-ce8c-459c-82e4-ed4380a4b228 | 1.0 GiB | san-blog      |
block    | e3275890-7d80-4af6-90cc-c7a0759f555a | online | false   |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
```

Si tiene LUN asignadas a iGroups que comparten un IQN con un IQN de nodo de Kubernetes, como se muestra en el ejemplo siguiente, recibirá el error: `LUN already mapped to initiator(s) in this group`. Deberá quitar el iniciador o desasignar la LUN para importar el volumen.

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

Ejemplos de ONTAP SAN-economy

Los siguientes ejemplos muestran cómo importar volúmenes gestionados y no gestionados para el `ontap-san-economy` backend.

Volumen gestionado

Al importar un volumen gestionado, Trident toma posesión del FlexVol y le cambia el nombre. Debes tener en cuenta este cambio de nombre cuando importes varios LUN del mismo FlexVol.

El siguiente ejemplo importa `lun1` desde el FlexVol `toimport` como un volumen gestionado llamado `vol-managed-saneco`:

```
tridentctl import volume vol-managed-saneco toimport/lun1 -f
import1.yaml
```

Después de importar `lun1`, Trident cambia el nombre del FlexVol (por ejemplo, a `trident_lun_pool_xyz`). Para importar LUN adicionales del mismo FlexVol, usa el nuevo nombre de FlexVol:

```
tridentctl import volume vol-managed-saneco trident_lun_pool_xyz/lun2
-f import2.yaml
```

NOTA El backend `ontap-san-economy` importa un LUN a la vez. Puedes automatizar varias importaciones usando un script.

Volumen no gestionado

Al importar un volumen no gestionado, Trident no se apropia del FlexVol. Sin embargo, el FlexVol y el LUN deben seguir las convenciones de nomenclatura de Trident.

Formato de nomenclatura de FlexVol

```
trident_lun_pool_STORAGEPREFIX_RANDOMSTRING
```

- `STORAGEPREFIX` es el valor de `storagePrefix` en tu configuración de backend. El valor predeterminado es `trident`.
- `RANDOMSTRING` es cualquier cadena que elijas.

Requisito de nomenclatura LUN

El LUN debe llamarse `lun0`.

Ejemplo

Si tu `storagePrefix` es `xyz`, la ruta completa al LUN es:

```
trident_lun_pool_xyz_randomstring/lun0
```

Elemento

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

`san` el controlador.

NOTA El controlador Element admite los nombres de volúmenes duplicados. Sin embargo, 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.

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

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

+----------------------------------------+--------+--------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+--------+--------------
+----------+----------------------------------------+--------+---------+
| pvc-970ce1ca-2096-4ecd-8545-ac7edc24a8fe | 10 GiB | basic-element |
block    | d3ba047a-ea0b-43f9-9c42-e38e58301c49 | online | true    |
+----------------------------------------+--------+--------------
+----------+----------------------------------------+--------+---------+
```

Azure NetApp Files

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

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

El siguiente ejemplo importa un `azure-netapp-files` volumen en el back-end `azurenetappfiles_40517` con la ruta del volumen `importvol1`.

```
tridentctl import volume azurenetappfiles_40517 importvol1 -f <path-to-
pvc-file> -n trident

+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
| pvc-0ee95d60-fd5c-448d-b505-b72901b3a4ab | 100 GiB | anf-storage   |
file     | 1c01274f-d94b-44a3-98a3-04c953c9a51e | online | true    |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
```

NetApp Volumes para Google Cloud

Trident admite la importación de volúmenes utilizando `google-cloud-netapp-volumes` el controlador.

El siguiente ejemplo importa un volumen en backend `backend-tbc-gcnv1` con el volumen `testvoleasiaeast1`.

```
tridentctl import volume backend-tbc-gcnv1 "testvoleasiaeast1" -f < path-
to-pvc> -n trident

+----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS
| PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
| pvc-a69cda19-218c-4ca9-a941-aea05dd13dc0 |  10 GiB | gcnv-nfs-sc-
identity | file     | 8c18cdf1-0770-4bc0-bcc5-c6295fe6d837 | online | true
|
+----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

En el siguiente ejemplo se importa `google-cloud-netapp-volumes` un volumen cuando hay dos volúmenes en la misma región:

```
tridentctl import volume backend-tbc-gcnv1
"projects/123456789100/locations/asia-east1-a/volumes/testvoleasiaeast1"
-f <path-to-pvc> -n trident

+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
|                  NAME                    |  SIZE   | STORAGE CLASS
| PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
| pvc-a69cda19-218c-4ca9-a941-aea05dd13dc0 |  10 GiB | gcnv-nfs-sc-
identity | file     | 8c18cdf1-0770-4bc0-bcc5-c6295fe6d837 | online | true
|
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

Personalizar nombres y etiquetas de volúmenes

Con Trident puede asignar nombres y etiquetas significativos a los volúmenes que cree. 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:

- Operaciones de creación, importación y clonado de volúmenes.
- En el caso de `ontap-nas-economy` En el controlador, solo el nombre del volumen Qtree cumple con la plantilla de nombre.
- En el caso de `ontap-san-economy` Controlador, solo el nombre de la LUN cumple con la plantilla de nombre.

Limitaciones

- Los nombres de volumen personalizados solo son compatibles con los controladores ONTAP locales.
- Las etiquetas personalizadas solo son compatibles con `ontap-san`, `ontap-nas`, y `ontap-nas-flexgroup` conductores.
- Los nombres de volumen personalizados no se aplican a los volúmenes existentes.

Comportamientos clave de los nombres de volúmenes personalizables

- 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.
- 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. 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, Trident añadirá algunos caracteres aleatorios para crear nombres de volúmenes únicos.
5. Si el nombre personalizado para un volumen económico NAS supera los 64 caracteres de longitud, Trident asignará un nombre a los volúmenes de acuerdo con 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

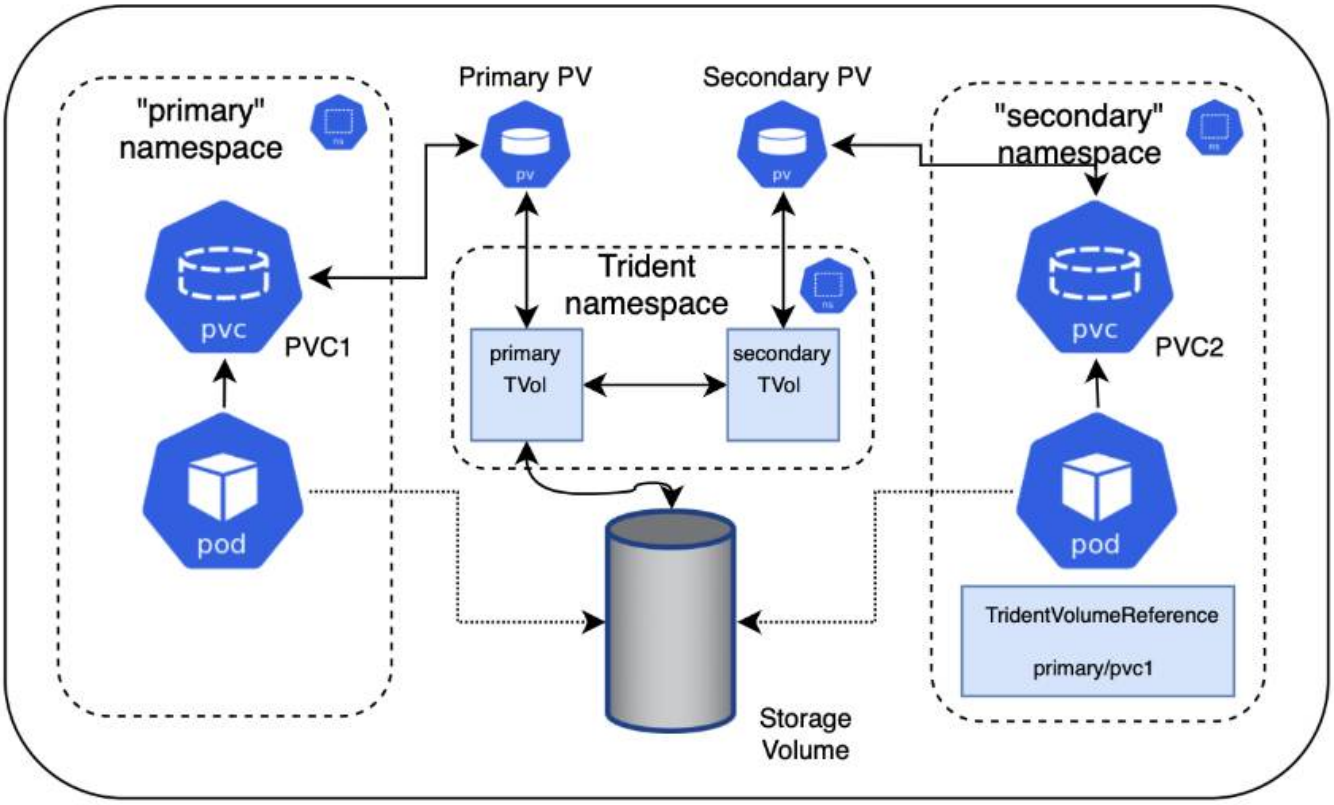
Con Trident, es posible crear un volumen en un espacio de nombres primario y compartirlo en uno o más espacios de nombres secundarios.

Funciones

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

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

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



Inicio rápido

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

1 **Configure la RVP de origen para compartir el volumen**

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

2 **Conceder permiso para crear una CR en el espacio de nombres de destino**

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

3 **Cree TridentVolumeReference en el espacio de nombres de destino**

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

4 **Cree el PVC subordinado en el espacio de nombres de destino**

El propietario del espacio de nombres de destino crea el PVC subordinado para utilizar el origen de datos desde el PVC de origen.

Configurar los espacios de nombres de origen y destino

Para garantizar la seguridad, el uso compartido entre espacios de nombres requiere la colaboración y la acción del propietario del espacio de nombres de origen, el administrador de clúster y el propietario del espacio de nombres de destino. La función de usuario se designa en cada paso.

Pasos

1. **Propietario del espacio de nombres de origen:** cree el PVC (`pvc1`) en el espacio de nombres de origen que concede permiso para compartir con el espacio de nombres de destino (`namespace2`) utilizando el `shareToNamespace` anotación.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1
  namespace: namespace1
  annotations:
    trident.netapp.io/shareToNamespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

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

   NOTA

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

2. **Administrador del clúster:** Asegúrese de que exista un RBAC adecuado para otorgar permiso al propietario del espacio de nombres de destino para crear la CR TridentVolumeReference en el espacio de nombres de destino.
3. **Propietario del espacio de nombres de destino:** cree un sistema TridentVolumeReference CR en el espacio de nombres de destino que haga referencia al espacio de nombres de origen `pvc1`.

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. **Propietario del espacio de nombres de destino:** cree un PVC (`pvc2`) en el espacio de nombres de destino (`namespace2`) utilizando el `shareFromPVC` Anotación para designar el PVC de origen.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  annotations:
    trident.netapp.io/shareFromPVC: namespace1/pvc1
  name: pvc2
  namespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

NOTA El tamaño del PVC de destino debe ser menor o igual que el PVC de origen.

Resultados

Trident lee la `shareFromPVC` anotación en la RVP de destino y crea el VP de destino como 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. Trident eliminará el acceso al volumen en el espacio de nombres de origen y mantendrá el acceso a otros espacios de nombres que compartan el volumen. Cuando se eliminan todos los espacios de nombres que hacen referencia al volumen, Trident lo elimina.

Uso `tridentctl get` para consultar volúmenes subordinados

Usando la utilidad[`tridentctl`, puedes ejecutar el comando `get` para obtener volúmenes subordinados. Para más información, consulta el enlace:../trident-reference/trident-cl.html[`tridentctl` comandos y

opciones].

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

Indicadores:

- `-h, --help`: Ayuda para volúmenes.
- `--parentOfSubordinate string`: Limite la consulta al volumen de origen subordinado.
- `--subordinateOf string`: Limite la consulta a las subordinadas del volumen.

Limitaciones

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

Si quiere más información

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

- Vea la demostración en "NetAppTV".

Clone volúmenes en espacios de nombres

Con Trident, puede crear nuevos volúmenes con volúmenes o copias de volúmenes existentes desde un espacio de nombres diferente dentro del mismo clúster de Kubernetes.

Requisitos previos

Antes de clonar volúmenes, asegúrese de que los back-ends de origen y de destino sean del mismo tipo y tengan la misma clase de almacenamiento.

NOTA	La clonación entre espacios de nombres solo se admite para `ontap-san` y `ontap-nas` controladores de almacenamiento. No se admiten clones de solo lectura.

Inicio rápido

Puede configurar el clonado de volúmenes en unos pocos pasos.

1. **Configure la PVC de origen para clonar el volumen**

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

origen.

2 Conceder permiso para crear una CR en el espacio de nombres de destino

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

3 Cree TridentVolumeReference en el espacio de nombres de destino

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

4 Cree la RVP del clon en el espacio de nombres de destino

El propietario del espacio de nombres de destino crea una RVP para clonar la RVP del espacio de nombres de origen.

Configurar los espacios de nombres de origen y destino

Para garantizar la seguridad, el clonado de volúmenes en espacios de nombres requiere la colaboración y acción del propietario del espacio de nombres de origen, el administrador del 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 (`namespace1`) (`pvc1`en el espacio de nombres de origen) que otorga permiso para compartir con el espacio de nombres de destino (`namespace2`) mediante la `cloneToNamespace` anotación.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1
  namespace: namespace1
  annotations:
    trident.netapp.io/cloneToNamespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

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

NOTA

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

2. **Administrador del clúster:** Asegúrese de que exista un RBAC adecuado para otorgar permiso al propietario del espacio de nombres de destino para crear la CR TridentVolumeReference en el espacio de nombres de destino(`namespace2`).
3. **Propietario del espacio de nombres de destino:** cree un sistema TridentVolumeReference CR en el espacio de nombres de destino que haga referencia al espacio de nombres de origen `pvc1`.

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. **Propietario del espacio de nombres de destino:** Crear un PVC (`namespace2`)(`pvc2` en el espacio de nombres de destino) utilizando el `cloneFromPVC` o `cloneFromSnapshot`, y `cloneFromNamespace` anotaciones para designar el PVC de origen.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  annotations:
    trident.netapp.io/cloneFromPVC: pvc1
    trident.netapp.io/cloneFromNamespace: namespace1
  name: pvc2
  namespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

Limitaciones

- En el caso de las RVP aprovisionadas con controladores para el sector económico ONTAP-nas, no se admiten clones de solo lectura.

Replicar volúmenes mediante SnapMirror

Trident admite las relaciones de mirroring entre un volumen de origen en un clúster y el volumen de destino en el clúster con relación entre iguales para replicar datos para la recuperación ante desastres. Puede utilizar una definición de recurso personalizado (CRD) con espacio de nombres, denominada relación de espejo Trident (TMR), para realizar las siguientes operaciones:

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

Requisitos previos de replicación

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

Clústeres ONTAP

- **Trident**: La versión 22,10 o posterior de Trident debe existir tanto en los clústeres de Kubernetes de origen como de destino que utilizan ONTAP como backend.
- **Licencias**: Las licencias asíncronas de SnapMirror de ONTAP que utilizan el paquete de protección de datos deben estar habilitadas en los clústeres de ONTAP de origen y de destino. Consulte "Información general sobre las licencias de SnapMirror en ONTAP" si desea obtener más información.

  A partir de ONTAP 9.10.1, todas las licencias se proporcionan como archivo de licencia de NetApp (NLF), que es un solo archivo que admite varias funciones. Consulte "Licencias incluidas con ONTAP One" si desea obtener más información.

NOTA	Sólo se admite la protección asincrónica SnapMirror.

Interconexión

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

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

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

Controladores compatibles

NetApp Trident admite la replicación de volúmenes con la tecnología NetApp SnapMirror utilizando clases de

almacenamiento respaldadas por los siguientes controladores: **`ontap-nas` : NFS** `ontap-san` : iSCSI **`ontap-san` :FC** `ontap-san` :NVMe/TCP (requiere la versión mínima de ONTAP 9.15.1)

NOTA	La replicación de volúmenes mediante SnapMirror no es compatible con los sistemas ASA r2. Para obtener información sobre los sistemas ASA r2, consulte "Obtenga información sobre los sistemas de almacenamiento R2 de ASA" .

Cree una RVP reflejada

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

Pasos

1. Realice los siguientes pasos en el clúster de Kubernetes principal:
   a. Cree un objeto StorageClass con el `trident.netapp.io/replication: true` parámetro.

      Ejemplo

      ```
      apiVersion: storage.k8s.io/v1
      kind: StorageClass
      metadata:
        name: csi-nas
      provisioner: csi.trident.netapp.io
      parameters:
        backendType: "ontap-nas"
        fsType: "nfs"
        trident.netapp.io/replication: "true"
      ```

   b. Cree una RVP con el tipo de almacenamiento creado anteriormente.

      Ejemplo

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

   c. Cree un CR de MirrorRelationship con información local.

Ejemplo

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

Trident recupera la información interna del volumen y el estado actual de protección de datos (DP) del volumen y, a continuación, rellena el campo de estado del MirrorRelationship.

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

```
kubectl get tmr csi-nas
```

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
  generation: 1
spec:
  state: promoted
  volumeMappings:
  - localPVCName: csi-nas
status:
  conditions:
  - state: promoted
    localVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
    localPVCName: csi-nas
    observedGeneration: 1
```

2. Realice los siguientes pasos en el clúster de Kubernetes secundario:

   a. Cree una StorageClass con el parámetro trident.netapp.io/replication: true.

Ejemplo

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-nas
provisioner: csi.trident.netapp.io
parameters:
  trident.netapp.io/replication: true
```

b. Cree un CR de MirrorRelationship con información de destino y origen.

Ejemplo

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: established
  volumeMappings:
  - localPVCName: csi-nas
    remoteVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
```

Trident creará una relación de SnapMirror con el nombre de la política de relaciones configurada (o por defecto para ONTAP) e inicializará la misma.

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

Ejemplo

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
  annotations:
    trident.netapp.io/mirrorRelationship: csi-nas
spec:
  accessModes:
  - ReadWriteMany
resources:
  requests:
    storage: 1Gi
storageClassName: csi-nas
```

Trident comprobará el CRD de TridentMirrorRelationship y no podrá crear el volumen si la relación no existe. Si existe la relación, Trident se asegurará de que el nuevo FlexVol volume se coloque en una SVM relacionada con la SVM remota definida en MirrorRelationship.

Estados de replicación de volúmenes

Una relación de mirroring de Trident (TMR) es un CRD que representa un extremo de una relación de replicación entre RVP. El TMR de destino tiene un estado que indica a Trident cuál es el estado deseado. El TMR de destino tiene los siguientes estados:

- **Establecido**: El PVC local es el volumen de destino de una relación de espejo, y esta es una nueva relación.
- **Promocionado**: El PVC local es ReadWrite y montable, sin relación de espejo actualmente en vigor.
- **Reestablecido**: El PVC local es el volumen de destino de una relación de espejo y también estaba anteriormente en esa relación de espejo.
  - El estado reestablecido se debe usar si el volumen de destino alguna vez mantuvo una relación con el volumen de origen debido a que sobrescribe el contenido del volumen de destino.
  - El estado reestablecido generará un error si el volumen no mantuvo una relación anteriormente con el origen.

Promocione la RVP secundaria durante una conmutación al respaldo no planificada

Realice el siguiente paso en el clúster de Kubernetes secundario:

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

Promocione la RVP secundaria durante una conmutación al respaldo planificada

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

Pasos

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

   Ejemplo

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

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

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

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

Pasos

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

Operaciones adicionales

Trident admite las siguientes operaciones en los volúmenes primarios y secundarios:

Replica la PVC primaria a una nueva PVC secundaria

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

Pasos

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

Cambie el tamaño de una RVP reflejada, primaria o secundaria

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

Elimine la replicación de una RVP

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

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

Eliminar una RVP (que se había duplicado previamente)

Trident comprueba si hay PVR replicadas y libera la relación de replicación antes de intentar eliminar el volumen.

Eliminar un TMR

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

Actualice las relaciones de reflejo cuando el ONTAP esté en línea

Las relaciones de reflejos se pueden actualizar en cualquier momento una vez establecidas. Puede utilizar los `state: promoted` campos o `state: reestablished` para actualizar las relaciones. Al promocionar un volumen de destino a un volumen de ReadWrite normal, se puede usar *promotedSnapshotHandle* para especificar una snapshot específica a la que restaurar el volumen actual.

Actualice las relaciones de reflejo cuando la ONTAP esté sin conexión

Puede utilizar un CRD para realizar una actualización de SnapMirror sin que Trident tenga conectividad directa al clúster de ONTAP. Consulte el siguiente formato de ejemplo de TridentActionMirrorUpdate:

Ejemplo

```
apiVersion: trident.netapp.io/v1
kind: TridentActionMirrorUpdate
metadata:
  name: update-mirror-b
spec:
  snapshotHandle: "pvc-1234/snapshot-1234"
  tridentMirrorRelationshipName: mirror-b
```

`status.state` Refleja el estado del CRD TridentActionMirrorUpdate. Puede tomar un valor de *succeeded*, *in progress* o *failed*.

Utilice Topología CSI

Trident puede crear y conectar volúmenes de forma selectiva a los nodos presentes en un clúster de Kubernetes 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 multizona, Trident utiliza la topología CSI.

CONSEJO Obtenga más información sobre la característica de topología CSI "aquí".

Kubernetes ofrece dos modos de enlace de volúmenes únicos:

- Con `VolumeBindingMode` Establecer en `Immediate`, Trident crea el volumen sin reconocimiento de 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` establezca en `WaitForFirstConsumer`, La creación y enlace de un volumen persistente para una RVP se retrasa hasta que se programa y crea un pod que usa la RVP. De esta forma, se crean volúmenes con el fin de cumplir las restricciones de programación que se aplican en los requisitos de topología.

NOTA	La `WaitForFirstConsumer` el modo de encuadernación no requiere etiquetas de topología. Esto se puede utilizar independientemente de la característica de topología CSI.

Lo que necesitará

Para utilizar la topología CSI, necesita lo siguiente:

- Un clúster de Kubernetes que ejecuta un "Compatible con la versión de Kubernetes"

```
kubectl version
Client Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:50:19Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedeafd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:41:49Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
```

- Los nodos del clúster deben tener etiquetas que introduzcan el reconocimiento de topología (`topology.kubernetes.io/region`y `topology.kubernetes.io/zone`). Estas etiquetas **deben estar presentes en los nodos del cluster** antes de instalar Trident para que Trident tenga en cuenta la topología.

```
kubectl get nodes -o=jsonpath='{range .items[*]}[{.metadata.name},
{.metadata.labels}]{"\n"}{end}' | grep --color "topology.kubernetes.io"
[node1,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node1","kubernetes.io/
os":"linux","node-
role.kubernetes.io/master":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-a"}]
[node2,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node2","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-b"}]
[node3,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kube
rnetes.io/arch":"amd64","kubernetes.io/hostname":"node3","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-c"}]
```

Paso 1: Cree un backend con detección de topología

Los back-ends de almacenamiento de Trident se pueden diseñar para aprovisionar volúmenes de forma selectiva según 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

```yaml
---
version: 1
storageDriverName: ontap-san
backendName: san-backend-us-east1
managementLIF: 192.168.27.5
svm: iscsi_svm
username: admin
password: password
supportedTopologies:
  - topology.kubernetes.io/region: us-east1
    topology.kubernetes.io/zone: us-east1-a
  - topology.kubernetes.io/region: us-east1
    topology.kubernetes.io/zone: us-east1-b
```

JSON

```json
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "san-backend-us-east1",
  "managementLIF": "192.168.27.5",
  "svm": "iscsi_svm",
  "username": "admin",
  "password": "password",
  "supportedTopologies": [
    {
      "topology.kubernetes.io/region": "us-east1",
      "topology.kubernetes.io/zone": "us-east1-a"
    },
    {
      "topology.kubernetes.io/region": "us-east1",
      "topology.kubernetes.io/zone": "us-east1-b"
    }
  ]
}
```

NOTA

`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. Para StorageClasses que contienen un subconjunto de las regiones y zonas proporcionadas en un backend, Trident crea un volumen en el backend.

Puede definir `supportedTopologies` por pool de almacenamiento también. Consulte el siguiente ejemplo:

```
---
version: 1
storageDriverName: ontap-nas
backendName: nas-backend-us-central1
managementLIF: 172.16.238.5
svm: nfs_svm
username: admin
password: password
supportedTopologies:
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-a
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-b
storage:
  - labels:
      workload: production
    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, la `region` y.. `zone` las etiquetas indican la ubicación del pool de almacenamiento. `topology.kubernetes.io/region` y.. `topology.kubernetes.io/zone` dicte desde donde se pueden consumir los pools de almacenamiento.

Paso 2: Defina las clases de almacenamiento que tienen en cuenta la topología

En función de las etiquetas de topología que se proporcionan a los nodos del clúster, se puede definir StorageClasse para que contenga información de topología. Esto determinará los pools de almacenamiento que sirven como candidatos para las solicitudes de RVP y el subconjunto de nodos que pueden usar los volúmenes aprovisionados mediante Trident.

Consulte el siguiente ejemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: netapp-san-us-east1
provisioner: csi.trident.netapp.io
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
  - matchLabelExpressions:
    - key: topology.kubernetes.io/zone
      values:
        - us-east1-a
        - us-east1-b
    - key: topology.kubernetes.io/region
      values:
        - us-east1
parameters:
  fsType: ext4
```

En la definición 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 crea RVP 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: null
name: pvc-san
spec: null
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: netapp-san-us-east1
```

La creación de una RVP con este manifiesto daría como resultado lo siguiente:

```
kubectl create -f pvc.yaml
persistentvolumeclaim/pvc-san created
kubectl get pvc
NAME      STATUS    VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS
AGE
pvc-san   Pending                                      netapp-san-us-east1
2s
kubectl describe pvc
Name:          pvc-san
Namespace:     default
StorageClass:  netapp-san-us-east1
Status:        Pending
Volume:
Labels:        <none>
Annotations:   <none>
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:
Access Modes:
VolumeMode:    Filesystem
Mounted By:    <none>
Events:
  Type    Reason                Age   From                         Message
  ----    ------                ----  ----                         -------
  Normal  WaitForFirstConsumer  6s    persistentvolume-controller  waiting
for first consumer to be created before binding
```

Para que Trident cree un volumen y lo enlace a la RVP, use la RVP en un pod. Consulte el siguiente ejemplo:

```
apiVersion: v1
kind: Pod
metadata:
  name: app-pod-1
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
        - matchExpressions:
          - key: topology.kubernetes.io/region
            operator: In
            values:
            - us-east1
      preferredDuringSchedulingIgnoredDuringExecution:
      - weight: 1
        preference:
          matchExpressions:
          - key: topology.kubernetes.io/zone
            operator: In
            values:
            - us-east1-a
            - us-east1-b
  securityContext:
    runAsUser: 1000
    runAsGroup: 3000
    fsGroup: 2000
  volumes:
  - name: vol1
    persistentVolumeClaim:
      claimName: pvc-san
  containers:
  - name: sec-ctx-demo
    image: busybox
    command: [ "sh", "-c", "sleep 1h" ]
    volumeMounts:
    - name: vol1
      mountPath: /data/demo
    securityContext:
      allowPrivilegeEscalation: false
```

Este podSpec indica a Kubernetes que programe el pod de los nodos presentes en el `us-east1` region y elija de cualquier nodo que esté presente en el `us-east1-a` o. `us-east1-b` zonas.

Consulte la siguiente salida:

```
kubectl get pods -o wide
NAME        READY   STATUS    RESTARTS   AGE   IP               NODE
NOMINATED NODE   READINESS GATES
app-pod-1   1/1     Running   0          19s   192.168.25.131   node2
<none>           <none>
kubectl get pvc -o wide
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS          AGE   VOLUMEMODE
pvc-san   Bound    pvc-ecb1e1a0-840c-463b-8b65-b3d033e2e62b   300Mi
RWO            netapp-san-us-east1   48s   Filesystem
```

Actualice los back-ends que se incluirán `supportedTopologies`

Se pueden actualizar los back-ends preexistentes para incluir una lista de `supportedTopologies` uso `tridentctl backend update`. Esto no afectará a los volúmenes que ya se han aprovisionado, y sólo se utilizarán en las siguientes CVP.

Obtenga más información

- "Gestione recursos para contenedores"
- "Selector de nodos"
- "Afinidad y anti-afinidad"
- "Tolerancias y taints"

Trabajar con instantáneas

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 Trident, importar una copia de Snapshot creada fuera de Trident, crear un volumen nuevo a partir de una copia de Snapshot existente y recuperar datos de volumen de copias de Snapshot.

Descripción general

La instantánea de volumen es compatible con `ontap-nas` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy` , `solidfire-san` , `azure-netapp-files` , y `google-cloud-netapp-volumes` 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 ni los CRD, consulte Implemente una controladora Snapshot de volumen.

NOTA	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".
   - Los `driver` puntos al controlador CSI de Trident.
   - `deletionPolicy` puede ser `Delete` o. `Retain`. Cuando se establece en `Retain`, la instantánea física subyacente en el clúster de almacenamiento se conserva incluso cuando `VolumeSnapshot` el objeto se ha eliminado.

     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 copia de snapshot se establece en `pvc1-snap`. Un VolumeSnapshot es análogo a un PVC y está asociado a un `VolumeSnapshotContent` objeto que representa la instantánea real.

```
kubectl create -f snap.yaml
volumesnapshot.snapshot.storage.k8s.io/pvc1-snap created

kubectl get volumesnapshots
NAME                    AGE
pvc1-snap               50s
```

- Puede identificar el `VolumeSnapshotContent` objeto para `pvc1-snap` VolumeSnapshot, describiéndolo. La `Snapshot Content Name` Identifica el objeto VolumeSnapshotContent que sirve esta snapshot. La `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 utilizar `dataSource` Para crear una RVP con una Snapshot de volumen llamada `<pvc-name>` como la fuente de los datos. Una vez creada la RVP, se puede conectar a un pod y utilizarla como cualquier otro PVC.

ADVERTENCIA	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 con `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

Trident admite "Proceso de snapshot aprovisionado previamente de Kubernetes"para permitir que el administrador de clúster cree `VolumeSnapshotContent` un objeto e importe copias de Snapshot creadas fuera de Trident.

Antes de empezar

Trident debe haber creado o importado el volumen principal de la snapshot.

Pasos

1. **Cluster admin:** Crear un `VolumeSnapshotContent` objeto que haga referencia a la instantánea backend. De esta forma, se inicia el flujo de trabajo Snapshot en Trident.
   - Especifique el nombre de la instantánea de backend en `annotations` como `trident.netapp.io/internalSnapshotName: <"backend-snapshot-name">`.
   - Especifique `<name-of-parent-volume-in-trident>/<volume-snapshot-content-name>` en `snapshotHandle`. Esta es la única información proporcionada a Trident por el Snapshotter externo en la `ListSnapshots` llamada.

     NOTA La `<volumeSnapshotContentName>` No siempre se 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>
  volumeSnapshotRef:
    name: import-snap
    namespace: default
```

2. **Administrador del clúster:** Crear el `VolumeSnapshot` CR que hace referencia al `VolumeSnapshotContent` objeto. Esto solicita acceso para utilizar el `VolumeSnapshot` en un espacio de nombres determinado.

   Ejemplo

   En el siguiente ejemplo se crea un `VolumeSnapshot` CR con nombre `import-snap` que hace referencia a la `VolumeSnapshotContent` nombre `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. Trident crea el `TridentSnapshot`.
   - El dispositivo de instantáneas externo establece el `VolumeSnapshotContent` para `readyToUse` y la `VolumeSnapshot` para `true`.
   - Trident vuelve `readyToUse=true`.
4. **Cualquier usuario:** Crear a. `PersistentVolumeClaim` para hacer referencia al nuevo `VolumeSnapshot`, donde `spec.dataSource` (o. `spec.dataSourceRef`) nombre es el `VolumeSnapshot` nombre.

Ejemplo

En el siguiente ejemplo se crea una RVP que hace referencia al `VolumeSnapshot` nombre `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 con el `ontap-nas` y.. `ontap-nas-economy` de windows Habilite el `.snapshot` directorio para recuperar datos de snapshots directamente.

Use la interfaz de línea de comandos de ONTAP para restaurar un volumen en un estado registrado en una snapshot anterior.

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

NOTA	Cuando se restaura una copia Snapshot, se sobrescribe la configuración de volúmenes existente. Se pierden los cambios que se hagan en los datos del volumen después de crear la copia Snapshot.

Restauración de volumen sin movimiento a partir de una copia de Snapshot

Trident permite restaurar volumen rápida y in situ a partir de una snapshot mediante `TridentActionSnapshotRestore` (TASR) CR. Esta CR funciona como una acción imprescindible de Kubernetes y no persiste una vez que finaliza la operación.

Trident admite la restauración de instantáneas en el `ontap-san` , `ontap-san-economy` , `ontap-nas` , `ontap-nas-flexgroup` , `azure-netapp-files` , `google-cloud-netapp-volumes` , y `solidfire-san` conductores.

Antes de empezar

Debe tener una snapshot de volumen disponible y la RVP vinculada.

- Compruebe que el estado de la RVP es de enlace.

```
kubectl get pvc
```

- Compruebe que la copia de Snapshot de volumen esté lista para utilizarse.

```
kubectl get vs
```

Pasos

1. Cree el CR de TASR. En este ejemplo, se crea una CR para la RVP `pvc1` y una instantánea de volumen `pvc1-snapshot`.

   NOTA El TASR CR debe estar en un espacio de nombres donde exista la PVC y VS.

```
cat tasr-pvc1-snapshot.yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentActionSnapshotRestore
metadata:
  name: trident-snap
  namespace: trident
spec:
  pvcName: pvc1
  volumeSnapshotName: pvc1-snapshot
```

2. Aplique el CR para restaurar a partir de la instantánea. Este ejemplo restaura desde la instantánea `pvc1`.

```
kubectl create -f tasr-pvc1-snapshot.yaml
```

```
tridentactionsnapshotrestore.trident.netapp.io/trident-snap created
```

Resultados

Trident restaura los datos a partir de la copia Snapshot. Puede verificar el estado de restauración de la Snapshot:

```
kubectl get tasr -o yaml
```

```
apiVersion: trident.netapp.io/v1
items:
- apiVersion: trident.netapp.io/v1
  kind: TridentActionSnapshotRestore
  metadata:
    creationTimestamp: "2023-04-14T00:20:33Z"
    generation: 3
    name: trident-snap
    namespace: trident
    resourceVersion: "3453847"
    uid: <uid>
  spec:
    pvcName: pvc1
    volumeSnapshotName: pvc1-snapshot
  status:
    startTime: "2023-04-14T00:20:34Z"
    completionTime: "2023-04-14T00:20:37Z"
    state: Succeeded
kind: List
metadata:
  resourceVersion: ""
```

NOTA

- En la mayoría de los casos, Trident no vuelve a intentar automáticamente la operación en caso de fallo. Deberá realizar la operación de nuevo.
- Es posible que el administrador deba conceder permiso al usuario de Kubernetes sin acceso de administrador para crear una CR TASR en su espacio de nombres de la aplicación.

Eliminar un VP con snapshots asociadas

Cuando se elimina un volumen persistente con snapshots asociadas, el volumen Trident correspondiente se actualiza a un estado «Eliminado». Quite las copias de Snapshot de volumen para eliminar el volumen de 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
```

NOTA Si es necesario, abra `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` y actualícelo `namespace` en el espacio de nombres.

Enlaces relacionados

- "Copias de Snapshot de volumen"
- "VolumeSnapshotClass"

Trabajar con instantáneas de grupos de volúmenes

Instantáneas de grupos de volúmenes de Kubernetes de volúmenes persistentes (PV). NetApp Trident permite crear instantáneas de varios volúmenes (un grupo de instantáneas de volumen). Esta instantánea de grupo de volúmenes representa copias de varios volúmenes tomadas al mismo tiempo.

NOTA VolumeGroupSnapshot es una función beta de Kubernetes con API beta. Kubernetes 1.32 es la versión mínima requerida para VolumeGroupSnapshot.

Crear instantáneas de grupos de volúmenes

La instantánea de grupo de volúmenes es compatible con los siguientes controladores de almacenamiento:

- `ontap-san`Controlador: solo para los protocolos iSCSI y FC, no para el protocolo NVMe/TCP.
- `ontap-san-economy` - únicamente para el protocolo iSCSI.
- `ontap-nas`

NOTA	La función de instantáneas de grupos de volúmenes no es compatible con los sistemas de almacenamiento NetApp ASA r2 o AFX.

Antes de empezar

- Asegúrese de que su versión de Kubernetes sea K8s 1.32 o superior.
- 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 externo y los CRD, consulte Implemente una controladora Snapshot de volumen .

NOTA	No cree un controlador de instantáneas si crea instantáneas de grupos de volúmenes a pedido en un entorno de GKE. GKE utiliza un controlador de instantáneas oculto integrado.

- En el controlador de instantáneas YAML, configure el `CSIVolumeGroupSnapshot` Establezca la puerta de función en 'verdadero' para garantizar que la instantánea del grupo de volúmenes esté habilitada.
- Cree las clases de instantáneas del grupo de volúmenes necesarias antes de crear una instantánea del grupo de volúmenes.
- Asegúrese de que todos los PVC/volúmenes estén en el mismo SVM para poder crear VolumeGroupSnapshot.

Pasos

- Cree una VolumeGroupSnapshotClass antes de crear un VolumeGroupSnapshot. Para obtener más información, consulte "Clase de instantánea de grupo de volúmenes".

```
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshotClass
metadata:
  name: csi-group-snap-class
  annotations:
    kubernetes.io/description: "Trident group snapshot class"
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

- Cree PVC con las etiquetas requeridas utilizando clases de almacenamiento existentes o agregue estas etiquetas a los PVC existentes.

  El siguiente ejemplo crea el PVC utilizando `pvc1-group-snap` como fuente de datos y etiqueta

`consistentGroupSnapshot: groupA` . Defina la clave y el valor de la etiqueta según sus requisitos.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1-group-snap
  labels:
    consistentGroupSnapshot: groupA
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Mi
  storageClassName: sc1-1
```

- Cree un VolumeGroupSnapshot con la misma etiqueta (`consistentGroupSnapshot: groupA`) especificado en el PVC.

  Este ejemplo crea una instantánea de un grupo de volúmenes:

```
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshot
metadata:
  name: "vgs1"
  namespace: trident
spec:
  volumeGroupSnapshotClassName: csi-group-snap-class
  source:
    selector:
      matchLabels:
        consistentGroupSnapshot: groupA
```

Recuperar datos de volumen mediante una instantánea de grupo

Puede restaurar volúmenes persistentes individuales utilizando las instantáneas individuales creadas como parte de la instantánea del grupo de volúmenes. No puede recuperar la instantánea del grupo de volúmenes como una unidad.

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
```

NOTA	Cuando se restaura una copia Snapshot, se sobrescribe la configuración de volúmenes existente. Se pierden los cambios que se hagan en los datos del volumen después de crear la copia Snapshot.

Restauración de volumen sin movimiento a partir de una copia de Snapshot

Trident permite restaurar volumen rápida y in situ a partir de una snapshot mediante `TridentActionSnapshotRestore` (TASR) CR. Esta CR funciona como una acción imprescindible de Kubernetes y no persiste una vez que finaliza la operación.

Para obtener más información, consulte "Restauración de volumen sin movimiento a partir de una copia de Snapshot".

Eliminar un PV con instantáneas de grupo asociadas

Al eliminar una instantánea de volumen de grupo:

- Puede eliminar VolumeGroupSnapshots en su totalidad, no instantáneas individuales del grupo.
- Si se eliminan PersistentVolumes mientras existe una instantánea para ese PersistentVolume, Trident moverá ese volumen a un estado de "eliminación" porque la instantánea debe eliminarse antes de que el volumen pueda eliminarse de manera segura.
- Si se ha creado un clon utilizando una instantánea agrupada y luego se desea eliminar el grupo, comenzará una operación de división en clonación y el grupo no se podrá eliminar hasta que se complete la división.

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-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotcl
asses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotco
ntents.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshots.
yaml
```

2. Cree la controladora Snapshot.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-8.2/deploy/kubernetes/snapshot-
controller/rbac-snapshot-controller.yaml
```

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-8.2/deploy/kubernetes/snapshot-
controller/setup-snapshot-controller.yaml
```

NOTA Si es necesario, abra `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` y actualícelo `namespace` en el espacio de nombres.

Enlaces relacionados

- "Clase de instantánea de grupo de volúmenes"
- "Copias de Snapshot de volumen"

Gestione y supervise Trident

Actualice Trident

Actualice Trident

A partir de la versión 24,02, Trident sigue una cadencia de lanzamiento de cuatro meses, entregando 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 actualizar al menos una vez al año para aprovechar las nuevas funciones de Trident.

Consideraciones antes de la actualización

Al actualizar a la versión más reciente de Trident, tenga en cuenta lo siguiente:

- Solo debe haber una instancia de Trident instalada en todos los espacios de nombres de un clúster de Kubernetes determinado.
- Trident 23,07 y versiones posteriores requieren v1 copias Snapshot de volumen y ya no admite instantáneas alfa o beta.
- Al actualizar, es importante que proporcione `parameter.fsType` el `StorageClasses` que utiliza Trident. Puede eliminar y volver a crear `StorageClasses` sin interrumpir los volúmenes existentes anteriores.
  - Este es un **requisito** para hacer cumplir "contextos de seguridad" Para volúmenes SAN.
  - El directorio sample input contiene ejemplos, como `storage-class-basic.yaml.templ` y `storage-class-bronze-default.yaml#`.
  - Para obtener más información, consulte "Problemas conocidos".

Paso 1: Seleccione una versión

Las versiones de Trident siguen una convención de nomenclatura basada en fechas `YY.MM`, donde “YY” es 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 24,06 (o cualquier versión de 24,06 puntos) a 25,06.
- 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 23,07 y desea actualizar a la versión 25,06:
  a. Primera actualización de 23.07 a 24.06.
  b. Luego actualice de 24.06 a 25.06.

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

Paso 2: Determine el método de instalación original

Para determinar qué versión utilizaba para instalar originalmente Trident:

1. Uso `kubectl get pods -n trident` para examinar los pods.
   - Si no hay ningún pod de operador, se instaló Trident utilizando `tridentctl`.
   - Si hay un pod de operador, se instaló Trident usando el operador Trident manualmente o usando Helm.
2. Si hay un pod de operador, utilice `kubectl describe torc` para determinar si Trident se instaló con Helm.
   - Si hay una etiqueta Helm, Trident se instaló usando Helm.
   - Si no hay ninguna etiqueta Helm, Trident se instaló manualmente usando el operador Trident.

Paso 3: Seleccione un método de actualización

Generalmente, deberías actualizar usando el mismo método que usaste para la instalación inicial, aunque puedes "desplazarse entre los métodos de instalación". Hay dos opciones para actualizar Trident.

- "Actualice con el operador Trident"

CONSEJO	Le sugerimos que 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 Trident, debe comprender los procesos en segundo plano que ocurren 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" para instalar y actualizar Trident es el manejo automático de objetos de Trident y Kubernetes sin interrumpir los volúmenes montados existentes. De esta manera, Trident puede soportar actualizaciones con cero tiempo de inactividad o "*actualizaciones sucesivas*". En particular, el operador de 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.

CONSEJO	Para obtener más información sobre la arquitectura de Trident en el clúster de Kubernetes, consulte "Arquitectura de 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 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.

Actualice una instalación de Trident con el operador Trident o Helm

Puedes actualizar Trident usando el operador Trident, ya sea manualmente o usando Helm. Puedes actualizar desde una instalación del operador Trident a otra instalación del operador Trident, o actualizar desde una instalación de `tridentctl` a una versión del operador Trident. Revisa "Seleccione un método de actualización" antes de actualizar una instalación del operador Trident.

Actualizar una instalación manual

Puede actualizar desde una instalación de operador Trident con ámbito de clúster a otra instalación de operador Trident con ámbito de clúster. Todas las versiones de Trident utilizan un operador con ámbito de clúster.

NOTA	Para actualizar desde Trident que se instaló con el operador de ámbito de espacio de nombres (versiones 20,07 a 20,10), use las instrucciones de actualización de de "la versión instalada"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 1,25 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 Trident:

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

2. Actualizar el `operator.yaml` , `tridentorchestrator_cr.yaml` , y `post_1_25_bundle.yaml` con el registro y las rutas de imagen de la versión a la que está actualizando (por ejemplo, 25.06) y el secreto correcto.
3. Elimine el operador Trident que se utilizó para instalar la instancia actual de Trident . Por ejemplo, si está actualizando desde la versión 25.02, ejecute el siguiente comando:

```
kubectl delete -f 25.02.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

4. Si ha personalizado la instalación inicial mediante `TridentOrchestrator` atributos, puede editar `TridentOrchestrator` objeto para modificar los parámetros de instalación. Esto podría incluir cambios realizados para especificar registros de imágenes de Trident y CSI reflejados para el modo sin conexión, habilitar registros de depuración o especificar secretos de extracción de imágenes.
5. Instale Trident utilizando el archivo YAML del paquete correcto para su entorno, donde *<bundle.yaml>* es `bundle_pre_1_25.yaml` o `bundle_post_1_25.yaml` basado en su versión de Kubernetes. Por ejemplo, si está instalando Trident 25.06.0, ejecute el siguiente comando:

```
kubectl create -f 25.06.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

6. Edite el torque tridente para incluir la imagen 25.06.0.

Actualizar una instalación Helm

Puede actualizar una instalación de Trident Helm.

ADVERTENCIA	Cuando actualice un clúster de Kubernetes de 1,24 a 1,25 o una versión posterior que tiene Trident instalado, debe actualizar los valores.yaml para establecer `excludePodSecurityPolicy` `true` o agregar `--set excludePodSecurityPolicy=true` al `helm upgrade` comando antes de poder actualizar el clúster.

Si ya has actualizado el clúster de Kubernetes de 1,24 a 1,25 sin actualizar el timón de Trident, la actualización de helm fallará. Para que la actualización de HELM se realice, realice estos pasos como requisitos previos:

1. Instale el plugin helm-mapkubeapis desde https://github.com/helm/helm-mapkubeapis.
2. Realizar una ejecución en seco para la versión de Trident en el espacio de nombres donde está instalado

Trident. Esto enumera los recursos, que se limpiarán.

```
helm mapkubeapis --dry-run trident  --namespace trident
```

3. Realice una carrera completa con el timón para realizar la limpieza.

```
helm mapkubeapis trident  --namespace trident
```

Pasos

1. Si "Trident instalado usando Helm", puedes usar `helm upgrade trident netapp-trident/trident-operator --version 100.2602.0` para actualizar en un solo paso. Si no agregaste el repositorio de Helm o no puedes usarlo para actualizar:
   a. Descargue la versión más reciente de Trident en "La sección *Assets* de GitHub".
   b. Usa el `helm upgrade` comando donde `trident-operator-26.06.0.tgz` refleja la versión a la que quieres actualizar.

```
helm upgrade <name> trident-operator-26.06.0.tgz
```

NOTA	Si establece opciones personalizadas durante la instalación inicial (como especificar registros privados reflejados para imágenes de Trident y CSI), agregue la `helm upgrade` comando que utiliza `--set` para asegurarse de que estas opciones están incluidas en el comando upgrade, de lo contrario, los valores se restablecerán a los valores predeterminados.

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

Actualizar desde a. `tridentctl` Instalación para el operador Trident

Puede actualizarlo a la versión más reciente del operador de Trident desde un `tridentctl` instalación. Los back-ends y EVs existentes estarán disponibles automáticamente.

NOTA	Antes de cambiar entre métodos de instalación, revisa "Moverse entre los métodos de instalación".

Pasos

1. Descargue la versión más reciente de Trident.

```
# Download the release required [26.06.0]
mkdir 26.06.0
cd 26.06.0
wget
https://github.com/NetApp/trident/releases/download/v26.06.0/trident-
installer-26.06.0.tar.gz
tar -xf trident-installer-26.06.0.tar.gz
cd trident-installer
```

2. Cree el `tridentorchestrator` CRD del manifiesto.

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

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

```
kubectl create -f deploy/<bundle-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. Cree un `TridentOrchestrator` CR para instalar 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:                v26.06.0
```

Actualice con trimentctl

Puede actualizar fácilmente una instalación existente de Trident usando `tridentctl`.

Acerca de esta tarea

La desinstalación y la reinstalación de Trident actúa como una actualización. Al desinstalar Trident, la reclamación de volumen persistente (RVP) y el volumen persistente (PV) que utiliza la implementación de Trident no se eliminan. Los VP que ya se hayan aprovisionado permanecerán disponibles mientras Trident esté desconectado, y Trident aprovisionará volúmenes para cualquier RVP que se creen en ese momento tras su nuevo funcionamiento.

Antes de empezar

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

Pasos

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

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

2. Vuelve a instalar Trident. Consulta "Instale Trident usando tridentctl".

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

Gestione Trident con tridentctl

```
https://github.com/NetApp/trident/releases["Paquete de instalación de
Trident"^]Incluye la `tridentctl` utilidad de línea de comandos para
ofrecer un acceso simple a Trident. Los usuarios de Kubernetes con
suficiente Privileges pueden usarlo para instalar Trident o gestionar el
espacio de nombres que contiene el pod de Trident.
```

Comandos e indicadores globales

Puede ejecutar `tridentctl help` para obtener una lista de comandos disponibles para `tridentctl` o añada la `--help` marque cualquier comando para obtener una lista de opciones e indicadores para ese comando específico.

```
tridentctl [command] [--optional-flag]
```

La utilidad Trident `tridentctl` admite los siguientes comandos e indicadores globales.

Comandos

`create`
Añadir un recurso a Trident.

`delete`
Quite uno o más recursos de Trident.

`get`
Obtenga uno o más recursos de Trident.

`help`
Ayuda sobre cualquier comando.

`images`
Imprima una tabla de las imágenes de contenedor que Trident necesita.

`import`
Importe un recurso existente a Trident.

`install`
Instale Trident.

`logs`
Imprima los registros desde Trident.

`send`
Envíe un recurso desde Trident.

`uninstall`
Desinstale Trident.

`update`
Modifique un recurso en Trident.

`update backend state`
Suspender temporalmente las operaciones de backend.

`upgrade`
Actualice un recurso en Trident.

`version`
Imprima la versión de Trident.

Indicadores globales

`-d, --debug`
Salida de depuración.

`-h, --help`
Ayuda de `tridentctl`.

`-k, --kubeconfig string`
Especifique el `KUBECONFIG` Ruta para ejecutar comandos localmente o desde un clúster de Kubernetes a otro.

> **NOTA** Como alternativa, puede exportar el `KUBECONFIG` Variable para apuntar a un clúster y un problema de Kubernetes específicos `tridentctl` comandos para ese clúster.

`-n, --namespace string`
Puesta en marcha de espacio de nombres de 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 Trident.

> **ADVERTENCIA** 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 agregar un recurso a Trident.

```
tridentctl create [option]
```

Opciones

`backend`: Añadir un backend a Trident.

eliminar

Utilice `delete` el comando para quitar uno o varios recursos de Trident.

```
tridentctl delete [option]
```

Opciones

`backend`: Eliminar uno o más back-ends de almacenamiento de Trident.
`snapshot`: Elimine una o varias snapshots de volumen de Trident.
`storageclass`: Eliminar una o más clases de almacenamiento de Trident.

`volume`: Eliminar uno o varios volúmenes de almacenamiento de Trident.

obtenga

Utilice `get` el comando para obtener uno o varios recursos de Trident.

```
tridentctl get [option]
```

Opciones

`backend`: Obtenga uno o más backends de almacenamiento de Trident.
`snapshot`: Obtenga una o más instantáneas de Trident.
`storageclass`: Obtenga una o más clases de almacenamiento de Trident.
`volume`: Obtenga uno o más volúmenes de Trident.

Indicadores

`-h`, `--help`: Ayuda para volúmenes.
`--parentOfSubordinate string`: Limite la consulta al volumen de origen subordinado.
`--subordinateOf string`: Limite la consulta a las subordinadas del volumen.

imágenes

Utilice `images` indicadores para imprimir una tabla de las imágenes de contenedor que necesita Trident.

```
tridentctl images [flags]
```

Indicadores

`-h`, `--help`: Ayuda para imágenes.
`-v`, `--k8s-version string`: Versión semántica del clúster de Kubernetes.

importe volumen

Utilice `import volume` el comando para importar un volumen existente a Trident.

```
tridentctl import volume <backendName> <volumeName> [flags]
```

Alias

`volume`, `v`

Indicadores

`-f`, `--filename string`: Ruta al archivo YLMA o JSON PVC.
`-h`, `--help`: Ayuda para el volumen.
`--no-manage`: Cree sólo PV/PVC. No asuma que se gestiona el ciclo de vida de los volúmenes.

instale

Utilice los `install` indicadores para instalar Trident.

```
tridentctl install [flags]
```

Indicadores

`--autosupport-image string`:La imagen del contenedor para Autosupport Telemetry (valor predeterminado "netapp/trident autosupport:<current-version>").
`--autosupport-proxy string` :La dirección/puerto de un proxy para enviar telemetría de soporte automático.
`--enable-node-prep` :Intente instalar los paquetes necesarios en los nodos.
`--generate-custom-yaml` :Genera archivos YAML sin instalar nada.
`-h` , `--help` :Ayuda para la instalación.
`--http-request-timeout` :Anula el tiempo de espera de la solicitud HTTP para la API REST del controlador Trident (valor predeterminado 1 m 30 s).
`--image-registry string` :La dirección/puerto de un registro de imágenes interno.
`--k8s-timeout duration` :El tiempo de espera para todas las operaciones de Kubernetes (valor predeterminado: 3 m0 s).
`--kubelet-dir string` :La ubicación del host del estado interno de kubelet (predeterminado "/var/lib/kubelet").
`--log-format string` :El formato de registro de Trident (texto, json) (valor predeterminado "texto").
`--node-prep` :Permite a Trident preparar los nodos del clúster Kubernetes para administrar volúmenes utilizando el protocolo de almacenamiento de datos especificado. **Actualmente, `iscsi` Es el único valor admitido. A partir de OpenShift 4.19, la versión mínima de Trident compatible con esta función es 25.06.1.**
`--pv string` :El nombre del PV legado utilizado por Trident, asegura que este no exista (predeterminado "trident").
`--pvc string` :El nombre del PVC heredado utilizado por Trident, asegura que este no exista (predeterminado "trident").
`--silence-autosupport` :No envíe paquetes de soporte automático a NetApp automáticamente (valor predeterminado: verdadero).
`--silent` :Desactivar la mayoría de las salidas durante la instalación.
`--trident-image string` :La imagen de Trident para instalar.
`--k8s-api-qps` :El límite de consultas por segundo (QPS) para las solicitudes de API de Kubernetes (valor predeterminado 100; opcional).
`--use-custom-yaml` :Utilice cualquier archivo YAML existente que se encuentre en el directorio de instalación.
`--use-ipv6` :Utilice IPv6 para la comunicación de Trident.

registros

Utilice `logs` los indicadores para imprimir los registros de 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 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 Trident.

```
tridentctl send [option]
```

Opciones

`autosupport`: Enviar un fichero AutoSupport a NetApp.

desinstalar

Utilice `uninstall` los indicadores para desinstalar Trident.

```
tridentctl uninstall [flags]
```

Indicadores

`-h, --help`: Ayuda para la desinstalación.
`--silent`: Desactiva la mayoría de la salida durante la desinstalación.

actualizar

Utilice `update` el comando para modificar un recurso en Trident.

```
tridentctl update [option]
```

Opciones

`backend`: Actualizar un backend en Trident.

actualizar estado de backend

Utilice la `update backend state` 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 configurar el `userState` tridentctl vía tbc, el campo debe eliminarse del tbc `userState`. Esto se puede hacer usando `kubectl edit tbc` el comando. Una vez `userState` eliminado el 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 en `suspended` para pausar las operaciones de backend. Establezca en

`normal` para reanudar las operaciones de back-end. 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 usando el `tridentctl update backend` comando y la ruta a la actualización `backend.json` archivo.

   Ejemplo: `tridentctl update backend -f /<path to backend JSON file>/backend.json -n trident`

```
{
  "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

Uso `version` indicadores para imprimir la versión de `tridentctl` Y el servicio Trident que se ejecuta.

```
tridentctl version [flags]
```

Indicadores

`--client`: Sólo versión de cliente (no se necesita ningún servidor).
`-h, --help`: Ayuda para la versión.

Compatibilidad con complementos

Tridentctl soporta plugins similares a kubectl. Tridentctl detecta un plugin si el nombre del archivo binario del plugin sigue el esquema “tridentctl-<plugin>”, y el binario se encuentra en una carpeta que enumera la variable de entorno PATH. Todos los plugins detectados se enumeran en la sección de plugins de la ayuda tridentctl. Opcionalmente, también puede limitar la búsqueda especificando una carpeta de plugin en la variable de entorno TRIDENTCTL_PLUGIN_PATH (Ejemplo `TRIDENTCTL_PLUGIN_PATH=~/tridentctl-plugins/`:). Si se utiliza la variable, tridenctl busca solo en la carpeta especificada.

Supervisar Trident

Trident proporciona un conjunto de puntos finales de métricas de Prometheus que puede utilizar para supervisar el rendimiento de Trident.

Descripción general

Las métricas que proporciona Trident le permiten hacer lo siguiente:

- Estar al tanto del estado y la configuración de 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. Puede ver cuánto tiempo tarda Trident en comunicarse con los back-ends y realizar operaciones.

NOTA	Por defecto, las métricas de Trident se exponen en el puerto de destino. `8001` al `/metrics` punto final. Estas métricas están **habilitadas por defecto** cuando se instala Trident . Puedes configurar el consumo de métricas de Trident a través de HTTPS en el puerto `8444` también.

Lo que necesitará

- Un clúster de Kubernetes con Trident instalado.
- Una instancia Prometheus. Esto puede ser un "Puesta en marcha de Prometeo en contenedores" También puede optar por ejecutar Prometheus como a. "aplicación nativa".

Paso 1: Definir un objetivo Prometheus

Debes definir un objetivo de Prometheus para recopilar las métricas y obtener información sobre los backends que gestiona Trident , los volúmenes que crea, etc. Ver"Documentación del operador de Prometheus" .

Paso 2: Cree un Prometheus ServiceMonitor

Para usar las métricas de Trident, debe crear un Prometheus ServiceMonitor que vaya a ver el `trident-csi` el servicio y escucha el `metrics` puerto. Un ejemplo de ServiceMonitor tiene este aspecto:

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: trident-sm
  namespace: monitoring
  labels:
    release: prom-operator
spec:
  jobLabel: trident
  selector:
    matchLabels:
      app: controller.csi.trident.netapp.io
  namespaceSelector:
    matchNames:
      - trident
  endpoints:
    - port: metrics
      interval: 15s
```

Esta definición de ServiceMonitor recupera las métricas devueltas por el `trident-csi` servicio y específicamente busca el `metrics` punto final del servicio. Como resultado, Prometheus ahora está configurado para comprender las métricas de Trident.

Además de las métricas disponibles directamente desde 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".

Consumir métricas de Trident a través de HTTPS

Para consumir métricas de Trident a través de HTTPS (puerto 8444), debe modificar la definición de ServiceMonitor para incluir la configuración TLS. También necesitas copiar el `trident-csi` secreto de `trident` espacio de nombres al espacio de nombres donde se está ejecutando Prometheus. Puedes hacerlo utilizando el siguiente comando:

```
kubectl get secret trident-csi -n trident -o yaml | sed 's/namespace:
trident/namespace: monitoring/' | kubectl apply -f -
```

Un ejemplo de ServiceMonitor para métricas HTTPS tiene el siguiente 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:
    - interval: 15s
      path: /metrics
      port: https-metrics
      scheme: https
      tlsConfig:
        ca:
          secret:
            key: caCert
            name: trident-csi
        cert:
          secret:
            key: clientCert
            name: trident-csi
        keySecret:
          key: clientKey
          name: trident-csi
        serverName: trident-csi
```

Trident admite métricas HTTPS en todos los métodos de instalación: tridentctl, Helm chart y Operator:

- Si estás utilizando el `tridentctl install` comando, puedes pasar el `--https-metrics` Bandera para habilitar las métricas HTTPS.
- Si estás utilizando el gráfico de Helm, puedes configurarlo. `httpsMetrics` Parámetro para habilitar las métricas HTTPS.
- Si utiliza archivos YAML, puede agregar el `--https_metrics` bandera a la `trident-main` contenedor en el `trident-deployment.yaml` archivo.

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 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 REST de 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 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 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

NOTA 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
```

Más información sobre la telemetría de Trident AutoSupport

De forma predeterminada, Trident envía métricas de Prometheus e información básica de backend a NetApp en una cadencia diaria.

- Para evitar que Trident envíe métricas de Prometheus e información básica de back-end a NetApp, pase el `--silence-autosupport` indicador durante la instalación de Trident.
- Trident también puede enviar registros de contenedores al soporte de NetApp bajo demanda a través `tridentctl send autosupport`de . Deberá activar Trident para que cargue sus registros. Antes de enviar registros, debe aceptar NetApphttps://www.netapp.com/company/legal/privacy-policy/["política de privacidad"^].
- A menos que se especifique, 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 que se instala junto con Trident. Puede obtener la imagen del contenedor en "AutoSupport de Trident".
- Trident AutoSupport no recopila ni transmite información personal identificable (PII) ni información personal. Viene con una "CLUF" que no aplica a la propia imagen del contenedor de Trident. Puedes aprender más sobre el compromiso de NetApp con la seguridad y la confianza de los datos "aquí".

Un ejemplo de carga útil enviada por Trident tiene el siguiente aspecto:

```
---
items:
  - backendUUID: ff3852e1-18a5-4df4-b2d3-f59f829627ed
    protocol: file
    config:
      version: 1
      storageDriverName: ontap-nas
      debug: false
      debugTraceFlags: null
      disableDelete: false
      serialNumbers:
        - nwkvzfanek_SN
      limitVolumeSize: ""
    state: online
    online: true
```

- Los mensajes de AutoSupport se envían al extremo AutoSupport de NetApp. Si está utilizando un Registro privado para almacenar imágenes contenedoras, puede utilizar `--image-registry` bandera.
- También puede configurar direcciones URL proxy generando los archivos YLMA de instalación. Esto se puede hacer usando `tridentctl install --generate-custom-yaml` Para crear los archivos

YAML y agregar `--proxy-url` argumento para `trident-autosupport` contenedor en `trident-deployment.yaml`.

Deshabilitar las métricas de Trident

Para **desactivar las métricas** de ser reportadas, debe generar YAMLs personalizados (utilizando la `--generate-custom-yaml` y editarlas para eliminar `--metrics` no se invoca el indicador para el `trident-main`contenedor.

Desinstale Trident

Debe usar el mismo método para desinstalar Trident que utilizó para instalar Trident.

Acerca de esta tarea

- Si necesita una corrección para los errores observados después de una actualización, problemas de dependencia o una actualización incorrecta o incompleta, debe desinstalar Trident y volver a instalar la versión anterior usando 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, desinstalar Trident no elimina los CRD ni los objetos relacionados creados por Trident. Si necesita eliminar completamente Trident y todos sus datos, consulte "Eliminar completamente Trident y CRD".

Antes de empezar

Si va a decomisionar clústeres de Kubernetes, debe eliminar todas las aplicaciones que usan volúmenes creados por 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 utilizar el mismo método para desinstalar Trident que utilizó para instalarlo. Antes de desinstalar, verifique qué versión utilizó para instalar Trident originalmente.

1. Uso `kubectl get pods -n trident` para examinar los pods.
   - Si no hay ningún pod de operador, se instaló Trident utilizando `tridentctl`.
   - Si hay un pod de operador, se instaló Trident usando el operador Trident manualmente o usando Helm.
2. Si hay un pod de operador, utilice `kubectl describe tproc trident` para determinar si Trident se instaló con Helm.
   - Si hay una etiqueta Helm, Trident se instaló usando Helm.
   - Si no hay ninguna etiqueta Helm, Trident se instaló manualmente usando 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 ha instalado Trident utilizando el operador, puede desinstalarlo realizando una de las siguientes acciones:

1. **Edición `TridentOrchestrator` CR y establecer el indicador de desinstalación**:

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

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

2. **Eliminar `TridentOrchestrator`**: Al eliminar el `TridentOrchestrator` CR que se utilizó para implementar Trident, le indica al operador que desinstale Trident. El operador procesa la eliminación `TridentOrchestrator` y procede a eliminar el despliegue de 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 instaló Trident usando Helm, puede desinstalarlo usando `helm uninstall`.

```
#List the Helm release corresponding to the 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 un `tridentctl` instalación

Utilice `uninstall` el comando en `tridentctl` para eliminar todos los recursos asociados con Trident, excepto los CRD y los objetos relacionados:

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

Trident para Docker

Requisitos previos para la implementación

Tiene que instalar y configurar los requisitos previos del protocolo necesarios en su host antes de poder implementar Trident.

Compruebe los requisitos

- Compruebe que la implementación se adapte a todas las "requisitos".
- Compruebe que tiene instalada una versión compatible de Docker. Si la versión de Docker no está actualizada, "instálelo o actualícelo".

```
docker --version
```

- 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
```

ADVERTENCIA	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
```

> **NOTA** Asegúrese `etc/multipath.conf` contiene `find_multipaths no` inferior `defaults`.

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

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

6. Activar e iniciar `iscsi`:

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

Ubuntu

1. Instale los siguientes paquetes del sistema:

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

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

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

3. Configure el escaneo en manual:

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

4. Activar accesos múltiples:

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

NOTA Asegúrese `etc/multipath.conf` contiene `find_multipaths no` inferior `defaults`.

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

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

Herramientas de NVMe

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

NOTA

- 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
```

Herramientas de FC

Instale las herramientas de FC mediante los comandos del sistema operativo.

- Cuando se utilicen nodos de trabajo que ejecuten RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) con VP FC, especifique `discard` mountOption en StorageClass para realizar la recuperación de espacio inline. Consulte "Documentación de Red Hat".

RHEL 8 O POSTERIOR

1. Instale los siguientes paquetes del sistema:

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

2. Activar accesos múltiples:

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

NOTA Asegúrese `etc/multipath.conf` contiene `find_multipaths no` inferior `defaults`.

3. Asegúrese de que `multipathd` se está ejecutando:

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

Ubuntu

1. Instale los siguientes paquetes del sistema:

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

2. 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
```

NOTA Asegúrese `etc/multipath.conf` contiene `find_multipaths no` inferior `defaults`.

3. Asegúrese de que `multipath-tools` está activado y en ejecución:

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

Ponga en marcha Trident

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.

Se pueden ejecutar varias instancias de 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 implementar Trident.

Método de complemento gestionado por Docker (versión 1.13/17.03 y posteriores)

NOTA	**Antes de empezar** Si ha utilizado Trident pre Docker 1,13/17,03 en el método de daemon tradicional, asegúrese de detener el proceso de Trident y reiniciar el daemon de Docker antes de utilizar el método de complemento gestionado.

1. Detener todas las instancias en ejecución:

```
pkill /usr/local/bin/netappdvp
pkill /usr/local/bin/trident
```

2. Reinicie Docker.

```
systemctl restart docker
```

3. Asegúrese de tener instalado Docker Engine 17.03 (nuevo 1.13) o una versión posterior.

```
docker --version
```

Si su versión no está actualizada, "instale o actualice la instalación".

Pasos

1. Cree un archivo de configuración y especifique las opciones siguientes:
   - `config`: El nombre de archivo predeterminado es `config.json`, sin embargo, puede utilizar cualquier nombre que elija si especifica `config` opción con el nombre de archivo. El archivo de configuración debe estar ubicado en la `/etc/netappdvp` directorio en el sistema host.
   - `log-level`: Especifique el nivel de registro (`debug`, `info`, `warn`, `error`, `fatal`). El valor predeterminado es `info`.

- `debug`: Especifique si el registro de depuración está activado. El valor predeterminado es false. Reemplaza el nivel de registro si es TRUE.

  i. Cree una ubicación para el archivo de configuración:

  ```
  sudo mkdir -p /etc/netappdvp
  ```

  ii. Cree el archivo de configuración:

  ```
  cat << EOF > /etc/netappdvp/config.json
  ```

  ```
  {
    "version": 1,
    "storageDriverName": "ontap-nas",
    "managementLIF": "10.0.0.1",
    "dataLIF": "10.0.0.2",
    "svm": "svm_nfs",
    "username": "vsadmin",
    "password": "password",
    "aggregate": "aggr1"
  }
  EOF
  ```

2. Inicie Trident utilizando el sistema de complementos gestionados. 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 utilizar Trident para consumir almacenamiento del 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/10.0/trident-
      installer-26.02.0.tar.gz
      tar zxf trident-installer-26.02.0.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
```

NOTA A menos que se especifique, el nombre predeterminado para el controlador de volumen es NetApp.

Una vez iniciado el daemon, puede crear y gestionar volúmenes mediante la interfaz de la 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 Trident al iniciar el sistema

Puede encontrar un archivo de ejemplo de unidad para sistemas basados en el sistema en

`contrib/trident.service.example` En el Git repo. Para utilizar el archivo con RHEL, realice lo siguiente:

1. Copie el archivo en la ubicación correcta.

   Debe utilizar nombres únicos para los archivos de unidad si tiene más de una instancia en ejecución.

   ```
   cp contrib/trident.service.example
   /usr/lib/systemd/system/trident.service
   ```

2. Edite el archivo, cambie la descripción (línea 2) para que coincida con el nombre del controlador y la ruta del archivo de configuración (línea 9) para reflejar su entorno.
3. Vuelva a cargar systemd para que procese los cambios:

   ```
   systemctl daemon-reload
   ```

4. Active el servicio.

   Este nombre varía en función de lo que haya nombrado el archivo en el `/usr/lib/systemd/system` directorio.

   ```
   systemctl enable trident
   ```

5. Inicie el servicio.

   ```
   systemctl start trident
   ```

6. Ver el estado.

   ```
   systemctl status trident
   ```

NOTA Siempre que modifique el archivo de unidad, ejecute el `systemctl daemon-reload` comando para que tenga en cuenta los cambios.

Actualice o desinstale Trident

Puede actualizar Trident para Docker de forma segura sin afectar a los volúmenes que se utilizan. 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 Trident para 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
```

NOTA La versión 18,01 de Trident sustituye al 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
```

IMPORTANTE	Si va a actualizar desde una versión anterior de Trident (anterior a 20,10) a Trident 20,10 o posterior, puede que se produzca un error. Para obtener más información, consulte "Problemas conocidos". Si se encuentra con el error, primero debe deshabilitar el plugin, luego quitar el plugin y luego instalar la versión de 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 los pasos que se indican a continuación para desinstalar Trident para 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

Es posible crear, clonar y quitar volúmenes fácilmente mediante los comandos estándar `docker volume` con el nombre de controlador de Trident especificado cuando se necesita.

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 Trident:

```
docker volume create -d ntap_bronze --name bronzeVolume
```

NOTA Si no especifica ninguna "opciones", se utilizan los valores predeterminados del controlador.

- Anule el tamaño de volumen predeterminado. Consulte el siguiente ejemplo para crear un volumen de 20 GiB con un controlador:

```
docker volume create -d netapp --name my_vol --opt size=20G
```

CONSEJO Los tamaños de volumen se expresan como cadenas que contienen un valor entero con unidades opcionales (por ejemplo: 10G, 20 GB, 3 TIB). Si no se especifica ninguna unidad, el valor predeterminado es G. Las unidades de tamaño se pueden expresar como potencias de 2 (B, KiB, MIB, GIB, TIB) o de 10 (B, KB, MB, GB, TB). Las unidades abreviadas utilizan potencias de 2 (G = GIB, T = TIB, …).

Quitar un volumen

- Quite el volumen como cualquier otro volumen de Docker:

```
docker volume rm firstVolume
```

IMPORTANTE Cuando utilice la `solidfire-san` driver, el ejemplo anterior elimina y purga el volumen.

Realice los siguientes pasos para actualizar Trident para Docker.

Clonar un volumen

Al usar el `ontap-nas` , `ontap-san` , y `solidfire-san` Los controladores de almacenamiento, Trident puede clonar volúmenes. Al usar el `ontap-nas-flexgroup` o `ontap-nas-economy` controladores, la clonación no es compatible. La creación de un nuevo volumen a partir de un volumen existente dará como resultado la creación de una nueva instantánea.

- Examine el volumen para enumerar las instantáneas:

```
docker volume inspect <volume_name>
```

- Cree un volumen nuevo a partir de un volumen existente. Esto dará como resultado la creación de una nueva snapshot:

```
docker volume create -d <driver_name> --name <new_name> -o from
=<source_docker_volume>
```

- Cree un volumen nuevo a partir de una snapshot existente en un volumen. Esto no creará una nueva snapshot:

```
docker volume create -d <driver_name> --name <new_name> -o from
=<source_docker_volume> -o fromSnapshot=<source_snap_name>
```

Ejemplo

```
docker volume inspect firstVolume

[
  {
    "Driver": "ontap-nas",
    "Labels": null,
    "Mountpoint": "/var/lib/docker-volumes/ontap-
nas/netappdvp_firstVolume",
    "Name": "firstVolume",
    "Options": {},
    "Scope": "global",
    "Status": {
      "Snapshots": [
        {
          "Created": "2017-02-10T19:05:00Z",
          "Name": "hourly.2017-02-10_1505"
        }
      ]
    }
  }
]

docker volume create -d ontap-nas --name clonedVolume -o from=firstVolume
clonedVolume

docker volume rm clonedVolume
docker volume create -d ontap-nas --name volFromSnap -o from=firstVolume
-o fromSnapshot=hourly.2017-02-10_1505
volFromSnap

docker volume rm volFromSnap
```

Acceso a volúmenes creados externamente

Puede acceder a dispositivos de bloque creados externamente (o a sus clones) mediante contenedores utilizando Trident **Only** si no tienen particiones y si su sistema de archivos es compatible con Trident (por ejemplo: Un `ext4`-formateado `/dev/sdc1` no será accesible a través de Trident).

Opciones de volumen específicas del controlador

Cada controlador de almacenamiento tiene un conjunto diferente de opciones que se pueden especificar al crear un volumen para personalizar el resultado. Consulte a continuación las opciones que se aplican al sistema de almacenamiento configurado.

Usar estas opciones durante la operación de creación de volúmenes es simple. Proporcione la opción y el valor con `-o` Operador durante el funcionamiento de la CLI. Estos sustituyen cualquier valor equivalente al archivo de configuración JSON.

Opciones de volumen de ONTAP

Las opciones de creación de volúmenes para NFS, iSCSI y FC son las siguientes:

Opción	Descripción
`size`	El tamaño del volumen, de manera predeterminada es 1 GIB.
`spaceReserve`	Aprovisione el volumen de manera thin o thick, de manera predeterminada, es thin. Los valores válidos son `none` (con thin provisioning) y. `volume` (thick-provisioning).
`snapshotPolicy`	Esto establecerá la política de instantáneas en el valor deseado. El valor predeterminado es `none`, lo que significa que no se crearán instantáneas automáticamente para el volumen. A menos que el administrador de almacenamiento lo modifique, existe una política denominada «predeterminada» en todos los sistemas de ONTAP, que crea y retiene seis snapshots cada hora, dos diarios y dos semanales. Los datos conservados en una instantánea se pueden recuperar si se navega hasta `.snapshot` el directorio de cualquier directorio del volumen.

Opción	Descripción
`snapshotReserve`	Esto establecerá la reserva de instantáneas en el porcentaje deseado. El valor predeterminado es no, lo que significa que ONTAP seleccionará la reserva de copias Snapshot (generalmente 5 %) si se seleccionó una política de copias Snapshot o 0 % si la política de copias Snapshot no es ninguna. Es posible establecer el valor predeterminado de snapshotReserve en el archivo de configuración para todos los back-ends de ONTAP, y se puede usar como opción de creación de volúmenes para todos los back-ends de ONTAP excepto ontap-nas-Economy.
`splitOnClone`	Al clonar un volumen, ONTAP dividirá inmediatamente el clon de su principal. El valor predeterminado es `false`. Algunos casos de uso para el clonado de volúmenes se sirven mejor dividiendo el clon de su elemento principal inmediatamente después de la creación, ya que es poco probable que haya ninguna oportunidad para la eficiencia del almacenamiento. Por ejemplo, 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 volumen nuevo; el valor predeterminado es `false`. Para usar esta opción, debe tener una licencia para NVE y habilitarse en el clúster. Si NAE está habilitado en el back-end, cualquier volumen aprovisionado en Trident será habilitado NAE. Para obtener más información, consulte: "Cómo funciona Trident con NVE y NAE".
`tieringPolicy`	Establece la política de organización en niveles que se utilizará para el volumen. Esto decide si los datos se mueven al nivel de cloud cuando quedan inactivos (inactivos).

Las siguientes opciones adicionales son para NFS **sólo**:

Opción	Descripción
`unixPermissions`	Esto controla el conjunto de permisos para el propio volumen. De forma predeterminada, los permisos se establecerán en `---rwxr-xr-x, o en notación numérica 0755, y. `root` será el propietario. El texto o el formato numérico funcionará.
`snapshotDir`	Configuración de esta opción en `true` hará la `.snapshot` directorio visible para los clientes que acceden al volumen. El valor predeterminado es `false`, que significa la visibilidad del `.snapshot` el directorio está desactivado de forma predeterminada. Algunas imágenes, por ejemplo, la imagen oficial de MySQL, no funcionan como se esperaba cuando el `.snapshot` el directorio es visible.
`exportPolicy`	Establece la política de exportación que se utilizará para el volumen. El valor predeterminado es `default`.
`securityStyle`	Configura el estilo de seguridad que se usará para acceder al volumen. El valor predeterminado es `unix`. Los valores válidos son `unix` y.. `mixed`.

Las siguientes opciones adicionales son para iSCSI **sólo**:

Opción	Descripción
`fileSystemType`	Configura el sistema de archivos utilizado para formatear volúmenes iSCSI. El valor predeterminado es `ext4`. Los valores válidos son `ext3`, `ext4`, y. `xfs`.
`spaceAllocation`	Configuración de esta opción en `false` Desactivará la función de asignación de espacio de la LUN. El valor predeterminado es `true`, Es decir, ONTAP notifica al host cuando el volumen se ha quedado sin espacio y el LUN del volumen no puede aceptar escrituras. Esta opción también permite que ONTAP reclame espacio automáticamente cuando el host elimina los datos.

Ejemplos

Vea los ejemplos siguientes:

- Cree un volumen de 10 GiB:

```
docker volume create -d netapp --name demo -o size=10G -o
encryption=true
```

- Cree un volumen de 100 GiB con instantáneas:

```
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 mínimo del volumen es 20 MiB.

Si no se especifica la reserva de instantáneas y la política de instantáneas es `none`, Trident utilice 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 Snapshot y acepte la reserva de snapshots predeterminada de la ONTAP (generalmente 5 %):

```
docker volume create -d netapp --name my_vol --opt
snapshotPolicy=myPolicy
```

Opciones de volumen del software Element

Las opciones del software Element exponen las políticas de tamaño y calidad de servicio asociadas con el volumen. Cuando se crea el volumen, la política de calidad de servicio asociada con él se especifica mediante el `-o type=service_level` terminología

El primer paso para definir un nivel de servicio de calidad de servicio con el controlador de Element es crear al

menos un tipo y especificar las IOPS mínimas, máximas y de ráfaga asociadas con un nombre en el archivo de configuración.

Otras opciones de creación de volúmenes del software Element incluyen las siguientes:

Opción	Descripción
`size`	El tamaño del volumen, por defecto es 1 GiB o entrada de configuración… "defaults": {"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 está ejecutando Trident usando el método de plugin administrado recomendado (es decir, usando `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 host `/var/log/netappdvp` directorio. Para habilitar el registro de depuración, especifique `-debug` al ejecutar el plugin.

Sugerencias generales para la solución de problemas

- El problema más común en el que se ejecutan los nuevos usuarios es una configuración errónea que impide que el plugin se inicialice. Cuando esto sucede, es probable que vea un mensaje como este cuando intente instalar o activar el plugin:

  ```
  Error response from daemon: dial unix /run/docker/plugins/<id>/netapp.sock:
  connect: no such file or directory
  ```

  Esto significa que el plugin no se pudo iniciar. Afortunadamente, el complemento se ha creado con una completa capacidad de registro que le ayudará a diagnosticar la mayoría de los problemas que es probable que se encuentren.

- Si hay problemas con el montaje de un PV en un contenedor, asegúrese de que `rpcbind` está instalado y en ejecución. Utilice el administrador de paquetes necesario para el sistema operativo del host y compruebe si `rpcbind` está en ejecución. Puede comprobar el estado del servicio rpcbind ejecutando un `systemctl status rpcbind` o su equivalente.

Gestione múltiples instancias de Trident

Se necesitan varias instancias de Trident cuando se desean que varias configuraciones de almacenamiento estén disponibles de forma simultánea. La clave para varias instancias es darles nombres diferentes mediante el `--alias` opción con el plugin en contenedor, o. `--volume-driver` Opción al crear una instancia de Trident en el host.

Pasos para el complemento gestionado de Docker (versión 1.13/17.03 o posterior)

1. Inicie la primera instancia que especifique un alias y un archivo de configuración.

   ```
   docker plugin install --grant-all-permissions --alias silver
   netapp/trident-plugin:21.07 config=silver.json
   ```

2. Inicie la segunda instancia, especificando un alias y un archivo de configuración distintos.

```
docker plugin install --grant-all-permissions --alias gold
netapp/trident-plugin:21.07 config=gold.json
```

3. Cree volúmenes que especifiquen el alias como el nombre del controlador.

   Por ejemplo, para el volumen Gold:

   ```
   docker volume create -d gold --name ntapGold
   ```

   Por ejemplo, en el caso del volumen Silver:

   ```
   docker volume create -d silver --name ntapSilver
   ```

Pasos para tradicional (versión 1.12 o anterior)

1. Inicie el plugin con una configuración NFS mediante un ID de controlador personalizado:

   ```
   sudo trident --volume-driver=netapp-nas --config=/path/to/config
   -nfs.json
   ```

2. Inicie el plugin con una configuración iSCSI mediante un ID de controlador personalizado:

   ```
   sudo trident --volume-driver=netapp-san --config=/path/to/config
   -iscsi.json
   ```

3. Aprovisione volúmenes Docker para cada instancia de controlador:

   Por ejemplo, para NFS:

   ```
   docker volume create -d netapp-nas --name my_nfs_vol
   ```

   Por ejemplo, para iSCSI:

   ```
   docker volume create -d netapp-san --name my_iscsi_vol
   ```

Opciones de configuración de almacenamiento

Consulte las opciones de configuración disponibles para las configuraciones de Trident.

Opciones de configuración global

Estas opciones de configuración se aplican a todas las configuraciones de Trident, independientemente de la plataforma de almacenamiento en uso.

Opción	Descripción	Ejemplo
`version`	Número de versión del archivo de configuración	`1`
`storageDriverName`	Nombre del controlador de almacenamiento	`ontap-nas`, `ontap-san`, `ontap-nas-economy`, `ontap-nas-flexgroup`, `solidfire-san`
`storagePrefix`	Prefijo opcional para los nombres de volúmenes. Valor predeterminado: `netappdvp_`.	`staging_`
`limitVolumeSize`	Restricción opcional de los tamaños de volumen. Valor por defecto: '' (no forzado)	`10g`

CONSEJO

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. NetApp recomienda usar un tenantID específico para la asignación de volúmenes de Docker o el uso de los datos del atributo que se rellenan con la versión de Docker, la información del controlador y el nombre sin procesar de Docker en casos en los que se haya utilizado cualquier munging del nombre.

Las opciones predeterminadas están disponibles para evitar tener que especificarlas en cada volumen que cree. La `size` la opción está disponible para todos los tipos de controladoras. Consulte la sección ONTAP Configuration para obtener un ejemplo de cómo establecer el tamaño de volumen predeterminado.

Opción	Descripción	Ejemplo
`size`	Tamaño predeterminado opcional para los nuevos volúmenes. Valor predeterminado: `1G`	`10G`

Configuración de ONTAP

Además de los valores de configuración global anteriores, al utilizar ONTAP, están disponibles las siguientes opciones de nivel superior.

Opción	Descripción	Ejemplo
`managementLIF`	Dirección IP de LIF de gestión de ONTAP. Es posible especificar un nombre de dominio completo (FQDN).	`10.0.0.1`
`dataLIF`	Dirección IP de LIF de protocolo. **Controladores NAS de ONTAP**: NetApp recomienda especificar `dataLIF`. Si no se proporciona, Trident recupera las LIF de datos de la SVM. Puede especificar un nombre de dominio completo (FQDN) que se utilice para las operaciones de montaje de NFS, lo que permite crear un DNS por turnos para equilibrar la carga en varias LIF de datos. **Controladores SAN ONTAP**: No especifique para iSCSI o FC. Trident utiliza "Asignación de LUN selectiva de ONTAP" para detectar las LIF iSCSI o FC 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 a la SVM se usan para aprovisionar un volumen FlexGroup.	`aggr1`

Opción	Descripción	Ejemplo
`limitAggregateUsage`	Opcional, fallo en el aprovisionamiento si el uso supera este porcentaje	`75%`
`nfsMountOptions`	Control detallado de las opciones de montaje de NFS; se define de forma predeterminada en «-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`	Trident crea y gestiona por nodo `igroups` como `netappdvp`. Este valor no se puede cambiar ni omitir. **Disponible sólo para `ontap-san` conductor**.	`netappdvp`
`limitVolumeSize`	Tamaño máximo 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 `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` para iSCSI, `nvme` para NVMe/TCP o `fcp` para SCSI over Fibre Channel (FC).	`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` (con thin provisioning) o `volume` (grueso)	`none`
`snapshotPolicy`	La política de Snapshot que se va a utilizar, el valor predeterminado es `none`	`none`
`snapshotReserve`	El porcentaje de reserva de Snapshot, el valor predeterminado es « » para aceptar el valor predeterminado de ONTAP	`10`
`splitOnClone`	Divida un clon de su elemento principal tras su creación (el valor predeterminado es `false`	`false`
`encryption`	Permite el cifrado de volúmenes de NetApp (NVE) en el volumen nuevo; los valores predeterminados son `false`. Para usar esta opción, debe tener una licencia para NVE y habilitarse en el clúster. Si NAE está habilitado en el back-end, cualquier volumen aprovisionado en Trident será habilitado NAE. Para obtener más información, consulte: "Cómo funciona 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.	Activa o desactiva el acceso al `.snapshot` directorio. Valores válidos: `true`, `false`. El comportamiento predeterminado puede variar según la versión de ONTAP y Trident. Configúralo explícitamente para asegurar un comportamiento consistente.
`exportPolicy`	La opción de NAS para la política de exportación de NFS que va a utilizar, de forma predeterminada a. `default`	`default`

Opción	Descripción	Ejemplo
securityStyle	Opción NAS para acceder al volumen NFS aprovisionado. Compatibilidad con NFS mixed y.. unix estilos de seguridad. El valor predeterminado es unix.	unix
fileSystemType	Opción SAN para seleccionar el tipo de sistema de archivos, de forma predeterminada a. ext4	xfs
tieringPolicy	Política de organización en niveles que se debe utilizar, el valor por defecto es none.	none
skipRecoveryQueue	Durante la eliminación de un volumen, omita la cola de recuperación en el almacenamiento y elimine el volumen inmediatamente.	"

Opciones de escala

Los ontap-nas controladores y ontap-san crean un ONTAP FlexVol para cada volumen Docker. ONTAP admite hasta 1000 FlexVols por nodo del clúster con un máximo de 12.000 volúmenes FlexVol Si los requisitos de volumen Docker se ajustan a esta limitación, ontap-nas el controlador es la solución NAS preferida debido a 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 pueden alojar los límites de FlexVol, seleccione la ontap-nas-economy o la ontap-san-economy controlador.

```
`ontap-nas-economy`El controlador crea volúmenes de Docker como qtrees de ONTAP dentro de un pool de volúmenes de 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.
```

NOTA `ontap-nas-economy`El controlador no es compatible actualmente con Docker Swarm, porque Docker 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 de 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 utiliza todos los agregados asignados a una SVM cuando se aprovisiona un volumen de FlexGroup. La compatibilidad con FlexGroup en Trident también tiene las siguientes consideraciones:

- Requiere ONTAP versión 9.2 o posterior.
- En el momento en el que se ha redactado este documento, FlexGroups solo admite NFS v3.
- Se recomienda habilitar los identificadores de NFSv3 de 64 bits para la SVM.
- El tamaño mínimo recomendado para el miembro/volumen de FlexGroup es 100 GiB.
- No se admite la clonado para los volúmenes de FlexGroup.

Para obtener información sobre FlexGroups y las cargas de trabajo que son adecuadas para FlexGroups, consulta la "Prácticas recomendadas y guía de implementación de NetApp FlexGroup para volúmenes".

Para obtener características avanzadas y a 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`.

Rol de ONTAP personalizado para Trident

Puede crear un rol de clúster de ONTAP con un Privileges mínimo de modo que no tenga que utilizar el rol de administrador de ONTAP para realizar operaciones en Trident. Cuando incluye el nombre de usuario en una configuración de back-end de Trident, Trident utiliza el rol de clúster de ONTAP que creó para realizar las operaciones.

Consulte "Generador de roles personalizados de Trident"para obtener más información sobre la creación de roles personalizados de Trident.

Con la CLI de ONTAP

1. Cree un rol nuevo mediante el siguiente comando:

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

2. Cree un nombre de usuario para el usuario de Trident:

```
security login create -username <user_name\> -application ontapi
-authmethod password -role <name_of_role_in_step_1\> -vserver <svm_name\>
-comment "user_description"
security login create -username <user_name\> -application http -authmethod
password -role <name_of_role_in_step_1\> -vserver <svm_name\> -comment
"user_description"
```

3. Asignar el rol al usuario:

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

Mediante System Manager

Realice los pasos siguientes en ONTAP System Manager:

1. **Crear un rol personalizado**:
   a. Para crear un rol personalizado a nivel de clúster, seleccione **Cluster > Settings**.

      (O) Para crear un rol personalizado en el nivel de SVM, seleccione **Almacenamiento > Storage VMs > `required SVM`> Settings > Users and Roles**.

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

Consulte las siguientes páginas si quiere más información:

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

Archivos de configuración de ONTAP de ejemplo

Ejemplo de NFS para <code>ontap-nas</code> el controlador

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "defaults": {
    "size": "10G",
    "spaceReserve": "none",
    "exportPolicy": "default"
  }
}
```

Ejemplo de NFS para <code>ontap-nas-flexgroup</code> el controlador

```
{
  "version": 1,
  "storageDriverName": "ontap-nas-flexgroup",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "defaults": {
    "size": "100G",
    "spaceReserve": "none",
    "exportPolicy": "default"
  }
}
```

Ejemplo de NFS para <code>ontap-nas-economy</code> el controlador

```
{
  "version": 1,
  "storageDriverName": "ontap-nas-economy",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1"
}
```

Ejemplo de iSCSI para <code>ontap-san</code> el controlador

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.3",
  "svm": "svm_iscsi",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "igroupName": "netappdvp"
}
```

Ejemplo de NFS para <code>ontap-san-economy</code> el controlador

```
{
  "version": 1,
  "storageDriverName": "ontap-san-economy",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.3",
  "svm": "svm_iscsi_eco",
  "username": "vsadmin",
  "password": "password",
  "aggregate": "aggr1",
  "igroupName": "netappdvp"
}
```

Ejemplo de NVMe/TCP para <code>ontap-san</code> controlador

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

Ejemplo de SCSI sobre FC para el controlador <code> ONTAP-san</code>

```
{
  "version": 1,
  "backendName": "ontap-san-backend",
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "sanType": "fcp",
  "svm": "trident_svm",
  "username": "vsadmin",
  "password": "password",
  "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`

Opción	Descripción	Ejemplo
`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-`

La `solidfire-san` El controlador no es compatible con Docker Swarm.

Ejemplo del archivo de configuración del software Element

```
{
  "version": 1,
  "storageDriverName": "solidfire-san",
  "Endpoint": "https://admin:admin@192.168.160.3/json-rpc/8.0",
  "SVIP": "10.0.0.7:3260",
  "TenantName": "docker",
  "InitiatorIFace": "default",
  "Types": [
    {
      "Type": "Bronze",
      "Qos": {
        "minIOPS": 1000,
        "maxIOPS": 2000,
        "burstIOPS": 4000
      }
    },
    {
      "Type": "Silver",
      "Qos": {
        "minIOPS": 4000,
        "maxIOPS": 6000,
        "burstIOPS": 8000
      }
    },
    {
      "Type": "Gold",
      "Qos": {
        "minIOPS": 6000,
        "maxIOPS": 8000,
        "burstIOPS": 10000
      }
    }
  ]
}
```

Problemas y limitaciones conocidos

Busque información sobre problemas conocidos y limitaciones cuando use Trident con Docker.

Si se actualiza el complemento Trident Docker Volume Plugin a 20.10 y versiones posteriores, se produce un error de actualización sin dicho archivo o directorio.

Solución alternativa

1. Desactivar el plugin.

```
docker plugin disable -f netapp:latest
```

2. Quitar el plugin.

```
docker plugin rm -f netapp:latest
```

3. Vuelva a instalar el plugin proporcionando el extra `config` parámetro.

```
docker plugin install netapp/trident-plugin:20.10 --alias netapp --grant
-all-permissions config=config.json
```

Los nombres de volumen deben tener una longitud mínima de 2 caracteres.

NOTA 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 ciertos comportamientos que impiden que Trident ofrezca compatibilidad con cada combinación de almacenamiento y controladores.

- 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 plugins de volúmenes (incluido Trident) deben ejecutarse de forma independiente en cada nodo de un clúster de 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 pilotos están sujetos a problemas como condiciones de carrera que pueden resultar en la creación de un gran número de volúmenes para una sola solicitud sin un claro “ganador”; por ejemplo, Element tiene una característica que permite que los volúmenes tengan el mismo nombre pero diferentes ID.

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 al implementar 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. Trident proporciona recursos PV a todos los espacios de nombres del clúster de Kubernetes y, en consecuencia, aprovecha una cuenta de servicio que ha elevado el número de Privileges.

Además, el acceso al pod de Trident puede permitir a un usuario acceder a las credenciales del sistema de almacenamiento y a otra información confidencial. Es importante asegurarse de que los usuarios de aplicaciones y aplicaciones de gestión no tengan la capacidad de acceder a las definiciones de objetos de Trident o a los pods mismos.

Utilice cuotas y límites de rango para controlar el consumo de almacenamiento

Kubernetes cuenta con dos funciones que, al combinarse, ofrecen un potente mecanismo que limita el consumo de recursos que consumen las aplicaciones. La "mecanismo de cuotas de almacenamiento" permite al administrador implementar límites de consumo globales, específicos para las clases de almacenamiento, de capacidad y de recuento de objetos en función del espacio de nombres. Además, utilizando un "límite de rango" Garantiza que las solicitudes de PVC se encuentren dentro de un valor mínimo y máximo antes de reenviar la solicitud al aprovisionador.

Estos valores se definen por espacio de nombres, lo que significa que cada espacio de nombres debe tener valores definidos que se ajustan a los requisitos de sus recursos. Consulte aquí para obtener más información acerca de "cómo aprovechar las cuotas".

Configuración del almacenamiento

Cada plataforma de almacenamiento de la cartera de NetApp tiene unas funciones únicas que benefician a las aplicaciones, en contenedores o no.

Descripción general de la plataforma

Trident funciona con ONTAP y Element. No existe una plataforma que se adapte mejor a todas las aplicaciones y escenarios que otra, sin embargo, las necesidades de la aplicación y el equipo que administra el dispositivo deben tenerse en cuenta al elegir una plataforma.

Debe seguir las prácticas recomendadas de base para el sistema operativo del host con el protocolo que está aprovechando. Opcionalmente, es posible que desee considerar la incorporación de prácticas recomendadas para las aplicaciones, cuando esté disponible, con configuración de back-end, clase de almacenamiento y RVP para optimizar el almacenamiento para aplicaciones específicas.

Prácticas recomendadas para ONTAP y Cloud Volumes ONTAP

Conozca las prácticas recomendadas para configurar ONTAP y Cloud Volumes ONTAP para Trident.

Las siguientes recomendaciones son directrices para configurar ONTAP para cargas de trabajo en contenedores, que consumen volúmenes aprovisionados de forma dinámica por Trident. Cada uno de ellos debe considerarse y evaluarse según la idoneidad de su entorno.

Utilice SVM dedicadas a Trident

Las máquinas virtuales de almacenamiento (SVM) proporcionan separación de tareas administrativas y de aislamiento entre clientes en un sistema ONTAP. Dedicar una SVM a las aplicaciones permite delegar privilegios y aplicar prácticas recomendadas para limitar el consumo de recursos.

Existen varias opciones disponibles para la gestión de la SVM:

- Proporcione la interfaz de gestión del clúster en la configuración del back-end, junto con las credenciales adecuadas, y especifique el nombre de la SVM.
- Cree una interfaz de gestión dedicada para la SVM mediante ONTAP System Manager o la CLI.
- Comparta la función de gestión con una interfaz de datos NFS.

En cada caso, la interfaz debe estar en DNS, y se debe usar el nombre DNS al configurar Trident. Esto permite facilitar algunas situaciones de recuperación ante desastres, por ejemplo, SVM-DR sin retención de identidad de red.

No tiene ninguna preferencia entre tener una LIF de gestión dedicada o compartida para la SVM, sin embargo, debe asegurarse de que las políticas de seguridad de red se alineen con el enfoque que elija. Sin embargo, debería accederse a la LIF de gestión a través de DNS para facilitar la máxima flexibilidad que debería tener "SVM-DR" Se podrá usar en combinación con Trident.

Limite el número máximo de volúmenes

Los sistemas de almacenamiento de ONTAP tienen un número máximo de volúmenes, que varía en función de la versión del software y la plataforma de hardware. Consulte "Hardware Universe de NetApp" Para su plataforma y versión de ONTAP específica, determine los límites exactos. Cuando se agota el número de volúmenes, las operaciones de aprovisionamiento fallan no solo para Trident, sino para todas las solicitudes de almacenamiento.

De Trident `ontap-nas` y.. `ontap-san` Los controladores aprovisionan un FlexVolume para cada volumen persistente (PV) de Kubernetes que se crea. La `ontap-nas-economy` El controlador crea aproximadamente un FlexVolume por cada 200 VP (configurable entre 50 y 300). La `ontap-san-economy` El controlador crea aproximadamente un FlexVolume por cada 100 VP (configurable entre 50 y 200). Para evitar que Trident consuma todos los volúmenes disponibles en el sistema de almacenamiento, debe establecer un límite en la SVM. Puede hacerlo desde la línea de comandos:

```
vserver modify -vserver <svm_name> -max-volumes <num_of_volumes>
```

Valor para `max-volumes` varía en función de varios criterios específicos de su entorno:

- El número de volúmenes existentes en el clúster de ONTAP
- El número de volúmenes que espera aprovisionar fuera de Trident para otras aplicaciones
- El número de volúmenes persistentes que tienen previsto consumir las aplicaciones de Kubernetes

La `max-volumes` El valor es el total de volúmenes aprovisionados en todos los nodos del clúster de ONTAP, no en un nodo ONTAP individual. Como resultado, es posible que encuentre algunas condiciones en las que

un nodo de un clúster de ONTAP pueda tener muchos más o menos volúmenes aprovisionados de Trident que otro nodo.

Por ejemplo, un clúster ONTAP de dos nodos puede alojar un máximo de 2000 volúmenes FlexVol. 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.**

Clonar un volumen

NetApp Trident admite la clonación de volúmenes al usar `ontap-nas` , `ontap-san` , y `solidfire-san` Controladores de almacenamiento. Al usar el `ontap-nas-flexgroup` o `ontap-nas-economy` controladores, la clonación no es compatible. La creación de un nuevo volumen a partir de un volumen existente dará como resultado la creación de una nueva instantánea.

ADVERTENCIA	Evite clonar una PVC asociada a una clase de almacenamiento diferente. Realice operaciones de clonación dentro de la misma clase de almacenamiento para garantizar la compatibilidad y evitar comportamientos inesperados.

Limite el tamaño máximo de los volúmenes que ha creado Trident

Para configurar el tamaño máximo de los volúmenes que Trident puede crear `limitVolumeSize` parámetro en la `backend.json` definición.

Además de controlar el tamaño del volumen en la cabina de almacenamiento, también se deben aprovechar las capacidades de Kubernetes.

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. Con el `useCHAP` Parámetro en la configuración de back-end, Trident autentica las conexiones iSCSI para los 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" O un contenedor fuera de control que no afecta a las cargas de trabajo fuera de la SVM de Trident.

Puede crear una política de calidad de servicio para la SVM en unos pasos. Consulte la documentación de su versión de ONTAP para obtener la información más precisa. El ejemplo siguiente crea una política de calidad de servicio que limita el total de IOPS disponibles para la SVM a 5000.

```
# create the policy group for the SVM
qos policy-group create -policy-group <policy_name> -vserver <svm_name>
-max-throughput 5000iops

# assign the policy group to the SVM, note this will not work
# if volumes or files in the SVM have existing QoS policies
vserver modify -vserver <svm_name> -qos-policy-group <policy_name>
```

Además, si su versión de ONTAP admite esta función, puede considerar el uso de una calidad de servicio mínima para garantizar un volumen del rendimiento para cargas de trabajo en contenedores. La calidad de servicio adaptativa no es compatible con una política de nivel de SVM.

El número de IOPS dedicado a las cargas de trabajo de los contenedores depende de muchos aspectos. Entre otras cosas, estas incluyen:

- Otras cargas de trabajo que utilizan la cabina de almacenamiento. Si hay otras cargas de trabajo, no relacionadas con la puesta en marcha de Kubernetes, y que utilizan los recursos de almacenamiento, se debe tener cuidado para garantizar que esas cargas de trabajo no se vean afectadas de forma accidental.
- Cargas de trabajo esperadas que se ejecutan en contenedores. Si las cargas de trabajo que tienen requisitos de IOPS elevados se ejecutan en contenedores, una política de calidad de servicio baja resulta en una mala experiencia.

Es importante recordar que una política de calidad de servicio asignada en el nivel de la SVM da como resultado que todos los volúmenes aprovisionados a la SVM compartan el mismo pool de IOPS. Si una, o una cifra pequeña, de las aplicaciones con contenedores tienen un requisito elevado de IOPS, podría convertirse en un problema para las otras cargas de trabajo con contenedores. Si este es el caso, puede que se desee considerar utilizar la automatización externa para asignar políticas de calidad de servicio por volumen.

IMPORTANTE	Debe asignar el grupo de políticas QoS al SVM **only** si la versión de ONTAP es anterior a 9.8.

Cree grupos de políticas de calidad de servicio para Trident

La calidad de servicio garantiza que el rendimiento de las cargas de trabajo críticas no se vea degradado por cargas de trabajo de la competencia. Los grupos de políticas de calidad de servicio de ONTAP proporcionan opciones de calidad de servicio para volúmenes y permiten a los usuarios definir el techo de rendimiento para una o más cargas de trabajo. Para obtener más información sobre la calidad de servicio, consulte "Rendimiento garantizado con QoS".
Puede especificar grupos de políticas de calidad de servicio en el back-end o en un pool de almacenamiento y se aplican a cada volumen creado en ese pool o back-end.

ONTAP tiene dos tipos de grupos de políticas de calidad de servicio: Tradicionales y adaptativos. Los grupos de políticas tradicionales proporcionan un rendimiento máximo (o mínimo, en versiones posteriores) plano en IOPS. La calidad de servicio adaptativa escala automáticamente el rendimiento al tamaño de la carga de trabajo y mantiene la ratio de IOPS en TB|GB a medida que el tamaño de la carga de trabajo cambia. Esto supone una ventaja significativa cuando se gestionan cientos o miles de cargas de trabajo en una puesta en marcha de gran tamaño.

Tenga en cuenta lo siguiente al crear grupos de políticas de calidad de servicio:

- Debe configurar la `qosPolicy` introduzca la `defaults` bloque de la configuración del back-end. Consulte

el siguiente ejemplo de configuración del back-end:

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 0.0.0.0
dataLIF: 0.0.0.0
svm: svm0
username: user
password: pass
defaults:
  qosPolicy: standard-pg
storage:
  - labels:
      performance: extreme
    defaults:
      adaptiveQosPolicy: extremely-adaptive-pg
  - labels:
      performance: premium
    defaults:
      qosPolicy: premium-pg
```

- Debe aplicar los grupos de políticas por volumen, de modo que cada volumen obtenga el rendimiento entero según lo especifique el grupo de políticas. No se admiten los grupos de políticas compartidas.

Para obtener más información acerca de los grupos de políticas de QoS, consulte "Referencia de comandos del ONTAP" .

Limite el acceso a recursos de almacenamiento a los miembros del clúster de Kubernetes

La limitación del acceso a los volúmenes de NFS, LUN de iSCSI y LUN de FC que ha creado Trident es un componente crucial de la política de seguridad de su puesta en marcha de Kubernetes. Si lo hace, se evita que los hosts que no forman parte del clúster de Kubernetes accedan a los volúmenes y que potencialmente modifiquen los datos de forma inesperada.

Es importante comprender que los espacios de nombres son el límite lógico de los recursos en Kubernetes. Se supone que los recursos del mismo espacio de nombres se pueden compartir; sin embargo, es importante destacar que no existe ninguna funcionalidad entre espacios de nombres. Esto significa que aunque los VP sean objetos globales, cuando están enlazados a una RVP solo pueden acceder a ellos mediante POD que están en el mismo espacio de nombres. **Es fundamental asegurarse de que los espacios de nombres se utilizan para proporcionar la separación cuando sea apropiado.**

La preocupación principal de la mayoría de las organizaciones con respecto a la seguridad de los datos en un contexto de Kubernetes es que un proceso en un contenedor puede acceder al almacenamiento montado en el host, pero que no está destinado al contenedor. "Espacios de nombres" están diseñados para evitar este tipo de compromiso. Sin embargo, hay una excepción: Contenedores privilegiados.

Un contenedor con privilegios es uno que se ejecuta con mucho más permisos de nivel de host de lo normal. No se deniegan de forma predeterminada, por lo que debe desactivar la funcionalidad utilizando "directivas de seguridad de pod".

Para los volúmenes en los que se desea obtener acceso tanto a los hosts de Kubernetes como a los externos, el almacenamiento se debe gestionar de forma tradicional, con el VP introducido por el administrador, y no gestionado por Trident. Esto garantiza que el volumen de almacenamiento se destruya solo cuando tanto los hosts de Kubernetes como los externos se desconectaron y ya no utilizan el volumen. Además, se puede aplicar una política de exportación personalizada, lo que permite el acceso desde los nodos del clúster de Kubernetes y los servidores objetivo fuera del clúster de Kubernetes.

Para las implementaciones que tienen nodos de infraestructura dedicados (por ejemplo, OpenShift) u otros nodos que no pueden programar aplicaciones de usuario, se deben utilizar directivas de exportación independientes para limitar aún más el acceso a los recursos de almacenamiento. Esto incluye la creación de una directiva de exportación para los servicios que se implementan en dichos nodos de infraestructura (por ejemplo, los servicios de registro y métricas de OpenShift) y aplicaciones estándar que se implementan en nodos que no son de infraestructura.

Usar una política de exportación dedicada

Debe asegurarse de que existe una política de exportación para cada back-end que solo permita el acceso a los nodos presentes en el clúster de Kubernetes. Trident puede crear y gestionar automáticamente políticas de exportación. De esta forma, Trident limita el acceso a los volúmenes que aprovisiona a los nodos en el clúster de Kubernetes y simplifica la adición o la eliminación de nodos.

También puede crear una política de exportación manualmente y rellenarla con una o varias reglas de exportación que procesarán cada solicitud de acceso a nodo:

- Utilice la `vserver export-policy create` Comando de la interfaz de línea de comandos de ONTAP para crear la política de exportación.
- Añada reglas a la política de exportación mediante la `vserver export-policy rule create` Comando de la CLI de ONTAP.

Si ejecuta estos comandos, puede restringir el acceso de los nodos de Kubernetes a los datos.

Desactivar `showmount` Para la SVM de la aplicación

```
`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 el y recibir una lista de los montajes disponibles, incluidos los a los que no tiene acceso. Aunque esto, por sí solo, no supone un compromiso con la seguridad, proporciona información innecesaria, potencialmente que ayuda a un usuario no autorizado a conectarse con una exportación NFS.
```

Debe desactivar `showmount` Con el comando CLI de ONTAP a nivel de la SVM:

```
vserver nfs modify -vserver <svm_name> -showmount disabled
```

Mejores prácticas para SolidFire

Conozca las prácticas recomendadas para configurar el almacenamiento de SolidFire para Trident.

Crear cuenta de SolidFire

Cada cuenta SolidFire representa un propietario de volumen único y recibe su propio conjunto de credenciales de protocolo de autenticación por desafío mutuo (CHAP). Es posible acceder a los volúmenes asignados a una cuenta mediante el nombre de cuenta y las credenciales CHAP relativas o un grupo de acceso de volúmenes. Una cuenta puede tener hasta 2000 volúmenes asignados, pero un volumen solo puede pertenecer a una cuenta.

Cree una política de calidad de servicio

Utilice las políticas de calidad de servicio de SolidFire si desea crear y guardar un ajuste de calidad de servicio estandarizado que se puede aplicar a muchos volúmenes.

Puede establecer parámetros de calidad de servicio por cada volumen. El rendimiento de cada volumen se puede garantizar mediante el establecimiento de tres parámetros configurables que definen la calidad de servicio: Min IOPS, Max IOPS y Burst IOPS.

Aquí pueden ver los valores mínimos, máximos y de ráfaga de IOPS en relación con el tamaño de bloque de 4 KB.

Parámetro IOPS	Definición	Espacio valor	Valor predeterminado	Capacidad Valor (4 KB)
IOPS mín	El nivel garantizado de rendimiento de un volumen.	50	50	15000
Tasa máx. De IOPS	El rendimiento no superará este límite.	50	15000	200,000
IOPS de ráfaga	IOPS máximo permitido en un escenario de ráfaga breve.	50	15000	200,000

NOTA 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 aumenta el tamaño de bloque, el sistema aumenta el ancho de banda hasta el nivel necesario para procesar los tamaños de bloque más grandes. A medida que aumenta el ancho de banda, disminuye el número de IOPS que el sistema puede alcanzar. Consulta "Calidad de servicio de SolidFire" para más información sobre QoS 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.

NOTA	Trident con el aprovisionador CSI mejorado admite el uso de la autenticación CHAP. Los VAG sólo deben utilizarse en el modo de funcionamiento tradicional no CSI.

La autenticación CHAP (verificación de que el iniciador es el usuario de volumen objetivo) solo se admite con control de acceso basado en la cuenta. Si se utiliza CHAP para la autenticación, hay dos opciones disponibles: CHAP unidireccional y CHAP bidireccional. CHAP unidireccional autentica el acceso al volumen mediante el nombre de cuenta de SolidFire y el secreto de iniciador. La opción CHAP bidireccional proporciona la manera más segura de autenticar el volumen, ya que el volumen autentica el host a través del nombre de cuenta y el secreto de iniciador, y luego el host autentica el volumen por medio del nombre de cuenta y el secreto de destino.

Sin embargo, si no se puede habilitar CHAP y se requieren los VAG, cree el grupo de acceso y añada los iniciadores de host y los volúmenes al grupo de acceso. Cada IQN que se añade a un grupo de acceso puede acceder a cada volumen del grupo con o sin autenticación CHAP. Si el iniciador de iSCSI está configurado para utilizar la autenticación CHAP, se utiliza el control de acceso basado en cuentas. Si el iniciador iSCSI no está configurado para utilizar la autenticación CHAP, se utiliza el control de acceso del grupo de acceso de volúmenes.

¿Dónde encontrar más información?

A continuación se enumeran algunas de las prácticas recomendadas. Busque en el "Biblioteca de NetApp" para las versiones más actuales.

ONTAP

- "Prácticas recomendadas y guía de implementación de NFS"
- "Administración de SAN" (Para iSCSI)
- "Configuración exprés de iSCSI para RHEL"

Software Element

- "Configuración de SolidFire para Linux"

NetApp HCI

- "Requisitos previos de la implementación de NetApp HCI"
- "Acceda al motor de implementación de NetApp"

Información sobre las prácticas recomendadas de la aplicación

- "Prácticas recomendadas para MySQL en ONTAP"
- "Prácticas recomendadas para MySQL en SolidFire"
- "NetApp SolidFire y Cassandra"
- "Prácticas recomendadas de Oracle en SolidFire"
- "Prácticas recomendadas de PostgreSQL en SolidFire"

No todas las aplicaciones tienen directrices específicas, es importante trabajar con su equipo de NetApp y utilizar el "Biblioteca de NetApp" para encontrar la documentación más actualizada.

Integre Trident

Para integrar Trident, los siguientes elementos de diseño y arquitectura requieren integración: Selección y puesta en marcha de controladores, diseño de la clase de almacenamiento, diseño de pools virtuales, reclamación de volumen persistente (RVP) impactos en el aprovisionamiento de almacenamiento, operaciones de volúmenes y puesta en marcha de servicios OpenShift mediante Trident.

Selección y despliegue del conductor

Seleccione e implemente un controlador de back-end para el sistema de almacenamiento.

Controladores de entorno de administración ONTAP

Los controladores de entorno de administración de ONTAP se diferencian por el protocolo utilizado y cómo se aprovisionan los volúmenes en el sistema de almacenamiento. Por lo tanto, tenga cuidado al decidir qué controlador implementar.

En un nivel superior, si la aplicación cuenta con componentes que necesitan almacenamiento compartido (varios POD que acceden al mismo PVC), los controladores basados en NAS serán la opción predeterminada, mientras que los controladores iSCSI basados en bloques satisfacen las necesidades de almacenamiento no compartido. Elija el protocolo según los requisitos de la aplicación y el nivel de comodidad de los equipos de almacenamiento e infraestructura. Por lo general, existe poca diferencia entre ellas para la mayoría de las aplicaciones, con tanta frecuencia la decisión se basa en si se necesita o no almacenamiento compartido (donde más de un pod necesitará acceso simultáneo).

Los controladores de entorno de administración de ONTAP disponibles son:

- `ontap-nas`: Cada PV aprovisionado es un FlexVolume ONTAP completo.
- `ontap-nas-economy`: Cada PV aprovisionado es un qtree, con un número configurable de qtrees por FlexVolume (el valor predeterminado es 200).
- `ontap-nas-flexgroup`: Cada VP aprovisionado como ONTAP FlexGroup completo y se utilizan todos los agregados asignados a una SVM.
- `ontap-san`: Cada PV aprovisionado es una LUN dentro de su propio FlexVolume.
- `ontap-san-economy`: Cada VP aprovisionado es una LUN, con un número configurable de LUN por FlexVolume (el valor predeterminado es 100).

La elección entre los tres controladores NAS tiene algunas ramificaciones a las funciones, que están disponibles para la aplicación.

Tenga en cuenta que, en las siguientes tablas, no todas las capacidades se exponen a través de Trident. El administrador de almacenamiento debe aplicar algunas después del aprovisionamiento si se desea disponer de esta funcionalidad. Las notas al pie de la superíndice distinguen la funcionalidad por característica y controlador.

Unidades NAS de ONTAP	Snapshot	Clones	Políticas de exportación dinámicas	Conexión múltiple	Calidad de servicio	Cambie el tamaño	Replicación
`ontap-nas`	Sí	Sí	Yespie de página:5[]	Sí	Yespie de página:1[]	Sí	Yespie de página:1[]
`ontap-nas-economy`	Nota de la oferta:3[]	Nota de la oferta:3[]	Yespie de página:5[]	Sí	Nota de la oferta:3[]	Sí	Nota de la oferta:3[]
`ontap-nas-flexgroup`	Yespie de página:1[]	NO	Yespie de página:5[]	Sí	Yespie de página:1[]	Sí	Yespie de página:1[]

Trident ofrece 2 controladores SAN para ONTAP, cuyas funcionalidades se muestran a continuación.

Unidades SAN de ONTAP	Snapshot	Clones	Conexión múltiple	CHAP bidireccional	Calidad de servicio	Cambie el tamaño	Replicación
`ontap-san`	Sí	Sí	Yespie de página:4[]	Sí	Yespie de página:1[]	Sí	Yespie de página:1[]
`ontap-san-economy`	Sí	Sí	Yespie de página:4[]	Sí	Nota de la oferta:3[]	Sí	Nota de la oferta:3[]

> Nota al pie de las tablas anteriores: Yes [1]: No administrado por Trident Yes [2]: Administrado por Trident, pero no por PV NO [3]: No gestionado por Trident y no por PV granular Yes [4]: Admitido para volúmenes de bloque sin procesar Yes [5]: Admitido por Trident

Las funciones que no son granulares en los VP se aplican a todo el FlexVolume y todos los VP (es decir, qtrees o LUN de FlexVols compartidos) compartirán un programa común.

Como podemos ver en las tablas anteriores, gran parte de la funcionalidad entre `ontap-nas` y.. `ontap-nas-economy` es lo mismo. Sin embargo, porque la `ontap-nas-economy` Esta unidad limita la capacidad de controlar la programación con una granularidad VP, lo que puede afectar a su planificación de backup y recuperación ante desastres en concreto. En el caso de los equipos de desarrollo que desean aprovechar la funcionalidad de clonado de PVC en el almacenamiento de ONTAP, esto solo es posible cuando se utiliza la `ontap-nas`, `ontap-san` o. `ontap-san-economy` de windows

NOTA La `solidfire-san` El controlador también es capaz de clonar EVs.

Controladores de entorno de administración Cloud Volumes ONTAP

Cloud Volumes ONTAP proporciona control de datos junto con funciones de almacenamiento empresarial para diversos casos de uso, como recursos compartidos de archivos y almacenamiento a nivel de bloque que presta servicio a protocolos NAS y SAN (NFS, SMB/CIFS e iSCSI). Los controladores compatibles para Cloud Volume ONTAP son `ontap-nas`, `ontap-nas-economy`, `ontap-san` y.. `ontap-san-economy`. Estos son aplicables a Cloud Volume ONTAP para Azure, Cloud Volume ONTAP para GCP.

Controladores de entorno de administración de Amazon FSX para ONTAP

Amazon FSx para 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

La `solidfire-san` El controlador que se utiliza con las plataformas HCI/SolidFire de NetApp, ayuda al administrador a configurar un back-end de Element para Trident según los límites de calidad de servicio. Si desea diseñar el back-end de modo que establezca los límites de calidad de servicio específicos en los volúmenes aprovisionados por Trident, utilice la `type` parámetro en el archivo back-end. El administrador también puede restringir el tamaño del volumen que podría crearse en el almacenamiento mediante el `limitVolumeSize` parámetro. Actualmente, las funciones de almacenamiento de Element, como el cambio de tamaño de volumen y la replicación de volumen, no se admiten mediante el `solidfire-san` controlador. Estas operaciones se deben realizar manualmente mediante la interfaz de usuario web del software Element.

Controlador SolidFire	Snapshot	Clones	Conexión múltiple	CHAP	Calidad de servicio	Cambie el tamaño	Replicación
`solidfire-san`	Sí	Sí	Yespie de página:2[]	Sí	Sí	Sí	Yespie de página:1[]

> Nota al pie de página: Yes [1]: No gestionado por Trident Yes [2]: Compatible con volúmenes de bloque sin procesar

Controladores de entorno de administración Azure NetApp Files

Trident utiliza el `azure-netapp-files` driver para gestionar el "Azure NetApp Files" service.

Puede encontrar más información sobre este controlador y cómo configurarlo en "Configuración de backend de 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	Yespie de página:1[]

> Nota al pie de página: YesFootnote:1[]: No administrado por Trident

Diseño de clase de almacenamiento

Las clases de almacenamiento individuales deben configurarse y aplicarse para crear un objeto de clase de almacenamiento Kubernetes. En esta sección se analiza cómo diseñar una clase de almacenamiento para su aplicación.

Utilización de back-end específica

El filtrado se puede usar en un objeto de clase de almacenamiento específico para determinar el pool o conjunto de pools de almacenamiento que se utilizarán con esa clase de almacenamiento específica. Se pueden establecer tres conjuntos de filtros en la clase de almacenamiento: `storagePools`, `additionalStoragePools`, y/o. `excludeStoragePools`.

```
`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 Trident utiliza para el aprovisionamiento junto con el
conjunto de pools seleccionados por los atributos y `storagePools`
parámetros. Es posible usar un parámetro de forma independiente o ambos
juntos para garantizar que se seleccione el conjunto adecuado de pools de
almacenamiento.
```

La `excludeStoragePools` el parámetro se utiliza para excluir específicamente el conjunto de pools enumerado que coincide con los atributos.

Emular las políticas de calidad de servicio

Si desea diseñar clases de almacenamiento para emular políticas de calidad de servicio, cree una clase de almacenamiento con la `media` atributo como `hdd` o. `ssd`. Según la `media` Atributo mencionado en la clase de almacenamiento, Trident seleccionará el back-end apropiado `hdd` o. `ssd` agregados para coincidir con el atributo de medios y, a continuación, dirigir el aprovisionamiento de los volúmenes al agregado específico. Por tanto, podemos crear UNA CLASE PREMIUM DE almacenamiento que tendría `media` atributo establecido como `ssd` Las cuales pueden clasificarse como política DE calidad DE servicio PREMIUM. Podemos crear otro ESTÁNDAR de clase de almacenamiento que tenga el conjunto de atributos de medios como "hdd", que podría clasificarse como política DE calidad DE servicio ESTÁNDAR. También podríamos usar el atributo "IOPS" en la clase de almacenamiento para redirigir el aprovisionamiento a un dispositivo Element que se puede definir como una Política de calidad de servicio.

Utilizar back-end basado en funciones específicas

Las clases de almacenamiento se pueden diseñar para dirigir el aprovisionamiento de volúmenes en un entorno de administración específico, donde se habilitan funciones como thin provisioning y thick, copias Snapshot, clones y cifrado. Para especificar qué almacenamiento se debe utilizar, cree clases de almacenamiento que especifiquen el back-end adecuado con la función necesaria habilitada.

Pools virtuales

Los pools virtuales están disponibles para todos los back-ends de Trident. Puede definir pools virtuales para cualquier backend, utilizando cualquier controlador que proporcione 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 etiquetas específicas, Trident elige un back-end que coincide con todas las etiquetas de selector para colocar el volumen. Si las etiquetas de selector de clase de almacenamiento coinciden con varios pools de

almacenamiento, Trident elegirá uno de ellos de los que aprovisionar el volumen.

Diseño de pool virtual

Al crear un backend, generalmente se puede especificar un conjunto de parámetros. Antes, el administrador no podía crear otro backend con las mismas credenciales de almacenamiento y un conjunto de parámetros diferente. Con la introducción de los grupos virtuales, este problema se ha solucionado. Un grupo virtual es una abstracción de nivel entre el backend y la clase de almacenamiento de Kubernetes, que permite al administrador definir parámetros y etiquetas, a los que se puede hacer referencia mediante clases de almacenamiento de Kubernetes como selector, de forma independiente del backend. Se pueden definir grupos virtuales para todos los backends de NetApp compatibles con Trident. Esa lista incluye SolidFire/ NetApp HCI, ONTAP, así como Azure NetApp Files.

NOTA	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. Configurado `servicelevel` aspecto al nivel de rendimiento apropiado y agregar otros aspectos requeridos en cada etiqueta. Ahora cree diferentes clases de almacenamiento de Kubernetes que se asignarán a diferentes pools virtuales. Con el `parameters.selector` Campo, cada clase de almacenamiento llama a qué pools virtuales se pueden utilizar para alojar un volumen.

Asignación de un conjunto específico de aspectos

Se pueden diseñar varios pools virtuales con un conjunto específico de aspectos a partir de un único back-end de almacenamiento. Para ello, configure el backend con varias etiquetas y defina los aspectos necesarios en cada etiqueta. Ahora cree diferentes clases de almacenamiento de Kubernetes usando `parameters.selector` campo que se asignará a diferentes pools virtuales. Los volúmenes que se aprovisionan en el back-end tendrán los aspectos definidos en el pool virtual elegido.

Las características de PVC que afectan al aprovisionamiento de almacenamiento

Algunos parámetros aparte de la clase de almacenamiento solicitada pueden afectar al proceso de decisiones de aprovisionamiento de 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.

Trident intentará hacer coincidir el protocolo de almacenamiento utilizado con el método de acceso especificado de acuerdo con la siguiente matriz. Es independiente de la plataforma de almacenamiento subyacente.

	ReadWriteOnce	ReadOnlyMany	ReadWriteMany
ISCSI	Sí	Sí	Sí (bloque sin formato)

	ReadWriteOnce	ReadOnlyMany	ReadWriteMany
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.

NOTA Los aprovisionadores en árbol de Kubernetes no admiten en este momento las operaciones de cambio de tamaño de volúmenes para VP FC, NFS o iSCSI. Trident admite la expansión de volúmenes NFS, iSCSI y FC.

Los detalles de conexión del VP no se pueden modificar una vez creado.

Cree snapshots de volumen bajo demanda

Trident admite la creación de instantáneas de volumen bajo demanda y la creación de RVP a partir de instantáneas mediante 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

Trident también admite la creación de PersistentVolumes a partir de las 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. Trident gestionará la RVP creando 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 a Trident ni al clúster de Kubernetes, siempre y cuando el agregado de destino sea una a la que tenga acceso la SVM a la que utiliza Trident. Lo que es más importante, si se acaba de añadir el agregado a la SVM, se deberá actualizar el back-end volviendo a añadirlo a Trident. Esto activará que Trident vuelva a inventariar la SVM con el fin de reconocer el nuevo agregado.

Sin embargo, Trident no admite automáticamente el movimiento de volúmenes entre back-ends. Esto incluye entre las SVM del mismo clúster, entre clústeres o en una plataforma de almacenamiento diferente (aunque dicho sistema de almacenamiento sea una que esté conectado a Trident).

Si se copia un volumen en otra ubicación, es posible que se use la función de importación de volúmenes para importar los volúmenes actuales a Trident.

Expanda los volúmenes

Trident admite el cambio de tamaño de los volúmenes persistentes (PV) NFS, iSCSI y FC. Esto permite a los usuarios cambiar el tamaño de sus volúmenes directamente a través de la capa de Kubernetes. La expansión de volumen es posible para todas las principales plataformas de almacenamiento de NetApp , incluyendo ONTAP y los backends SolidFire/ NetApp HCI . Para permitir una posible expansión posterior, establezca `allowVolumeExpansion` a `true` en su StorageClass asociada con el volumen. Siempre que sea necesario cambiar el tamaño del volumen persistente, edite el `spec.resources.requests.storage` anotación en la reclamación de volumen persistente al tamaño de volumen requerido. Trident se encargará automáticamente de redimensionar el 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 a un entorno Kubernetes. Actualmente, esto cuenta con el apoyo de `ontap-nas` , `ontap-nas-flexgroup` , `solidfire-san` , y `azure-netapp-files` conductores. Esta función resulta útil al migrar una aplicación existente a Kubernetes o durante escenarios de recuperación ante desastres.

Cuando use la ONTAP y `solidfire-san` los controladores, utilice el comando `tridentctl import volume <backend-name> <volume-name> -f /path/pvc.yaml` para importar un volumen existente a Kubernetes que gestionará Trident. El archivo RVP YAML o JSON utilizado en el comando de importación del volumen señala a una clase de almacenamiento que identifica a 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 el `azure-netapp-files` Se utiliza el controlador, utilice el comando `tridentctl import volume <backend-name> <volume path> -f /path/pvc.yaml` para importar el volumen a Kubernetes para que sea administrado por Trident. Esto garantiza una referencia de volumen única.

Cuando se ejecuta el comando anterior, Trident encontrará el volumen en el backend y leerá su tamaño. Añadirá (y sobrescribirá automáticamente si es necesario) el tamaño de volumen de la RVP configurada. A continuación, Trident crea el nuevo VP y Kubernetes enlaza la RVP al 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.

Servicio de registro

Se ha documentado en la implementación y administración del almacenamiento para el registro "netapp.io" en la "blog".

Servicio de registro

Al igual que otros servicios OpenShift, el servicio de registro se pone en marcha con Ansible, con parámetros de configuración suministrados por el archivo de inventario, también conocido como los hosts, que se proporcionan al libro de estrategia. Hay dos métodos de instalación que se tratarán: Implementar el registro durante la instalación inicial de OpenShift y desplegar el registro después de que OpenShift haya sido instalado.

PRECAUCIÓN	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, tendrá que establecer la variable Ansible `openshift_enable_unsupported_configurations` para `true` para evitar que el instalador falle.

Manos a la obra

Opcionalmente, el servicio de registro puede implementarse tanto para aplicaciones como para las operaciones principales del propio clúster OpenShift. Si decide implementar el registro de operaciones, especificando la variable `openshift_logging_use_ops` como `true`, se crearán dos instancias del servicio. Las variables que controlan la instancia de registro de las operaciones contienen "OPS" en ellas, mientras que la instancia de las aplicaciones no.

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.

NOTA	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 Red Hat 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 de NFS para el servicio de registro.
`openshift_logging_storage_host`	El nombre de host o la dirección IP del host NFS. Esto debe establecerse en la LIF de datos para la máquina virtual.
`openshift_logging_storage_nfs_directory`	La ruta de montaje para la exportación NFS. Por ejemplo, si el volumen se juntan como `/openshift_logging`, utilizaría esa ruta de acceso para esta variable.
`openshift_logging_storage_volume_name`	El nombre, por ejemplo `pv_ose_logs`, Del PV que se va a crear.
`openshift_logging_storage_volume_size`	Por ejemplo, el tamaño de la exportación NFS `100Gi`.

Si su clúster OpenShift ya se está ejecutando y, por lo tanto, Trident se ha implementado y configurado, el instalador puede utilizar el aprovisionamiento dinámico para crear los volúmenes. Será necesario configurar

las siguientes variables.

Variable	Detalles
`openshift_logging_es_pvc_dynamic`	Establezca esta opción en true para usar volúmenes aprovisionados dinámicamente.
`openshift_logging_es_pvc_storage_class_name`	El nombre de la clase de almacenamiento que se utilizará en la RVP.
`openshift_logging_es_pvc_size`	El tamaño del volumen solicitado en la RVP.
`openshift_logging_es_pvc_prefix`	Prefijo para los EVs que utiliza el servicio de registro.
`openshift_logging_es_ops_pvc_dynamic`	Establezca en `true` para utilizar volúmenes aprovisionados de forma dinámica para la instancia de registro de operaciones.
`openshift_logging_es_ops_pvc_storage_class_name`	Nombre de la clase de almacenamiento para la instancia de registro de operaciones.
`openshift_logging_es_ops_pvc_size`	El tamaño de la solicitud de volumen para la instancia de operaciones.
`openshift_logging_es_ops_pvc_prefix`	Prefijo para las RVP de instancia de OPS.

Despliegue la pila de registro

Si va a implementar el registro como parte del proceso de instalación inicial de OpenShift, sólo tendrá que seguir el proceso de implementación estándar. Ansible configurará y pondrá en marcha los servicios y los objetos de OpenShift necesarios para que el servicio esté disponible tan pronto como finalice Ansible.

No obstante, si se pone en marcha después de la instalación inicial, Ansible deberá usar el libro de estrategia de los componentes. Este proceso puede cambiar ligeramente con diferentes versiones de OpenShift, así que asegúrese de leer y seguir "Documentación de Red Hat OpenShift Container Platform 3,11" para su versión.

Servicio de métricas

El servicio de métricas proporciona al administrador información valiosa sobre el estado, la utilización de recursos y la disponibilidad del clúster OpenShift. También es necesaria para la funcionalidad de escala automática en pod y muchas organizaciones usan datos del servicio de mediciones para su cargo y/o para mostrar aplicaciones.

Al igual que sucede con el servicio de registro y OpenShift en su conjunto, Ansible se utiliza para poner en marcha el servicio de métricas. Además, al igual que el servicio de registro, el servicio de 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.

NOTA	Las siguientes tablas solo contienen las variables relevantes para la configuración del almacenamiento en cuanto se relaciona con el servicio de mediciones. Hay muchas otras opciones en la documentación que se deben revisar, configurar y utilizar de acuerdo con su implementación.

Variable	Detalles
`openshift_metrics_storage_kind`	Establezca en `nfs` Para que el instalador cree un PV de NFS para el servicio de registro.
`openshift_metrics_storage_host`	El nombre de host o la dirección IP del host NFS. Esto debe establecerse en la LIF de datos para la SVM.
`openshift_metrics_storage_nfs_directory`	La ruta de montaje para la exportación NFS. Por ejemplo, si el volumen se juntan como `/openshift_metrics`, utilizaría esa ruta de acceso para esta variable.
`openshift_metrics_storage_volume_name`	El nombre, por ejemplo `pv_ose_metrics`, Del PV que se va a crear.
`openshift_metrics_storage_volume_size`	Por ejemplo, el tamaño de la exportación NFS `100Gi`.

Si su clúster OpenShift ya se está ejecutando y, por lo tanto, Trident se ha implementado y configurado, el instalador puede utilizar el aprovisionamiento dinámico para crear los volúmenes. Será necesario configurar las siguientes variables.

Variable	Detalles
`openshift_metrics_cassandra_pvc_prefix`	Prefijo que se utiliza para las RVP de métricas.
`openshift_metrics_cassandra_pvc_size`	El tamaño de los volúmenes que se van a solicitar.
`openshift_metrics_cassandra_storage_type`	El tipo de almacenamiento que se utilizará para las métricas, debe establecerse una dinámica para que Ansible cree RVP con la clase de almacenamiento adecuada.
`openshift_metrics_cassanda_pvc_storage_class_name`	El nombre de la clase de almacenamiento que se va a utilizar.

Implementar el servicio de métricas

Con las variables de Ansible definidas en el archivo de hosts/inventario, ponga en marcha el servicio con Ansible. Si va a implementar en el momento de la instalación de OpenShift, el PV se creará y utilizará automáticamente. Si va a poner en marcha mediante los libros de estrategia de componentes, después de la instalación de OpenShift, Ansible crea las RVP necesarias y, después de que Trident haya aprovisionado almacenamiento para ellos, pone 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 Red Hat"la versión de modo que esté configurada para el entorno.

Protección de datos y recuperación ante desastres

Conozca las opciones de protección y recuperación para Trident y volúmenes creados mediante Trident. Debería tener una estrategia de protección y recuperación de datos para cada aplicación con un requisito de persistencia.

Replicación y recuperación de Trident

Puede crear un backup para restaurar Trident en caso de desastre.

Replicación de Trident

Trident utiliza CRD de Kubernetes para almacenar y gestionar su propio estado y el clúster etcd de Kubernetes para almacenar sus metadatos.

Pasos

1. Realice una copia de seguridad del clúster etcd de Kubernetes mediante "Kubernetes: Realizar backups de un clúster etcd".
2. Coloque los artefactos de backup en un FlexVol volume

NOTA	NetApp recomienda que proteja la SVM en la que reside FlexVol con una relación de SnapMirror con otra SVM.

Recuperación de Trident

Con los CRD de Kubernetes y la instantánea etcd del clúster de Kubernetes, puede recuperar Trident.

Pasos

1. Desde la SVM de destino, monte el volumen que contiene los certificados y archivos de datos ETCD de Kubernetes en el host que se configurará como nodo maestro.
2. Copie todos los certificados necesarios correspondientes al clúster de Kubernetes en `/etc/kubernetes/pki` y los archivos del miembro etcd debajo de `/var/lib/etcd`.
3. Restaure el clúster de Kubernetes desde el backup etcd mediante "Kubernetes: Restaurar un clúster ETCD".
4. Ejecución `kubectl get crd` Para verificar que todos los recursos personalizados de Trident han surgido y recuperado los objetos de Trident para verificar que todos los datos están disponibles.

Replicación y recuperación de SVM

Trident no puede configurar las 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 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 Trident

Trident no detecta automáticamente los fallos de SVM. En caso de desastre, el administrador puede iniciar manualmente la conmutación por error de Trident en la nueva SVM.

Pasos

1. Cancelar las transferencias programadas y continuas de SnapMirror, interrumpir la relación de replicación, detener la SVM de origen y, a continuación, activar la SVM de destino de SnapMirror.
2. Si especificó `-identity-preserve false` o. `-discard-config network` Al configurar la replicación de SVM, actualice el `managementLIF` y.. `dataLIF` En el archivo de definición de back-end de Trident.
3. Confirme `storagePrefix` Está presente en el archivo de definición de back-end de Trident. Este parámetro no puede cambiarse. Omitiendo `storagePrefix` provocará que la actualización de backend falle.
4. Actualice todos los back-ends requeridos para reflejar el nuevo nombre de la SVM de destino mediante:

```
./tridentctl update backend <backend-name> -f <backend-json-file> -n
<namespace>
```

5. Si especificó `-identity-preserve false` o. `discard-config network`, debe rebotar todos los pods de aplicación.

> **NOTA** Si especificó `-identity-preserve true`, todos los volúmenes aprovisionados por Trident comienzan a servir datos cuando se activa la SVM de destino.

Replicación y recuperación de volúmenes

Trident no puede configurar las relaciones de replicación de SnapMirror; sin embargo, el administrador de almacenamiento puede utilizar "Replicación y recuperación SnapMirror de ONTAP" para replicar volúmenes creados por Trident.

Luego, es posible importar los volúmenes recuperados a Trident con "importación de volumen tridentctl".

> **NOTA** La importación no es compatible con `ontap-nas-economy`, `ontap-san-economy`, o. `ontap-flexgroup-economy` de windows

Protección de datos Snapshot

Puede proteger y restaurar datos con:

- Un controlador snapshot externo y CRD para crear snapshots de volúmenes de Kubernetes de volúmenes persistentes (VP).

  "Copias de Snapshot de volumen"

- Snapshots de ONTAP para restaurar el contenido completo de un volumen o para recuperar archivos o LUN individuales.

  "Snapshots de ONTAP"

Automatización de la conmutación por error de aplicaciones con estado con Trident

La función de desconexión forzada de Trident permite desconectar automáticamente los volúmenes de los nodos defectuosos en un clúster de Kubernetes, evitando la corrupción de datos y garantizando la disponibilidad de las aplicaciones. Esta función resulta especialmente útil en escenarios donde los nodos dejan de responder o se desconectan para realizar tareas de mantenimiento.

Detalles acerca de forzar separación

La separación forzada está disponible para `ontap-san` , `ontap-san-economy` , `ontap-nas` , y `ontap-nas-economy` solo. Antes de habilitar la desconexión forzada, se debe habilitar el apagado de nodo no elegante (NGNS) en el clúster de Kubernetes. NGNS está habilitado de forma predeterminada para Kubernetes 1.28 y superiores. Para obtener más información, consulte "Kubernetes: Cierre de nodo sin gracia" .

NOTA	Cuando se utiliza `ontap-nas` el controlador o `ontap-nas-economy`, es necesario establecer el `autoExportPolicy` parámetro en la configuración de back-end para `true` que Trident pueda restringir el acceso desde el nodo Kubernetes con la contaminación aplicada mediante políticas de exportación gestionadas.
ADVERTENCIA	Dado que 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 del clúster de Kubernetes haya aplicado la `node.kubernetes.io/out-of-service=nodeshutdown:NoExecute` tinta al nodo y `enableForceDetach` se establezca en `true`, Trident determinará el estado del nodo y:

1. Detener el acceso de E/S de backend para los volúmenes montados en ese nodo.
2. Marque el objeto de nodo Trident como `dirty` (no es seguro para las nuevas publicaciones).

NOTA	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 Trident pueda verificar el nodo `clean` (seguro para las nuevas publicaciones).

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

1. Identifique y limpie las rutas publicadas obsoletas en el nodo.
2. Si el nodo está en un `cleanable` estado (se ha quitado el taint de fuera de servicio y el nodo está en `Ready` estado) y todas las rutas obsoletas publicadas están limpias, Trident readmitirá el nodo como `clean` y permitirá que los nuevos volúmenes publicados al nodo.

Detalles sobre la conmutación por error automatizada

Puedes automatizar el proceso de desconexión forzada mediante la integración con"operador de comprobación del estado del nodo (NHC)" . Cuando se produce un fallo en un nodo, NHC activa la remediación de nodos Trident (TNR) y la desconexión forzada automáticamente mediante la creación de un CR TridentNodeRemediation en el espacio de nombres de Trident que define el nodo que ha fallado. El TNR se crea únicamente en caso de fallo del nodo y el NHC lo elimina una vez que el nodo vuelve a estar en línea o se elimina el nodo.

Falló el proceso de eliminación del pod del nodo.

La conmutación por error automatizada selecciona las cargas de trabajo que se eliminarán del nodo que ha fallado. Cuando se crea un TNR, el controlador TNR marca el nodo como modificado, impidiendo cualquier publicación de nuevos volúmenes y comenzando a eliminar los pods compatibles con desconexión forzada y sus conexiones de volumen.

Todos los volúmenes/PVC compatibles con la desconexión forzada son compatibles con la conmutación por error automatizada:

- NAS y volúmenes NAS-economy que utilizan políticas de exportación automática (SMB aún no es compatible).
- SAN y volúmenes de economía SAN.

Referirse aDetalles acerca de forzar separación .

Comportamiento predeterminado:

- Los pods que utilizan volúmenes compatibles con force-detach se eliminan del nodo que ha fallado. Kubernetes reprogramará estas tareas en un nodo en buen estado.
- Los pods que utilizan un volumen no compatible con force-detach, incluidos los volúmenes que no son Trident, no se eliminan del nodo que ha fallado.
- Los pods sin estado (que no son PVC) no se eliminan del nodo fallido, a menos que la anotación del pod `trident.netapp.io/podRemediationPolicy: delete` está listo.

Anulación del comportamiento de eliminación de pods:

El comportamiento de eliminación de pods se puede personalizar mediante una anotación de pod: `trident.netapp.io/podRemediationPolicy[retain, delete]` . Estas anotaciones se examinan y

utilizan cuando se produce una conmutación por error. Aplique anotaciones a la especificación del pod de despliegue/conjunto de réplicas de Kubernetes para evitar que la anotación desaparezca después de una conmutación por error:

- `retain`- El pod NO se eliminará del nodo fallido durante una conmutación por error automatizada.
- `delete`- El pod SERÁ eliminado del nodo fallido durante una conmutación por error automatizada.

Estas anotaciones se pueden aplicar a cualquier pod.

ADVERTENCIA	• Las operaciones de E/S se bloquearán únicamente en los nodos fallidos para los volúmenes que admiten la desconexión forzada. • Para los volúmenes que no admiten la desconexión forzada, existe riesgo de corrupción de datos y problemas de conexión múltiple.

CR de remediación de nodos Trident

El CR TridentNodeRemediation (TNR) define un nodo fallido. El nombre del TNR es el nombre del nodo que falló.

Ejemplo de TNR:

```
apiVersion: trident.netapp.io/v1
kind: TridentNodeRemediation
metadata:
  name: <K8s-node-name>
spec: {}
```

Estado de TNR: Utilice los siguientes comandos para ver el estado de los TNR:
`kubectl get tnr <name> -n <trident-namespace>`

Los TNR pueden encontrarse en uno de los siguientes estados:

- *Remediación*:
  - Suspender el acceso de E/S de backend para los volúmenes compatibles con force-detach montados en ese nodo.
  - El objeto de nodo Trident está marcado como sucio (no es seguro para nuevas publicaciones).
  - Retire los pods y las conexiones de volumen del nodo.
- *Recuperación de nodo pendiente*:
  - El controlador está esperando a que el nodo vuelva a estar en línea.
  - Una vez que el nodo esté en línea, la aplicación de la publicación garantizará que el nodo esté limpio y listo para nuevas publicaciones de volúmenes.
- Si el nodo se elimina de K8s, el controlador TNR eliminará el TNR y detendrá la reconciliación.
- *Éxito*:
  - Todos los pasos de remediación y recuperación de nodos se completaron con éxito. El nodo está limpio y listo para nuevas publicaciones de volúmenes.

- *Fallido*:
  - Error irrecuperable. Los motivos de error se establecen en el campo status.message del CR.

Habilitación de la conmutación por error automática

Requisitos previos:

- Asegúrese de que la desconexión forzada esté habilitada antes de habilitar la conmutación por error automática. Para obtener más información, consulteDetalles acerca de forzar separación .
- Instale la comprobación del estado del nodo (NHC) en el clúster de Kubernetes.
  - "Instalar operator-sdk".
  - Instale Operator Lifecycle Manager (OLM) en el clúster si aún no está instalado: `operator-sdk olm install` .
  - Instalar el operador de comprobación del estado del nodo: `kubectl create -f https://operatorhub.io/install/node-healthcheck-operator.yaml` .

NOTA	También puedes usar formas alternativas para detectar fallas de nodos como se especifica en la sección de Integración de soluciones personalizadas de comprobación del estado de los nodos abajo.

Ver"Operador de comprobación del estado del nodo" Para más información.

Pasos

1. Cree un CR NodeHealthCheck (NHC) en el espacio de nombres Trident para supervisar los nodos de trabajo en el clúster. Ejemplo:

```
apiVersion: remediation.medik8s.io/v1alpha1
kind: NodeHealthCheck
metadata:
  name: <CR name>
spec:
  selector:
    matchExpressions:
      - key: node-role.kubernetes.io/control-plane
        operator: DoesNotExist
      - key: node-role.kubernetes.io/master
        operator: DoesNotExist
  remediationTemplate:
    apiVersion: trident.netapp.io/v1
    kind: TridentNodeRemediationTemplate
    namespace: <Trident installation namespace>
    name: trident-node-remediation-template
  minHealthy: 0 # Trigger force-detach upon one or more node failures
  unhealthyConditions:
    - type: Ready
      status: "False"
      duration: 0s
    - type: Ready
      status: Unknown
      duration: 0s
```

2. Aplique la comprobación de estado del nodo CR en el `trident` espacio de nombres.

```
kubectl apply -f <nhc-cr-file>.yaml -n <trident-namespace>
```

El CR anterior está configurado para vigilar los nodos de trabajo de K8s en busca de condiciones de nodo Ready: false y Unknown. La conmutación por error automática se activará cuando un nodo pase al estado Listo: falso o Listo: desconocido.

El `unhealthyConditions` En el CR se utiliza un período de gracia de 0 segundos. Esto provoca que la conmutación por error automatizada se active inmediatamente después de que K8s establezca la condición del nodo Ready: false, que se establece después de que K8s pierde el latido de un nodo. K8s tiene un tiempo de espera predeterminado de 40 segundos después del último latido antes de establecer Ready: false. Este período de gracia se puede personalizar en las opciones de implementación de K8s.

Para obtener opciones de configuración adicionales, consulte"Documentación del operador de comprobación de estado del nodo" .

Información de configuración adicional

Cuando Trident se instala con la opción de desconexión forzada habilitada, se crean automáticamente dos recursos adicionales en el espacio de nombres de Trident para facilitar la integración con NHC: TridentNodeRemediationTemplate (TNRT) y ClusterRole.

Plantilla de remediación de nodos Trident (TNRT):

El TNRT sirve como plantilla para el controlador NHC, que utiliza el TNRT para generar recursos TNR según sea necesario.

```
apiVersion: trident.netapp.io/v1
kind: TridentNodeRemediationTemplate
metadata:
  name: trident-node-remediation-template
  namespace: trident
spec:
  template:
    spec: {}
```

Rol de clúster:

También se agrega un rol de clúster durante la instalación cuando se habilita la desconexión forzada. Esto otorga a NHC permisos para TNR en el espacio de nombres Trident .

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  labels:
    rbac.ext-remediation/aggregate-to-ext-remediation: "true"
  name: tridentnoderemediation-access
rules:
- apiGroups:
  - trident.netapp.io
  resources:
  - tridentnoderemediationtemplates
  - tridentnoderemediations
  verbs:
  - get
  - list
  - watch
  - create
  - update
  - patch
  - delete
```

Actualizaciones y mantenimiento de clústeres K8s

Para evitar cualquier conmutación por error, pause la conmutación por error automatizada durante el mantenimiento o las actualizaciones de K8s, cuando se espera que los nodos se desconecten o se reinicien. Puedes pausar el CR del NHC (descrito anteriormente) parcheando su CR:

```
kubectl patch NodeHealthCheck <cr-name> --patch
'{"spec":{"pauseRequests":["<description-for-reason-of-pause>"]}}' --type=merge
```

Esto pausa el proceso de conmutación por error automática. Para volver a habilitar la conmutación por error automática, elimine las solicitudes de pausa de la especificación una vez finalizado el mantenimiento.

Limitaciones

- Las operaciones de E/S se impiden únicamente en los nodos fallidos para los volúmenes compatibles con force-detach. Solo se eliminan automáticamente los pods que utilizan volúmenes/PVC compatibles con force-detach.
- La conmutación por error automática y la desconexión forzada se ejecutan dentro del módulo controlador Trident. Si falla el nodo que aloja el controlador Trident, la conmutación por error automatizada se retrasará hasta que K8s mueva el pod a un nodo en buen estado.

Integración de soluciones personalizadas de comprobación del estado de los nodos

Puede reemplazar el operador de comprobación de estado del nodo con herramientas alternativas de detección de fallos de nodo para activar la conmutación por error automática. Para garantizar la compatibilidad con el mecanismo de conmutación por error automatizado, su solución personalizada debe:

- Cree un TNR cuando se detecte una falla de nodo, utilizando el nombre del nodo fallido como nombre CR del TNR.
- Elimine el TNR cuando el nodo se haya recuperado y el TNR esté en el estado Correcto.

Seguridad

Seguridad

Utilice las recomendaciones que se indican aquí para asegurarse de que la instalación de Trident es segura.

Ejecute 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 definiciones de objetos de Trident o los pods a fin de garantizar un almacenamiento fiable y bloquear posibles actividades maliciosas.

Para separar el resto de aplicaciones y usuarios de Trident, instale siempre Trident en su propio espacio de nombres de Kubernetes (`trident`). Al colocar Trident en su propio espacio de nombres, se garantiza que solo el personal administrativo de Kubernetes tenga acceso al pod de Trident y a los artefactos (como el back-end y los secretos CHAP, si procede) almacenados en los objetos CRD con nombres. Debe asegurarse de permitir solo el acceso de los administradores al espacio de nombres Trident y, por lo tanto, el acceso a la `tridentctl` aplicación.

Utilice la autenticación CHAP con los back-ends DE SAN de ONTAP

Trident admite la autenticación basada en CHAP para cargas de trabajo SAN de ONTAP (mediante `ontap-san` y `ontap-san-economy` controladores). NetApp recomienda el uso de CHAP bidireccional con Trident para la autenticación entre un host y el back-end de almacenamiento.

Para los back-ends ONTAP que utilizan los controladores de almacenamiento SAN, Trident puede configurar

CHAP bidireccional y gestionar nombres de usuario y secretos CHAP a través de `tridentctl`. Consulte "Prepárese para configurar el back-end con los controladores SAN de ONTAP"para comprender cómo configura Trident 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. Trident utiliza un objeto secreto que incluye dos contraseñas CHAP por inquilino. Cuando Trident está instalado, administra los secretos CHAP y los almacena en un `tridentvolume` objeto CR para el VP respectivo. Al crear un VP, Trident utiliza los secretos CHAP para iniciar una sesión iSCSI y comunicarse con el sistema NetApp HCI y SolidFire a través de CHAP.

NOTA Los volúmenes que crea Trident no se asocian con ningún grupo de acceso de volúmenes.

Utilice 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 Trident será habilitado para NAE.
  - Puede establecer la marca de cifrado de NVE `""` para crear volúmenes con la función NAE habilitada.
- Si NAE no está habilitado en el back-end, cualquier volumen aprovisionado en Trident será NVE habilitado a menos que la marca de cifrado de NVE esté establecida en `false` (el valor predeterminado) en la configuración del back-end.

NOTA

Los volúmenes creados en Trident en un back-end habilitado para NAE deben estar cifrados NVE o NAE.

- Puede establecer el indicador NVE Encryption como `true` En la configuración del back-end de Trident, con el fin de anular el cifrado NAE y utilizar una clave de cifrado específica por volumen.
- Al configurar la marca de cifrado de NVE `false` en un back-end habilitado para NAE, se crea un volumen con la función NAE habilitada. No se puede deshabilitar el cifrado NAE mediante la marca de cifrado de NVE en `false`.

- Se puede crear manualmente un volumen de NVE en Trident mediante la configuración explícita de 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 la configuración de clave unificada de Linux (LUKS) para cifrar los volúmenes de ECONOMÍA DE SAN de ONTAP y SAN de ONTAP en Trident. Trident admite la rotación de frase de acceso y la expansión de volumen para volúmenes

cifrados con LUKS.

En Trident, los volúmenes cifrados con LUKS utilizan el cifrado y el modo aes-xts-plain64, como recomienda "NIST".

NOTA	El cifrado LUKS no es compatible con los sistemas ASA r2. Para obtener información sobre los sistemas ASA r2, consulte "Obtenga información sobre los sistemas de almacenamiento R2 de ASA" .

Antes de empezar

- Los nodos de trabajo deben tener instalado cryptsetup 2.1 o superior (pero inferior a 3.0). Si desea más información, visite "Gitlab: Cryptsetup".
- Por motivos de rendimiento, NetApp recomienda que los nodos de trabajo admitan las instrucciones nuevas estándar de cifrado avanzado (AES-NI). Para verificar el soporte de AES-ni, ejecute el siguiente comando:

```
grep "aes" /proc/cpuinfo
```

Si no se devuelve nada, su procesador no admite AES-ni. Para obtener más información sobre AES-ni, visite: "Intel: Instrucciones estándar de cifrado avanzado (AES-ni)".

Active el cifrado LUKS

Puede habilitar el cifrado por volumen en el lado del host usando la configuración de clave unificada de Linux (LUKS) para SAN de ONTAP y volúmenes DE ECONOMÍA SAN de ONTAP.

Pasos

1. Defina los atributos de cifrado LUKS en la configuración de backend. Para obtener más información sobre las opciones de configuración del back-end para SAN de ONTAP, consulte "Opciones de configuración DE SAN de ONTAP".

```
{
  "storage": [
    {
      "labels": {
        "luks": "true"
      },
      "zone": "us_east_1a",
      "defaults": {
        "luksEncryption": "true"
      }
    },
    {
      "labels": {
        "luks": "false"
      },
      "zone": "us_east_1a",
      "defaults": {
        "luksEncryption": "false"
      }
    }
  ]
}
```

2. Uso `parameters.selector` Para definir los pools de almacenamiento mediante el cifrado LUKS. Por ejemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: 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 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.

ADVERTENCIA	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, Trident compara la frase de acceso LUKS del volumen con la frase de acceso activa del secreto.

- Si la clave de acceso del volumen no coincide con la clave de acceso activa en el secreto, se produce la rotación.
- Si la clave de acceso del volumen coincide con la clave de acceso activa en el secreto, el `previous-luks-passphrase` se ignora el parámetro.

Pasos

1. Añada el `node-publish-secret-name` y.. `node-publish-secret-namespace` Parámetros de StorageClass. Por ejemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-san
provisioner: csi.trident.netapp.io
parameters:
  trident.netapp.io/backendType: "ontap-san"
  csi.storage.k8s.io/node-stage-secret-name: luks
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
  csi.storage.k8s.io/node-publish-secret-name: luks
  csi.storage.k8s.io/node-publish-secret-namespace: ${pvc.namespace}
```

2. Identifique las bases de datos passhrases existentes en el volumen o la snapshot.

Volumen

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames:["A"]
```

Snapshot

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames:["A"]
```

3. Actualice el secreto LUKS del volumen para especificar las passphrases nuevas y anteriores. Asegúrese `previous-luke-passphrase-name` y.. `previous-luks-passphrase` coincidir con la frase de contraseña anterior.

```
apiVersion: v1
kind: Secret
metadata:
  name: luks-pvc1
stringData:
  luks-passphrase-name: B
  luks-passphrase: secretB
  previous-luks-passphrase-name: A
  previous-luks-passphrase: secretA
```

4. Cree un nuevo pod montando el volumen. Esto es necesario para iniciar la rotación.
5. Compruebe que se ha girado la frase de paso.

Volumen

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames:["B"]
```

Snapshot

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames:["B"]
```

Resultados

La frase de contraseña se giró cuando solo se devuelve la nueva frase de contraseña en el volumen y la instantánea.

NOTA	Si se devuelven dos passphrasas, por ejemplo `luksPassphraseNames: ["B", "A"]`, la rotación está incompleta. Puede activar un nuevo pod para intentar completar la rotación.

Habilite la expansión de volumen

Es posible habilitar la ampliación de volumen en un volumen cifrado LUKS.

Pasos

1. Habilite el `CSINodeExpandSecret` puerta de características (beta 1.25+). Consulte "Kubernetes 1.25: Use Secrets for Node-Driven Expansion of CSI Volumes" para obtener más detalles.
2. Añada el `node-expand-secret-name` y.. `node-expand-secret-namespace` Parámetros de StorageClass. Por ejemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: 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.

Cifrado en tránsito de Kerberos

Con el cifrado en tiempo real de Kerberos, puede mejorar la seguridad de acceso a los datos al habilitar el cifrado del tráfico entre su clúster gestionado y el entorno de administración de almacenamiento.

Trident es compatible con el cifrado Kerberos para ONTAP como back-end de almacenamiento:

- **ONTAP local** - Trident admite el cifrado Kerberos sobre conexiones NFSv3 y NFSv4 desde Red Hat OpenShift y clústeres Kubernetes upstream a volúmenes ONTAP locales.

Puede crear, eliminar, cambiar el tamaño, copiar, clonar, Clone de solo lectura e importe volúmenes que usen cifrado NFS.

Configure el cifrado de Kerberos en tránsito con volúmenes de ONTAP locales

Es posible habilitar el cifrado de Kerberos en el tráfico de almacenamiento entre el clúster gestionado y un back-end de almacenamiento de ONTAP en las instalaciones.

NOTA	El cifrado de Kerberos para el tráfico NFS con back-ends de almacenamiento de ONTAP en las instalaciones solo se admite mediante `ontap-nas` el controlador de almacenamiento.

Antes de empezar

- Asegúrese de tener acceso a la `tridentctl` utilidad.
- Asegúrese de tener acceso de administrador al back-end de almacenamiento de ONTAP.
- Asegúrese de conocer el nombre del volumen o los volúmenes que compartirá desde el back-end de almacenamiento ONTAP.
- Asegúrese de haber preparado la máquina virtual de almacenamiento de ONTAP para admitir el cifrado de Kerberos para los volúmenes de NFS. Consulte "Habilite Kerberos en una LIF de datos" para obtener instrucciones.
- Asegúrate de que cualquier volumen NFSv4 que uses con cifrado Kerberos esté configurado correctamente. Consulta la sección NetApp NFSv4 Domain Configuration (página 13) de "NetApp NFSv4 mejoras y guía de mejores prácticas".

Añada o modifique las políticas de exportación de ONTAP

Tiene que agregar reglas a políticas de exportación de ONTAP existentes o crear nuevas políticas de exportación que sean compatibles con el cifrado de Kerberos para el volumen raíz de la máquina virtual de almacenamiento de ONTAP, así como para cualquier volumen de ONTAP compartido con el clúster de Kubernetes ascendente. Las reglas de políticas de exportación que añada, o las nuevas políticas de exportación que cree, deben admitir los siguientes protocolos de acceso y permisos de acceso:

Protocolos de acceso

Configura la política de exportación con los protocolos de acceso NFS, NFSv3 y NFSv4.

Detalles de acceso

Puede configurar una de tres versiones diferentes de cifrado de Kerberos, según las necesidades del

volumen:

- **Kerberos 5** - (autenticación y cifrado)
- **Kerberos 5i** - (autenticación y encriptación con protección de identidad)
- **Kerberos 5p** - (autenticación y encriptación con protección de identidad y privacidad)

Configure la regla de política de exportación de ONTAP con los permisos de acceso adecuados. Por ejemplo, si los clústeres montarán los volúmenes NFS con una combinación de Kerberos 5i y cifrado Kerberos 5p, utilice los siguientes ajustes de acceso:

Tipo	Acceso de solo lectura	Acceso de lectura/escritura	Acceso de superusuario
UNIX	Activado	Activado	Activado
Kerberos 5i	Activado	Activado	Activado
Kerberos 5p	Activado	Activado	Activado

Consulte la siguiente documentación para saber cómo crear políticas de exportación de ONTAP y reglas de políticas de exportación:

- "Cree una política de exportación"
- "Añada una regla a una política de exportación"

Cree un back-end de almacenamiento

Puede crear una configuración de back-end de almacenamiento de Trident que incluya la funcionalidad de cifrado de Kerberos.

Acerca de esta tarea

Al crear un archivo de configuración de backend de almacenamiento que configure el cifrado Kerberos, puede especificar una de las tres versiones diferentes del cifrado Kerberos mediante el `spec.nfsMountOptions` parámetro:

- `spec.nfsMountOptions: sec=krb5` (autenticación y cifrado)
- `spec.nfsMountOptions: sec=krb5i` (autenticación y cifrado con protección de identidad)
- `spec.nfsMountOptions: sec=krb5p` (autenticación y encriptación con protección de identidad y privacidad)

Especifique solo un nivel de Kerberos. Si especifica más de un nivel de cifrado de Kerberos en la lista de parámetros, sólo se utilizará la primera opción.

Pasos

1. En el clúster gestionado, cree un archivo de configuración de back-end de almacenamiento utilizando el ejemplo siguiente. Sustituya los valores entre paréntesis <> por información de su entorno:

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-ontap-nas-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-ontap-nas
spec:
  version: 1
  storageDriverName: "ontap-nas"
  managementLIF: <STORAGE_VM_MGMT_LIF_IP_ADDRESS>
  dataLIF: <PROTOCOL_LIF_FQDN_OR_IP_ADDRESS>
  svm: <STORAGE_VM_NAME>
  username: <STORAGE_VM_USERNAME_CREDENTIAL>
  password: <STORAGE_VM_PASSWORD_CREDENTIAL>
  nasType: nfs
  nfsMountOptions: ["sec=krb5i"] #can be krb5, krb5i, or krb5p
  qtreesPerFlexvol:
  credentials:
    name: backend-ontap-nas-secret
```

2. Utilice el archivo de configuración que creó en el paso anterior para crear el backend:

```
tridentctl create backend -f <backend-configuration-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.

Cree una clase de almacenamiento

Puede crear una clase de almacenamiento para aprovisionar volúmenes con el cifrado de Kerberos.

Acerca de esta tarea

Al crear un objeto de clase de almacenamiento, puede especificar una de las tres versiones diferentes del cifrado de Kerberos mediante el `mountOptions` parámetro:

- `mountOptions: sec=krb5` (autenticación y cifrado)
- `mountOptions: sec=krb5i` (autenticación y cifrado con protección de identidad)
- `mountOptions: sec=krb5p` (autenticación y encriptación con protección de identidad y privacidad)

Especifique solo un nivel de Kerberos. Si especifica más de un nivel de cifrado de Kerberos en la lista de parámetros, sólo se utilizará la primera opción. Si el nivel de cifrado especificado en la configuración de backend de almacenamiento es diferente al nivel especificado en el objeto de clase de almacenamiento, el objeto de clase de almacenamiento tiene prioridad.

Pasos

1. Cree un objeto de Kubernetes StorageClass, mediante el siguiente ejemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-nas-sc
provisioner: csi.trident.netapp.io
mountOptions:
  - sec=krb5i #can be krb5, krb5i, or krb5p
parameters:
  backendType: ontap-nas
  storagePools: ontapnas_pool
  trident.netapp.io/nasType: nfs
allowVolumeExpansion: true
```

2. Cree la clase de almacenamiento:

```
kubectl create -f sample-input/storage-class-ontap-nas-sc.yaml
```

3. Asegúrese de que se ha creado la clase de almacenamiento:

```
kubectl get sc ontap-nas-sc
```

   Debería ver una salida similar a la siguiente:

```
NAME          PROVISIONER             AGE
ontap-nas-sc    csi.trident.netapp.io   15h
```

Aprovisione los volúmenes

Después de crear un back-end de almacenamiento y una clase de almacenamiento, ahora puede aprovisionar un volumen. Para obtener instrucciones, consulte "Aprovisione un volumen" .

Configure el cifrado de Kerberos en tránsito con volúmenes Azure NetApp Files

Puede habilitar el cifrado de Kerberos en el tráfico de almacenamiento entre su clúster gestionado y un solo back-end de almacenamiento de Azure NetApp Files o un pool virtual de back-ends de almacenamiento de Azure NetApp Files.

Antes de empezar

- Asegúrese de haber habilitado Trident en el clúster gestionado de Red Hat OpenShift.
- Asegúrese de tener acceso a la `tridentctl` utilidad.
- Asegúrese de haber preparado el back-end de almacenamiento de Azure NetApp Files para el cifrado Kerberos siguiendo los requisitos y siguiendo las instrucciones de "Documentación de Azure NetApp Files".
- Asegúrate de que cualquier volumen NFSv4 que uses con cifrado Kerberos esté configurado correctamente. Consulta la sección NetApp NFSv4 Domain Configuration (página 13) de "NetApp NFSv4 mejoras y guía de mejores prácticas".

Cree un back-end de almacenamiento

Puede crear una configuración de back-end de almacenamiento de Azure NetApp Files que incluya la funcionalidad de cifrado de Kerberos.

Acerca de esta tarea

Cuando crea un archivo de configuración de backend de almacenamiento que configura el cifrado Kerberos, puede definirlo para que se aplique en uno de los dos niveles posibles:

- El **storage backend level** usando el `spec.kerberos` campo
- El **nivel de pool virtual** usando el `spec.storage.kerberos` campo

Cuando se define la configuración en el nivel del pool virtual, el pool se selecciona con la etiqueta de la clase de almacenamiento.

En cualquier nivel, puede especificar una de las tres versiones diferentes del cifrado Kerberos:

- `kerberos: sec=krb5` (autenticación y cifrado)
- `kerberos: sec=krb5i` (autenticación y cifrado con protección de identidad)
- `kerberos: sec=krb5p` (autenticación y encriptación con protección de identidad y privacidad)

Pasos

1. En el clúster gestionado, cree un archivo de configuración de back-end de almacenamiento mediante uno de los siguientes ejemplos, en función del lugar donde necesite definir el back-end de almacenamiento (nivel de back-end de almacenamiento o nivel de pool virtual). Sustituya los valores entre paréntesis <> por información de su entorno:

Ejemplo de nivel de back-end de almacenamiento

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-secret
```

Ejemplo de nivel de pool virtual

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  storage:
    - labels:
        type: encryption
      kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-secret
```

2. Utilice el archivo de configuración que creó en el paso anterior para crear el backend:

```
tridentctl create backend -f <backend-configuration-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.

Cree una clase de almacenamiento

Puede crear una clase de almacenamiento para aprovisionar volúmenes con el cifrado de Kerberos.

Pasos

1. Cree un objeto de Kubernetes StorageClass, mediante el siguiente ejemplo:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-nfs
provisioner: csi.trident.netapp.io
parameters:
  backendType: azure-netapp-files
  trident.netapp.io/nasType: nfs
  selector: type=encryption
```

2. Cree la clase de almacenamiento:

```
kubectl create -f sample-input/storage-class-sc-nfs.yaml
```

3. Asegúrese de que se ha creado la clase de almacenamiento:

```
kubectl get sc -sc-nfs
```

Debería ver una salida similar a la siguiente:

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

Aprovisione los volúmenes

Después de crear un back-end de almacenamiento y una clase de almacenamiento, ahora puede aprovisionar un volumen. Para obtener instrucciones, consulte "Aprovisione un volumen" .

Proteja las aplicaciones con Trident Protect

Obtenga más información sobre Trident Protect

NetApp Trident Protect proporciona capacidades avanzadas de gestión de datos de aplicaciones que mejoran la funcionalidad y la disponibilidad de las aplicaciones Kubernetes con estado respaldadas por los sistemas de almacenamiento NetApp ONTAP y el aprovisionador de almacenamiento NetApp Trident CSI. Trident Protect simplifica la gestión, la protección y el movimiento de cargas de trabajo en contenedores en nubes públicas y entornos locales. También ofrece capacidades de automatización a través de su API y CLI.

Puede proteger aplicaciones con Trident Protect creando recursos personalizados (CR) o utilizando la CLI de Trident Protect.

El futuro

Puede obtener información sobre los requisitos de Trident Protect antes de instalarlo:

- "Requisitos de Trident Protect"

Instalar Trident Protect

Requisitos de Trident Protect

Comience por verificar que su entorno operativo, clústeres de aplicaciones, aplicaciones y licencias estén listos. Asegúrese de que su entorno cumpla con estos requisitos para implementar y operar Trident Protect.

Compatibilidad del clúster Kubernetes de Trident Protect

Trident Protect es compatible con una amplia gama de ofertas de Kubernetes totalmente administradas y autoadministradas, que incluyen:

- Amazon Elastic Kubernetes Service (EKS)
- Google Kubernetes Engine (GKE)
- Microsoft Azure Kubernetes Service (AKS)
- Red Hat OpenShift
- SUSE Harvester 1.7.0 (ONTAP iSCSI)
- SUSE Rancher
- Cartera de VMware Tanzania
- Subida de Kubernetes

NOTA

- Las copias de seguridad de Trident Protect solo se admiten en nodos de cómputo de Linux. Los nodos de cómputo de Windows no son compatibles con operaciones de copia de seguridad.
- Asegúrese de que el clúster en el que instala Trident Protect esté configurado con un controlador de instantáneas en ejecución y los CRD relacionados. Para instalar un controlador de instantáneas, consulte "estas instrucciones" .
- Asegúrate de que existe al menos un VolumeSnapshotClass. Para más información, consulta "VolumeSnapshotClass".
- Se requiere Helm 4.x o posterior para instalar Trident Protect.

Compatibilidad del backend de almacenamiento Trident Protect

Trident Protect admite los siguientes backends de almacenamiento:

- Amazon FSX para ONTAP de NetApp
- Cloud Volumes ONTAP
- Cabinas de almacenamiento ONTAP de NetApp
- NetApp Volumes para Google Cloud
- Azure NetApp Files

Asegúrese de que el back-end de almacenamiento cumple los siguientes requisitos:

- Asegúrese de que el almacenamiento de NetApp conectado al clúster utilice Trident 24.02 o más reciente (se recomienda Trident 24.10).
- Asegúrese de tener un back-end de almacenamiento NetApp ONTAP.
- Asegúrese de haber configurado un depósito de almacenamiento de objetos para almacenar backups.
- Cree los espacios de nombres de aplicación que planee utilizar para las aplicaciones o las operaciones de administración de datos de las aplicaciones. Trident Protect no crea estos espacios de nombres por usted; si especifica un espacio de nombres inexistente en un recurso personalizado, la operación fallará.

Requisitos para volúmenes de economía nas

Trident Protect admite operaciones de copia de seguridad y restauración en volúmenes nas-economy. Actualmente no se admiten instantáneas, clones ni replicación SnapMirror a volúmenes nas-economy. Debe habilitar un directorio de instantáneas para cada volumen nas-economy que planea usar con Trident Protect.

NOTA

Algunas aplicaciones no son compatibles con volúmenes que usan un directorio Snapshot. Para estas aplicaciones, debe ocultar el directorio Snapshot mediante la ejecución del siguiente comando en el sistema de almacenamiento de ONTAP:

```
nfs modify -vserver <svm> -v3-hide-snapshot enabled
```

Para habilitar el directorio snapshot, ejecute el siguiente comando para cada volumen nas-económico, sustituyéndolo `<volume-UUID>` por el UUID del volumen que desea cambiar:

```
tridentctl update volume <volume-UUID> --snapshot-dir=true --pool-level
=true -n trident
```

NOTA Es posible habilitar los directorios de snapshots de forma predeterminada para volúmenes nuevos si se configura la opción Trident backend configuration `snapshotDir` en `true`. Los volúmenes existentes no se ven afectados.

Protección de datos con máquinas virtuales de KubeVirt

Trident Protect proporciona capacidades de congelamiento y descongelamiento del sistema de archivos para máquinas virtuales KubeVirt durante operaciones de protección de datos para garantizar la consistencia de los datos. El método de configuración y el comportamiento predeterminado para las operaciones de congelamiento de VM varían según las versiones de Trident Protect; las versiones más nuevas ofrecen una configuración simplificada a través de los parámetros del gráfico de Helm.

NOTA Durante las operaciones de restauración, cualquier `VirtualMachineSnapshots` Los datos creados para una máquina virtual (VM) no se restauran.

Trident Protect 25.10 y versiones posteriores

Trident Protect congela y descongela automáticamente los sistemas de archivos de KubeVirt durante las operaciones de protección de datos para garantizar la coherencia. A partir de Trident Protect 25.10, puede desactivar este comportamiento mediante el `vm.freeze` parámetro durante la instalación del gráfico Helm. El parámetro está habilitado por defecto.

```
helm install ... --set vm.freeze=false ...
```

Trident Protect 24.10.1 a 25.06

A partir de Trident Protect 24.10.1, Trident Protect congela y descongela automáticamente los sistemas de archivos de KubeVirt durante las operaciones de protección de datos. Opcionalmente, puede desactivar este comportamiento automático mediante el siguiente comando:

```
kubectl set env deployment/trident-protect-controller-manager
NEPTUNE_VM_FREEZE=false -n trident-protect
```

Trident Protect 24.10

Trident Protect 24.10 no garantiza automáticamente un estado consistente para los sistemas de archivos de VM KubeVirt durante las operaciones de protección de datos. Si desea proteger los datos de su máquina virtual KubeVirt con Trident Protect 24.10, debe habilitar manualmente la funcionalidad de congelamiento/descongelamiento de los sistemas de archivos antes de la operación de protección de datos. Esto garantiza que los sistemas de archivos se encuentren en un estado consistente.

Puede configurar Trident Protect 24.10 para administrar la congelación y descongelación del sistema de archivos de la máquina virtual durante las operaciones de protección de datos."configurar la virtualización" y luego usando el siguiente comando:

```
kubectl set env deployment/trident-protect-controller-manager
NEPTUNE_VM_FREEZE=true -n trident-protect
```

Para saber cómo definir aplicaciones y restaurar máquinas virtuales con Trident Protect, consulta "Protege las máquinas virtuales de KubeVirt con Trident Protect".

Requisitos para la replicación de SnapMirror

La replicación de NetApp SnapMirror está disponible para su uso con Trident Protect para las siguientes soluciones ONTAP :

- Sistemas NetApp FAS, AFF y ASA en las instalaciones. La replicación SnapMirror con Trident protect no es compatible actualmente para los sistemas ASA r2.
- ONTAP Select de NetApp
- Cloud Volumes ONTAP de NetApp
- Amazon FSX para ONTAP de NetApp

Requisitos de clústeres de ONTAP para la replicación de SnapMirror

Asegúrese de que el clúster de ONTAP cumple los siguientes requisitos si tiene pensado utilizar la replicación de SnapMirror:

- * NetApp Trident*: NetApp Trident debe existir tanto en el clúster de Kubernetes de origen como en el de destino que utilizan ONTAP como backend. Trident Protect admite la replicación con la tecnología NetApp SnapMirror utilizando clases de almacenamiento respaldadas por los siguientes controladores:
  - `ontap-nas` : NFS
  - `ontap-san` : iSCSI
  - `ontap-san` :FC
  - `ontap-san` :NVMe/TCP (requiere la versión mínima de ONTAP 9.15.1)
- **Licencias**: Las licencias asíncronas de SnapMirror de ONTAP que utilizan el paquete de protección de datos deben estar habilitadas en los clústeres de ONTAP de origen y de destino. Consulte "Información general sobre las licencias de SnapMirror en ONTAP" si desea obtener más información.

  A partir de ONTAP 9.10.1, todas las licencias se proporcionan como archivo de licencia de NetApp (NLF), que es un solo archivo que admite varias funciones. Consulte "Licencias incluidas con ONTAP One" si

desea obtener más información.

NOTA	Sólo se admite la protección asincrónica SnapMirror.

Consideraciones sobre la relación de paridad para la replicación de SnapMirror

Compruebe que el entorno cumple los siguientes requisitos si piensa utilizar la paridad de back-end de almacenamiento:

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

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

- **NetApp Trident y SVM**: las SVM remotas emparejadas deben estar disponibles para NetApp Trident en el clúster de destino.
- **Backends administrados**: debe agregar y administrar backends de almacenamiento ONTAP en Trident Protect para crear una relación de replicación.

Configuración de Trident/ONTAP para la replicación de SnapMirror

Trident Protect requiere que configure al menos un backend de almacenamiento que admita la replicación para los clústeres de origen y destino. Si los clústeres de origen y destino son los mismos, la aplicación de destino debería utilizar un backend de almacenamiento diferente al de la aplicación de origen para lograr la mejor resiliencia.

Requisitos del clúster de Kubernetes para la replicación de SnapMirror

Asegúrese de que sus clústeres de Kubernetes cumplan con los siguientes requisitos:

- **Accesibilidad de AppVault**: tanto los clústeres de origen como de destino deben tener acceso a la red para leer y escribir en AppVault para la replicación de objetos de la aplicación.
- **Conectividad de red**: configure reglas de firewall, permisos de bucket y listas de IP permitidas para habilitar la comunicación entre ambos clústeres y AppVault a través de WAN.

NOTA	Muchos entornos empresariales implementan políticas de firewall estrictas en las conexiones WAN. Verifique estos requisitos de red con su equipo de infraestructura antes de configurar la replicación.

Instalar y configurar Trident Protect

Si su entorno cumple con los requisitos de Trident Protect, puede seguir estos pasos para instalar Trident Protect en su clúster. Puede obtener Trident Protect de NetApp o instalarlo desde su propio registro privado. Instalar desde un registro privado resulta útil si su clúster no puede acceder a Internet.

Instalar Trident Protect

Instalar Trident Protect de NetApp

Pasos

1. Añada el repositorio Helm de Trident:

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

2. Utilice Helm para instalar Trident Protect. Reemplazar `<name-of-cluster>` con un nombre de clúster, que se asignará al clúster y se utilizará para identificar las copias de seguridad y las instantáneas del clúster:

```
helm install trident-protect netapp-trident-protect/trident-protect
--set clusterName=<name-of-cluster> --version 100.2606.0 --create
-namespace --namespace trident-protect
```

3. Opcionalmente, para habilitar el registro de depuración (recomendado para la resolución de problemas), utilice:

```
helm install trident-protect netapp-trident-protect/trident-protect
--set clusterName=<name-of-cluster> --set logLevel=debug --version
100.2606.0 --create-namespace --namespace trident-protect
```

   El registro de depuración ayuda al soporte de NetApp a solucionar problemas sin necesidad de cambiar el nivel de registro ni reproducir el problema.

Instalar Trident Protect desde un registro privado

Puede instalar Trident Protect desde un registro de imágenes privado si su clúster de Kubernetes no puede acceder a Internet. En estos ejemplos, sustituya los valores entre corchetes por información de su entorno:

Pasos

1. Tire de las siguientes imágenes a su máquina local, actualice las etiquetas y, a continuación, empújelas en su registro privado:

```
docker.io/netapp/controller:26.06.0
docker.io/netapp/restic:26.06.0
docker.io/netapp/kopia:26.06.0
docker.io/netapp/kopiablockrestore:26.06.0
docker.io/netapp/trident-autosupport:26.06.0
docker.io/netapp/exechook:26.06.0
docker.io/netapp/resourcebackup:26.06.0
docker.io/netapp/resourcerestore:26.06.0
docker.io/netapp/resourcedelete:26.06.0
docker.io/netapp/trident-protect-utils:v1.0.0
```

Por ejemplo:

```
docker pull docker.io/netapp/controller:26.06.0
```

```
docker tag docker.io/netapp/controller:26.06.0 <private-registry-url>/controller:26.06.0
```

```
docker push <private-registry-url>/controller:26.06.0
```

NOTA Para obtener el gráfico Helm, primero descarga el gráfico Helm en una máquina con acceso a internet usando `helm pull trident-protect --version 100.2606.0 --repo https://netapp.github.io/trident-protect-helm-chart`, luego copia el archivo `trident-protect-100.2606.0.tgz` resultante a tu entorno sin conexión e instala usando `helm install trident-protect ./trident-protect-100.2606.0.tgz` en vez de la referencia del repositorio en el paso final.

2. Cree el espacio de nombres del sistema Trident Protect:

```
kubectl create ns trident-protect
```

3. Inicie sesión en el Registro:

```
helm registry login <private-registry-url> -u <account-id> -p <api-token>
```

4. Cree un secreto de extracción para utilizarlo en la autenticación del registro privado:

```
kubectl create secret docker-registry regcred --docker
-username=<registry-username> --docker-password=<api-token> -n
trident-protect --docker-server=<private-registry-url>
```

5. Añada el repositorio Helm de Trident:

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

6. Crea un archivo llamado `protectValues.yaml` . Asegúrese de que contenga las siguientes configuraciones de Trident Protect:

```
---
imageRegistry: <private-registry-url>
imagePullSecrets:
  - name: regcred
```

NOTA

El `imageRegistry` y `imagePullSecrets` Los valores se aplican a todas las imágenes de componentes, incluyendo `resourcebackup` y `resourcerestore` . Si envías imágenes a una ruta de repositorio específica dentro de tu registro (por ejemplo, `example.com:443/my-repo`), incluya la ruta completa en el campo del registro. Esto garantizará que todas las imágenes se extraigan de `<private-registry-url>/<image-name>:<tag>`.

7. Utilice Helm para instalar Trident Protect. Reemplazar `<name_of_cluster>` con un nombre de clúster, que se asignará al clúster y se utilizará para identificar las copias de seguridad y las instantáneas del clúster:

```
helm install trident-protect netapp-trident-protect/trident-protect
--set clusterName=<name_of_cluster> --version 100.2606.0 --create
-namespace --namespace trident-protect -f protectValues.yaml
```

8. Opcionalmente, para habilitar el registro de depuración (recomendado para la resolución de problemas), utilice:

```
helm install trident-protect netapp-trident-protect/trident-protect
--set clusterName=<name-of-cluster> --set logLevel=debug --version
100.2606.0 --create-namespace --namespace trident-protect -f
protectValues.yaml
```

El registro de depuración ayuda al soporte de NetApp a solucionar problemas sin necesidad de cambiar el nivel de registro ni reproducir el problema.

NOTA	Para obtener opciones de configuración adicionales para el gráfico de Helm, incluidos los ajustes de AutoSupport y el filtrado de espacios de nombres, consulte "Personalizar la instalación de Trident Protect".

Instalar el complemento CLI de Trident Protect

Puede utilizar el complemento de línea de comandos Trident Protect, que es una extensión de Trident `tridentctl` utilidad para crear e interactuar con recursos personalizados (CR) de Trident Protect.

Instalar el complemento CLI de Trident Protect

Antes de utilizar la utilidad de línea de comandos, debe instalarla en la máquina que utiliza para acceder al clúster. Siga estos pasos, dependiendo de si su máquina utiliza una CPU x64 o ARM.

Descargar plugin para CPU Linux AMD64

Pasos

1. Descargue el complemento CLI de Trident Protect:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/26.06.0/tridentctl-protect-linux-amd64
```

Descargar plugin para CPU Linux ARM64

Pasos

1. Descargue el complemento CLI de Trident Protect:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/26.06.0/tridentctl-protect-linux-arm64
```

Descargar plugin para CPU Mac AMD64

Pasos

1. Descargue el complemento CLI de Trident Protect:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/26.06.0/tridentctl-protect-macos-amd64
```

Descargar plugin para CPU Mac ARM64

Pasos

1. Descargue el complemento CLI de Trident Protect:

```
curl -L -o tridentctl-protect https://github.com/NetApp/tridentctl-protect/releases/download/26.06.0/tridentctl-protect-macos-arm64
```

1. Active los permisos de ejecución para el binario del plugin:

```
chmod +x tridentctl-protect
```

2. Copie el binario del plugin a una ubicación definida en su variable PATH. Por ejemplo, `/usr/bin` o `/usr/local/bin` (puede que necesite Privileges elevado):

```
cp ./tridentctl-protect /usr/local/bin/
```

3. Opcionalmente, puede copiar el binario del plugin a una ubicación en su directorio principal. En este caso, se recomienda asegurarse de que la ubicación forma parte de la variable PATH:

```
cp ./tridentctl-protect ~/bin/
```

NOTA Copiar el plugin a una ubicación en su variable PATH le permite usar el plugin escribiendo `tridentctl-protect` o `tridentctl protect` desde cualquier ubicación.

Consulte la ayuda del complemento de la CLI de Trident

Puede utilizar las funciones de ayuda del plugin incorporado para obtener ayuda detallada sobre las capacidades del plugin:

Pasos

1. Utilice la función de ayuda para ver la guía de uso:

```
tridentctl-protect help
```

Habilite el autocompletado de comandos

Después de haber instalado el complemento CLI de Trident Protect, puede habilitar el autocompletado para ciertos comandos.

Active la finalización automática del shell Bash

Pasos

1. Crea el script de finalización:

```
tridentctl-protect completion bash > tridentctl-completion.bash
```

2. Cree un nuevo directorio en el directorio principal para que contenga el script:

```
mkdir -p ~/.bash/completions
```

3. Mueva el script descargado al `~/.bash/completions` directorio:

```
mv tridentctl-completion.bash ~/.bash/completions/
```

4. Añada la siguiente línea al `~/.bashrc` archivo en su directorio principal:

```
source ~/.bash/completions/tridentctl-completion.bash
```

Active la finalización automática del shell Z

Pasos

1. Crea el script de finalización:

```
tridentctl-protect completion zsh > tridentctl-completion.zsh
```

2. Cree un nuevo directorio en el directorio principal para que contenga el script:

```
mkdir -p ~/.zsh/completions
```

3. Mueva el script descargado al `~/.zsh/completions` directorio:

```
mv tridentctl-completion.zsh ~/.zsh/completions/
```

4. Añada la siguiente línea al `~/.zprofile` archivo en su directorio principal:

```
source ~/.zsh/completions/tridentctl-completion.zsh
```

Resultado

En su próximo inicio de sesión en el shell, puede utilizar el comando auto-completado con el plugin tridentctl-Protect.

Personalizar la instalación de Trident Protect

Puede personalizar la configuración predeterminada de Trident Protect para satisfacer los requisitos específicos de su entorno.

Especificar los límites de recursos del contenedor Trident Protect

Puede utilizar un archivo de configuración para especificar límites de recursos para los contenedores de Trident Protect después de instalar Trident Protect. Establecer límites de recursos le permite controlar qué cantidad de recursos del clúster consumen las operaciones de Trident Protect.

Pasos

1. Crear un archivo llamado `resourceLimits.yaml`.
2. Complete el archivo con opciones de límite de recursos para los contenedores de Trident Protect según las necesidades de su entorno.

   El siguiente archivo de configuración de ejemplo muestra la configuración disponible y contiene los valores predeterminados para cada límite de recursos:

```
---
jobResources:
  defaults:
    limits:
      cpu: 8000m
      memory: 10000Mi
      ephemeralStorage: ""
    requests:
      cpu: 100m
      memory: 100Mi
      ephemeralStorage: ""
  resticVolumeBackup:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  resticVolumeRestore:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
```

```
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  kopiaVolumeBackup:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
  kopiaVolumeRestore:
    limits:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
    requests:
      cpu: ""
      memory: ""
      ephemeralStorage: ""
```

3. Aplique los valores del `resourceLimits.yaml` archivo:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f resourceLimits.yaml --reuse-values
```

Personalizar restricciones de contexto de seguridad

Puede utilizar un archivo de configuración para modificar la restricción de contexto de seguridad (SCC) de OpenShift para los contenedores de Trident Protect después de instalar Trident Protect. Estas restricciones definen las limitaciones de seguridad para los pods en un clúster de Red Hat OpenShift.

Pasos

1. Crear un archivo llamado `sccconfig.yaml`.
2. Agregue la opción SCC al archivo y modifique los parámetros según las necesidades de su entorno.

   El siguiente ejemplo muestra los valores predeterminados de los parámetros para la opción SCC:

```
scc:
  create: true
  name: trident-protect-job
  priority: 1
```

En esta tabla se describen los parámetros de la opción SCC:

Parámetro	Descripción	Predeterminado
cree	Determina si se puede crear un recurso SCC. Un recurso de SCC se creará sólo si `scc.create` se establece en `true` y el proceso de instalación de Helm identifica un entorno de OpenShift. Si no funciona en OpenShift, o si `scc.create` está establecido en `false`, no se creará ningún recurso SCC.	verdadero
nombre	Especifica el nombre del SCC.	Trident-protect-job
prioridad	Define la prioridad del SCC. Los SCCS con valores de prioridad más altos se evalúan antes que aquellos con valores más bajos.	1

3. Aplique los valores del `sccconfig.yaml` archivo:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f sccconfig.yaml --reuse-values
```

Esto reemplazará los valores predeterminados por los especificados en el `sccconfig.yaml` archivo.

Configurar ajustes adicionales del gráfico de timón Trident Protect

Puedes personalizar la configuración de AutoSupport, el filtrado de espacios de nombres y la gestión de certificados de webhooks de OpenShift para adaptarlos a tus necesidades específicas. En la siguiente tabla se describen los parámetros de configuración disponibles:

Parámetro	Tipo	Descripción
autoSupport.proxy	cadena	Configura una URL de proxy para conexiones de NetApp AutoSupport . Use esto para enrutar las cargas de paquetes de soporte a través de un servidor proxy. Ejemplo: `http://my.proxy.url` .
autoSupport.inseguro	booleano	Omite la verificación TLS para las conexiones proxy de AutoSupport cuando se configura en `true` . Úselo sólo para conexiones proxy inseguras. (por defecto: `false`)

Parámetro	Tipo	Descripción
autoSupport.enabled	booleano	Habilita o deshabilita las cargas diarias de paquetes de AutoSupport de Trident Protect. Cuando se configura para `false` Las cargas diarias programadas están desactivadas, pero aún puede generar manualmente paquetes de soporte. (por defecto: `true`)
restaurar anotaciones de SkipNamespace	cadena	Lista separada por comas de anotaciones de espacios de nombres para excluir de las operaciones de copia de seguridad y restauración. Le permite filtrar espacios de nombres según anotaciones.
restaurar etiquetas de espacios de nombres de saltos	cadena	Lista separada por comas de etiquetas de espacios de nombres para excluir de las operaciones de copia de seguridad y restauración. Le permite filtrar espacios de nombres según etiquetas.
useOpenshiftServiceCA	booleano	Permite la integración con el Service CA Operator nativo de OpenShift para la creación e inyección automáticas de certificados TLS para el webhook de admisión de Trident Protect. Cuando se establece en `true`, el Service CA Operator gestiona automáticamente los certificados TLS del webhook. Úsalo solo en clústeres de OpenShift. (predeterminado: `false`)

Puede configurar estas opciones mediante un archivo de configuración YAML o indicadores de línea de comandos:

Utilice el archivo YAML

Pasos

1. Crea un archivo de configuración y nómbralo `values.yaml` .
2. En el archivo que ha creado, agregue las opciones de configuración que desea personalizar.

```
autoSupport:
  enabled: false
  proxy: http://my.proxy.url
  insecure: true
restoreSkipNamespaceAnnotations: "annotation1,annotation2"
restoreSkipNamespaceLabels: "label1,label2"
useOpenshiftServiceCA: true
```

3. Después de rellenar el `values.yaml` archivo con los valores correctos, aplique el archivo de configuración:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f values.yaml --reuse-values
```

Usar la bandera CLI

Pasos

1. Utilice el siguiente comando con el `--set` bandera para especificar parámetros individuales:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect \
  --set autoSupport.enabled=false \
  --set autoSupport.proxy=http://my.proxy.url \
  --set-string
restoreSkipNamespaceAnnotations="{annotation1,annotation2}" \
  --set-string restoreSkipNamespaceLabels="{label1,label2}" \
  --set useOpenshiftServiceCA=true \
  --reuse-values
```

Restringir los pods de Trident Protect a nodos específicos

Puede utilizar la restricción de selección de nodos nodeSelector de Kubernetes para controlar cuáles de sus nodos son elegibles para ejecutar pods de Trident Protect, según las etiquetas de los nodos. De forma predeterminada, Trident Protect está restringido a los nodos que ejecutan Linux. Puedes personalizar aún más estas restricciones según tus necesidades.

Pasos

1. Crear un archivo llamado `nodeSelectorConfig.yaml`.
2. Agregue la opción nodeSelector al archivo y modifique el archivo para agregar o cambiar etiquetas de nodo para restringir según las necesidades del entorno. Por ejemplo, el siguiente archivo contiene la restricción predeterminada del sistema operativo, pero también se dirige a una región y un nombre de aplicación específicos:

```
nodeSelector:
  kubernetes.io/os: linux
  region: us-west
  app.kubernetes.io/name: mysql
```

3. Aplique los valores del `nodeSelectorConfig.yaml` archivo:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect -f nodeSelectorConfig.yaml --reuse-values
```

Esto reemplaza las restricciones predeterminadas por las especificadas en el `nodeSelectorConfig.yaml` archivo.

Administrar Trident Protect

Administrar la autorización y el control de acceso de Trident Protect

Trident Protect utiliza el modelo Kubernetes de control de acceso basado en roles (RBAC). De forma predeterminada, Trident Protect proporciona un único espacio de nombres de sistema y su cuenta de servicio predeterminada asociada. Si tiene una organización con muchos usuarios o necesidades de seguridad específicas, puede utilizar las funciones RBAC de Trident Protect para obtener un control más granular sobre el acceso a los recursos y espacios de nombres.

El administrador de clúster siempre tiene acceso a los recursos del espacio de nombres predeterminado `trident-protect` y también puede acceder a los recursos en el resto de espacios de nombres. Para controlar el acceso a recursos y aplicaciones, es necesario crear espacios de nombres adicionales y agregar recursos y aplicaciones a esos espacios de nombres.

Tenga en cuenta que ningún usuario puede crear CRS de gestión de datos de aplicaciones en el espacio de nombres predeterminado `trident-protect`. Debe crear CRS de gestión de datos de aplicaciones en un espacio de nombres de aplicaciones (como práctica recomendada, crear CRS de gestión de datos de aplicaciones en el mismo espacio de nombres que la aplicación asociada).

NOTA

Solo los administradores deben tener acceso a los objetos de recursos personalizados privilegiados de Trident Protect, que incluyen:

- **AppVault**: Requiere datos de credenciales de bucket
- **AutoSupportBundle**: Recopila métricas, registros y otros datos confidenciales de Trident Protect
- **BundleSchedule**: Gestiona los horarios de recopilación de registros

Como práctica recomendada, use RBAC para restringir el acceso a los objetos con privilegios a los administradores.

Para obtener más información sobre cómo el RBAC regula el acceso a los recursos y espacios de nombres, consulte la "Documentación de RBAC de Kubernetes".

Para obtener información sobre las cuentas de servicio, consulte la "Documentación de la cuenta de servicio de Kubernetes".

Ejemplo: Administrar el acceso para dos grupos de usuarios

Por ejemplo, una organización tiene un administrador de clústeres, un grupo de usuarios de ingeniería y un grupo de usuarios de marketing. El administrador del clúster debe realizar las siguientes tareas para crear un entorno en el que el grupo de ingeniería y el grupo de marketing tengan acceso solo a los recursos asignados a sus respectivos espacios de nombres.

Paso 1: Crear un espacio de nombres para contener recursos para cada grupo

La creación de un espacio de nombres permite separar los recursos de forma lógica y controlar mejor quién tiene acceso a dichos recursos.

Pasos

1. Cree un espacio de nombres para el grupo de ingeniería:

```
kubectl create ns engineering-ns
```

2. Cree un espacio de nombres para el grupo de marketing:

```
kubectl create ns marketing-ns
```

Paso 2: Crear nuevas cuentas de servicio para interactuar con los recursos de cada espacio de nombres

Cada nuevo espacio de nombres que cree viene con una cuenta de servicio predeterminada, pero debe crear una cuenta de servicio para cada grupo de usuarios para que pueda dividir aún más Privileges entre grupos en el futuro si es necesario.

Pasos

1. Cree una cuenta de servicio para el grupo de ingeniería:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: eng-user
  namespace: engineering-ns
```

2. Cree una cuenta de servicio para el grupo de marketing:

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: mkt-user
  namespace: marketing-ns
```

Paso 3: Crear un secreto para cada nueva cuenta de servicio

Un secreto de cuenta de servicio se utiliza para autenticarse con la cuenta de servicio, y se puede eliminar y volver a crear fácilmente si está comprometido.

Pasos

1. Cree un secreto para la cuenta de servicio de ingeniería:

```
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: eng-user
  name: eng-user-secret
  namespace: engineering-ns
type: kubernetes.io/service-account-token
```

2. Cree un secreto para la cuenta de servicio de marketing:

```
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: mkt-user
  name: mkt-user-secret
  namespace: marketing-ns
type: kubernetes.io/service-account-token
```

Paso 4: Cree un objeto RoleBinding para enlazar el objeto ClusterRole a cada nueva cuenta de servicio

Se crea un objeto ClusterRole predeterminado cuando instala Trident Protect. Puede vincular este ClusterRole a la cuenta de servicio creando y aplicando un objeto RoleBinding.

Pasos

1. Enlazar ClusterRole a la cuenta de servicio de ingeniería:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: engineering-ns-tenant-rolebinding
  namespace: engineering-ns
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: trident-protect-tenant-cluster-role
subjects:
- kind: ServiceAccount
  name: eng-user
  namespace: engineering-ns
```

2. Enlazar ClusterRole a la cuenta de servicio de marketing:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: marketing-ns-tenant-rolebinding
  namespace: marketing-ns
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: trident-protect-tenant-cluster-role
subjects:
- kind: ServiceAccount
  name: mkt-user
  namespace: marketing-ns
```

Paso 5: Probar permisos

Compruebe que los permisos son correctos.

Pasos

1. Confirme que los usuarios de ingeniería pueden acceder a los recursos de ingeniería:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get applications.protect.trident.netapp.io -n engineering-ns
```

2. Confirme que los usuarios de ingeniería no pueden acceder a los recursos de marketing:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get applications.protect.trident.netapp.io -n marketing-ns
```

Paso 6: Otorgar acceso a los objetos de AppVault

Para realizar tareas de gestión de datos, como backups e instantáneas, el administrador de clúster debe conceder acceso a los objetos de AppVault a usuarios individuales.

Pasos

1. Cree y aplique un archivo YAML de combinación secreta y AppVault que otorgue acceso a un usuario a un AppVault. Por ejemplo, el siguiente CR otorga acceso a un AppVault al usuario `eng-user`:

```
apiVersion: v1
data:
  accessKeyID: <ID_value>
  secretAccessKey: <key_value>
kind: Secret
metadata:
  name: appvault-for-eng-user-only-secret
  namespace: trident-protect
type: Opaque
---
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: appvault-for-eng-user-only
  namespace: trident-protect # Trident Protect system namespace
spec:
  providerConfig:
    azure:
      accountName: ""
      bucketName: ""
      endpoint: ""
    gcp:
      bucketName: ""
      projectID: ""
    s3:
      bucketName: testbucket
      endpoint: 192.168.0.1:30000
      secure: "false"
      skipCertValidation: "true"
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: appvault-for-eng-user-only-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: appvault-for-eng-user-only-secret
  providerType: GenericS3
```

2. Cree y aplique un CR de rol para permitir que los administradores del cluster concedan acceso a recursos específicos en un espacio de nombres. Por ejemplo:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: eng-user-appvault-reader
  namespace: trident-protect
rules:
- apiGroups:
  - protect.trident.netapp.io
  resourceNames:
  - appvault-for-enguser-only
  resources:
  - appvaults
  verbs:
  - get
```

3. Cree y aplique un CR de RoleBinding para enlazar los permisos al usuario eng-user. Por ejemplo:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: eng-user-read-appvault-binding
  namespace: trident-protect
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: eng-user-appvault-reader
subjects:
- kind: ServiceAccount
  name: eng-user
  namespace: engineering-ns
```

4. Compruebe que los permisos son correctos.

   a. Se ha intentado recuperar la información del objeto AppVault para todos los espacios de nombres:

   ```
   kubectl get appvaults -n trident-protect
   --as=system:serviceaccount:engineering-ns:eng-user
   ```

   Debería ver una salida similar a la siguiente:

```
Error from server (Forbidden): appvaults.protect.trident.netapp.io is
forbidden: User "system:serviceaccount:engineering-ns:eng-user"
cannot list resource "appvaults" in API group
"protect.trident.netapp.io" in the namespace "trident-protect"
```

b. Prueba para ver si el usuario puede obtener la información de AppVault a la que ahora tiene permiso para acceder:

```
kubectl auth can-i --as=system:serviceaccount:engineering-ns:eng-user
get appvaults.protect.trident.netapp.io/appvault-for-eng-user-only -n
trident-protect
```

Debería ver una salida similar a la siguiente:

```
yes
```

Resultado

Los usuarios a los que ha otorgado permisos de AppVault deben poder usar objetos de AppVault autorizados para operaciones de gestión de datos de aplicaciones y no deben poder acceder a ningún recurso fuera de los espacios de nombres asignados ni crear nuevos recursos a los que no tengan acceso.

Supervisar los recursos de Trident Protect

Puede utilizar las herramientas de código abierto kube-state-metrics, Prometheus y Alertmanager para supervisar el estado de los recursos protegidos por Trident Protect.

El servicio kube-state-metrics genera métricas a partir de la comunicación con la API de Kubernetes. Usarlo con Trident Protect expone información útil sobre el estado de los recursos en su entorno.

Prometheus es un conjunto de herramientas que puede ingerir los datos generados por kube-state-metrics y presentarlos como información fácilmente legible sobre estos objetos. Juntos, kube-state-metrics y Prometheus le brindan una manera de monitorear la salud y el estado de los recursos que administra con Trident Protect.

Alertmanager es un servicio que ingiere las alertas enviadas por herramientas como Prometheus y las redirige a los destinos que configure.

NOTA

Las configuraciones y directrices incluidas en estos pasos son solo ejemplos; debe personalizarlas para que se adapten a su entorno. Consulte la siguiente documentación oficial para obtener instrucciones específicas y soporte:

- "documentación sobre métricas del estado de kube"
- "Documentación de Prometheus"
- "Documentación de Alertmanager"

Paso 1: Instale las herramientas de monitoreo

Para habilitar la supervisión de recursos en Trident Protect, debe instalar y configurar kube-state-metrics, Promethus y Alertmanager.

Instale kube-state-metrics

Puede instalar kube-state-metrics usando Helm.

Pasos

1. Agregue el gráfico Helm de métricas de estado-kube. Por ejemplo:

```
helm repo add prometheus-community https://prometheus-
community.github.io/helm-charts
helm repo update
```

2. Aplique el CRD de Prometheus ServiceMonitor al clúster:

```
kubectl apply -f https://raw.githubusercontent.com/prometheus-
operator/prometheus-operator/main/example/prometheus-operator-
crd/monitoring.coreos.com_servicemonitors.yaml
```

3. Cree un archivo de configuración para el diagrama Helm (por ejemplo, `metrics-config.yaml`). Puede personalizar la siguiente configuración de ejemplo para que coincida con su entorno:

Metrics-config.yaml: Configuración del gráfico Helm de métricas de estado de kube

```
---
extraArgs:
  # Collect only custom metrics
  - --custom-resource-state-only=true

customResourceState:
  enabled: true
  config:
    kind: CustomResourceStateMetrics
    spec:
      resources:
      - groupVersionKind:
          group: protect.trident.netapp.io
          kind: "Backup"
          version: "v1"
        labelsFromPath:
          backup_uid: [metadata, uid]
          backup_name: [metadata, name]
          creation_time: [metadata, creationTimestamp]
        metrics:
        - name: backup_info
          help: "Exposes details about the Backup state"
          each:
            type: Info
            info:
              labelsFromPath:
                appVaultReference: ["spec", "appVaultRef"]
                appReference: ["spec", "applicationRef"]
rbac:
  extraRules:
  - apiGroups: ["protect.trident.netapp.io"]
    resources: ["backups"]
    verbs: ["list", "watch"]

# Collect metrics from all namespaces
namespaces: ""

# Ensure that the metrics are collected by Prometheus
prometheus:
  monitor:
    enabled: true
```

4. Instale kube-state-metrics mediante el despliegue del gráfico Helm. Por ejemplo:

```
helm install custom-resource -f metrics-config.yaml prometheus-
community/kube-state-metrics --version 5.21.0
```

5. Configure kube-state-metrics para generar métricas para los recursos personalizados utilizados por Trident Protect siguiendo las instrucciones en "documentación de recursos personalizados de kube-state-metrics" .

Instale Prometheus

Puede instalar Prometheus siguiendo las instrucciones del "Documentación de Prometheus" .

Instale Alertmanager

Puede instalar Alertmanager siguiendo las instrucciones del "Documentación de Alertmanager" .

Paso 2: Configure las herramientas de monitoreo para que funcionen juntas

Después de instalar las herramientas de supervisión, debe configurarlas para que funcionen juntas.

Pasos

1. Integre métricas de estado-kube con Prometheus. Edite el archivo de configuración Prometheus (`prometheus.yaml`) y agregue la información del servicio kube-state-metrics. Por ejemplo:

   prometheus.yaml: integración del servicio kube-state-metrics con Prometheus

```
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: prometheus-config
  namespace: trident-protect
data:
  prometheus.yaml: |
    global:
      scrape_interval: 15s
    scrape_configs:
      - job_name: 'kube-state-metrics'
        static_configs:
          - targets: ['kube-state-metrics.trident-protect.svc:8080']
```

2. Configurar Prometheus para enrutar alertas a Alertmanager. Edite el archivo de configuración de Prometheus (`prometheus.yaml`) y agregue la siguiente sección:

prometheus.yaml: Enviar alertas a Alertmanager

```
alerting:
  alertmanagers:
    - static_configs:
        - targets:
            - alertmanager.trident-protect.svc:9093
```

Resultado

Ahora Prometheus puede recopilar métricas de kube-state-metrics y puede enviar alertas a Alertmanager. Ahora está listo para configurar qué condiciones desencadenan una alerta y dónde se deben enviar las alertas.

Paso 3: Configure las alertas y los destinos de alertas

Después de configurar las herramientas para que funcionen juntas, debe configurar qué tipo de información activa alertas y dónde se deben enviar las alertas.

Ejemplo de alerta: Fallo de backup

En el siguiente ejemplo se define una alerta crucial que se activa cuando el estado del recurso personalizado de backup se establece en `Error` 5 segundos o más. Puede personalizar este ejemplo para que coincida con su entorno e incluir este fragmento de YAML en su `prometheus.yaml` archivo de configuración:

rules.yaml: Define una alerta de Prometheus para copias de seguridad fallidas

```
rules.yaml: |
  groups:
    - name: fail-backup
        rules:
          - alert: BackupFailed
            expr: kube_customresource_backup_info{status="Error"}
            for: 5s
            labels:
              severity: critical
            annotations:
              summary: "Backup failed"
              description: "A backup has failed."
```

Configure Alertmanager para que envíe alertas a otros canales

Puede configurar Alertmanager para que envíe notificaciones a otros canales, como correo electrónico, PagerDuty, Microsoft Teams u otros servicios de notificación especificando la configuración respectiva en `alertmanager.yaml` el archivo.

El siguiente ejemplo configura Alertmanager para enviar notificaciones a un canal de Slack. Para personalizar este ejemplo a su entorno, reemplace el valor de `api_url` la clave por la URL del webhook de Slack utilizada en su entorno:

alertmanager.yaml: envía alertas a un canal de Slack

```
data:
  alertmanager.yaml: |
    global:
      resolve_timeout: 5m
    route:
      receiver: 'slack-notifications'
    receivers:
      - name: 'slack-notifications'
        slack_configs:
          - api_url: '<your-slack-webhook-url>'
            channel: '#failed-backups-channel'
            send_resolved: false
```

Generar un paquete de soporte de Trident Protect

Trident Protect permite a los administradores generar paquetes que incluyen información útil para el soporte de NetApp , incluidos registros, métricas e información de topología sobre los clústeres y las aplicaciones bajo administración. Si está conectado a Internet, puede cargar paquetes de soporte en el sitio de soporte de NetApp (NSS) mediante un archivo de recursos personalizados (CR).

Cree un paquete de soporte mediante un CR

Pasos

1. Cree el archivo de recursos personalizados (CR) y asígnele un nombre (por ejemplo, `trident-protect-support-bundle.yaml`).
2. Configure los siguientes atributos:
   - **metadata.name**: (*required*) El nombre de este recurso personalizado; elija un nombre único y sensible para su entorno.
   - **Spec.triggerType**: (*required*) Determina si el paquete de soporte se genera inmediatamente o se programa. La generación de paquetes programada se tiene lugar a LAS 12am UTC. Los posibles valores son los siguientes:
     - Programado
     - Manual
   - **SPEC.uploadEnabled**: (*Opcional*) Controla si el paquete de soporte debe cargarse en el sitio de soporte de NetApp después de que se genere. Si no se especifica, el valor por defecto es `false`. Los posibles valores son los siguientes:
     - verdadero
     - false (predeterminado)
   - **Spec.dataWindowStart**: (*Optional*) Una cadena de fecha en formato RFC 3339 que especifica la fecha y la hora en que debe comenzar la ventana de datos incluidos en el paquete de soporte. Si no se especifica, el valor predeterminado es hace 24 horas. La fecha de ventana más antigua que puede especificar es hace 7 días.

     Ejemplo YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: AutoSupportBundle
metadata:
  name: trident-protect-support-bundle
spec:
  triggerType: Manual
  uploadEnabled: true
  dataWindowStart: 2024-05-05T12:30:00Z
```

3. Después de rellenar el `trident-protect-support-bundle.yaml` archivo con los valores correctos, aplicar la CR:

```
kubectl apply -f trident-protect-support-bundle.yaml -n trident-protect
```

Cree un bundle de soporte mediante la CLI

Pasos

1. Cree el paquete de soporte, reemplazando valores entre paréntesis con la información del entorno. `trigger-type`Determina si el grupo se crea inmediatamente o si la hora de creación está determinada por la programación, y puede ser `Manual` o. `Scheduled` El valor predeterminado es `Manual`.

   Por ejemplo:

   ```
   tridentctl-protect create autosupportbundle <my-bundle-name>
   --trigger-type <trigger-type> -n trident-protect
   ```

Supervisar y recuperar el paquete de soporte

Después de crear un paquete de soporte utilizando cualquiera de los métodos, puede supervisar el progreso de su generación y recuperarlo en su sistema local.

Pasos

1. Espera a que `status.generationState` Para alcanzar `Completed` estado. Puede monitorear el progreso de la generación con el siguiente comando:

   ```
   kubectl get autosupportbundle trident-protect-support-bundle -n trident-protect
   ```

2. Recupere el paquete de soporte para su sistema local. Obtenga el comando de copia del paquete AutoSupport completado:

   ```
   kubectl describe autosupportbundle trident-protect-support-bundle -n trident-protect
   ```

   Encuentra el `kubectl cp` comando desde la salida y ejecútelo, reemplazando el argumento de destino con su directorio local preferido.

Actualizar Trident Protect

Puede actualizar Trident Protect a la última versión para aprovechar nuevas funciones o correcciones de errores.

NOTA

- Al actualizar desde la versión 24.10, las instantáneas que se ejecutan durante la actualización podrían fallar. Este fallo no impide la creación de instantáneas futuras, ya sean manuales o programadas. Si una instantánea falla durante la actualización, puede crear una nueva manualmente para garantizar la protección de su aplicación.

  Para evitar posibles fallos, puede desactivar todas las instantáneas programadas antes de la actualización y volver a activarlas después. Sin embargo, esto puede provocar que se pierdan las instantáneas programadas durante el periodo de actualización.

- Para instalaciones en registros privados, asegúrese de que el gráfico Helm y las imágenes necesarias para la versión de destino estén disponibles en su registro privado y verifique que sus valores Helm personalizados sean compatibles con la nueva versión del gráfico. Para obtener más información, consulte"Instalar Trident Protect desde un registro privado" .

Paso 1: Seleccione una versión

Las versiones de Trident Protect siguen una convención de nomenclatura `YY.MM` basada en fechas, donde "YY" son los dos últimos dígitos del año y "MM" es el mes. Las versiones de punto siguen una `YY.MM.X` convención, donde "X" es el nivel de parche. Vas a seleccionar la versión a la que quieres actualizar según la versión desde la que estás actualizando.

- Puedes hacer una actualización directa a cualquier versión de destino que esté dentro de una ventana de cuatro versiones respecto a la versión que tienes instalada. Por ejemplo, puedes actualizar directamente de 24.10 (o cualquier versión puntual de 24.10) a 25.10.
- Si estás actualizando desde una versión fuera del periodo de cuatro versiones, haz una actualización en varios pasos. Usa las instrucciones de actualización para la "versión anterior" de la que estás actualizando para pasar a la versión más reciente que entre en el periodo de cuatro versiones. Por ejemplo, si estás usando la versión 24.10 y quieres actualizar a la 26.02:
  a. Primero actualiza de 24.10 a 25.02.
  b. Luego actualiza de 25.02 a 26.02.

Paso 2: actualizar Trident Protect

Para actualizar Trident Protect, realice los siguientes pasos.

Pasos

1. Actualice el repositorio de Trident Helm:

```
helm repo update
```

2. Actualice los CRD Trident Protect:

NOTA Este paso es necesario si estás actualizando desde una versión anterior a la 25.06, ya que los CRD ahora están incluidos en la tabla de Trident Protect Helm.

   a. Ejecute este comando para cambiar la administración de CRD de `trident-protect-crds` a `trident-protect` :

```
kubectl get crd | grep protect.trident.netapp.io | awk '{print $1}' |
xargs -I {} kubectl patch crd {} --type merge -p '{"metadata":
{"annotations":{"meta.helm.sh/release-name": "trident-protect"}}}'
```

b. Ejecute este comando para eliminar el secreto de Helm para el `trident-protect-crds` cuadro:

NOTA No desinstale el `trident-protect-crds` Gráfico usando Helm, ya que esto podría eliminar sus CRD y cualquier dato relacionado.

```
kubectl delete secret -n trident-protect -l name=trident-protect-
crds,owner=helm
```

3. Mejora Trident Protect:

```
helm upgrade trident-protect netapp-trident-protect/trident-protect
--version 100.2606.0 --namespace trident-protect
```

NOTA Puede configurar el nivel de registro durante la actualización añadiendo `--set logLevel=debug` al comando de actualización. El nivel de registro predeterminado es `warn` . Se recomienda el registro de depuración para la resolución de problemas, ya que ayuda al soporte de NetApp a diagnosticar problemas sin necesidad de cambios en el nivel de registro ni de reproducir el problema.

Gestione y proteja aplicaciones

Utilice objetos Trident Protect AppVault para administrar depósitos

El recurso personalizado (CR) de depósito para Trident Protect se conoce como AppVault. Los objetos AppVault son la representación declarativa del flujo de trabajo de Kubernetes de un depósito de almacenamiento. Un CR de AppVault contiene las configuraciones necesarias para que un bucket se utilice en operaciones de protección, como copias de seguridad, instantáneas, operaciones de restauración y replicación de SnapMirror . Solo los administradores pueden crear AppVaults.

Debe crear una CR de AppVault manualmente o desde la línea de comandos al realizar operaciones de protección de datos en una aplicación. La CR de AppVault es específica de su entorno, y puede usar los ejemplos de esta página como guía al crearlas.

NOTA Asegúrese de que AppVault CR esté en el clúster donde está instalado Trident Protect. Si el CR de AppVault no existe o no puede acceder a él, la línea de comandos mostrará un error.

Configurar la autenticación y las contraseñas de AppVault

Antes de crear un CR de AppVault, asegúrese de que AppVault y el transportador de datos que elija puedan autenticarse con el proveedor y cualquier recurso relacionado.

Contraseñas del repositorio de Data Mover

Cuando crea objetos de AppVault utilizando CR o el complemento CLI de Trident Protect, puede especificar un secreto de Kubernetes con contraseñas personalizadas para el cifrado de Restic y Kopia. Si no especifica un secreto, Trident Protect utiliza una contraseña predeterminada.

- Al crear manualmente CR de AppVault, utilice el campo **spec.dataMoverPasswordSecretRef** para especificar el secreto.
- Al crear objetos de AppVault mediante la CLI de Trident Protect, utilice el `--data-mover-password-secret-ref` argumento para especificar el secreto.

Cree un secreto de contraseña del repositorio de Data Mover

Utilice los siguientes ejemplos para crear la contraseña secreta. Cuando crea objetos de AppVault, puede indicarle a Trident Protect que use este secreto para autenticarse con el repositorio de transferencia de datos.

NOTA

- Dependiendo de qué transmisor de datos esté utilizando, solo necesita incluir la contraseña correspondiente para ese transmisor de datos. Por ejemplo, si está utilizando Restic y no planea usar KOPIA en el futuro, puede incluir solo la contraseña de Restic cuando cree el secreto.
- Guarde la contraseña en un lugar seguro. La necesitará para restaurar datos en el mismo clúster o en uno diferente. Si el clúster o el… `trident-protect` Si se elimina el espacio de nombres, no podrá restaurar sus copias de seguridad o instantáneas sin la contraseña.

Utilice un CR

```
---
apiVersion: v1
data:
  KOPIA_PASSWORD: <base64-encoded-password>
  RESTIC_PASSWORD: <base64-encoded-password>
kind: Secret
metadata:
  name: my-optional-data-mover-secret
  namespace: trident-protect
type: Opaque
```

Utilice la CLI

```
kubectl create secret generic my-optional-data-mover-secret \
--from-literal=KOPIA_PASSWORD=<plain-text-password> \
--from-literal=RESTIC_PASSWORD=<plain-text-password> \
-n trident-protect
```

Permisos de IAM de almacenamiento compatibles con S3

Cuando accede a almacenamiento compatible con S3, como Amazon S3, Generic S3, "StorageGRID S3" , o "ONTAP S3" Al utilizar Trident Protect, debe asegurarse de que las credenciales de usuario que proporcione tengan los permisos necesarios para acceder al depósito. El siguiente es un ejemplo de una política que otorga los permisos mínimos requeridos para acceder con Trident Protect. Puede aplicar esta política al usuario que administra las políticas de bucket compatibles con S3.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:ListBucket",
        "s3:DeleteObject"
      ],
      "Resource": "*"
    }
  ]
}
```

Para obtener más información sobre las políticas de Amazon S3, consulte los ejemplos en la "Documentación de Amazon S3" .

Identidad de pod EKS para la autenticación de Amazon S3 (AWS)

Trident Protect admite EKS Pod Identity para operaciones de transferencia de datos de Kopia. Esta función permite el acceso seguro a los buckets S3 sin almacenar las credenciales de AWS en los secretos de Kubernetes.

Requisitos para la identidad del pod EKS con Trident Protect

Antes de utilizar EKS Pod Identity con Trident Protect, asegúrese de lo siguiente:

- Su clúster EKS tiene Identidad de Pod habilitada.
- Ha creado una función de IAM con los permisos de bucket S3 necesarios. Para obtener más información, consulte"Permisos de IAM de almacenamiento compatibles con S3" .
- La función IAM está asociada con las siguientes cuentas de servicio de Trident Protect:
  - `<trident-protect>-controller-manager`
  - `<trident-protect>-resource-backup`
  - `<trident-protect>-resource-restore`
  - `<trident-protect>-resource-delete`

Para obtener instrucciones detalladas sobre cómo habilitar Pod Identity y asociar roles de IAM con cuentas de servicio, consulte la "Documentación de identidad de pod de AWS EKS" .

Configuración de AppVault Al usar EKS Pod Identity, configure su AppVault CR con la `useIAM: true` bandera en lugar de credenciales explícitas:

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: eks-protect-vault
  namespace: trident-protect
spec:
  providerType: AWS
  providerConfig:
    s3:
      bucketName: trident-protect-aws
      endpoint: s3.example.com
      useIAM: true
```

Ejemplos de generación de claves de AppVault para proveedores de cloud

Al definir un CR de AppVault, debe incluir credenciales para acceder a los recursos alojados por el proveedor, a menos que utilice la autenticación IAM. La forma de generar las claves para las credenciales variará según el proveedor. Los siguientes son ejemplos de generación de claves de línea de comandos para varios proveedores. Puede utilizar los siguientes ejemplos para crear claves para las credenciales de cada proveedor de nube.

Google Cloud

```
kubectl create secret generic <secret-name> \
--from-file=credentials=<mycreds-file.json> \
-n trident-protect
```

Amazon S3 (AWS)

```
kubectl create secret generic <secret-name> \
--from-literal=accessKeyID=<objectstorage-accesskey> \
--from-literal=secretAccessKey=<amazon-s3-trident-protect-src-bucket
-secret> \
-n trident-protect
```

Microsoft Azure

```
kubectl create secret generic <secret-name> \
--from-literal=accountKey=<secret-name> \
-n trident-protect
```

Genérico S3

```
kubectl create secret generic <secret-name> \
--from-literal=accessKeyID=<objectstorage-accesskey> \
--from-literal=secretAccessKey=<generic-s3-trident-protect-src-bucket
-secret> \
-n trident-protect
```

ONTAP S3

```
kubectl create secret generic <secret-name> \
--from-literal=accessKeyID=<objectstorage-accesskey> \
--from-literal=secretAccessKey=<ontap-s3-trident-protect-src-bucket
-secret> \
-n trident-protect
```

StorageGRID S3

```
kubectl create secret generic <secret-name> \
--from-literal=accessKeyID=<objectstorage-accesskey> \
--from-literal=secretAccessKey=<storagegrid-s3-trident-protect-src
-bucket-secret> \
-n trident-protect
```

Ejemplos de creación de AppVault

A continuación se muestran ejemplos de definiciones de AppVault para cada proveedor.

Ejemplos de AppVault CR

Puede utilizar los siguientes ejemplos de CR para crear objetos de AppVault para cada proveedor de cloud.

NOTA

- Opcionalmente, puede especificar un secreto de Kubernetes que contenga contraseñas personalizadas para el cifrado del repositorio de Restic y KOPIA. Consulte Contraseñas del repositorio de Data Mover si desea obtener más información.
- Para los objetos AppVault de Amazon S3 (AWS), puede especificar opcionalmente un sessionToken, lo que resulta útil si utiliza el inicio de sesión único (SSO) para la autenticación. Este token se crea cuando se generan claves para el proveedor en Ejemplos de generación de claves de AppVault para proveedores de cloud.
- Para los objetos S3 AppVault, puede especificar opcionalmente una URL de proxy de salida para el tráfico S3 saliente mediante la `spec.providerConfig.S3.proxyURL` clave.

Google Cloud

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: gcp-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: GCP
  providerConfig:
    gcp:
      bucketName: trident-protect-src-bucket
      projectID: project-id
  providerCredentials:
    credentials:
      valueFromSecret:
        key: credentials
        name: gcp-trident-protect-src-bucket-secret
```

Amazon S3 (AWS)

```
---
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: amazon-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: AWS
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
    sessionToken:
      valueFromSecret:
        key: sessionToken
        name: s3-secret
```

NOTA Para entornos EKS que utilizan Pod Identity con el transportador de datos Kopia, puede eliminar el `providerCredentials` sección y agregar `useIAM: true` bajo el `s3` configuración en su lugar.

Microsoft Azure

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: azure-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: Azure
  providerConfig:
    azure:
      accountName: account-name
      bucketName: trident-protect-src-bucket
  providerCredentials:
    accountKey:
      valueFromSecret:
        key: accountKey
        name: azure-trident-protect-src-bucket-secret
```

Genérico S3

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: generic-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: GenericS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
```

ONTAP S3

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: ontap-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: OntapS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
```

StorageGRID S3

```
apiVersion: protect.trident.netapp.io/v1
kind: AppVault
metadata:
  name: storagegrid-s3-trident-protect-src-bucket
  namespace: trident-protect
spec:
  dataMoverPasswordSecretRef: my-optional-data-mover-secret
  providerType: StorageGridS3
  providerConfig:
    s3:
      bucketName: trident-protect-src-bucket
      endpoint: s3.example.com
      proxyURL: http://10.1.1.1:3128
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        key: accessKeyID
        name: s3-secret
    secretAccessKey:
      valueFromSecret:
        key: secretAccessKey
        name: s3-secret
```

Ejemplos de creación de AppVault mediante la CLI de Trident Protect

Puede utilizar los siguientes ejemplos de comandos CLI para crear AppVault CRS para cada proveedor.

NOTA

- Opcionalmente, puede especificar un secreto de Kubernetes que contenga contraseñas personalizadas para el cifrado del repositorio de Restic y KOPIA. Consulte Contraseñas del repositorio de Data Mover si desea obtener más información.
- Para los objetos S3 AppVault, puede especificar opcionalmente una URL de proxy de salida para el tráfico S3 saliente mediante el `--proxy-url <ip_address:port>` argumento.

Google Cloud

```
tridentctl-protect create vault GCP <vault-name> \
--bucket <mybucket> \
--project <my-gcp-project> \
--secret <secret-name>/credentials \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

Amazon S3 (AWS)

```
tridentctl-protect create vault AWS <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

Microsoft Azure

```
tridentctl-protect create vault Azure <vault-name> \
--account <account-name> \
--bucket <bucket-name> \
--secret <secret-name> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

Genérico S3

```
tridentctl-protect create vault GenericS3 <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

ONTAP S3

```
tridentctl-protect create vault OntapS3 <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

StorageGRID S3

```
tridentctl-protect create vault StorageGridS3 <vault-name> \
--bucket <bucket-name> \
--secret  <secret-name>  \
--endpoint <s3-endpoint> \
--data-mover-password-secret-ref <my-optional-data-mover-secret> \
-n trident-protect
```

Apoyado `providerConfig.s3` opciones de configuración

Consulte la siguiente tabla para conocer las opciones de configuración del proveedor S3:

Parámetro	Descripción	Predeterminado	Ejemplo
`providerConfig.s3.skipCertValidation`	Deshabilitar la verificación del certificado SSL/TLS.	falso	"verdadero", "falso"
`providerConfig.s3.secure`	Habilite la comunicación HTTPS segura con el punto final S3.	verdadero	"verdadero", "falso"
`providerConfig.s3.proxyURL`	Especifique la URL del servidor proxy utilizado para conectarse a S3.	Ninguno	http://proxy.example.com:8080
`providerConfig.s3.rootCA`	Proporcione un certificado CA raíz personalizado para la verificación SSL/TLS.	Ninguno	"CN=MiCAPersonalizada"
`providerConfig.s3.useIAM`	Habilite la autenticación IAM para acceder a los buckets S3. Aplicable para la identidad del pod EKS.	falso	verdadero, falso

Ver información de AppVault

Puede utilizar el complemento CLI de Trident Protect para ver información sobre los objetos de AppVault que ha creado en el clúster.

Pasos

1. Ver el contenido de un objeto AppVault:

```
tridentctl-protect get appvaultcontent gcp-vault \
--show-resources all \
-n trident-protect
```

Ejemplo de salida:

```
+-------------+-------+----------+----------------------------
+---------------------------+
|   CLUSTER   |  APP  |   TYPE   |            NAME             |
TIMESTAMP         |
+-------------+-------+----------+----------------------------
+---------------------------+
|             | mysql | snapshot | mysnap                      | 2024-
08-09 21:02:11 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815180300 | 2024-
08-15 18:03:06 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815190300 | 2024-
08-15 19:03:06 (UTC) |
| production1 | mysql | snapshot | hourly-e7db6-20240815200300 | 2024-
08-15 20:03:06 (UTC) |
| production1 | mysql | backup   | hourly-e7db6-20240815180300 | 2024-
08-15 18:04:25 (UTC) |
| production1 | mysql | backup   | hourly-e7db6-20240815190300 | 2024-
08-15 19:03:30 (UTC) |
| production1 | mysql | backup   | hourly-e7db6-20240815200300 | 2024-
08-15 20:04:21 (UTC) |
| production1 | mysql | backup   | mybackup5                   | 2024-
08-09 22:25:13 (UTC) |
|             | mysql | backup   | mybackup                    | 2024-
08-09 21:02:52 (UTC) |
+-------------+-------+----------+----------------------------
+---------------------------+
```

2. Opcionalmente, para ver AppVaultPath para cada recurso, utilice el indicador `--show-paths`.

   El nombre del clúster en la primera columna de la tabla solo está disponible si se especificó un nombre de clúster en la instalación de Trident Protect Helm. Por ejemplo: `--set clusterName=production1`.

Eliminar un AppVault

Puede eliminar un objeto AppVault en cualquier momento.

NOTA No elimine `finalizers` la clave de AppVault CR antes de eliminar el objeto AppVault. Si lo hace, puede dar como resultado datos residuales en el bucket de AppVault y recursos huérfanos en el cluster.

Antes de empezar

Asegúrese de haber eliminado todos los CRS de instantánea y copia de seguridad que utiliza el AppVault que desea eliminar.

Quite un AppVault con la CLI de Kubernetes

1. Elimine el objeto AppVault, sustituyéndolo `appvault-name` por el nombre del objeto AppVault que desea eliminar:

```
kubectl delete appvault <appvault-name> \
-n trident-protect
```

Eliminar un AppVault mediante la CLI de Trident Protect

1. Elimine el objeto AppVault, sustituyéndolo `appvault-name` por el nombre del objeto AppVault que desea eliminar:

```
tridentctl-protect delete appvault <appvault-name> \
-n trident-protect
```

Definir una aplicación para la gestión con Trident Protect

Puede definir una aplicación que desee administrar con Trident Protect creando un CR de aplicación y un CR de AppVault asociado.

Cree un CR de AppVault

Debe crear un CR de AppVault que se utilizará al realizar operaciones de protección de datos en la aplicación, y el CR de AppVault debe residir en el clúster donde está instalado Trident Protect. La solicitud de cambio (CR) de AppVault es específica de su entorno; para ver ejemplos de solicitudes de cambio de cambio de AppVault, consulte"Recursos personalizados de AppVault."

Defina una aplicación

Debe definir cada aplicación que desee administrar con Trident Protect. Puede definir una aplicación para su administración ya sea creando manualmente un CR de aplicación o utilizando la CLI de Trident Protect.

NOTA Para definir una aplicación con ámbito de máquina virtual (VM) que proteja máquinas virtuales (VM) específicas de KubeVirt por su nombre en lugar de espacios de nombres completos, usa el campo `spec.includedVirtualMachines` en el Application CR o la bandera `--virtual-machines` en la CLI. Para detalles, referencia de campos y ejemplos en YAML, consulta "Protege las máquinas virtuales de KubeVirt con Trident Protect".

Agregar una aplicación mediante un CR

Pasos

1. Cree el archivo CR de la aplicación de destino:
   a. Cree el archivo de recursos personalizados (CR) y asígnele un nombre (por ejemplo, `maria-app.yaml`).
   b. Configure los siguientes atributos:
      - **metadata.name**: (*required*) El nombre del recurso personalizado de la aplicación. Tenga en cuenta el nombre que elija porque otros archivos CR necesarios para las operaciones de protección hacen referencia a este valor.
      - **spec.includedNamespaces**: (*required*) Utilice el espacio de nombres y el selector de etiquetas para especificar los espacios de nombres y recursos que utiliza la aplicación. El espacio de nombres de la aplicación debe formar parte de esta lista. El selector de etiquetas es opcional y se puede utilizar para filtrar recursos dentro de cada espacio de nombres especificado.
      - **spec.includedClusterScopedResources**: (*Optional*) Utilice este atributo para especificar los recursos de ámbito de cluster que se incluirán en la definición de la aplicación. Este atributo le permite seleccionar estos recursos en función de su grupo, versión, tipo y etiquetas.
        - **GroupVersionKind**: (*required*) Especifica el grupo API, la versión y el tipo del recurso de ámbito de cluster.
        - **LabelSelector**: (*Optional*) Filtra los recursos de ámbito de cluster en función de sus etiquetas.
      - **metadata.annotations.protect.trident.netapp.io/skip-vm-freeze**: (*Opcional*) Esta anotación solo es aplicable a aplicaciones definidas desde máquinas virtuales, como en entornos KubeVirt, donde las congelaciones del sistema de archivos ocurren antes de las instantáneas. Especifique si esta aplicación puede escribir en el sistema de archivos durante una instantánea. Si se establece en verdadero, la aplicación ignora la configuración global y puede escribir en el sistema de archivos durante una instantánea. Si se establece en falso, la aplicación ignora la configuración global y el sistema de archivos se congela durante una instantánea. Si se especifica pero la aplicación no tiene máquinas virtuales en la definición de la aplicación, la anotación se ignora. Si no se especifica, la aplicación sigue el procedimiento establecido."Configuración de congelación global de Trident Protect" .

Si necesita aplicar esta anotación después de crear una aplicación, puede utilizar el siguiente comando:

```
kubectl annotate application -n <application CR namespace> <application CR name> protect.trident.netapp.io/skip-vm-freeze="true"
```

+
Ejemplo YAML:

+

```
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  annotations:
    protect.trident.netapp.io/skip-vm-freeze: "false"
  name: my-app-name
  namespace: my-app-namespace
spec:
  includedNamespaces:
    - namespace: namespace-1
      labelSelector:
        matchLabels:
          app: example-app
    - namespace: namespace-2
      labelSelector:
        matchLabels:
          app: another-example-app
  includedClusterScopedResources:
    - groupVersionKind:
        group: rbac.authorization.k8s.io
        kind: ClusterRole
        version: v1
      labelSelector:
        matchLabels:
          mylabel: test
```

1. (*Opcional*) Si lo necesitas, puedes añadir filtrado de recursos al mismo CR para incluir o excluir recursos específicos:

   - **Ejemplo de filtro genérico:**

     - **ResourceFilter.resourceSelectionCriteria**: (Requerido para filtrar) Usar `Include` o `Exclude` incluir o excluir un recurso definido en resourceMatchers. Agregue los siguientes parámetros resourceMatchers para definir los recursos que se van a incluir o excluir:

       - **ResourceFilter.resourceMatchers**: Una matriz de objetos resourceMatcher. Si define varios elementos en esta matriz, coinciden como una OPERACIÓN OR y los campos dentro de cada elemento (grupo, tipo, versión) coinciden como una operación AND.
       - **ResourceMatchers[].group**: (*Optional*) Grupo del recurso a filtrar.
       - **ResourceMatchers[].kind**: (*Optional*) Tipo de recurso a filtrar.
       - **ResourceMatchers[].version**: (*Optional*) Versión del recurso que se va a filtrar.

- **ResourceMatchers[].names**: (*Optional*) Nombres en el campo Kubernetes metadata.name del recurso que se va a filtrar.
- **ResourceMatchers[].namespaces**: (*Optional*) Espacios de nombres en el campo Kubernetes metadata.name del recurso que se va a filtrar.
- **ResourceMatchers[].labelSelectors**: (*Optional*) Cadena de selector de etiquetas en el campo Kubernetes metadata.name del recurso tal como se define en el "Documentación de Kubernetes". Por ejemplo `"trident.netapp.io/os=linux"`: .

NOTA	Cuando ambos `resourceFilter` y `labelSelector` se utilizan, `resourceFilter` corre primero y luego `labelSelector` Se aplica a los recursos resultantes.

Por ejemplo:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

◦ **Ejemplo de filtro solo de PVC:**

Para definir una aplicación sólo para PVC, también debes incluir `PersistentVolume` y `VolumeSnapshotClass` en el filtro de recursos. Las operaciones de instantáneas y copias de seguridad dependen de `PersistentVolume` (el volumen de ámbito de clúster vinculado a cada PVC) y `VolumeSnapshotClass` (el controlador de instantáneas), y fallarán sin ellos. Por ejemplo:

```
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  name: my-pvc-app
  namespace: my-app-namespace
spec:
  includedNamespaces:
  - namespace: my-app-namespace
  resourceFilter:
    resourceMatchers:
    - kind: PersistentVolumeClaim
      version: v1
    - kind: PersistentVolume
      version: v1
    - kind: VolumeSnapshotClass
      version: v1
    resourceSelectionCriteria: Include
```

2. Después de crear la CR de la aplicación para que coincida con su entorno, aplique la CR. Por ejemplo:

```
kubectl apply -f maria-app.yaml
```

Pasos

1. Cree y aplique la definición de la aplicación utilizando uno de los siguientes ejemplos, sustituyendo valores entre paréntesis por información de su entorno. Puede incluir espacios de nombres y recursos en la definición de la aplicación mediante listas separadas por comas con los argumentos que se muestran en los ejemplos.

Opcionalmente, puede usar una anotación al crear una aplicación para especificar si la aplicación puede escribir en el sistema de archivos durante una instantánea. Esto solo es aplicable a aplicaciones definidas desde máquinas virtuales, como en entornos KubeVirt, donde se producen congelaciones del sistema de archivos antes de las instantáneas. Si configura la anotación a `true` La aplicación ignora la configuración global y puede escribir en el sistema de archivos durante una instantánea. Si lo configuras para `false` La aplicación ignora la configuración global y el sistema de archivos se congela durante la toma de una instantánea. Si utiliza la anotación pero la aplicación no tiene máquinas virtuales en la definición de la aplicación, la anotación se ignora. Si no utilizas la anotación, la aplicación sigue el comportamiento esperado."Configuración de congelación global de Trident Protect" .

+ Para especificar la anotación cuando usas la CLI para crear una aplicación, puedes usar la bandera `--annotation`.

+ * Crea la aplicación y utiliza la configuración global para el comportamiento de congelación del sistema de archivos:

+

```
tridentctl-protect create application <my_new_app_cr_name> --namespaces
<namespaces_to_include> --csr <cluster_scoped_resources_to_include>
--namespace <my-app-namespace>
```

+ * Crea la aplicación y configura la configuración local de la aplicación para el comportamiento de congelación del sistema de archivos:

+

```
tridentctl-protect create application <my_new_app_cr_name> --namespaces
<namespaces_to_include> --csr <cluster_scoped_resources_to_include>
--namespace <my-app-namespace> --annotation
protect.trident.netapp.io/skip-vm-freeze=<"true"|"false">
```

+ * Puedes usar `--resource-filter-include` y `--resource-filter-exclude` para incluir o excluir recursos según `resourceSelectionCriteria` como grupo, tipo, versión, etiquetas, nombres y namespaces, como se muestra en el siguiente ejemplo:

+

```
tridentctl-protect create application <my_new_app_cr_name> --namespaces
<namespaces_to_include> --csr <cluster_scoped_resources_to_include>
--namespace <my-app-namespace> --resource-filter-include
'[{"Group":"apps","Kind":"Deployment","Version":"v1","Names":["my-
deployment"],"Namespaces":["my-namespace"],"LabelSelectors":["app=my-
app"]}]'
```

+ * Para definir una aplicación que utilice únicamente PVC, también debes incluir `PersistentVolume` y `VolumeSnapshotClass` en el filtro de recursos. Las operaciones de instantáneas y copias de seguridad dependen de `PersistentVolume` (el volumen de ámbito de clúster vinculado a cada PVC) y `VolumeSnapshotClass` (el controlador de instantáneas), y fallarán si no se incluyen. Por ejemplo:

+

```
tridentctl-protect create app my-pvc-app --namespaces <my-app-
namespace> --resource-filter-include
'[{"Kind":"PersistentVolumeClaim","Version":"v1"},{"Kind":"PersistentVo
lume","Version":"v1"},{"Kind":"VolumeSnapshotClass","Version":"v1"}]'
-n <my-app-namespace>
```

Proteja las aplicaciones con Trident Protect

Puede proteger todas las aplicaciones administradas por Trident Protect tomando

instantáneas y copias de seguridad mediante una política de protección automatizada o de forma ad hoc.

NOTA	Puede configurar Trident Protect para congelar y descongelar sistemas de archivos durante operaciones de protección de datos. "Obtenga más información sobre cómo configurar la congelación del sistema de archivos con Trident Protect".

Crear una snapshot bajo demanda

Puede crear una snapshot bajo demanda en cualquier momento.

NOTA	Los recursos de ámbito de clúster se incluyen en un backup, una copia de Snapshot o un clon si se hace referencia explícitamente a estos en la definición de la aplicación o si tienen referencias a cualquiera de los espacios de nombres de la aplicación.

Cree una instantánea con un CR

Pasos

1. Cree el archivo de recursos personalizados (CR) y asígnele un nombre `trident-protect-snapshot-cr.yaml`.
2. En el archivo creado, configure los siguientes atributos:
   - **metadata.name**: (*required*) El nombre de este recurso personalizado; elija un nombre único y sensible para su entorno.
   - **Spec.applicationRef**: El nombre de Kubernetes de la aplicación a la instantánea.
   - **Spec.appVaultRef**: (*required*) El nombre del AppVault donde se debe almacenar el contenido de la instantánea (metadatos).
   - **Spec.reclaimer Policy**: (*Optional*) define lo que sucede con el AppArchive de una instantánea cuando se elimina el CR de la instantánea. Esto significa que incluso cuando se define en `Retain`, la instantánea se suprimirá. Opciones válidas:
     - `Retain` (predeterminado)
     - `Delete`

     ```
     apiVersion: protect.trident.netapp.io/v1
     kind: Snapshot
     metadata:
       namespace: my-app-namespace
       name: my-cr-name
     spec:
       applicationRef: my-application
       appVaultRef: appvault-name
       reclaimPolicy: Delete
     ```
3. Después de rellenar `trident-protect-snapshot-cr.yaml` el archivo con los valores correctos, aplique el CR:

   ```
   kubectl apply -f trident-protect-snapshot-cr.yaml
   ```

Cree una copia Snapshot mediante la CLI

Pasos

1. Cree la instantánea, reemplazando valores entre paréntesis con información de su entorno. Por ejemplo:

   ```
   tridentctl-protect create snapshot <my_snapshot_name> --appvault
   <my_appvault_name> --app <name_of_app_to_snapshot> -n
   <application_namespace>
   ```

Cree un backup bajo demanda

Puede realizar una copia de seguridad de una aplicación en cualquier momento.

NOTA	Los recursos de ámbito de clúster se incluyen en un backup, una copia de Snapshot o un clon si se hace referencia explícitamente a estos en la definición de la aplicación o si tienen referencias a cualquiera de los espacios de nombres de la aplicación.

Antes de empezar

Asegúrese de que la caducidad del token de sesión de AWS sea suficiente para cualquier operación de copia de seguridad de S3 que se ejecute durante mucho tiempo. Si el token caduca durante la operación de copia de seguridad, la operación puede fallar.

- Consulte el "Documentación de la API de AWS" para obtener más información sobre la comprobación de la caducidad del token de sesión actual.
- Consulte el documento para "Documentación de AWS IAM" obtener más información acerca de las credenciales con recursos de AWS.

Cree una copia de seguridad con un CR

Pasos

1. Cree el archivo de recursos personalizados (CR) y asígnele un nombre `trident-protect-backup-cr.yaml`.
2. En el archivo creado, configure los siguientes atributos:
   - **metadata.name**: (*required*) El nombre de este recurso personalizado; elija un nombre único y sensible para su entorno.
   - **Spec.applicationRef**: (*required*) El nombre de Kubernetes de la aplicación para realizar una copia de seguridad.
   - **Spec.appVaultRef**: (*required*) El nombre del AppVault donde se debe almacenar el contenido de la copia de seguridad.
   - **SPEC.DATAMOVER**: (*Optional*) Una cadena que indica qué herramienta de copia de seguridad usar para la operación de copia de seguridad. Valores posibles (distingue mayúsculas de minúsculas):
     - `Restic`
     - `Kopia` (predeterminado)
   - **Spec.reclaimer Policy**: (*Optional*) define lo que sucede con una copia de seguridad cuando se libera de su reclamación. Los posibles valores son los siguientes:
     - `Delete`
     - `Retain` (predeterminado)
   - **spec.snapshotRef**: (*Opcional*): Nombre de la instantánea que se utilizará como origen de la copia de seguridad. Si no se proporciona, se creará una instantánea temporal y se realizará una copia de seguridad.

     Ejemplo YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Backup
metadata:
  namespace: my-app-namespace
  name: my-cr-name
spec:
  applicationRef: my-application
  appVaultRef: appvault-name
  dataMover: Kopia
```

3. Después de rellenar `trident-protect-backup-cr.yaml` el archivo con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-backup-cr.yaml
```

Cree un backup con la interfaz de línea de comandos

Pasos

1. Cree el backup sustituyendo valores entre paréntesis con información de su entorno. Por ejemplo:

```
tridentctl-protect create backup <my_backup_name> --appvault <my-vault-name> --app <name_of_app_to_back_up> --data-mover <Kopia_or_Restic> -n <application_namespace>
```

   Opcionalmente, puede utilizar `--full-backup` el indicador para especificar si una copia de seguridad debe ser no incremental. Por defecto, todos los backups son incrementales. Cuando se utiliza este indicador, la copia de seguridad pasa a ser no incremental. Es mejor realizar backups completos de forma periódica y después realizar backups incrementales entre backups completos para minimizar el riesgo asociado a las restauraciones.

Anotaciones de respaldo admitidas

La siguiente tabla describe las anotaciones que puede utilizar al crear una copia de seguridad de CR:

Anotación	Tipo	Descripción	Valor predeterminado
proteger.trident.netapp.io/copia-de-seguridad-completa	cadena	Especifica si una copia de seguridad debe ser no incremental. Empezar a `true` para crear una copia de seguridad no incremental. La mejor práctica es realizar una copia de seguridad completa periódicamente y luego realizar copias de seguridad incrementales entre las copias de seguridad completas para minimizar el riesgo asociado con las restauraciones.	"falso"
protect.trident.netapp.io/snapshot-complete-timeout	cadena	Tiempo máximo permitido para que se complete la operación de instantánea general.	"60 m"
protect.trident.netapp.io/volume-snapshots-ready-to-use-timeout	cadena	Tiempo máximo permitido para que las instantáneas de volumen alcancen el estado listo para usar.	"30 m"
protect.trident.netapp.io/volume-snapshots-created-timeout	cadena	Tiempo máximo permitido para la creación de instantáneas de volumen.	"5 m"
protect.trident.netapp.io/pvc-bind-timeout-sec	cadena	Tiempo máximo (en segundos) que se debe esperar para que cualquier PersistentVolumeClaim (PVC) recién creado llegue al `Bound` fase antes de que fallen las operaciones.	"1200" (20 minutos)

Cree un programa de protección de datos

Una política de protección protege una aplicación mediante la creación de instantáneas, copias de seguridad o ambas según un cronograma definido. Puede elegir crear instantáneas y copias de seguridad cada hora, día,

semana o mes, y puede especificar la cantidad de copias que desea conservar. Puede programar una copia de seguridad completa no incremental mediante la anotación full-backup-rule. De forma predeterminada, todas las copias de seguridad son incrementales. Realizar una copia de seguridad completa periódicamente, junto con copias de seguridad incrementales entre ellas, ayuda a reducir el riesgo asociado con las restauraciones.

NOTA

- Puede crear programaciones para instantáneas únicamente configurando `backupRetention` a cero y `snapshotRetention` a un valor mayor que cero. Configuración `snapshotRetention` Poner a cero significa que cualquier copia de seguridad programada seguirá creando instantáneas, pero éstas serán temporales y se eliminarán inmediatamente después de que se complete la copia de seguridad.
- Los recursos de ámbito de clúster se incluyen en un backup, una copia de Snapshot o un clon si se hace referencia explícitamente a estos en la definición de la aplicación o si tienen referencias a cualquiera de los espacios de nombres de la aplicación.

Crear un horario mediante un CR

Pasos

1. Cree el archivo de recursos personalizados (CR) y asígnele un nombre `trident-protect-schedule-cr.yaml`.
2. En el archivo creado, configure los siguientes atributos:
   - **metadata.name**: (*required*) El nombre de este recurso personalizado; elija un nombre único y sensible para su entorno.
   - **SPEC.DATAMOVER**: (*Optional*) Una cadena que indica qué herramienta de copia de seguridad usar para la operación de copia de seguridad. Valores posibles (distingue mayúsculas de minúsculas):
     - `Restic`
     - `Kopia` (predeterminado)
   - **Spec.applicationRef**: El nombre de Kubernetes de la aplicación para realizar una copia de seguridad.
   - **Spec.appVaultRef**: (*required*) El nombre del AppVault donde se debe almacenar el contenido de la copia de seguridad.
   - **spec.backupRetention**: (*Obligatorio*) La cantidad de copias de seguridad a conservar. Cero indica que no se deben crear copias de seguridad (solo instantáneas).
   - **spec.backupReclaimPolicy**: (*Opcional*) Determina qué sucede con una copia de seguridad si el CR de la copia de seguridad se elimina durante su período de retención. Después del período de retención, las copias de seguridad siempre se eliminan. Valores posibles (distingue entre mayúsculas y minúsculas):
     - `Retain` (predeterminado)
     - `Delete`
   - **spec.snapshotRetention**: (*Obligatorio*) La cantidad de instantáneas a conservar. El valor cero indica que no se deben crear instantáneas.
   - **spec.snapshotReclaimPolicy**: (*Opcional*) Determina qué sucede con una instantánea si el CR de la instantánea se elimina durante su período de retención. Después del período de retención, las instantáneas siempre se eliminan. Valores posibles (distingue entre mayúsculas y minúsculas):
     - `Retain`
     - `Delete`(por defecto)
   - **spec.granularity**: La frecuencia con la que debe ejecutarse el horario. Los posibles valores, junto con los campos asociados necesarios:
     - `Hourly`(requiere que usted especifique `spec.minute`)
     - `Daily`(requiere que usted especifique `spec.minute` y `spec.hour`)
     - `Weekly`(requiere que usted especifique `spec.minute, spec.hour` , y `spec.dayOfWeek`)
     - `Monthly`(requiere que usted especifique `spec.minute, spec.hour` , y `spec.dayOfMonth`)
     - `Custom`

- **spec.dayOfMonth**: (*Opcional*) El día del mes (1 - 31) en que debe ejecutarse la programación. Este campo es obligatorio si la granularidad está establecida en `Monthly` . El valor debe proporcionarse como una cadena.
- **spec.dayOfWeek**: (*Opcional*) El día de la semana (0 - 7) en que debe ejecutarse la programación. Los valores de 0 o 7 indican domingo. Este campo es obligatorio si la granularidad está establecida en `Weekly` . El valor debe proporcionarse como una cadena.
- **spec.hour**: (*Opcional*) La hora del día (0 - 23) en que debe ejecutarse el programa. Este campo es obligatorio si la granularidad está establecida en `Daily` , `Weekly` , o `Monthly` . El valor debe proporcionarse como una cadena.
- **spec.minute**: (*Opcional*) El minuto de la hora (0 - 59) en que debe ejecutarse el programa. Este campo es obligatorio si la granularidad está establecida en `Hourly` , `Daily` , `Weekly` , o `Monthly` . El valor debe proporcionarse como una cadena.
- **spec.runImmediately**: (*Opcional*) Establece en `true` para activar una ejecución de línea de base única e inmediata (copia de seguridad y/o instantánea según la configuración de retención) cuando se crea el programa. Por defecto es `false`. Esto no modifica la recurrencia posterior.

  Ejemplo de YAML para programación de copias de seguridad y instantáneas:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  namespace: my-app-namespace
  name: my-cr-name
spec:
  dataMover: Kopia
  applicationRef: my-application
  appVaultRef: appvault-name
  backupRetention: "15"
  snapshotRetention: "15"
  granularity: Daily
  hour: "0"
  minute: "0"
```

  Ejemplo de YAML para una programación de solo instantáneas:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  namespace: my-app-namespace
  name: my-snapshot-schedule
spec:
  applicationRef: my-application
  appVaultRef: appvault-name
  backupRetention: "0"
  snapshotRetention: "15"
  granularity: Daily
  hour: "2"
  minute: "0"
```

Ejemplo YAML para programación con ejecución inmediata:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  namespace: my-app-namespace
  name: my-daily-schedule-run-immediately
spec:
  applicationRef: my-application
  appVaultRef: appvault-name
  backupRetention: "7"
  snapshotRetention: "7"
  granularity: Daily
  hour: "3"
  minute: "0"
  runImmediately: true
```

3. Después de rellenar `trident-protect-schedule-cr.yaml` el archivo con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-schedule-cr.yaml
```

Cree una programación con la CLI

Pasos

1. Cree el programa de protección, reemplazando los valores entre paréntesis con información de su entorno. Por ejemplo:

NOTA Puede usar `tridentctl-protect create schedule --help` para ver información de ayuda detallada de este comando.

```
tridentctl-protect create schedule <my_schedule_name> \
  --appvault <my_appvault_name> \
  --app <name_of_app_to_snapshot> \
  --backup-retention <how_many_backups_to_retain> \
  --backup-reclaim-policy <Retain|Delete (default Retain)> \
  --data-mover <Kopia_or_Restic> \
  --day-of-month <day_of_month_to_run_schedule> \
  --day-of-week <day_of_week_to_run_schedule> \
  --granularity <frequency_to_run> \
  --hour <hour_of_day_to_run> \
  --minute <minute_of_hour_to_run> \
  --recurrence-rule <recurrence> \
  --snapshot-retention <how_many_snapshots_to_retain> \
  --snapshot-reclaim-policy <Retain|Delete (default Delete)> \
  --full-backup-rule <string> \
  --run-immediately <true|false> \
  -n <application_namespace>
```

Las siguientes banderas proporcionan control adicional sobre su programación:

- **Programación de copia de seguridad completa**: utilice la `--full-backup-rule` bandera para programar copias de seguridad completas no incrementales. Esta bandera sólo funciona con `--granularity Daily`. Valores posibles:
  - `Always`:Cree una copia de seguridad completa todos los días.
  - Días de la semana específicos: especifique uno o más días separados por comas (por ejemplo, `"Monday,Thursday"`). Valores válidos: lunes, martes, miércoles, jueves, viernes, sábado, domingo.

    NOTA El `--full-backup-rule` La bandera no funciona con granularidad horaria, semanal o mensual.

- **Protección inicial inmediata**: Usa `--run-immediately true` para crear una copia de seguridad o instantánea inicial inmediatamente cuando se crea la programación, en lugar de esperar a la primera hora de ejecución programada. El valor predeterminado es `false`.
- **Horarios solo de instantáneas**: Establecer `--backup-retention 0` y especifique un valor mayor que cero para `--snapshot-retention`.

Anotaciones de programación admitidas

La siguiente tabla describe las anotaciones que puede utilizar al crear una solicitud de cambio de programación:

Anotación	Tipo	Descripción	Valor predeterminado
proteger.trident.netapp.io/regla-de-copia-de-seguridad-completa	cadena	Especifica la regla para programar copias de seguridad completas. Puedes configurarlo para `Always` para realizar copias de seguridad completas y constantes o personalizarlas según sus necesidades. Por ejemplo, si elige granularidad diaria, puede especificar los días de la semana en los que debe realizarse la copia de seguridad completa (por ejemplo, `"Monday,Thursday"`). Los valores válidos para días de la semana son: lunes, martes, miércoles, jueves, viernes, sábado y domingo. Tenga en cuenta que esta anotación solo se puede utilizar con programaciones que tengan `granularity` empezar a `Daily`.	No configurado (todas las copias de seguridad son incrementales)
protect.trident.netapp.io/snapshot-complete-timeout	cadena	Tiempo máximo permitido para que se complete la operación de instantánea general.	"60 m"
protect.trident.netapp.io/volume-snapshots-ready-to-use-timeout	cadena	Tiempo máximo permitido para que las instantáneas de volumen alcancen el estado listo para usar.	"30 m"
protect.trident.netapp.io/volume-snapshots-created-timeout	cadena	Tiempo máximo permitido para la creación de instantáneas de volumen.	"5 m"
protect.trident.netapp.io/pvc-bind-timeout-sec	cadena	Tiempo máximo (en segundos) que se debe esperar para que cualquier PersistentVolumeClaim (PVC) recién creado llegue al `Bound` fase antes de que fallen las operaciones.	"1200" (20 minutos)

Eliminar una copia de Snapshot

Elimine las snapshots programadas o bajo demanda que ya no necesite.

Pasos

1. Elimine el CR de instantánea asociado a la instantánea:

```
kubectl delete snapshot <snapshot_name> -n my-app-namespace
```

Eliminar una copia de seguridad

Elimine los backups programados o bajo demanda que ya no necesita.

NOTA	Asegúrese de que la política de recuperación esté configurada en `Delete` Para eliminar todos los datos de respaldo del almacenamiento de objetos. La configuración predeterminada de la política es `Retain` Para evitar la pérdida accidental de datos. Si la política no se modifica a `Delete` Los datos de respaldo permanecerán en el almacenamiento de objetos y será necesario eliminarlos manualmente.

Pasos

1. Elimine el CR de backup asociado con el backup:

```
kubectl delete backup <backup_name> -n my-app-namespace
```

Compruebe el estado de una operación de backup

Puede usar la línea de comandos para comprobar el estado de una operación de backup que está en curso, se completa o tiene errores.

Pasos

1. Utilice el siguiente comando para recuperar el estado de la operación de copia de seguridad, sustituyendo los valores entre corchetes por información de su entorno:

```
kubectl get backup -n <namespace_name> <my_backup_cr_name> -o jsonpath
='{.status}'
```

Permita el backup y la restauración para las operaciones de azure-NetApp-files (ANF)

Si ha instalado Trident Protect, puede habilitar la funcionalidad de copia de seguridad y restauración que ahorra espacio para los backends de almacenamiento que usan la clase de almacenamiento azure-netapp-files y se crearon antes de Trident 24.06. Esta funcionalidad funciona con volúmenes NFSv4 y no consume espacio adicional del grupo de capacidad.

Antes de empezar

Asegúrese de lo siguiente:

- Ha instalado Trident Protect.
- Ha definido una aplicación en Trident Protect. Esta aplicación tendrá una funcionalidad de protección limitada hasta que complete este procedimiento.
- `azure-netapp-files`Seleccionó como clase de almacenamiento predeterminada para el back-end de almacenamiento.

Expanda para obtener pasos de configuración

1. Haga lo siguiente en Trident si el volumen ANF se creó antes de actualizar a Trident 24,10:

   a. Habilite el directorio de instantáneas para cada VP basado en azure-NetApp-files y asociado con la aplicación:

   ```
   tridentctl update volume <pv name> --snapshot-dir=true -n trident
   ```

   b. Confirme que el directorio de snapshots se haya habilitado para cada VP asociado:

   ```
   tridentctl get volume <pv name> -n trident -o yaml | grep
   snapshotDir
   ```

   Respuesta:

   ```
   snapshotDirectory: "true"
   ```

   +
   Cuando el directorio de instantáneas no está habilitado, Trident Protect elige la funcionalidad de copia de seguridad regular, que consume temporalmente espacio en el grupo de capacidad durante el proceso de copia de seguridad. En este caso, asegúrese de que haya suficiente espacio disponible en el grupo de capacidad para crear un volumen temporal del tamaño del volumen del que se está realizando la copia de seguridad.

Resultado

La aplicación está lista para realizar copias de seguridad y restaurar mediante Trident Protect. Cada PVC también está disponible para ser utilizado por otras aplicaciones para copias de seguridad y restauraciones.

Restaure las aplicaciones

Restaurar aplicaciones usando Trident Protect

Puede utilizar Trident Protect para restaurar su aplicación desde una instantánea o una copia de seguridad. Restaurar desde una instantánea existente será más rápido al restaurar la aplicación en el mismo clúster.

NOTA

- Al restaurar una aplicación, todos los ganchos de ejecución configurados para la aplicación se restauran con la aplicación. Si hay un enlace de ejecución posterior a la restauración, se ejecuta automáticamente como parte de la operación de restauración.
- Se permite restaurar desde una copia de seguridad a un espacio de nombres diferente o al original en volúmenes qtree. Sin embargo, no se permite restaurar desde una instantánea a un espacio de nombres diferente o al original en volúmenes qtree.
- Puede utilizar la configuración avanzada para personalizar las operaciones de restauración. Para obtener más información, consulte "Utilice la configuración de restauración avanzada de Trident Protect".
- Para restaurar máquinas virtuales (VM) específicas de KubeVirt desde un archivo sin restaurar todos los recursos del espacio de nombres, usa el campo `includedVirtualMachines` en un CR de restauración. Para CR compatibles, referencia de campos y ejemplos en YAML, consulta "Protege las máquinas virtuales de KubeVirt con Trident Protect".

Restauración desde un backup a un espacio de nombres diferente

Cuando restaura una copia de seguridad en un espacio de nombres diferente mediante un CR de BackupRestore, Trident Protect restaura la aplicación en un nuevo espacio de nombres y crea un CR de aplicación para la aplicación restaurada. Para proteger la aplicación restaurada, cree copias de seguridad o instantáneas bajo demanda, o establezca un programa de protección.

NOTA

- Al restaurar un backup en un espacio de nombres diferente con los recursos existentes, no se alterará ningún recurso que comparta los nombres con los que aparecen en el backup. Para restaurar todos los recursos del backup, elimine y vuelva a crear el espacio de nombres objetivo, o restaure el backup en un nuevo espacio de nombres.
- Al utilizar una CR para restaurar a un nuevo espacio de nombres, debe crear manualmente el espacio de nombres de destino antes de aplicar la CR. Trident Protect crea espacios de nombres automáticamente solo cuando se usa la CLI.

Antes de empezar

Asegúrese de que la caducidad del token de sesión de AWS sea suficiente para las operaciones de restauración de S3 que se ejecuten durante mucho tiempo. Si el token caduca durante la operación de restauración, puede fallar la operación.

- Consulte el "Documentación de la API de AWS" para obtener más información sobre la comprobación de la caducidad del token de sesión actual.
- Consulte el documento para "Documentación de AWS IAM" obtener más información acerca de las credenciales con recursos de AWS.

NOTA

Al restaurar copias de seguridad utilizando Kopia como transportador de datos, puede especificar opcionalmente anotaciones en la CR o usar la CLI para controlar el comportamiento del almacenamiento temporal utilizado por Kopia. Consulte el "Documentación de KOPIA" Para obtener más información sobre las opciones que puede configurar. Utilice el `tridentctl-protect create --help` Comando para obtener más información sobre cómo especificar anotaciones con la CLI de Trident Protect.

Utilice un CR

Pasos

1. Cree el archivo de recursos personalizados (CR) y asígnele un nombre `trident-protect-backup-restore-cr.yaml`.
2. En el archivo creado, configure los siguientes atributos:
   - **metadata.name**: (*required*) El nombre de este recurso personalizado; elija un nombre único y sensible para su entorno.
   - **Spec.appArchivePath**: La ruta dentro de AppVault donde se almacena el contenido de la copia de seguridad. Puede utilizar el siguiente comando para buscar esta ruta:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

   - **Spec.appVaultRef**: (*required*) El nombre del AppVault donde se almacena el contenido de la copia de seguridad.
   - **spec.destinationApplicationName**: (*opcional*) el nombre para la aplicación restaurada. Si se proporciona, la aplicación restaurada usa este nombre. Si no se proporciona, la aplicación restaurada usa el nombre de la aplicación de origen.
   - **spec.namespaceMapping**: La asignación del espacio de nombres de origen de la operación de restauración al espacio de nombres de destino. Reemplace `my-source-namespace` y `my-destination-namespace` con la información de su entorno.

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: BackupRestore
     metadata:
       name: my-cr-name
       namespace: my-destination-namespace
     spec:
       appArchivePath: my-backup-path
       appVaultRef: appvault-name
       destinationApplicationName: my-new-app-name
       namespaceMapping: [{"source": "my-source-namespace",
     "destination": "my-destination-namespace"}]
     ```

3. (*Optional*) Si necesita seleccionar solo ciertos recursos de la aplicación para restaurar, agregue filtros que incluyan o excluyan recursos marcados con etiquetas particulares:

   NOTA Trident Protect selecciona algunos recursos automáticamente debido a su relación con los recursos que usted selecciona. Por ejemplo, si selecciona un recurso de reclamo de volumen persistente y tiene un pod asociado, Trident Protect también restaurará el pod asociado.

   - **ResourceFilter.resourceSelectionCriteria**: (Requerido para filtrar) Usar `Include` o `Exclude`

incluir o excluir un recurso definido en resourceMatchers. Agregue los siguientes parámetros resourceMatchers para definir los recursos que se van a incluir o excluir:

- **ResourceFilter.resourceMatchers**: Una matriz de objetos resourceMatcher. Si define varios elementos en esta matriz, coinciden como una OPERACIÓN OR y los campos dentro de cada elemento (grupo, tipo, versión) coinciden como una operación AND.
  - **ResourceMatchers[].group**: (*Optional*) Grupo del recurso a filtrar.
  - **ResourceMatchers[].kind**: (*Optional*) Tipo de recurso a filtrar.
  - **ResourceMatchers[].version**: (*Optional*) Versión del recurso que se va a filtrar.
  - **ResourceMatchers[].names**: (*Optional*) Nombres en el campo Kubernetes metadata.name del recurso que se va a filtrar.
  - **ResourceMatchers[].namespaces**: (*Optional*) Espacios de nombres en el campo Kubernetes metadata.name del recurso que se va a filtrar.
  - **ResourceMatchers[].labelSelectors**: (*Optional*) Cadena de selector de etiquetas en el campo Kubernetes metadata.name del recurso tal como se define en el "Documentación de Kubernetes". Por ejemplo `"trident.netapp.io/os=linux"`: .

    Por ejemplo:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Después de rellenar `trident-protect-backup-restore-cr.yaml` el archivo con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-backup-restore-cr.yaml
```

Utilice la CLI

Pasos

1. Restaure la copia de seguridad en un espacio de nombres diferente, sustituyendo valores entre paréntesis por información de su entorno. El `namespace-mapping` argumento utiliza espacios de

nombres separados por dos puntos para asignar espacios de nombres de origen a los espacios de nombres de destino correctos en el formato `source1:dest1,source2:dest2`. Por ejemplo:

```
tridentctl-protect create backuprestore <my_restore_name> \
--backup <backup_namespace>/<backup_to_restore> \
--namespace-mapping <source_to_destination_namespace_mapping> \
--destination-app-name<custom_app_name>\
-n <application_namespace>
```

Restaure desde un backup al espacio de nombres original

Puedes restaurar una copia de seguridad en el espacio de nombres original en cualquier momento. Cuando realizas una restaurar sin movimiento, Trident Protect gestiona automáticamente los programas de protección y las operaciones en curso para evitar puntos de recuperación no válidos:

- Todos los programas de protección activados para la aplicación se desactivan antes de que comience la restauración. Esto evita que se ejecuten copias de seguridad o instantáneas programadas mientras se restauran los recursos de la aplicación.
- Después de que la restauración se complete correctamente, solo se vuelven a habilitar las programaciones que estaban habilitadas antes de la restauración. Las programaciones que ya estaban deshabilitadas permanecen deshabilitadas.
- Cualquier operación de copia de seguridad o instantánea en curso se cancela antes de que comience la restauración. Si una operación no se cancela en 5 minutos, la restauración continúa y registra una advertencia en el estado de restauración CR.

Antes de empezar

Asegúrese de que la caducidad del token de sesión de AWS sea suficiente para las operaciones de restauración de S3 que se ejecuten durante mucho tiempo. Si el token caduca durante la operación de restauración, puede fallar la operación.

- Consulte el "Documentación de la API de AWS" para obtener más información sobre la comprobación de la caducidad del token de sesión actual.
- Consulte el documento para "Documentación de AWS IAM" obtener más información acerca de las credenciales con recursos de AWS.

NOTA

Al restaurar copias de seguridad utilizando Kopia como transportador de datos, puede especificar opcionalmente anotaciones en la CR o usar la CLI para controlar el comportamiento del almacenamiento temporal utilizado por Kopia. Consulte el "Documentación de KOPIA" Para obtener más información sobre las opciones que puede configurar. Utilice el `tridentctl-protect create --help` Comando para obtener más información sobre cómo especificar anotaciones con la CLI de Trident Protect.

Utilice un CR

Pasos

1. Cree el archivo de recursos personalizados (CR) y asígnele un nombre `trident-protect-backup-ipr-cr.yaml`.
2. En el archivo creado, configure los siguientes atributos:
   - **metadata.name**: (*required*) El nombre de este recurso personalizado; elija un nombre único y sensible para su entorno.
   - **Spec.appArchivePath**: La ruta dentro de AppVault donde se almacena el contenido de la copia de seguridad. Puede utilizar el siguiente comando para buscar esta ruta:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o jsonpath='{.status.appArchivePath}'
     ```

   - **Spec.appVaultRef**: (*required*) El nombre del AppVault donde se almacena el contenido de la copia de seguridad.

     Por ejemplo:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: BackupInplaceRestore
metadata:
  name: my-cr-name
  namespace: my-app-namespace
spec:
  appArchivePath: my-backup-path
  appVaultRef: appvault-name
```

3. (*Optional*) Si necesita seleccionar solo ciertos recursos de la aplicación para restaurar, agregue filtros que incluyan o excluyan recursos marcados con etiquetas particulares:

NOTA	Trident Protect selecciona algunos recursos automáticamente debido a su relación con los recursos que usted selecciona. Por ejemplo, si selecciona un recurso de reclamo de volumen persistente y tiene un pod asociado, Trident Protect también restaurará el pod asociado.

   - **ResourceFilter.resourceSelectionCriteria**: (Requerido para filtrar) Usar `Include` o `Exclude` incluir o excluir un recurso definido en resourceMatchers. Agregue los siguientes parámetros resourceMatchers para definir los recursos que se van a incluir o excluir:
     - **ResourceFilter.resourceMatchers**: Una matriz de objetos resourceMatcher. Si define varios elementos en esta matriz, coinciden como una OPERACIÓN OR y los campos dentro de cada elemento (grupo, tipo, versión) coinciden como una operación AND.
       - **ResourceMatchers[].group**: (*Optional*) Grupo del recurso a filtrar.
       - **ResourceMatchers[].kind**: (*Optional*) Tipo de recurso a filtrar.

- **ResourceMatchers[].version**: (*Optional*) Versión del recurso que se va a filtrar.
- **ResourceMatchers[].names**: (*Optional*) Nombres en el campo Kubernetes metadata.name del recurso que se va a filtrar.
- **ResourceMatchers[].namespaces**: (*Optional*) Espacios de nombres en el campo Kubernetes metadata.name del recurso que se va a filtrar.
- **ResourceMatchers[].labelSelectors**: (*Optional*) Cadena de selector de etiquetas en el campo Kubernetes metadata.name del recurso tal como se define en el "Documentación de Kubernetes". Por ejemplo `"trident.netapp.io/os=linux"`: .

  Por ejemplo:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Después de rellenar `trident-protect-backup-ipr-cr.yaml` el archivo con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-backup-ipr-cr.yaml
```

Utilice la CLI

Pasos

1. Restaure la copia de seguridad en el espacio de nombres original, sustituyendo valores entre paréntesis por información de su entorno. El `backup` argumento utiliza un espacio de nombres y un nombre de copia de seguridad en el formato `<namespace>/<name>`. Por ejemplo:

```
tridentctl-protect create backupinplacerestore <my_restore_name> \
--backup <namespace/backup_to_restore> \
-n <application_namespace>
```

Restauración desde un backup en otro clúster

Puede restaurar un backup a otro clúster si hay un problema con el clúster original.

NOTA

- Al restaurar copias de seguridad utilizando Kopia como transportador de datos, puede especificar opcionalmente anotaciones en la CR o usar la CLI para controlar el comportamiento del almacenamiento temporal utilizado por Kopia. Consulte el "Documentación de KOPIA" Para obtener más información sobre las opciones que puede configurar. Utilice el `tridentctl-protect create --help` Comando para obtener más información sobre cómo especificar anotaciones con la CLI de Trident Protect.
- Al utilizar una CR para restaurar a un nuevo espacio de nombres, debe crear manualmente el espacio de nombres de destino antes de aplicar la CR. Trident Protect crea espacios de nombres automáticamente solo cuando se usa la CLI.

Antes de empezar

Asegúrese de que se cumplen los siguientes requisitos previos:

- El clúster de destino tiene Trident Protect instalado.
- El clúster de destino tiene acceso a la ruta de bloque de la misma AppVault que el clúster de origen, en la que se almacena el backup.
- Asegúrese de que su entorno local pueda conectarse al bucket de almacenamiento de objetos definido en el CR de AppVault al ejecutar el proceso. `tridentctl-protect get appvaultcontent` dominio. Si las restricciones de red impiden el acceso, ejecute la CLI de Trident Protect desde dentro de un pod en el clúster de destino.
- Asegúrese de que la caducidad del token de sesión de AWS sea suficiente para las operaciones de restauración que se ejecuten durante mucho tiempo. Si el token caduca durante la operación de restauración, puede fallar la operación.
  - Consulte el "Documentación de la API de AWS" para obtener más información sobre la comprobación de la caducidad del token de sesión actual.
  - Consulte el documento para "Documentación de AWS" obtener más información acerca de las credenciales con recursos de AWS.

Pasos

1. Verifica que el AppVault CR existe en el clúster de destino usando el complemento CLI de Trident Protect:

```
tridentctl-protect get appvault --context <destination_cluster_name>
```

NOTA

Si el AppVault CR no existe en el clúster de destino, créalo siguiendo los pasos en "Utilice objetos Trident Protect AppVault para administrar depósitos".

2. Visualiza el contenido de la copia de seguridad disponible en AppVault en el clúster de destino y anota `appArchivePath` de la copia de seguridad que quieres restaurar:

```
tridentctl-protect get appvaultcontent <appvault_name> \
--show-resources backup \
--show-paths \
--context <destination_cluster_name>
```

Al ejecutar este comando, se muestran las copias de seguridad disponibles en AppVault, incluidos sus clústeres de origen, los nombres de aplicaciones correspondientes, las marcas de tiempo y las rutas de archivo.

Ejemplo de salida:

```
+-------------+-----------+--------+-----------------
+--------------------------+-------------+
|   CLUSTER   |    APP    |  TYPE  |      NAME       |         TIMESTAMP
|    PATH     |
+-------------+-----------+--------+-----------------
+--------------------------+-------------+
| production1 | wordpress | backup | wordpress-bkup-1| 2024-10-30
08:37:40 (UTC)| backuppath1 |
| production1 | wordpress | backup | wordpress-bkup-2| 2024-10-30
08:37:40 (UTC)| backuppath2 |
+-------------+-----------+--------+-----------------
+--------------------------+-------------+
```

3. Restaure la aplicación en el clúster de destino mediante el nombre de AppVault y la ruta de archivo:

NOTA Cuando utilices un CR, asegúrate de que el espacio de nombres previsto para la restauración de la aplicación existe en el clúster de destino.

Utilice un CR

1. Cree el archivo de recursos personalizados (CR) y asígnele un nombre `trident-protect-backup-restore-cr.yaml`.
2. En el archivo creado, configure los siguientes atributos:
   - **metadata.name**: (*required*) El nombre de este recurso personalizado; elija un nombre único y sensible para su entorno.
   - **Spec.appVaultRef**: (*required*) El nombre del AppVault donde se almacena el contenido de la copia de seguridad.
   - **spec.appArchivePath**: (*Obligatorio*) La ruta dentro de AppVault donde se almacena el contenido de la copia de seguridad. Usa el comando del paso 2 para ver el contenido de la copia de seguridad y encontrar `appArchivePath` la copia de seguridad que quieres restaurar.
   - **spec.destinationApplicationName**: (*opcional*) el nombre para la aplicación restaurada. Si se proporciona, la aplicación restaurada usa este nombre. Si no se proporciona, la aplicación restaurada usa el nombre de la aplicación de origen.
   - **spec.namespaceMapping**: La asignación del espacio de nombres de origen de la operación de restauración al espacio de nombres de destino. Reemplace `my-source-namespace` y `my-destination-namespace` con la información de su entorno.

     Por ejemplo:

```
apiVersion: protect.trident.netapp.io/v1
kind: BackupRestore
metadata:
  name: my-cr-name
  namespace: my-destination-namespace
spec:
  appVaultRef: appvault-name
  appArchivePath: my-backup-path
  destinationApplicationName: my-new-app-name
  namespaceMapping: [{"source": "my-source-namespace", "
destination": "my-destination-namespace"}]
```

3. Después de rellenar `trident-protect-backup-restore-cr.yaml` el archivo con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-backup-restore-cr.yaml
```

Utilice la CLI

1. Utilice el siguiente comando para restaurar la aplicación, sustituyendo valores entre paréntesis por información de su entorno. El argumento de asignación de espacio de nombres utiliza espacios de nombres separados por dos puntos para asignar espacios de nombres de origen a los espacios de nombres de destino correctos con el formato source1:DEST1,source2:DEST2. Por ejemplo:

```
tridentctl-protect create backuprestore <restore_name> \
--namespace-mapping <source_to_destination_namespace_mapping> \
--appvault <appvault_name> \
--path <backup_path> \
--destination-app-name <custom_app_name> \
--context <destination_cluster_name> \
-n <application_namespace>
```

Restauración desde una copia snapshot a un espacio de nombres diferente

Puede restaurar datos desde una instantánea utilizando un archivo de recursos personalizados (CR) ya sea en un espacio de nombres diferente o en el espacio de nombres de origen original. Cuando restaura una instantánea a un espacio de nombres diferente mediante un CR de SnapshotRestore, Trident Protect restaura la aplicación en un nuevo espacio de nombres y crea un CR de aplicación para la aplicación restaurada. Para proteger la aplicación restaurada, cree copias de seguridad o instantáneas bajo demanda, o establezca un programa de protección.

NOTA

- SnapshotRestore admite el `spec.storageClassMapping` atributo, pero solo cuando las clases de almacenamiento de origen y destino utilizan el mismo backend de almacenamiento. Si intenta restaurar a un `StorageClass` que utiliza un backend de almacenamiento diferente, la operación de restauración fallará.
- Al utilizar una CR para restaurar a un nuevo espacio de nombres, debe crear manualmente el espacio de nombres de destino antes de aplicar la CR. Trident Protect crea espacios de nombres automáticamente solo cuando se usa la CLI.

Antes de empezar

Asegúrese de que la caducidad del token de sesión de AWS sea suficiente para las operaciones de restauración de S3 que se ejecuten durante mucho tiempo. Si el token caduca durante la operación de restauración, puede fallar la operación.

- Consulte el "Documentación de la API de AWS" para obtener más información sobre la comprobación de la caducidad del token de sesión actual.
- Consulte el documento para "Documentación de AWS IAM" obtener más información acerca de las credenciales con recursos de AWS.

Utilice un CR

Pasos

1. Cree el archivo de recursos personalizados (CR) y asígnele un nombre `trident-protect-snapshot-restore-cr.yaml`.
2. En el archivo creado, configure los siguientes atributos:
   - **metadata.name**: (*required*) El nombre de este recurso personalizado; elija un nombre único y sensible para su entorno.
   - **Spec.appVaultRef**: (*required*) El nombre del AppVault donde se almacena el contenido de la instantánea.
   - **Spec.appArchivePath**: La ruta dentro de AppVault donde se almacena el contenido de la instantánea. Puede utilizar el siguiente comando para buscar esta ruta:

     ```
     kubectl get snapshots <SNAPHOT_NAME> -n my-app-namespace -o
     jsonpath='{.status.appArchivePath}'
     ```

   - **spec.destinationApplicationName**: (*opcional*) el nombre para la aplicación restaurada. Si se proporciona, la aplicación restaurada usa este nombre. Si no se proporciona, la aplicación restaurada usa el nombre de la aplicación de origen.
   - **spec.namespaceMapping**: La asignación del espacio de nombres de origen de la operación de restauración al espacio de nombres de destino. Reemplace `my-source-namespace` y `my-destination-namespace` con la información de su entorno.

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: SnapshotRestore
     metadata:
       name: my-cr-name
       namespace: my-destination-namespace
     spec:
       appVaultRef: appvault-name
       appArchivePath: my-snapshot-path
       namespaceMapping: [{"source": "my-source-namespace",
     "destination": "my-destination-namespace"}]
     ```

3. (*Optional*) Si necesita seleccionar solo ciertos recursos de la aplicación para restaurar, agregue filtros que incluyan o excluyan recursos marcados con etiquetas particulares:

   NOTA Trident Protect selecciona algunos recursos automáticamente debido a su relación con los recursos que usted selecciona. Por ejemplo, si selecciona un recurso de reclamo de volumen persistente y tiene un pod asociado, Trident Protect también restaurará el pod asociado.

   - **ResourceFilter.resourceSelectionCriteria**: (Requerido para filtrar) Usar `Include` o `Exclude` incluir o excluir un recurso definido en resourceMatchers. Agregue los siguientes parámetros

resourceMatchers para definir los recursos que se van a incluir o excluir:

- **ResourceFilter.resourceMatchers**: Una matriz de objetos resourceMatcher. Si define varios elementos en esta matriz, coinciden como una OPERACIÓN OR y los campos dentro de cada elemento (grupo, tipo, versión) coinciden como una operación AND.
  - **ResourceMatchers[].group**: (*Optional*) Grupo del recurso a filtrar.
  - **ResourceMatchers[].kind**: (*Optional*) Tipo de recurso a filtrar.
  - **ResourceMatchers[].version**: (*Optional*) Versión del recurso que se va a filtrar.
  - **ResourceMatchers[].names**: (*Optional*) Nombres en el campo Kubernetes metadata.name del recurso que se va a filtrar.
  - **ResourceMatchers[].namespaces**: (*Optional*) Espacios de nombres en el campo Kubernetes metadata.name del recurso que se va a filtrar.
  - **ResourceMatchers[].labelSelectors**: (*Optional*) Cadena de selector de etiquetas en el campo Kubernetes metadata.name del recurso tal como se define en el "Documentación de Kubernetes". Por ejemplo `"trident.netapp.io/os=linux"`: .

    Por ejemplo:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Después de rellenar `trident-protect-snapshot-restore-cr.yaml` el archivo con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-snapshot-restore-cr.yaml
```

Utilice la CLI

Pasos

1. Restaure la instantánea en un espacio de nombres diferente, reemplazando los valores entre paréntesis por información de su entorno.

- El `snapshot` argumento utiliza un espacio de nombres y un nombre de instantánea en el formato `<namespace>/<name>`.
- El `namespace-mapping` argumento utiliza espacios de nombres separados por dos puntos para asignar espacios de nombres de origen a los espacios de nombres de destino correctos en el formato `source1:dest1,source2:dest2`.

  Por ejemplo:

```
tridentctl-protect create snapshotrestore <my_restore_name> \
--snapshot <namespace/snapshot_to_restore> \
--namespace-mapping <source_to_destination_namespace_mapping> \
--destination-app-name <custom_app_name> \
-n <application_namespace>
```

Restauración desde una copia Snapshot al espacio de nombres original

Puedes restaurar una instantánea en el espacio de nombres original en cualquier momento. Cuando realizas una restaurar sin movimiento, Trident Protect gestiona automáticamente los programas de protección y las operaciones en curso para evitar puntos de recuperación no válidos:

- Todos los programas de protección activados para la aplicación se desactivan antes de que comience la restauración. Esto evita que se ejecuten copias de seguridad o instantáneas programadas mientras se restauran los recursos de la aplicación.
- Después de que la restauración se complete correctamente, solo se vuelven a habilitar las programaciones que estaban habilitadas antes de la restauración. Las programaciones que ya estaban deshabilitadas permanecen deshabilitadas.
- Cualquier operación de copia de seguridad o instantánea en curso se cancela antes de que comience la restauración. Si una operación no se cancela en 5 minutos, la restauración continúa y registra una advertencia en el estado de restauración CR.

Antes de empezar

Asegúrese de que la caducidad del token de sesión de AWS sea suficiente para las operaciones de restauración de S3 que se ejecuten durante mucho tiempo. Si el token caduca durante la operación de restauración, puede fallar la operación.

- Consulte el "Documentación de la API de AWS" para obtener más información sobre la comprobación de la caducidad del token de sesión actual.
- Consulte el documento para "Documentación de AWS IAM" obtener más información acerca de las credenciales con recursos de AWS.

Utilice un CR

Pasos

1. Cree el archivo de recursos personalizados (CR) y asígnele un nombre `trident-protect-snapshot-ipr-cr.yaml`.
2. En el archivo creado, configure los siguientes atributos:
   - **metadata.name**: (*required*) El nombre de este recurso personalizado; elija un nombre único y sensible para su entorno.
   - **Spec.appVaultRef**: (*required*) El nombre del AppVault donde se almacena el contenido de la instantánea.
   - **Spec.appArchivePath**: La ruta dentro de AppVault donde se almacena el contenido de la instantánea. Puede utilizar el siguiente comando para buscar esta ruta:

     ```
     kubectl get snapshots <SNAPSHOT_NAME> -n my-app-namespace -o jsonpath='{.status.appArchivePath}'
     ```

```
---
apiVersion: protect.trident.netapp.io/v1
kind: SnapshotInplaceRestore
metadata:
  name: my-cr-name
  namespace: my-app-namespace
spec:
  appVaultRef: appvault-name
  appArchivePath: my-snapshot-path
```

+ . (*Opcional*) Si necesitas seleccionar solo determinados recursos de la aplicación para restaurarlos, añade un filtro que incluya o excluya los recursos marcados con etiquetas concretas:

+ NOTA: Trident Protect selecciona automáticamente algunos recursos debido a su relación con los recursos que tú seleccionas. Por ejemplo, si seleccionas un recurso de reclamación de volumen persistente y este tiene un pod asociado, Trident Protect también restaurará el pod asociado.

+ * **resourceFilter.resourceSelectionCriteria**: (Obligatorio para el filtrado) Utiliza `Include` o `Exclude` para incluir o excluir un recurso definido en resourceMatchers. Agrega los siguientes parámetros resourceMatchers para definir los recursos que se van a incluir o excluir: **resourceFilter.resourceMatchers: Una matriz de objetos resourceMatcher. Si defines varios elementos en este array, se combinan como una operación OR, y los campos dentro de cada elemento (group, kind, version) se combinan como una operación AND.** * **resourceMatchers[].group**: (*Opcional*) Grupo del recurso que se va a filtrar. * **resourceMatchers[].kind: (*Opcional*) Tipo del recurso que se va a filtrar.** * **resourceMatchers[].version**: (*Opcional*) Versión del recurso que se va a filtrar. * **resourceMatchers[].names: (*Opcional*) Nombres en el campo metadata.name de Kubernetes del recurso que se va a filtrar.** * **resourceMatchers[].namespaces**: (*Opcional*) Namespaces en el campo metadata.name de Kubernetes del recurso que se va a filtrar. * **resourceMatchers[].labelSelectors**: (*Opcional*) Cadena de selector de etiquetas en el campo metadata.name de Kubernetes del recurso, tal

como se define en "Documentación de Kubernetes". Por ejemplo: `"trident.netapp.io/os=linux"`.

+ Por ejemplo:

+

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "Include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

+ . Después de que completes el archivo `trident-protect-snapshot-ipr-cr.yaml` con los valores correctos, aplica el CR:

+

```
kubectl apply -f trident-protect-snapshot-ipr-cr.yaml
```

Utilice la CLI

Pasos

1. Restaure la instantánea en el espacio de nombres original, reemplazando los valores entre paréntesis por información de su entorno. Por ejemplo:

```
tridentctl-protect create snapshotinplacerestore <my_restore_name> \
--snapshot <namespace/snapshot_to_restore> \
-n <application_namespace>
```

Compruebe el estado de una operación de restauración

Puede usar la línea de comandos para comprobar el estado de una operación de restauración en curso, que se completó o con errores.

Pasos

1. Utilice el siguiente comando para recuperar el estado de la operación de restauración, sustituyendo valores de entre corchetes con información de su entorno:

```
kubectl get backuprestore -n <namespace_name> <my_restore_cr_name> -o
jsonpath='{.status}'
```

Utilice la configuración de restauración avanzada de Trident Protect

Puede personalizar las operaciones de restauración utilizando configuraciones avanzadas, como anotaciones, configuraciones de espacios de nombres y opciones de almacenamiento para satisfacer sus requisitos específicos.

Etiquetas y anotaciones del espacio de nombres durante las operaciones de restauración y conmutación al nodo de respaldo

Durante las operaciones de restauración y conmutación al nodo de respaldo, se realizan etiquetas y anotaciones en el espacio de nombres de destino que coincidan con las etiquetas y anotaciones en el espacio de nombres de origen. Se añaden etiquetas o anotaciones del espacio de nombres origen que no existen en el espacio de nombres destino, y las etiquetas o anotaciones que ya existan se sobrescriben para que coincidan con el valor del espacio de nombres origen. Las etiquetas o anotaciones que sólo existen en el espacio de nombres de destino permanecen sin cambios.

NOTA Si utiliza Red Hat OpenShift, es importante tener en cuenta el papel fundamental que desempeñan las anotaciones de espacios de nombres en los entornos OpenShift. Las anotaciones de espacio de nombres garantizan que los pods restaurados cumplan con los permisos y las configuraciones de seguridad adecuados definidos por las restricciones de contexto de seguridad (SCC) de OpenShift y puedan acceder a los volúmenes sin problemas de permisos. Para obtener más información, consulte la"Documentación de restricciones de contexto de seguridad de OpenShift" .

Puede evitar que se sobrescriban anotaciones específicas en el espacio de nombres de destino mediante el establecimiento de la variable de entorno de Kubernetes `RESTORE_SKIP_NAMESPACE_ANNOTATIONS` antes de llevar a cabo la operación de restauración o conmutación por error. Por ejemplo:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect \
  --set-string
restoreSkipNamespaceAnnotations="{<annotation_key_to_skip_1>,<annotation_k
ey_to_skip_2>}" \
  --reuse-values
```

NOTA Al realizar una operación de restauración o conmutación por error, se tendrán en cuenta las anotaciones y etiquetas de espacio de nombres especificadas en `restoreSkipNamespaceAnnotations` y `restoreSkipNamespaceLabels` quedan excluidos de la operación de restauración o conmutación por error. Asegúrese de que estos ajustes se configuren durante la instalación inicial de Helm. Para obtener más información, consulte "Configurar ajustes adicionales del gráfico de timón Trident Protect".

Si instalaste la aplicación de origen usando Helm con el `--create-namespace` bandera, se le da un trato especial a la `name` Clave de etiqueta. Durante el proceso de restauración o conmutación por error, Trident Protect copia esta etiqueta en el espacio de nombres de destino, pero actualiza el valor al valor del espacio de nombres de destino si el valor del origen coincide con el espacio de nombres de origen. Si este valor no coincide con el espacio de nombres de origen, se copia al espacio de nombres de destino sin cambios.

Ejemplo

El siguiente ejemplo presenta un espacio de nombres de origen y destino, cada uno con anotaciones y etiquetas diferentes. Puede ver el estado del espacio de nombres de destino antes y después de la operación, así como cómo las anotaciones y etiquetas se combinan o sobrescriben en el espacio de nombres de destino.

Antes de la operación de restauración o conmutación por error

En la siguiente tabla se muestra el estado del ejemplo de espacios de nombres de origen y destino antes de la operación de restauración o conmutación por error:

Espacio de nombres	Anotaciones	Etiquetas
Espacio de nombres ns-1 (origen)	• anotación.uno/clave: “updatedvalue” • anotación.dos/clave: “verdadero”	• entorno=producción • cumplimiento=hipaa • name=ns-1
Espacio de nombres ns-2 (destino)	• anotación.uno/tecla: “verdadero” • anotación.tres/clave: “falso”	• role=base de datos

Después de la operación de restauración

En la siguiente tabla se muestra el estado del espacio de nombres de destino de ejemplo después de la operación de restauración o conmutación por error. Se han agregado algunas claves, algunas se han sobrescrito y la `name` etiqueta se ha actualizado para que coincida con el espacio de nombres de destino:

Espacio de nombres	Anotaciones	Etiquetas
Espacio de nombres ns-2 (destino)	• anotación.uno/clave: “updatedvalue” • anotación.dos/clave: “verdadero” • anotación.tres/clave: “falso”	• name=ns-2 • cumplimiento=hipaa • entorno=producción • role=base de datos

Campos admitidos

Esta sección describe campos adicionales disponibles para operaciones de restauración.

Mapeo de clases de almacenamiento

El `spec.storageClassMapping` El atributo define una asignación de una clase de almacenamiento presente en la aplicación de origen a una nueva clase de almacenamiento en el clúster de destino. Puede

usar esto al migrar aplicaciones entre clústeres con diferentes clases de almacenamiento o al cambiar el backend de almacenamiento para operaciones de BackupRestore.

Ejemplo:

```
storageClassMapping:
  - destination: "destinationStorageClass1"
    source: "sourceStorageClass1"
  - destination: "destinationStorageClass2"
    source: "sourceStorageClass2"
```

Anotaciones admitidas

Esta sección enumera las anotaciones compatibles para configurar diversos comportamientos en el sistema. Si el usuario no configura una anotación explícitamente, el sistema utilizará el valor predeterminado.

Anotación	Tipo	Descripción	Valor predeterminado
protect.trident.netapp.io/tiempo-de-espera-del-transportador-de-datos-sec	cadena	El tiempo máximo (en segundos) permitido para que la operación de transferencia de datos permanezca detenida.	"300"
protect.trident.netapp.io/kopia-content-cache-size-limit-mb	cadena	El límite de tamaño máximo (en megabytes) para el caché de contenido de Kopia.	"1000"
protect.trident.netapp.io/pvc-bind-timeout-sec	cadena	Tiempo máximo (en segundos) que se debe esperar para que cualquier PersistentVolumeClaim (PVC) recién creado llegue al `Bound` fase antes de que fallen las operaciones. Se aplica a todos los tipos de restauración de CR (BackupRestore, BackupInplaceRestore, SnapshotRestore, SnapshotInplaceRestore). Utilice un valor más alto si su backend de almacenamiento o clúster requiere con frecuencia más tiempo.	"1200" (20 minutos)

Modifica los recursos a medida que los restauras usando recursos personalizados

Las transformaciones de recursos te permiten modificar un recurso mientras se está restaurando. Esto resulta útil cuando la versión restaurada necesita ser diferente de la original, por ejemplo, al actualizar un nombre de host en un ConfigMap al restaurar una aplicación en un clúster diferente.

Cómo funciona la modificación de recursos

El campo `transformations` en `SnapshotRestore`, `BackupRestore`, `AppMirrorRelationship` y otros recursos de restauración te permite modificar los recursos de Kubernetes durante el proceso de restauración. Esto es útil cuando la aplicación restaurada necesita configurarse de manera diferente a la original, por

ejemplo, cambiando nombres de host, URLs de registros, límites de recursos o variables de entorno.

Las transformaciones de recursos usan operaciones "RFC 6902 JSON Patch" y rutas "RFC 6901 JSON Pointer" para seleccionar y modificar campos específicos dentro de los recursos de Kubernetes.

A continuación se muestra la estructura básica de un objeto de restaurar sin movimiento que contiene transformaciones de recursos:

```
apiVersion: protect.trident.netapp.io/v1
kind: SnapshotRestore
metadata:
  name: my-restore
  namespace: target-namespace
spec:
  appVaultRef: my-vault
  appArchivePath: /path/to/snapshot
  namespaceMapping:
    - source: source-ns
      destination: target-ns
  transformations:
    - resource:
        kind: ConfigMap       # Required: resource kind
        group: ""             # Optional: API group (empty for core
resources)
        version: ""           # Optional: API version
        name: ""              # Optional: specific resource name
      operations:
        - op: replace         # Operation type
          path: "/data/key"   # JSON Pointer path
          value: "new-value"  # New value (for add/replace/test)
```

Recursos compatibles

Puedes usar transformaciones de recursos con recursos que cumplan los siguientes criterios:

- **kind** (obligatorio): el tipo de recurso de Kubernetes (por ejemplo, `ConfigMap`, `Deployment`, `Pod`)
- **grupo** (opcional): el grupo de la API (por ejemplo, `apps`, `route.openshift.io`) — omítelo para los recursos principales
- **versión** (opcional): la versión de la API (por ejemplo, `v1`, `v1beta1`)
- **nombre** (opcional): aplica solo a un recurso específico por su nombre

NOTA	Actualmente no es posible modificar PersistentVolumeClaims ni los espacios de nombres.

Operaciones compatibles

Puedes utilizar las siguientes operaciones para modificar recursos: `add`, `copy`, `move`, `remove`, `replace`, y `test`.

Agrega un valor a un recurso

Usa la operación `add` para añadir un nuevo campo o valor en la ruta especificada. Puedes añadir datos a objetos o arrays.

Utilice un CR

El siguiente ejemplo añade un selector de nodo a un recurso `Deployment`:

```
transformations:
  - resource:
      kind: Deployment
    operations:
      - op: add
        path: "/spec/template/spec/nodeSelector"
        value:
          "topology.kubernetes.io/zone": "us-east-1a"
          disktype: "ssd"
```

Utilice la CLI

Usa el siguiente comando para ejecutar esta transformación usando la línea de comandos:

```
tridentctl-protect create snapshotrestore <restore_name> \
  --snapshot <namespace/snapshot-name> \
  --namespace-mapping <source-ns>:<dest-ns> \
  --transformation
'apps,v1,Deployment:add{"path":"/spec/template/spec/nodeSelector","valu
e":{"topology.kubernetes.io/zone":"us-east-1a","disktype":"ssd"}}'
```

Copia un valor dentro de un recurso

Usa la operación `copy` para copiar un valor de una ruta a otra dentro del mismo recurso. El origen no se modifica.

Utilice un CR

El siguiente ejemplo duplica una clave de datos para un objeto `ConfigMap`:

```
transformations:
  - resource:
      kind: ConfigMap
    operations:
      - op: copy
        from: "/data/source-key"
        path: "/data/backup-key"
```

Utilice la CLI

Usa el siguiente comando para ejecutar esta transformación usando la línea de comandos:

```
tridentctl-protect create snapshotrestore <restore_name> \
  --snapshot <namespace/snapshot-name> \
  --namespace-mapping <source-ns>:<dest-ns> \
  --transformation ',v1,ConfigMap:copy{"from":"/data/source-
key","path":"/data/backup-key"}'
```

Mover un valor dentro de un recurso

Usa la operación `move` para mover un valor de una ruta a otra dentro del mismo recurso. El origen se elimina y el valor se coloca en el destino.

Utilice un CR

El siguiente ejemplo cambia el nombre de una clave de datos para un objeto `ConfigMap`:

```
transformations:
  - resource:
      kind: ConfigMap
    operations:
      - op: move
        from: "/data/OLD_KEY"
        path: "/data/NEW_KEY"
```

Utilice la CLI

Usa el siguiente comando para ejecutar esta transformación usando la línea de comandos:

```
tridentctl-protect create snapshotrestore <restore_name> \
  --snapshot <namespace/snapshot-name> \
  --namespace-mapping <source-ns>:<dest-ns> \
  --transformation
',v1,ConfigMap:move{"from":"/data/OLD_KEY","path":"/data/NEW_KEY"}'
```

Eliminar un valor de un recurso

Usa la operación `remove` para eliminar un campo o valor en la ruta especificada.

Utilice un CR

El siguiente ejemplo elimina una anotación de un recurso `ConfigMap`:

```
transformations:
  - resource:
      kind: ConfigMap
    operations:
      - op: remove
        path: "/metadata/annotations/kubectl.kubernetes.io~1last-
applied-configuration"
```

NOTA En la ruta del ejemplo anterior, `~1` es la secuencia de escape del puntero JSON para `/`.

Utilice la CLI

Usa el siguiente comando para ejecutar esta transformación usando la línea de comandos:

```
tridentctl-protect create snapshotrestore <restore_name> \
  --snapshot <namespace/snapshot-name> \
  --namespace-mapping <source-ns>:<dest-ns> \
  --transformation
',v1,ConfigMap:remove{"path":"/metadata/annotations/kubectl.kubernetes.
io~1last-applied-configuration"}'
```

Sustituir un valor dentro de un recurso

Usa la operación `replace` para reemplazar un valor existente dentro de un recurso en la ruta especificada. La ruta JSON ya debe existir.

Utilice un CR

El siguiente ejemplo cambia el nombre de host de un objeto `Route`:

```
transformations:
  - resource:
      kind: Route
      group: route.openshift.io
    operations:
      - op: replace
        path: "/spec/host"
        value: "prod.example.com"
```

Utilice la CLI

Usa el siguiente comando para ejecutar esta transformación usando la línea de comandos:

```
tridentctl-protect create snapshotrestore <restore_name> \
  --snapshot <namespace/snapshot-name> \
  --namespace-mapping <source-ns>:<dest-ns> \
  --transformation
'route.openshift.io,v1,Route:replace{"path":"/spec/host","value":"prod.
example.com"}'
```

Probar la modificación de los recursos

Utiliza la operación `test` para comprobar que el valor de una ruta coincide con el valor esperado antes de que se apliquen las demás operaciones. Si la comprobación falla, se revierte toda la modificación.

Utilice un CR

El siguiente ejemplo actualiza `database-host` solo si `environment` está establecido en `staging`:

```
transformations:
  - resource:
      kind: ConfigMap
    operations:
      - op: test
        path: "/data/environment"
        value: "staging"
      - op: replace
        path: "/data/database-host"
        value: "prod-db.example.com"
```

Utilice la CLI

Usa el siguiente comando para ejecutar esta transformación usando la línea de comandos:

```
tridentctl-protect create snapshotrestore <restore_name> \
  --snapshot <namespace/snapshot-name> \
  --namespace-mapping <source-ns>:<dest-ns> \
  --transformation
',v1,ConfigMap:test{"path":"/data/environment","value":"staging"},repla
ce{"path":"/data/database-host","value":"prod-db.example.com"}'
```

Replicar aplicaciones utilizando NetApp SnapMirror y Trident Protect

Con Trident Protect, puede utilizar las capacidades de replicación asincrónica de la tecnología SnapMirror de NetApp para replicar cambios de datos y aplicaciones de un backend de almacenamiento a otro, en el mismo clúster o entre diferentes clústeres.

Etiquetas y anotaciones del espacio de nombres durante las operaciones de restauración y conmutación al nodo de respaldo

Durante las operaciones de restauración y conmutación al nodo de respaldo, se realizan etiquetas y anotaciones en el espacio de nombres de destino que coincidan con las etiquetas y anotaciones en el espacio de nombres de origen. Se añaden etiquetas o anotaciones del espacio de nombres origen que no existen en el espacio de nombres destino, y las etiquetas o anotaciones que ya existan se sobrescriben para que coincidan con el valor del espacio de nombres origen. Las etiquetas o anotaciones que sólo existen en el espacio de nombres de destino permanecen sin cambios.

NOTA Si utiliza Red Hat OpenShift, es importante tener en cuenta el papel fundamental que desempeñan las anotaciones de espacios de nombres en los entornos OpenShift. Las anotaciones de espacio de nombres garantizan que los pods restaurados cumplan con los permisos y las configuraciones de seguridad adecuados definidos por las restricciones de contexto de seguridad (SCC) de OpenShift y puedan acceder a los volúmenes sin problemas de permisos. Para obtener más información, consulte la"Documentación de restricciones de contexto de seguridad de OpenShift" .

Puede evitar que se sobrescriban anotaciones específicas en el espacio de nombres de destino mediante el establecimiento de la variable de entorno de Kubernetes `RESTORE_SKIP_NAMESPACE_ANNOTATIONS` antes de llevar a cabo la operación de restauración o conmutación por error. Por ejemplo:

```
helm upgrade trident-protect -n trident-protect netapp-trident-
protect/trident-protect \
  --set-string
restoreSkipNamespaceAnnotations="{<annotation_key_to_skip_1>,<annotation_k
ey_to_skip_2>}" \
  --reuse-values
```

NOTA Al realizar una operación de restauración o conmutación por error, se tendrán en cuenta las anotaciones y etiquetas de espacio de nombres especificadas en `restoreSkipNamespaceAnnotations` y `restoreSkipNamespaceLabels` quedan excluidos de la operación de restauración o conmutación por error. Asegúrese de que estos ajustes se configuren durante la instalación inicial de Helm. Para obtener más información, consulte "Configurar ajustes adicionales del gráfico de timón Trident Protect".

Si instalaste la aplicación de origen usando Helm con el `--create-namespace` bandera, se le da un trato especial a la `name` Clave de etiqueta. Durante el proceso de restauración o conmutación por error, Trident Protect copia esta etiqueta en el espacio de nombres de destino, pero actualiza el valor al valor del espacio de nombres de destino si el valor del origen coincide con el espacio de nombres de origen. Si este valor no coincide con el espacio de nombres de origen, se copia al espacio de nombres de destino sin cambios.

Ejemplo

El siguiente ejemplo presenta un espacio de nombres de origen y destino, cada uno con anotaciones y etiquetas diferentes. Puede ver el estado del espacio de nombres de destino antes y después de la operación, así como cómo las anotaciones y etiquetas se combinan o sobrescriben en el espacio de nombres de destino.

Antes de la operación de restauración o conmutación por error

En la siguiente tabla se muestra el estado del ejemplo de espacios de nombres de origen y destino antes de la operación de restauración o conmutación por error:

Espacio de nombres	Anotaciones	Etiquetas
Espacio de nombres ns-1 (origen)	• anotación.uno/clave: “updatedvalue” • anotación.dos/clave: “verdadero”	• entorno=producción • cumplimiento=hipaa • name=ns-1

Espacio de nombres	Anotaciones	Etiquetas
Espacio de nombres ns-2 (destino)	• anotación.uno/tecla: “verdadero” • anotación.tres/clave: “falso”	• role=base de datos

Después de la operación de restauración

En la siguiente tabla se muestra el estado del espacio de nombres de destino de ejemplo después de la operación de restauración o conmutación por error. Se han agregado algunas claves, algunas se han sobrescrito y la `name` etiqueta se ha actualizado para que coincida con el espacio de nombres de destino:

Espacio de nombres	Anotaciones	Etiquetas
Espacio de nombres ns-2 (destino)	• anotación.uno/clave: “updatedvalue” • anotación.dos/clave: “verdadero” • anotación.tres/clave: “falso”	• name=ns-2 • cumplimiento=hipaa • entorno=producción • role=base de datos

NOTA Puede configurar Trident Protect para congelar y descongelar sistemas de archivos durante operaciones de protección de datos. "Obtenga más información sobre cómo configurar la congelación del sistema de archivos con Trident Protect".

Ganchos de ejecución durante operaciones de conmutación por error e inversas

Al usar la relación AppMirror para proteger su aplicación, hay comportamientos específicos relacionados con los ganchos de ejecución que debe tener en cuenta durante las operaciones de conmutación por error y de reversión.

- Durante la conmutación por error, los ganchos de ejecución se copian automáticamente del clúster de origen al de destino. No es necesario volver a crearlos manualmente. Tras la conmutación por error, los ganchos de ejecución permanecen en la aplicación y se ejecutarán durante cualquier acción relevante.
- Durante la resincronización inversa, se eliminan todos los ganchos de ejecución existentes en la aplicación. Cuando la aplicación de origen se convierte en la aplicación de destino, estos ganchos de ejecución dejan de ser válidos y se eliminan para impedir su ejecución.

Para obtener más información sobre los ganchos de ejecución, consulte"Administrar los ganchos de ejecución de Trident Protect" .

Configurar una relación de replicación

La configuración de una relación de replicación implica lo siguiente:

- Elegir la frecuencia con la que desea que Trident Protect tome una instantánea de la aplicación (que incluye los recursos de Kubernetes de la aplicación, así como las instantáneas de volumen para cada uno de los volúmenes de la aplicación)
- Elegir el programa de replicación (incluye recursos de Kubernetes, así como datos de volumen persistentes)

- Establecer la hora para que se realice la snapshot

Pasos

1. En el clúster de origen, cree un AppVault para la aplicación de origen. Según su proveedor de almacenamiento, modifique un ejemplo en "Recursos personalizados de AppVault" Para adaptarse a su entorno:

Cree un AppVault con un CR

a. Cree el archivo de recursos personalizados (CR) y asígnele un nombre (por ejemplo, `trident-protect-appvault-primary-source.yaml`).

b. Configure los siguientes atributos:

- **metadata.name**: (*required*) El nombre del recurso personalizado de AppVault. Tome nota del nombre que elija, ya que otros archivos CR necesarios para una relación de replicación hacen referencia a este valor.
- **spec.providerConfig**: (*required*) almacena la configuración necesaria para acceder a AppVault utilizando el proveedor especificado. Elija un nombre de bucketName y cualquier otro detalle necesario para su proveedor. Tome nota de los valores que elija, ya que otros archivos CR necesarios para una relación de replicación hacen referencia a estos valores. Consulte "Recursos personalizados de AppVault"para ver ejemplos de CRS de AppVault con otros proveedores.
- **spec.providerCredentials**: (*required*) almacena referencias a cualquier credencial necesaria para acceder a AppVault utilizando el proveedor especificado.
  - **spec.providerCredentials.valueFromSecret**: (*required*) indica que el valor de la credencial debe provenir de un secreto.
    - **KEY**: (*REQUIRED*) La clave válida del secreto para seleccionar.
    - **Name**: (*required*) Nombre del secreto que contiene el valor para este campo. Debe estar en el mismo espacio de nombres.
  - **spec.providerCredentials.secretAccessKey**: (*required*) La clave de acceso utilizada para acceder al proveedor. El **name** debe coincidir con **spec.providerCredentials.valueFromSecret.name**.
- **spec.providerType**: (*required*) determina qué proporciona la copia de seguridad; por ejemplo, NetApp ONTAP S3, S3 genérico, Google Cloud o Microsoft Azure. Los posibles valores son los siguientes:
  - aws
  - azure
  - gcp
  - genérico-s3
  - ONTAP-s3
  - StorageGRID-s3

c. Después de rellenar `trident-protect-appvault-primary-source.yaml` el archivo con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-appvault-primary-source.yaml -n
trident-protect
```

Cree un AppVault con la CLI

a. Cree AppVault, reemplazando los valores entre paréntesis con información de su entorno:

```
tridentctl-protect create vault Azure <vault-name> --account
<account-name> --bucket <bucket-name> --secret <secret-name> -n
trident-protect
```

2. En el clúster de origen, cree la aplicación de origen CR:

Cree la aplicación de origen mediante un CR

a. Cree el archivo de recursos personalizados (CR) y asígnele un nombre (por ejemplo, `trident-protect-app-source.yaml`).

b. Configure los siguientes atributos:

- **metadata.name**: (*required*) El nombre del recurso personalizado de la aplicación. Tome nota del nombre que elija, ya que otros archivos CR necesarios para una relación de replicación hacen referencia a este valor.
- **spec.includedNamespaces**: (*required*) Una matriz de espacios de nombres y etiquetas asociadas. Utilice nombres de espacio de nombres y, opcionalmente, reduzca el ámbito de los espacios de nombres con etiquetas para especificar los recursos que existen en los espacios de nombres que se muestran aquí. El espacio de nombres de la aplicación debe formar parte de esta cabina.

  Ejemplo YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  name: my-app-name
  namespace: my-app-namespace
spec:
  includedNamespaces:
    - namespace: my-app-namespace
      labelSelector: {}
```

c. Después de rellenar `trident-protect-app-source.yaml` el archivo con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-app-source.yaml -n my-app-namespace
```

Cree la aplicación de origen mediante la CLI

a. Cree la aplicación de origen. Por ejemplo:

```
tridentctl-protect create app <my-app-name> --namespaces
<namespaces-to-be-included> -n <my-app-namespace>
```

3. Opcionalmente, en el clúster de origen, tome una instantánea de la aplicación de origen. Esta instantánea se utiliza como base para la aplicación en el clúster de destino. Si omite este paso, deberá esperar a que se ejecute la siguiente instantánea programada para tener una instantánea reciente. Para crear una instantánea bajo demanda, consulte "Crear una snapshot bajo demanda".

4. En el clúster de origen, cree la programación de replicación CR:

NOTA

Además del programa que se proporciona a continuación, se recomienda crear un programa de instantáneas diarias independiente con un período de retención de 7 días para mantener una instantánea común entre los clústeres de ONTAP emparejados. Esto garantiza que las instantáneas estén disponibles hasta por 7 días, pero el período de retención se puede personalizar según las necesidades del usuario.

Si se produce una conmutación por error, el sistema puede usar estas instantáneas hasta por 7 días para operaciones inversas. Este enfoque agiliza y optimiza el proceso, ya que solo se transfieren los cambios realizados desde la última instantánea, no todos los datos.

Si un programa existente para la aplicación ya cumple con los requisitos de retención deseados, no se requieren programas adicionales.

Cree la programación de replicación utilizando un CR.

a. Cree un programa de replicación para la aplicación de origen:

i. Cree el archivo de recursos personalizados (CR) y asígnele un nombre (por ejemplo, `trident-protect-schedule.yaml`).

ii. Configure los siguientes atributos:

- **metadata.name**: (*required*) El nombre del recurso personalizado de horario.
- **spec.appVaultRef**: (*Obligatorio*) Este valor debe coincidir con el campo metadata.name del AppVault para la aplicación de origen.
- **spec.applicationRef**: (*Obligatorio*) Este valor debe coincidir con el campo metadata.name del CR de la aplicación de origen.
- **Spec.backupRetention**: (*required*) Este campo es obligatorio y el valor debe establecerse en 0.
- **Spec.enabled**: Debe establecerse en true.
- **spec.granularity**: debe ser establecido en `Custom`.
- **Spec.recurrenceRule**: Define una fecha de inicio en la hora UTC y un intervalo de recurrencia.
- **Spec.snapshotRetention**: Debe establecerse en 2.

Ejemplo YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: Schedule
metadata:
  name: appmirror-schedule
  namespace: my-app-namespace
spec:
  appVaultRef: my-appvault-name
  applicationRef: my-app-name
  backupRetention: "0"
  enabled: true
  granularity: Custom
  recurrenceRule: |-
    DTSTART:20220101T000200Z
    RRULE:FREQ=MINUTELY;INTERVAL=5
  snapshotRetention: "2"
```

i. Después de rellenar `trident-protect-schedule.yaml` el archivo con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-schedule.yaml -n my-app-
namespace
```

Cree la programación de replicación mediante la CLI.

a. Cree la planificación de replicación, reemplazando los valores entre corchetes con información de su entorno:

```
tridentctl-protect create schedule --name appmirror-schedule
--app <my_app_name> --appvault <my_app_vault> --granularity
Custom --recurrence-rule <rule> --snapshot-retention
<snapshot_retention_count> -n <my_app_namespace>
```

Ejemplo:

```
tridentctl-protect create schedule --name appmirror-schedule
--app <my_app_name> --appvault <my_app_vault> --granularity
Custom --recurrence-rule  "DTSTART:20220101T000200Z
\nRRULE:FREQ=MINUTELY;INTERVAL=5" --snapshot-retention 2 -n
<my_app_namespace>
```

5. En el clúster de destino, cree una aplicación de origen AppVault CR idéntica a la aplicación AppVault CR que aplicó en el clúster de origen y asígnele el nombre (por ejemplo, `trident-protect-appvault-primary-destination.yaml`).
6. Aplicar el CR:

```
kubectl apply -f trident-protect-appvault-primary-destination.yaml -n
trident-protect
```

7. Cree un AppVault CR de destino para la aplicación de destino en el clúster de destino. Según su proveedor de almacenamiento, modifique un ejemplo en "Recursos personalizados de AppVault" Para adaptarse a su entorno:
   a. Cree el archivo de recursos personalizados (CR) y asígnele un nombre (por ejemplo, `trident-protect-appvault-secondary-destination.yaml`).
   b. Configure los siguientes atributos:
      - **metadata.name**: (*required*) El nombre del recurso personalizado de AppVault. Tome nota del nombre que elija, ya que otros archivos CR necesarios para una relación de replicación hacen referencia a este valor.
      - **spec.providerConfig**: (*required*) almacena la configuración necesaria para acceder a AppVault utilizando el proveedor especificado. Elija un `bucketName` y cualquier otro detalle necesario para su proveedor. Tome nota de los valores que elija, ya que otros archivos CR necesarios para una relación de replicación hacen referencia a estos valores. Consulte "Recursos personalizados de

AppVault"para ver ejemplos de CRS de AppVault con otros proveedores.

- **spec.providerCredentials**: (*required*) almacena referencias a cualquier credencial necesaria para acceder a AppVault utilizando el proveedor especificado.
  - **spec.providerCredentials.valueFromSecret**: (*required*) indica que el valor de la credencial debe provenir de un secreto.
    - **KEY**: (*REQUIRED*) La clave válida del secreto para seleccionar.
    - **Name**: (*required*) Nombre del secreto que contiene el valor para este campo. Debe estar en el mismo espacio de nombres.
  - **spec.providerCredentials.secretAccessKey**: (*required*) La clave de acceso utilizada para acceder al proveedor. El **name** debe coincidir con **spec.providerCredentials.valueFromSecret.name**.
- **spec.providerType**: (*required*) determina qué proporciona la copia de seguridad; por ejemplo, NetApp ONTAP S3, S3 genérico, Google Cloud o Microsoft Azure. Los posibles valores son los siguientes:
  - aws
  - azure
  - gcp
  - genérico-s3
  - ONTAP-s3
  - StorageGRID-s3

c. Después de rellenar `trident-protect-appvault-secondary-destination.yaml` el archivo con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-appvault-secondary-destination.yaml
-n trident-protect
```

8. En el clúster de destino, cree un archivo CR AppMirrorRelationship.

NOTA	Al utilizar una CR, cree manualmente el espacio de nombres de destino antes de aplicar la CR. Trident Protect crea espacios de nombres automáticamente solo cuando se usa la CLI.

Cree una AppMirrorRelationship con un CR

a. Cree el archivo de recursos personalizados (CR) y asígnele un nombre (por ejemplo, `trident-protect-relationship.yaml`).

b. Configure los siguientes atributos:

- **metadata.name:** (requerido) El nombre del recurso personalizado AppMirrorRelationship.
- **spec.destinationAppVaultRef**: (*required*) Este valor debe coincidir con el nombre de AppVault para la aplicación de destino en el clúster de destino.
- **spec.namespaceMapping**: (*required*) Los espacios de nombres de destino y origen deben coincidir con el espacio de nombres de aplicación definido en la aplicación CR respectiva.
- **Spec.sourceAppVaultRef**: (*required*) Este valor debe coincidir con el nombre de AppVault para la aplicación de origen.
- **Spec.sourceApplicationName**: (*required*) Este valor debe coincidir con el nombre de la aplicación de origen que definió en la aplicación de origen CR.
- **spec.sourceApplicationUID**: (Obligatorio) Este valor debe coincidir con el UID de la aplicación de origen que definió en el CR de la aplicación de origen.
- **spec.storageClassName**: (*Opcional*) Elija el nombre de una clase de almacenamiento válida en el clúster. La clase de almacenamiento debe estar vinculada a una máquina virtual de almacenamiento ONTAP que esté emparejada con el entorno de origen. Si no se proporciona la clase de almacenamiento, se utilizará por defecto la clase de almacenamiento predeterminada del clúster.
- **spec.storageClassMapping**: (*Opcional*) Asigna las clases de almacenamiento de origen a las de destino para PVC individuales dentro del mismo AMR. Cada entrada requiere un `source` y un nombre de clase de almacenamiento `destination`. Cuando se configura, `storageClassMapping` anula por completo `storageClassName`; los PVC que no tengan una entrada de origen coincidente usan la clase de almacenamiento predeterminada del clúster, no `storageClassName`.

  Al usar `storageClassMapping` en un CR de AppMirrorRelationship, ten en cuenta las siguientes restricciones:

  - `storageClassMapping` anula `storageClassName` por completo. Cuando ambos campos están configurados, `storageClassName` se ignora.
  - Los PVC con clases de almacenamiento de origen que no tienen una entrada correspondiente en `storageClassMapping` usan la clase de almacenamiento predeterminada del clúster, no `storageClassName`.
  - Cada entrada requiere tanto un `source` como un `destination` valor.
  - Cambiar la clase de almacenamiento de destino puede provocar la eliminación y recreación de los PVC de destino existentes.
- **Spec.recurrenceRule**: Define una fecha de inicio en la hora UTC y un intervalo de recurrencia.

  Ejemplo YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: AppMirrorRelationship
metadata:
  name: amr-16061e80-1b05-4e80-9d26-d326dc1953d8
  namespace: my-app-namespace
spec:
  desiredState: Established
  destinationAppVaultRef: generic-s3-trident-protect-dst-bucket-
8fe0b902-f369-4317-93d1-ad7f2edc02b5
  namespaceMapping:
    - destination: my-app-namespace
      source: my-app-namespace
  recurrenceRule: |-
    DTSTART:20220101T000200Z
    RRULE:FREQ=MINUTELY;INTERVAL=5
  sourceAppVaultRef: generic-s3-trident-protect-src-bucket-
b643cc50-0429-4ad5-971f-ac4a83621922
  sourceApplicationName: my-app-name
  sourceApplicationUID: 7498d32c-328e-4ddd-9029-122540866aeb
  storageClassName: sc-vsim-2
```

c. Después de rellenar `trident-protect-relationship.yaml` el archivo con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-
namespace
```

Cree una AppMirrorRelationship con la CLI

a. Crea y aplica el objeto AppMirrorRelationship, reemplazando los valores entre corchetes con información de tu entorno:

```
tridentctl-protect create appmirrorrelationship
<name_of_appmirorrelationship> --destination-app-vault
<my_vault_name> --source-app-vault <my_vault_name> --recurrence
-rule <rule> --namespace-mapping <ns_mapping> --source-app-id
<source_app_UID> --source-app <my_source_app_name> --storage
-class <storage_class_name> --storageclass-mapping
<src_class:dst_class> -n <application_namespace>
```

NOTA	Usa `--storageclass-mapping` para asignar clases de almacenamiento de origen individuales a clases de almacenamiento de destino. Especifica uno o más pares `source:destination` separados por comas (por ejemplo, `"class-a:class-b,class-c:class-d"`). Usa `*:destination-class` como comodín para asignar todos los PVC sin asignar a una sola clase de almacenamiento de destino. Cuando `--storageclass-mapping` está configurado, `--storage-class` se ignora.

Ejemplo con una única clase de almacenamiento de destino (`--storage-class`):

```
tridentctl-protect create appmirrorrelationship my-amr
--destination-app-vault appvault2 --source-app-vault appvault1
--recurrence-rule
"DTSTART:20220101T000200Z\nRRULE:FREQ=MINUTELY;INTERVAL=5"
--source-app my-app --namespace-mapping "my-source-ns1:my-dest-
ns1,my-source-ns2:my-dest-ns2" --source-app-id 373f24c1-5769-
404c-93c3-5538af6ccc36 --storage-class my-storage-class -n my-
dest-ns1
```

9. (*Optional*) En el clúster de destino, compruebe el estado y el estado de la relación de replicación:

```
kubectl get amr -n my-app-namespace <relationship name> -o=jsonpath
='{.status}' | jq
```

Conmutación por error al clúster de destino

Con Trident Protect, puede conmutar por error las aplicaciones replicadas a un clúster de destino. Este procedimiento detiene la relación de replicación y pone la aplicación en línea en el clúster de destino. Trident Protect no detiene la aplicación en el clúster de origen si estaba operativa.

Pasos

1. En el clúster de destino, edite el archivo CR de AppMirrorRelationship (por ejemplo, `trident-protect-relationship.yaml`) y cambie el valor de **spec.desiredState** a `Promoted`.
2. Guarde el archivo CR.
3. Aplicar el CR:

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-namespace
```

4. (*Optional*) Cree cualquier programación de protección que necesite en la aplicación con fallos.
5. (*Optional*) Compruebe el estado y el estado de la relación de replicación:

```
kubectl get amr -n my-app-namespace <relationship name> -o=jsonpath
='{.status}' | jq
```

Resincronizar una relación de replicación con fallo

La operación de resincronización vuelve a establecer la relación de replicación. Después de realizar una operación de resincronización, la aplicación de origen original se convierte en la aplicación en ejecución y se descartan todos los cambios realizados en la aplicación en ejecución en el clúster de destino.

El proceso detiene la aplicación en el clúster de destino antes de restablecer la replicación.

IMPORTANTE	Se perderán todos los datos escritos en la aplicación de destino durante la conmutación al respaldo.

Pasos

1. Opcional: En el clúster de origen, cree una copia Snapshot de la aplicación de origen. De esta forma se garantiza que se capturen los cambios más recientes del clúster de origen.
2. En el clúster de destino, edite el archivo CR de AppMirrorRelationship (por ejemplo, `trident-protect-relationship.yaml`) y cambie el valor de spec.desiredState a `Established`.
3. Guarde el archivo CR.
4. Aplicar el CR:

```
kubectl apply -f trident-protect-relationship.yaml -n my-app-namespace
```

5. Si ha creado cualquier programación de protección en el clúster de destino para proteger la aplicación con errores, elimínela. Cualquier programación que permanezca provoca errores de snapshots de volumen.

Resincronización inversa de una relación de replicación fallida

Cuando se realiza una resincronización inversa de una relación de replicación fallida, la aplicación de destino se convierte en la aplicación de origen y el origen se convierte en el destino. Se mantienen los cambios realizados en la aplicación de destino durante la conmutación por error.

Pasos

1. En el clúster de destino original, elimine el CR de AppMirrorRelationship. Esto hace que el destino se convierta en el origen. Si queda alguna programación de protección en el nuevo clúster de destino, elimínela.
2. Configure una relación de replicación aplicando los archivos CR que utilizó originalmente para configurar la relación con los clusters opuestos.
3. Asegúrese de que el nuevo destino (cluster de origen original) está configurado con los CRS de AppVault.
4. Configure una relación de replicación en el cluster opuesto, configurando valores para la dirección inversa.

Invertir dirección de replicación de aplicaciones

Cuando invierte la dirección de replicación, Trident Protect mueve la aplicación al backend de almacenamiento de destino mientras continúa replicando al backend de almacenamiento de origen original. Trident Protect

detiene la aplicación de origen y replica los datos en el destino antes de conmutar por error a la aplicación de destino.

En esta situación, está intercambiando el origen y el destino.

Pasos

1. En el clúster de origen, cree una snapshot de apagado:

Cree una instantánea de cierre con un CR

a. Desactive las programaciones de políticas de protección para la aplicación de origen.

b. Crear un archivo CR de ShutdownSnapshot:

   i. Cree el archivo de recursos personalizados (CR) y asígnele un nombre (por ejemplo, `trident-protect-shutdownsnapshot.yaml`).

   ii. Configure los siguientes atributos:

   - **metadata.name**: (*required*) El nombre del recurso personalizado.
   - **Spec.AppVaultRef**: (*required*) Este valor debe coincidir con el campo metadata.name del AppVault para la aplicación de origen.
   - **Spec.ApplicationRef**: (*required*) Este valor debe coincidir con el campo metadata.name del archivo CR de la aplicación de origen.

     Ejemplo YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: ShutdownSnapshot
metadata:
  name: replication-shutdown-snapshot-afc4c564-e700-4b72-86c3-
c08a5dbe844e
  namespace: my-app-namespace
spec:
  appVaultRef: generic-s3-trident-protect-src-bucket-04b6b4ec-
46a3-420a-b351-45795e1b5e34
  applicationRef: my-app-name
```

c. Después de rellenar `trident-protect-shutdownsnapshot.yaml` el archivo con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-shutdownsnapshot.yaml -n my-app-
namespace
```

Cree una snapshot apagada con la CLI

a. Cree la instantánea de cierre, reemplazando valores entre paréntesis con información de su entorno. Por ejemplo:

```
tridentctl-protect create shutdownsnapshot <my_shutdown_snapshot>
--appvault <my_vault> --app <app_to_snapshot> -n
<application_namespace>
```

2. En el clúster de origen, cuando se complete la snapshot de apagado, obtenga el estado de la snapshot de apagado:

```
kubectl get shutdownsnapshot -n my-app-namespace
<shutdown_snapshot_name> -o yaml
```

3. En el clúster de origen, busque el valor de **shutdownsnapshot.status.appArchivePath** usando el siguiente comando, y registre la última parte de la ruta del archivo (también llamada nombre base; esto será todo después de la última barra diagonal):

```
k get shutdownsnapshot -n my-app-namespace <shutdown_snapshot_name> -o
jsonpath='{.status.appArchivePath}'
```

4. Realice una conmutación por error del nuevo clúster de destino al nuevo clúster de origen, con el siguiente cambio:

NOTA	En el paso 2 del procedimiento de conmutación por error, incluya el `spec.promotedSnapshot` campo en el archivo AppMirrorRelationship CR y establezca su valor en el nombre base que registró en el paso 3 anterior.

5. Realice los pasos de resincronización inversa en Resincronización inversa de una relación de replicación fallida.
6. Habilite las programaciones de protección en el nuevo clúster de origen.

Resultado

Las siguientes acciones se producen debido a la replicación inversa:

- Se toma una instantánea de los recursos de Kubernetes de la aplicación de origen original.
- Los pods de la aplicación de origen originales se detienen con dignidad al eliminar los recursos de Kubernetes de la aplicación (dejando las RVP y los VP en funcionamiento).
- Después de que los pods se cierran, se toman y replican instantáneas de los volúmenes de la aplicación.
- Las relaciones de SnapMirror se rompen, lo que hace que los volúmenes de destino estén listos para la lectura/escritura.
- Los recursos de Kubernetes de la aplicación se restauran a partir de la instantánea previa al cierre, utilizando los datos del volumen replicados después de que se cerró la aplicación de origen original.
- La replicación se restablece en la dirección inversa.

Conmutación tras error de las aplicaciones al clúster de origen original

Al utilizar Trident Protect, puede lograr una "conmutación por error" después de una operación de conmutación por error mediante la siguiente secuencia de operaciones. En este flujo de trabajo para restaurar la dirección de replicación original, Trident Protect replica (resincroniza) cualquier cambio de aplicación a la aplicación de origen original antes de revertir la dirección de replicación.

Este proceso se inicia desde una relación que ha completado una conmutación al nodo de respaldo a un destino e implica los siguientes pasos:

- Comience con un estado de conmutación al respaldo.
- Vuelva a sincronizar la relación de replicación.

PRECAUCIÓN	No realice una operación de resincronización normal, ya que esto descartará los datos escritos en el clúster de destino durante el procedimiento de conmutación por error.

- Invierta la dirección de replicación.

Pasos

1. Realice Resincronización inversa de una relación de replicación fallidalos pasos.
2. Realice Invertir dirección de replicación de aplicacioneslos pasos.

Eliminar una relación de replicación

Puede eliminar una relación de replicación en cualquier momento. Al eliminar la relación de replicación de la aplicación, se crean dos aplicaciones independientes sin relación entre ellas.

Pasos

1. En el clúster de eliminación actual, elimine el CR de AppMirrorRelationship:

```
kubectl delete -f trident-protect-relationship.yaml -n my-app-namespace
```

Migrar aplicaciones usando Trident Protect

Puede migrar sus aplicaciones entre clústeres o a diferentes clases de almacenamiento restaurando datos de respaldo.

NOTA	Al migrar una aplicación, todos los ganchos de ejecución configurados para la aplicación se migran con la aplicación. Si hay un enlace de ejecución posterior a la restauración, se ejecuta automáticamente como parte de la operación de restauración.

Operaciones de backup y restauración

Para realizar operaciones de backup y restauración para las siguientes situaciones, se pueden automatizar tareas de backup y restauración específicas.

Clone en el mismo clúster

Para clonar una aplicación en el mismo clúster, cree una copia Snapshot o un backup y restaure los datos en el mismo clúster.

Pasos

1. Debe realizar una de las siguientes acciones:
   a. "Crear una copia de Snapshot".
   b. "Cree un backup".
2. En el mismo clúster, realice una de las siguientes acciones, según si se ha creado una snapshot o un backup:

a. "Restaure los datos desde la copia Snapshot".

b. "Restaure los datos del backup".

Clone en otro clúster

Para clonar una aplicación en un clúster diferente (realizar una clonación entre clústeres), cree una copia de seguridad en el clúster de origen y, a continuación, restaure la copia de seguridad en un clúster diferente. Asegúrese de que Trident Protect esté instalado en el clúster de destino.

NOTA	Puede replicar una aplicación entre diferentes clusters mediante "Replicación de SnapMirror".

Pasos

1. "Cree un backup".
2. Asegúrese de que el CR de AppVault para el depósito de almacenamiento de objetos que contiene la copia de seguridad se haya configurado en el clúster de destino.
3. En el clúster de destino, "restaure los datos del backup".

Migre aplicaciones de una clase de almacenamiento a otra clase de almacenamiento

Puede migrar aplicaciones de una clase de almacenamiento a una clase de almacenamiento diferente restaurando una copia de seguridad en la clase de almacenamiento de destino.

Por ejemplo (excluyendo los secretos de la CR de restauración):

```
apiVersion: protect.trident.netapp.io/v1
kind: SnapshotRestore
metadata:
  name: "${snapshotRestoreCRName}"
spec:
  appArchivePath: "${snapshotArchivePath}"
  appVaultRef: "${appVaultCRName}"
  namespaceMapping:
    - destination: "${destinationNamespace}"
      source: "${sourceNamespace}"
  storageClassMapping:
    - destination: "${destinationStorageClass}"
      source: "${sourceStorageClass}"
  resourceFilter:
    resourceMatchers:
      kind: Secret
      version: v1
    resourceSelectionCriteria: exclude
```

Restaure la instantánea con un CR

Pasos

1. Cree el archivo de recursos personalizados (CR) y asígnele un nombre `trident-protect-snapshot-restore-cr.yaml`.
2. En el archivo creado, configure los siguientes atributos:
   - **metadata.name**: (*required*) El nombre de este recurso personalizado; elija un nombre único y sensible para su entorno.
   - **Spec.appArchivePath**: La ruta dentro de AppVault donde se almacena el contenido de la instantánea. Puede utilizar el siguiente comando para buscar esta ruta:

     ```
     kubectl get snapshots <my-snapshot-name> -n trident-protect -o jsonpath='{.status.appArchivePath}'
     ```

   - **Spec.appVaultRef**: (*required*) El nombre del AppVault donde se almacena el contenido de la instantánea.
   - **spec.namespaceMapping**: La asignación del espacio de nombres de origen de la operación de restauración al espacio de nombres de destino. Reemplace `my-source-namespace` y `my-destination-namespace` con la información de su entorno.

     ```
     ---
     apiVersion: protect.trident.netapp.io/v1
     kind: SnapshotRestore
     metadata:
       name: my-cr-name
       namespace: trident-protect
     spec:
       appArchivePath: my-snapshot-path
       appVaultRef: appvault-name
       namespaceMapping: [{"source": "my-source-namespace",
     "destination": "my-destination-namespace"}]
     ```

3. Opcionalmente, si necesita seleccionar solo ciertos recursos de la aplicación para restaurar, agregue filtros que incluyan o excluyan recursos marcados con etiquetas concretas:
   - **ResourceFilter.resourceSelectionCriteria**: (Requerido para filtrar) Usa `include or exclude` para incluir o excluir un recurso definido en resourceMatchers. Agregue los siguientes parámetros resourceMatchers para definir los recursos que se van a incluir o excluir:
     - **ResourceFilter.resourceMatchers**: Una matriz de objetos resourceMatcher. Si define varios elementos en esta matriz, coinciden como una OPERACIÓN OR y los campos dentro de cada elemento (grupo, tipo, versión) coinciden como una operación AND.
       - **ResourceMatchers[].group**: (*Optional*) Grupo del recurso a filtrar.
       - **ResourceMatchers[].kind**: (*Optional*) Tipo de recurso a filtrar.
       - **ResourceMatchers[].version**: (*Optional*) Versión del recurso que se va a filtrar.

- **ResourceMatchers[].names**: (*Optional*) Nombres en el campo Kubernetes metadata.name del recurso que se va a filtrar.
- **ResourceMatchers[].namespaces**: (*Optional*) Espacios de nombres en el campo Kubernetes metadata.name del recurso que se va a filtrar.
- **ResourceMatchers[].labelSelectors**: (*Optional*) Cadena de selector de etiquetas en el campo Kubernetes metadata.name del recurso tal como se define en el "Documentación de Kubernetes". Por ejemplo `"trident.netapp.io/os=linux"`: .

  Por ejemplo:

```
spec:
  resourceFilter:
    resourceSelectionCriteria: "include"
    resourceMatchers:
      - group: my-resource-group-1
        kind: my-resource-kind-1
        version: my-resource-version-1
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
      - group: my-resource-group-2
        kind: my-resource-kind-2
        version: my-resource-version-2
        names: ["my-resource-names"]
        namespaces: ["my-resource-namespaces"]
        labelSelectors: ["trident.netapp.io/os=linux"]
```

4. Después de rellenar `trident-protect-snapshot-restore-cr.yaml` el archivo con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-snapshot-restore-cr.yaml
```

Restaure la instantánea mediante la interfaz de línea de comandos

Pasos

1. Restaure la instantánea en un espacio de nombres diferente, reemplazando los valores entre paréntesis por información de su entorno.

   - El `snapshot` argumento utiliza un espacio de nombres y un nombre de instantánea en el formato `<namespace>/<name>`.
   - El `namespace-mapping` argumento utiliza espacios de nombres separados por dos puntos para asignar espacios de nombres de origen a los espacios de nombres de destino correctos en el formato `source1:dest1,source2:dest2`.

     Por ejemplo:

```
tridentctl-protect create snapshotrestore <my_restore_name>
--snapshot <namespace/snapshot_to_restore> --namespace-mapping
<source_to_destination_namespace_mapping>
```

Protege las máquinas virtuales de KubeVirt con Trident Protect

Trident Protect te permite hacer copias de seguridad y restaurar máquinas virtuales (VM) de KubeVirt que se ejecutan en OpenShift Virtualization. Puedes proteger todas las VMs en uno o varios espacios de nombres a la vez, o usar el campo `includedVirtualMachines` para seleccionar VMs específicas por nombre. Trident Protect recopila automáticamente todos los recursos dependientes de las VMs durante la copia de seguridad y, al restaurar, puedes recuperar todo el archivo o restaurar de forma selectiva VMs individuales.

KubeVirt modos de protección

Trident Protect admite dos formas de definir qué máquinas virtuales de KubeVirt se deben proteger. Elige uno de estos modos al definir una aplicación:

Modo	Campo	Descripción
Basado en namespace	`includedNamespaces`	Protege todos los recursos coincidentes en uno o varios espacios de nombres. Usa un `labelSelector` en la entrada del espacio de nombres para controlar qué máquinas virtuales se incluyen dinámicamente.
De ámbito de la máquina virtual	`includedVirtualMachines`	Protege máquinas virtuales específicas de KubeVirt por espacio de nombres y nombre. Utiliza este modo cuando las máquinas virtuales estén repartidas por distintos espacios de nombres o cuando no desees proteger todos los recursos del espacio de nombres.

Debes especificar exactamente un campo (`includedNamespaces` o `includedVirtualMachines`) en el `ApplicationSpec`. No puedes utilizar ambos en la misma aplicación.

Ambos modos admiten filtros adicionales:

- Utiliza `resourceFilter` para incluir o excluir recursos específicos.
- Utiliza `includedClusterScopedResources` para incluir recursos compatibles con el ámbito del clúster, como objetos `StorageClass`.

NOTA Trident Protect puede congelar y descongelar los sistemas de archivos de las máquinas virtuales de KubeVirt durante las operaciones de protección de datos para garantizar la coherencia. Para obtener detalles de configuración, consulta "Protección de datos con máquinas virtuales de KubeVirt".

Recursos recopilados durante la copia de seguridad y la instantánea

Cuando creas una copia de seguridad o una instantánea para una aplicación que incluye KubeVirt VMs, Trident Protect recopila los recursos de las VMs seleccionadas y descubre automáticamente sus recursos dependientes.

El comportamiento de la recopilación se aplica en dos casos:

- Aplicaciones con ámbito de máquina virtual que utilizan `includedVirtualMachines`.
- Aplicaciones basadas en espacios de nombres que contienen recursos de KubeVirt `VirtualMachine`.

Recopilación con ámbito de máquina virtual

En el caso de las aplicaciones con ámbito de máquina virtual, Trident Protect comienza con cada `VirtualMachine` listado en `includedVirtualMachines` y descubre los recursos dependientes necesarios para restaurar esa máquina virtual.

Tipo de recurso	Ámbito	Cómo se descubre
`VirtualMachine`	Con espacio de nombres	Coincidencia directa de la `includedVirtualMachines` lista.
`VirtualMachineInstance`	Con espacio de nombres	El mismo espacio de nombres y nombre que la máquina virtual, si la máquina virtual está en ejecución.
`VirtualMachineInstancetype`	Con espacio de nombres	Citado por `spec.instancetype` cuando `kind: VirtualMachineInstancetype`.
`VirtualMachineClusterInstancetype`	Clúster	Referenciado por `spec.instancetype` cuando `kind` no está definido o `kind: VirtualMachineClusterInstancetype`.
`VirtualMachinePreference`	Con espacio de nombres	Referenciado por `spec.preference` cuando `kind: VirtualMachinePreference`.
`VirtualMachineClusterPreference`	Clúster	Referenciado por `spec.preference` cuando `kind` no está definido o `kind: VirtualMachineClusterPreference`.
`DataVolume`	Con espacio de nombres	Citado en `spec.template.spec.volumes[].dataVolume.name`.
`PersistentVolumeClaim`	Con espacio de nombres	Descubierto a partir de PVCs respaldados por DataVolume, que usan el mismo nombre que el DataVolume; referencias directas a PVC en `persistentVolumeClaim.claimName`; PVCs de volcado de memoria en `memoryDump.claimName`; y PVCs efímeros en `ephemeral.persistentVolumeClaim.claimName`.
`PersistentVolume`	Clúster	Detectado a partir del PV de respaldo para cada PVC recopilado usando `spec.volumeName`.

Tipo de recurso	Ámbito	Cómo se descubre
`Secret`	Con espacio de nombres	Descubiertos a partir de los secretos de volumen en `secret.secretName`; los secretos de descarga de imágenes en containerDisk; las referencias a secretos de datos de usuario y de red en CloudInitNoCloud y CloudInitConfigDrive; los secretos de Sysprep; los secretos de descarga de imágenes de contenedores en KernelBoot; y los secretos de SSH o contraseñas en AccessCredentials.
`ConfigMap`	Con espacio de nombres	Detectado desde el volumen ConfigMaps en `configMap.name` y Sysprep ConfigMaps.
`ServiceAccount`	Con espacio de nombres	Descubierto a partir de referencias de volúmenes `serviceAccount.serviceAccountName`.
`NetworkAttachmentDefinition`	Con espacio de nombres	Se detectan a partir de las redes Multus en `spec.template.spec.networks[].multus.networkName` cuando el nombre de la red no contiene `/`. Solo se recogen las referencias de NetworkAttachmentDefinition que pertenecen al mismo espacio de nombres. Las referencias entre espacios de nombres se omiten.
`Pod`	Con espacio de nombres	Detectado para `virt-launcher` Pods usando el selector de etiquetas `kubevirt.io=virt-launcher,vm.kubevirt.io/name=<vmName>`.
Clase de almacenamiento, volúmenes Trident y backends Trident	Clúster	Se recopilan de la misma forma que las aplicaciones basadas en espacios de nombres, a partir de los PVC detectados para la VM.

NOTA Una vez recopilados todos los recursos con espacio de nombres de ámbito de máquina virtual, Trident Protect aplica la estándar `resourceFilter`, si está configurada. Luego, la recopilación de recursos con ámbito de clúster continúa con normalidad.

Colección basada en espacios de nombres

En el caso de las aplicaciones basadas en espacios de nombres, Trident Protect analiza los espacios de nombres seleccionados y recopila todos los recursos que coinciden. Si se encuentran objetos KubeVirt `VirtualMachine`, Trident Protect recopila automáticamente sus recursos dependientes, incluidos los tipos de instancia, las preferencias y los Pods `virt-launcher`, del mismo modo que lo hace con las aplicaciones con ámbito de máquina virtual.

Esto significa que los recursos específicos de la máquina virtual se incluyen en tu copia de seguridad o instantánea, incluso cuando la aplicación se define con `includedNamespaces`.

Protege máquinas virtuales usando definiciones de aplicaciones basadas en espacios de nombres

Al crear una aplicación basada en espacios de nombres, Trident Protect analiza los espacios de nombres seleccionados y recopila todos los recursos, incluidas las máquinas virtuales de KubeVirt. Para cada máquina virtual encontrada, Trident Protect también recopila los recursos dependientes necesarios para realizar la copia de seguridad y la restauración de dicha máquina virtual.

Ejemplo de CR de aplicación basada en espacios de nombres:

```
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  name: <application_name>
  namespace: <application_namespace>
spec:
  includedNamespaces:
    - namespace: <vm_namespace>
```

Después de crear la aplicación, utiliza los flujos de trabajo estándar de Trident Protect para las copias de seguridad y las instantáneas:

- Para ver ejemplos de archivos YAML de instantáneas, consulta "Crear una snapshot bajo demanda".
- Para ver ejemplos de archivos YAML de copia de seguridad, consulta "Cree un backup bajo demanda".

Comportamiento de restauración para aplicaciones de máquinas virtuales basadas en espacios de nombres

Dado que las aplicaciones basadas en espacios de nombres recopilan recursos de los espacios de nombres seleccionados, la copia de seguridad o la instantánea pueden incluir objetos de tiempo de ejecución de KubeVirt y metadatos gestionados por KubeVirt. Algunos de estos objetos no deben restaurarse directamente porque representan el estado de tiempo de ejecución del clúster de origen.

Durante la restauración, Trident Protect aplica transformaciones compatibles con KubeVirt para eliminar los metadatos del clúster de origen y omitir los recursos transitorios de KubeVirt. Esto ayuda a KubeVirt a recrear la máquina virtual de forma limpia en el entorno restaurado.

Recurso	Transformar
VirtualMachine	Elimina las etiquetas y anotaciones que empiezan con `kubevirt.io/`.
VirtualMachineInstance	Elimina las etiquetas y anotaciones que empiezan con `kubevirt.io/`.
VirtualMachineInstanceMigration	Ignorado y no restaurado.
VirtualMachineSnapshot	Ignorado y no restaurado.
VirtualMachineSnapshotContent	Ignorado y no restaurado.
Pod (`virt-launcher`)	Se omite cuando el pod tiene la etiqueta `kubevirt.io=virt-launcher`.

NOTA Trident Protect restaura los recursos de KubeVirt en un orden definido para evitar problemas de dependencia. CDI DataVolumes (`cdi.kubevirt.io`) se restauran primero para que los datos del disco de la máquina virtual estén disponibles antes de que se creen los objetos de la máquina virtual. Después de eso, VirtualMachines (`kubevirt.io`) se restauran antes que VirtualMachineInstances.

Protección dinámica de máquinas virtuales con selectores de etiquetas

Si quieres proteger VMs de forma dinámica sin tener que especificar el nombre de cada VM, utiliza definiciones de aplicaciones basadas en espacios de nombres con selectores de etiquetas.

Solo el CR VirtualMachine necesita etiquetas. Los recursos dependientes como DataVolumes, PVC, secretos y ConfigMaps se recopilan automáticamente.

Pasos

1. Etiqueta los CR de las máquinas virtuales con una etiqueta compartida:

```
kubectl label virtualmachine <vm_name 1> app=<label_value> -n
<vm_namespace>
kubectl label virtualmachine <vm_name_2> app=<label_value> -n
<vm_namespace>
```

2. Crea un CR de aplicación con `includedNamespaces` y `labelSelector`:

```
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  name: <application_name>
  namespace: <application_namespace>
spec:
  includedNamespaces:
    - namespace: <vm_namespace>
      labelSelector:
        matchLabels:
          app: <label_value>
```

3. Crea copias de seguridad y restauraciones usando el flujo de trabajo estándar de Trident Protect.

Más detalles:

- Puedes añadir o eliminar máquinas virtuales de la protección en cualquier momento cambiando las etiquetas:

```
kubectl label virtualmachine <vm_name_to_add> app=<label_value> -n
<vm_namespace>
kubectl label virtualmachine <vm_name_to_remove> app
=<different_label_value> -n <vm_namespace>
```

- También puedes utilizar selectores de etiquetas en varios espacios de nombres:

```
spec:
  includedNamespaces:
    - namespace: <vm_namespace_1>
      labelSelector:
        matchLabels:
          <label_key>: <label_value>
    - namespace: <vm_namespace_2>
      labelSelector:
        matchLabels:
          <label_key>: <label_value>
```

Protege máquinas virtuales concretas con includedVirtualMachines

Utiliza `includedVirtualMachines` cuando quieras proteger un conjunto nombrado de VirtualMachines en uno o varios espacios de nombres y restaurar de forma selectiva VMs específicas desde un archivo.

Tanto los CR de aplicación como los de restauración utilizan el mismo formato de selector `includedVirtualMachines`.

Utiliza este selector en diferentes contextos:

- En una solicitud CR, define qué máquinas virtuales están protegidas.
- En una solicitud de restaurar sin movimiento (CR), se define qué máquinas virtuales se restauran desde la copia de seguridad o el archivo de instantánea.

NOTA En las restauraciones de CR, `includedVirtualMachines` es opcional. Si no incluyes este campo, se restaura todo el archivo. Cuando lo especificas, solo se restauran las máquinas virtuales enumeradas y sus recursos dependientes.

El YAML que aparece a continuación muestra la estructura del selector compartido:

```
includedVirtualMachines:
  - namespace: <vm_namespace_1>
    names:
      - <vm_name_1>
      - <vm_name_2>
  - namespace: <vm_namespace_2>
    names:
      - <vm_name_3>
```

NOTA Varias entradas en `includedVirtualMachines` pueden hacer referencia a diferentes espacios de nombres, lo que permite definir y restaurar aplicaciones compatibles con máquinas virtuales en varios espacios de nombres.

Campo	Tipo	Descripción
espacio de nombres	cadena	Espacio de nombres de Kubernetes que contiene los objetos de máquina virtual de destino.
nombres	lista de cadenas	Uno o varios nombres de máquinas virtuales en ese espacio de nombres.

Ejemplo de aplicación basada en máquinas virtuales:

```
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  name: <application_name>
  namespace: <application_namespace>
spec:
  includedVirtualMachines:
    - namespace: <vm_namespace_1>
      names:
        - <vm_name_1>
        - <vm_name_2>
    - namespace: <vm_namespace_2>
      names:
        - <vm_name_3>
  includedClusterScopedResources:
    - group: storage.k8s.io
      kind: StorageClass
```

Restaurar máquinas virtuales específicas de KubeVirt

Puedes usar `includedVirtualMachines` en los CR de restauración para restaurar solo las máquinas virtuales seleccionadas de KubeVirt desde un backup, una instantánea o un archivo de instantáneas replicadas.

Cuando `includedVirtualMachines` se especifica en un CR de restauración, Trident Protect restaura solo las VirtualMachines indicadas y sus recursos dependientes, como `VirtualMachineInstance`, `DataVolume`, `PersistentVolumeClaim`, `PersistentVolume`, `Secret`, `ConfigMap`, `ServiceAccount` y otros objetos relacionados con las máquinas virtuales.

Si no especificas `includedVirtualMachines` en la CR de restauración, Trident Protect restaura el archivo completo según la configuración de la CR de restauración.

CR compatibles con la restauración y comandos CLI

La siguiente tabla muestra los CR de restauración que son compatibles con `includedVirtualMachines` y sus comandos CLI correspondientes.

Restaurar CR	Comando de la CLI	Descripción
`BackupRestore`	`tridentctl-protect create backuprestore`	Restaura desde una copia de seguridad a asignaciones de espacios de nombres nuevos o diferentes. Admite `namespaceMapping` y `storageClassMapping`.
`BackupInplaceRestore`	`tridentctl-protect create backupinplacerestore`	Restaura desde una copia de seguridad en el espacio de nombres original o en los espacios de nombres originales.
`SnapshotRestore`	`tridentctl-protect create snapshotrestore`	Restaura desde una instantánea a asignaciones de espacios de nombres nuevos o diferentes. Admite `namespaceMapping` y `storageClassMapping`.
`SnapshotInplaceRestore`	`tridentctl-protect create snapshotinplacerestore`	Restaura desde una instantánea en el espacio de nombres original o en los espacios de nombres originales.
`ReplicateSnapshotRestore`	Solo YAML, o utiliza el comando CLI compatible si está disponible en tu entorno	Restaura a partir de una instantánea replicada. Admite `namespaceMapping`, `storageClassMapping` y restaurar sin movimiento usando el indicador `inPlaceRestore`.

NOTA La `tridentctl-protect` CLI no tiene un flag para `includedVirtualMachines` en los comandos de restaurar. Para restaurar máquinas virtuales específicas, debes aplicar directamente un archivo YAML de CR de restaurar usando `kubectl apply`. Puedes escribir el YAML desde cero o usar la CLI con `--dry-run --output yaml` para generar un archivo inicial, agregar el campo `includedVirtualMachines` bajo `spec` y luego aplicarlo.

Comportamiento de la restauración en el ámbito de la máquina virtual

El `includedVirtualMachines` campo de un CR de restauración utiliza el mismo formato de selector que el CR de la aplicación:

```
includedVirtualMachines:
  - namespace: <source_vm_namespace>
    names:
      - <vm_name_1>
      - <vm_name_2>
```

Para restaurar CR, los valores de los espacios de nombres en `includedVirtualMachines` se refieren a los espacios de nombres de origen tal como existen en la copia de seguridad, la instantánea o el archivo de instantánea replicada.

Si utilizas `namespaceMapping`, Trident Protect selecciona primero las máquinas virtuales solicitadas del archivo de origen y luego aplica las asignaciones de espacios de nombres durante la restauración. Si utilizas `storageClassMapping`, Trident Protect aplica las asignaciones de clases de almacenamiento durante la fase de transformación de la restauración.

Genera el archivo YAML de restauración con la CLI

Puedes utilizar `tridentctl-protect` para generar un CR de restauración básica y luego editar el archivo YAML para añadir `includedVirtualMachines`.

```
tridentctl-protect create <restore_command> <restore_name> \
  <restore_options> \
  --dry-run --output yaml > <restore_name>.yaml
```

Edita el archivo YAML generado y añade el campo `includedVirtualMachines` debajo de `spec`, luego aplica el archivo:

```
kubectl apply -f <restore_name>.yaml
```

Ejemplo: definir una aplicación basada en una máquina virtual

La siguiente aplicación protege tres máquinas virtuales concretas repartidas en dos espacios de nombres.

```
apiVersion: protect.trident.netapp.io/v1
kind: Application
metadata:
  name: production-vms
  namespace: prod
spec:
  includedVirtualMachines:
    - namespace: prod
      names:
        - app-server
        - database-server
    - namespace: monitoring
      names:
        - prometheus-vm
  includedClusterScopedResources:
    - group: storage.k8s.io
      kind: StorageClass
```

Ejemplo: crear una copia de seguridad

Crear una copia de seguridad para una aplicación basada en VM usa el CR de copia de seguridad estándar. No se requieren campos específicos de VM en el CR de copia de seguridad.

```yaml
apiVersion: protect.trident.netapp.io/v1
kind: Backup
metadata:
  name: daily-backup
  namespace: prod
spec:
  applicationRef: production-vms
  appVaultRef: s3-vault
```

Ejemplo: restaurar una máquina virtual a partir de una copia de seguridad

En el siguiente ejemplo solo se restaura `database-server` desde el archivo de copia de seguridad. Las demás máquinas virtuales del archivo, como `app-server` y `prometheus-vm`, se omiten.

```yaml
apiVersion: protect.trident.netapp.io/v1
kind: BackupRestore
metadata:
  name: restore-db-only
  namespace: prod-vms-restore
spec:
  appVaultRef: s3-vault
  appArchivePath: backups/production-vms/2026-03-10T00-00-00Z
  namespaceMapping:
    - source: prod
      destination: prod-dr
  includedVirtualMachines:
    - namespace: prod
      names:
        - database-server
```

```
kubectl apply -f restore-db-only.yaml
```

Solo `database-server` y sus recursos dependientes, como DataVolumes, PVC, PV, Secrets, ConfigMaps y los recursos relacionados de KubeVirt, se restauran.

Ejemplo: restaurar una única máquina virtual sin movimiento desde una copia de seguridad

El siguiente ejemplo restaura únicamente `database-server` en su espacio de nombres original desde el archivo de backup.

```
apiVersion: protect.trident.netapp.io/v1
kind: BackupInplaceRestore
metadata:
  name: ipr-db-only
  namespace: prod
spec:
  appVaultRef: s3-vault
  appArchivePath: backups/production-vms/2026-03-10T00-00-00Z
  includedVirtualMachines:
    - namespace: prod
      names:
        - database-server
```

```
kubectl apply -f ipr-db-only.yaml
```

Ejemplo: restaurar una máquina virtual a partir de una instantánea

En el siguiente ejemplo solo se restaura `ubuntu-vm-blue-a627be38` de un archivo de instantánea.

```
apiVersion: protect.trident.netapp.io/v1
kind: SnapshotRestore
metadata:
  name: blue-vm-snap-restore
  namespace: blue
spec:
  appVaultRef: my-appvault
  appArchivePath: snapshots/protectctl-blue-vm/snap-1
  includedVirtualMachines:
    - namespace: blue
      names:
        - ubuntu-vm-blue-a627be38
```

```
kubectl apply -f blue-vm-snap-restore.yaml
```

Combina la restauración de máquinas virtuales con otros filtros

Puedes utilizar `includedVirtualMachines` junto con `resourceFilter`, `namespaceMapping` y `storageClassMapping` en el mismo CR de restaurar sin movimiento para afinar qué se restaura.

Cuando añades un `resourceFilter`, Trident Protect primero selecciona las máquinas virtuales listadas en `includedVirtualMachines` y sus recursos dependientes. Luego aplica el filtro de recursos para acotar aún más lo que se incluye o excluye de esos resultados.

```
spec:
  includedVirtualMachines:
    - namespace: <vm_namespace>
      names:
        - <vm_name>
  resourceFilter:
    resourceSelectionCriteria: Include
    resourceMatchers:
      - kinds: ["PersistentVolumeClaim"]
        names: ["<pvc_name>"]
```

En el caso de `BackupRestore`, `SnapshotRestore` y `ReplicateSnapshotRestore`, `namespaceMapping` y `storageClassMapping` funcionan junto con `includedVirtualMachines`. Las referencias a los espacios de nombres en `includedVirtualMachines` se corresponden con los espacios de nombres de origen tal y como aparecen en el archivo. Las asignaciones de espacios de nombres y clases de almacenamiento se aplican durante la fase de transformación de la restauración, una vez finalizado el filtrado.

Para obtener más información sobre el comportamiento de la restauración, consulta "Restaurar aplicaciones usando Trident Protect".

Crear una aplicación con ámbito de máquina virtual mediante la CLI

Usa la `--virtual-machines` flag para limitar una aplicación a máquinas virtuales específicas de KubeVirt en vez de a espacios de nombres completos. La `--virtual-machines` flag es mutuamente excluyente con `--namespaces`.

Pasos

1. Crea la aplicación ejecutando uno de los siguientes comandos y sustituyendo los valores entre corchetes por la información de tu entorno. El formato para la `--virtual-machines` opción es `<namespace>(<vm_name>)`. Para varias máquinas virtuales, usa nombres separados por comas dentro de los paréntesis:
   - Una sola máquina virtual en un espacio de nombres:

     ```
     tridentctl-protect create application <application_name> \
       -n <application_namespace> \
       --virtual-machines "<vm_namespace>(<vm_name>)"
     ```

   - Varias máquinas virtuales en distintos espacios de nombres:

     ```
     tridentctl-protect create application <application_name> \
       -n <application_namespace> \
       --virtual-machines "<vm_namespace_1>(<vm_name_1>,<vm_name_2>
     ),<vm_namespace_2>(<vm_name_3>)"
     ```

Información relacionada

- "Defina una aplicación para la gestión"
- "Proteja las aplicaciones"
- "Restaurar aplicaciones usando Trident Protect"
- "Usa opciones avanzadas de restauración"
- "Instalar el complemento CLI de Trident Protect"
- "Utilice objetos de AppVault para gestionar buckets"

Administrar los ganchos de ejecución de Trident Protect

Un enlace de ejecución es una acción personalizada que puede configurar para que se ejecute junto con una operación de protección de datos de una aplicación gestionada. Por ejemplo, si dispone de una aplicación de base de datos, puede utilizar un enlace de ejecución para pausar todas las transacciones de la base de datos antes de realizar una instantánea y reanudar las transacciones una vez completada la instantánea. De este modo se garantiza la creación de instantáneas coherentes con la aplicación.

Tipos de enlaces de ejecución

Trident Protect admite los siguientes tipos de ganchos de ejecución, según cuándo se puedan ejecutar:

- Copia previa de Snapshot
- Possnapshot
- Previo al backup
- Después del backup
- Después de la restauración
- Después de la conmutación al respaldo

Orden de ejecución

Cuando se ejecuta una operación de protección de datos, los eventos de enlace de ejecución tienen lugar en el siguiente orden:

1. Los ganchos de ejecución de preoperación personalizados aplicables se ejecutan en los contenedores adecuados. Puede crear y ejecutar tantos ganchos de prefuncionamiento personalizados como necesite, pero el orden de ejecución de estos enlaces antes de la operación no está garantizado ni configurable.
2. Si corresponde, se producen bloqueos del sistema de archivos. "Obtenga más información sobre cómo configurar la congelación del sistema de archivos con Trident Protect".
3. Se realiza la operación de protección de datos.
4. Los sistemas de archivos congelados se descongelan, si corresponde.
5. Los enlaces de ejecución de post-operación personalizados aplicables se ejecutan en los contenedores adecuados. Puede crear y ejecutar tantos enlaces de post-operación personalizados como necesite, pero el orden de ejecución de estos enlaces después de la operación no está garantizado ni configurable.

Si crea varios enlaces de ejecución del mismo tipo (por ejemplo, presnapshot), no se garantiza el orden de ejecución de esos enlaces. Sin embargo, el orden de ejecución de ganchos de diferentes tipos está

garantizado. Por ejemplo, el siguiente es el orden de ejecución de una configuración que tiene todos los diferentes tipos de ganchos:

1. Ganchos presnapshot ejecutados
2. Ganchos posteriores a la instantánea ejecutados
3. Ganchos de precopia de seguridad ejecutados
4. Se han ejecutado los enlaces posteriores a la copia de seguridad

NOTA	El ejemplo de orden anterior solo se aplica cuando se ejecuta un backup que no utiliza una snapshot existente.

NOTA	Siempre debe probar sus secuencias de comandos de ejecución de enlace antes de habilitarlas en un entorno de producción. Puede utilizar el comando 'kubectl exec' para probar cómodamente los scripts. Después de habilitar los enlaces de ejecución en un entorno de producción, pruebe las copias Snapshot y backups resultantes para garantizar que sean coherentes. Para ello, puede clonar la aplicación en un espacio de nombres temporal, restaurar la instantánea o la copia de seguridad y, a continuación, probar la aplicación.

NOTA	Si un gancho de ejecución previo a la instantánea agrega, cambia o elimina recursos de Kubernetes, esos cambios se incluyen en la instantánea o la copia de seguridad y en cualquier operación de restauración posterior.

Notas importantes sobre los enlaces de ejecución personalizados

Tenga en cuenta lo siguiente al planificar enlaces de ejecución para sus aplicaciones.

- Un enlace de ejecución debe utilizar una secuencia de comandos para realizar acciones. Muchos enlaces de ejecución pueden hacer referencia al mismo script.
- Trident Protect requiere que los scripts que utilizan los ganchos de ejecución estén escritos en el formato de scripts de shell ejecutables.
- El tamaño del script está limitado a 96 KB.
- Trident Protect utiliza configuraciones de gancho de ejecución y cualquier criterio coincidente para determinar qué ganchos son aplicables a una operación de instantánea, copia de seguridad o restauración.

NOTA	Debido a que los enlaces de ejecución a menudo reducen o desactivan por completo la funcionalidad de la aplicación con la que se ejecutan, siempre debe intentar minimizar el tiempo que tardan los enlaces de ejecución personalizados. Si inicia una operación de copia de seguridad o de instantánea con los enlaces de ejecución asociados pero, a continuación, la cancela, los ganchos pueden ejecutarse si ya se ha iniciado la operación de copia de seguridad o de Snapshot. Esto significa que la lógica utilizada en un enlace de ejecución posterior a la copia de seguridad no puede suponer que la copia de seguridad se ha completado.

Filtros de gancho de ejecución

Al agregar o editar un enlace de ejecución para una aplicación, puede agregar filtros al enlace de ejecución para gestionar qué contenedores coincidirá el enlace. Los filtros son útiles para aplicaciones que usan la misma imagen de contenedor en todos los contenedores, pero pueden usar cada imagen para un propósito diferente (como Elasticsearch). Los filtros le permiten crear escenarios donde los enlaces de ejecución se

ejecutan en algunos, pero no necesariamente todos los contenedores idénticos. Si crea varios filtros para un único enlace de ejecución, se combinan con un operador y lógico. Puede tener hasta 10 filtros activos por gancho de ejecución.

Cada filtro que agrega a un gancho de ejecución utiliza una expresión regular para que coincida con los contenedores en su clúster. Cuando un gancho coincide con un contenedor, el gancho ejecutará su script asociado en ese contenedor. Las expresiones regulares para filtros utilizan la sintaxis de Expresión regular 2 (RE2), que no admite la creación de un filtro que excluya contenedores de la lista de coincidencias. Para obtener información sobre la sintaxis que Trident Protect admite para expresiones regulares en filtros de gancho de ejecución, consulte "Soporte de sintaxis de expresión regular 2 (RE2)" .

NOTA	Si se agrega un filtro de espacio de nombres a un enlace de ejecución que se ejecuta después de una operación de restauración o clonado y el origen y destino de la restauración o clonado se encuentran en diferentes espacios de nombres, el filtro de espacio de nombres solo se aplica al espacio de nombres de destino.

Ejemplos de gancho de ejecución

Visite el "Proyecto Verda GitHub de NetApp" para descargar ganchos de ejecución reales para aplicaciones populares como Apache Cassandra y Elasticsearch. También puede ver ejemplos y obtener ideas para estructurar sus propios enlaces de ejecución personalizados.

NOTA	Los scripts del repositorio de Verda se proporcionan «tal cual» y no cuentan con soporte oficial por parte de NetApp. NetApp solo abarca el framework de hooks de ejecución dentro de Trident Protect.

Cree un enlace de ejecución

Puede crear un gancho de ejecución personalizado para una aplicación usando Trident Protect. Necesitas tener permisos de Propietario, Administrador o Miembro para crear ganchos de ejecución.

Utilice un CR

Pasos

1. Cree el archivo de recursos personalizados (CR) y asígnele un nombre `trident-protect-hook.yaml`.
2. Configure los siguientes atributos para que coincidan con su entorno de Trident Protect y la configuración del clúster:
   - **metadata.name**: (*required*) El nombre de este recurso personalizado; elija un nombre único y sensible para su entorno.
   - **Spec.applicationRef**: (*required*) El nombre de Kubernetes de la aplicación para la que ejecutar el hook de ejecución.
   - **Spec.stage**: (*required*) Una cadena que indica qué etapa durante la acción debe ejecutarse el gancho de ejecución. Los posibles valores son los siguientes:
     - Pre
     - Publicación
   - **Spec.action**: (*required*) Una cadena que indica qué acción tomará el gancho de ejecución, asumiendo que los filtros de enlace de ejecución especificados coinciden. Los posibles valores son los siguientes:
     - Snapshot
     - Backup
     - Restaurar
     - Conmutación al respaldo
   - **Spec.enabled**: (*Optional*) Indica si este enlace de ejecución está habilitado o desactivado. Si no se especifica, el valor predeterminado es TRUE.
   - **Spec.hookSource**: (*required*) Una cadena que contiene el script hook codificado en base64.
   - **SPEC.TIMEOUT**: (*Optional*) Un número que define cuánto tiempo en minutos se permite ejecutar el gancho de ejecución. El valor mínimo es 1 minuto y el valor predeterminado es 25 minutos si no se especifica.
   - **Spec.arguments**: (*Optional*) Una lista YAML de argumentos que puede especificar para el enlace de ejecución.
   - **Spec.matchingCriteria**: (*Optional*) Una lista opcional de pares de valores clave de criterios, cada par que forma un filtro de enlace de ejecución. Puede agregar hasta 10 filtros por gancho de ejecución.
   - **Spec.matchingCriteria.type**: (*Optional*) Una cadena que identifica el tipo de filtro de gancho de ejecución. Los posibles valores son los siguientes:
     - ConteneerImage
     - Nombre del contenedor
     - PodName
     - PodLabel
     - Nombre del espacio de nombre
   - **Spec.matchingCriteria.value**: (*Optional*) Una cadena o expresión regular que identifica el valor del filtro de enlace de ejecución.

Ejemplo YAML:

```
apiVersion: protect.trident.netapp.io/v1
kind: ExecHook
metadata:
  name: example-hook-cr
  namespace: my-app-namespace
  annotations:
    astra.netapp.io/astra-control-hook-source-id:
/account/test/hookSource/id
spec:
  applicationRef: my-app-name
  stage: Pre
  action: Snapshot
  enabled: true
  hookSource: IyEvYmluL2Jhc2gKZWNobyAiZXhhbXBsZSBzY3JpcHQiCg==
  timeout: 10
  arguments:
    - FirstExampleArg
    - SecondExampleArg
  matchingCriteria:
    - type: containerName
      value: mysql
    - type: containerImage
      value: bitnami/mysql
    - type: podName
      value: mysql
    - type: namespaceName
      value: mysql-a
    - type: podLabel
      value: app.kubernetes.io/component=primary
    - type: podLabel
      value: helm.sh/chart=mysql-10.1.0
    - type: podLabel
      value: deployment-type=production
```

3. Después de rellenar el archivo CR con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-hook.yaml
```

Utilice la CLI

Pasos

1. Cree el enlace de ejecución, sustituyendo los valores entre paréntesis por información de su entorno. Por ejemplo:

```
tridentctl-protect create exechook <my_exec_hook_name> --action
<action_type> --app <app_to_use_hook> --stage <pre_or_post_stage>
--source-file <script-file> -n <application_namespace>
```

Ejecute manualmente un enlace de ejecución

Puede ejecutar manualmente un enlace de ejecución para fines de prueba o si necesita volver a ejecutar el enlace manualmente después de un fallo. Debe tener permisos de Propietario, Administrador o Miembro para ejecutar manualmente los ganchos de ejecución.

La ejecución manual de un gancho de ejecución consta de dos pasos básicos:

1. Cree un backup de recursos, que recopila recursos y crea un backup de ellos, determinando dónde se ejecutará el enlace
2. Ejecute el enlace de ejecución en la copia de seguridad

Paso 1: Crear una copia de seguridad de recursos

Utilice un CR

Pasos

1. Cree el archivo de recursos personalizados (CR) y asígnele un nombre `trident-protect-resource-backup.yaml`.
2. Configure los siguientes atributos para que coincidan con su entorno de Trident Protect y la configuración del clúster:
   - **metadata.name**: (*required*) El nombre de este recurso personalizado; elija un nombre único y sensible para su entorno.
   - **Spec.applicationRef**: (*required*) El nombre de Kubernetes de la aplicación para la que crear la copia de seguridad del recurso.
   - **Spec.appVaultRef**: (*required*) El nombre del AppVault donde se almacena el contenido de la copia de seguridad.
   - **Spec.appArchivePath**: La ruta dentro de AppVault donde se almacena el contenido de la copia de seguridad. Puede utilizar el siguiente comando para buscar esta ruta:

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o jsonpath='{.status.appArchivePath}'
     ```

     Ejemplo YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: ResourceBackup
metadata:
  name: example-resource-backup
spec:
  applicationRef: my-app-name
  appVaultRef: my-appvault-name
  appArchivePath: example-resource-backup
```

3. Después de rellenar el archivo CR con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-resource-backup.yaml
```

Utilice la CLI

Pasos

1. Cree el backup sustituyendo valores entre paréntesis con información de su entorno. Por ejemplo:

```
tridentctl protect create resourcebackup <my_backup_name> --app
<my_app_name> --appvault <my_appvault_name> -n
<my_app_namespace> --app-archive-path <app_archive_path>
```

2. Vea el estado del backup. Puede usar este comando de ejemplo varias veces hasta que se complete la operación:

```
tridentctl protect get resourcebackup -n <my_app_namespace>
<my_backup_name>
```

3. Compruebe que el backup se ha realizado correctamente:

```
kubectl describe resourcebackup <my_backup_name>
```

Paso 2: Ejecute el enlace de ejecución

Utilice un CR

Pasos

1. Cree el archivo de recursos personalizados (CR) y asígnele un nombre `trident-protect-hook-run.yaml`.
2. Configure los siguientes atributos para que coincidan con su entorno de Trident Protect y la configuración del clúster:
   - **metadata.name**: (*required*) El nombre de este recurso personalizado; elija un nombre único y sensible para su entorno.
   - **Spec.applicationRef**: (*required*) Asegúrese de que este valor coincida con el nombre de la aplicación de ResourceBackup CR que creó en el paso 1.
   - **Spec.appVaultRef**: (*required*) Asegúrese de que este valor coincida con appVaultRef del ResourceBackup CR que creó en el paso 1.
   - **Spec.appArchivePath**: Asegúrate de que este valor coincida con el appArchivePath del ResourceBackup CR que creaste en el paso 1.

     ```
     kubectl get backups <BACKUP_NAME> -n my-app-namespace -o jsonpath='{.status.appArchivePath}'
     ```

   - **Spec.action**: (*required*) Una cadena que indica qué acción tomará el gancho de ejecución, asumiendo que los filtros de enlace de ejecución especificados coinciden. Los posibles valores son los siguientes:
     - Snapshot
     - Backup
     - Restaurar
     - Conmutación al respaldo
   - **Spec.stage**: (*required*) Una cadena que indica qué etapa durante la acción debe ejecutarse el gancho de ejecución. Esta secuencia de gancho no ejecutará ganchos en ninguna otra etapa. Los posibles valores son los siguientes:
     - Pre
     - Publicación

     Ejemplo YAML:

```
---
apiVersion: protect.trident.netapp.io/v1
kind: ExecHooksRun
metadata:
  name: example-hook-run
spec:
  applicationRef: my-app-name
  appVaultRef: my-appvault-name
  appArchivePath: example-resource-backup
  stage: Post
  action: Failover
```

3. Después de rellenar el archivo CR con los valores correctos, aplique el CR:

```
kubectl apply -f trident-protect-hook-run.yaml
```

Utilice la CLI

Pasos

1. Cree la solicitud de ejecución de enlace de ejecución manual:

```
tridentctl protect create exechooksrun <my_exec_hook_run_name>
-n <my_app_namespace> --action snapshot --stage <pre_or_post>
--app <my_app_name> --appvault <my_appvault_name> --path
<my_backup_name>
```

2. Compruebe el estado de la ejecución del enlace de ejecución. Este comando se puede ejecutar varias veces hasta que se complete la operación:

```
tridentctl protect get exechooksrun -n <my_app_namespace>
<my_exec_hook_run_name>
```

3. Describa el objeto exechooksrun para ver los detalles y el estado finales:

```
kubectl -n <my_app_namespace> describe exechooksrun
<my_exec_hook_run_name>
```

Desinstalar Trident Protect

Es posible que necesites eliminar los componentes de Trident Protect si estás actualizando desde una versión de prueba a una versión completa del producto.

Para eliminar Trident Protect, realice los siguientes pasos.

Pasos

1. Eliminar los archivos CR de Trident Protect:

   NOTA Este paso no es necesario para la versión 25.06 y posteriores.

   ```
   helm uninstall -n trident-protect trident-protect-crds
   ```

2. Eliminar Trident Protect:

   ```
   helm uninstall -n trident-protect trident-protect
   ```

3. Eliminar el espacio de nombres Trident Protect:

   ```
   kubectl delete ns trident-protect
   ```

Blogs de Trident y Trident Protect

Puede encontrar algunos excelentes blogs NetApp Trident y Trident Protect aquí:

Blogs de Trident

- 16 de octubre de 2025:"Soluciones de almacenamiento avanzadas para Kubernetes"
- 19 de agosto de 2025:"Mejora de la consistencia de los datos: instantáneas de grupos de volúmenes en la virtualización de OpenShift con Trident"
- 9 de mayo de 2025: "Configuración automática del backend de Trident para FSx para ONTAP con el complemento Amazon EKS"
- 15 de abril de 2025: "NetApp Trident con Google Cloud NetApp Volumes para el protocolo SMB"
- 14 de abril de 2025: "Aprovechamiento del protocolo de canal de fibra con Trident 25.02 para el almacenamiento persistente en Kubernetes"
- 14 de abril de 2025: "Desbloqueo del poder de los sistemas NetApp ASA r2 para el almacenamiento en bloque de Kubernetes"
- 31 de marzo de 2025: "Simplificación de la instalación de Trident en Red Hat OpenShift con el nuevo operador certificado"
- 27 de marzo de 2025: "Aprovisionamiento de Trident para pymes con Google Cloud NetApp Volumes"
- 5 de marzo de 2025: "Desbloquee la integración perfecta del almacenamiento iSCSI: Una guía para FSxN en ROSA Clusters para AWS"
- 27 de febrero de 2025: "Implementación de identidad en la nube con Trident, GKE y Google Cloud NetApp Volumes"
- 12 de diciembre de 2024: "Presentamos el soporte de Fibre Channel en Trident"
- 11 de noviembre de 2024: "NetApp Trident con Google Cloud NetApp Volumes"
- 29 de octubre de 2024: "Amazon FSx for NetApp ONTAP con Red Hat OpenShift Service en AWS (ROSA) mediante Trident"
- 29 de octubre de 2024: "Migración en vivo de máquinas virtuales con OpenShift Virtualization en ROSA y Amazon FSx for NetApp ONTAP"
- 8 de julio de 2024: "Uso de NVMe/TCP para consumir almacenamiento ONTAP para sus aplicaciones modernas en contenedores en Amazon EKS"
- 1 de julio de 2024: "Almacenamiento de Kubernetes sin interrupciones con Google Cloud NetApp Volumes Flex y Astra Trident"
- 11 de junio de 2024: "ONTAP como almacenamiento backend para el registro de imágenes integrado en OpenShift"

Blogs de Trident Protect

- 16 de mayo de 2025:"Automatización de la conmutación por error del registro para la recuperación ante desastres con los ganchos posteriores a la restauración de Trident Protect"
- 16 de mayo de 2025: "Recuperación ante desastres de virtualización de OpenShift con NetApp Trident Protect"
- 13 de mayo de 2025:"Migración de clases de almacenamiento con copia de seguridad y restauración de Trident Protect"

- 9 de mayo de 2025:"Reescalar aplicaciones de Kubernetes con los ganchos posteriores a la restauración de Trident Protect"
- 3 de abril de 2025: "Trident Protect Power Up: Replicación de Kubernetes para protección y recuperación ante desastres"
- 13 de marzo de 2025: "Operaciones de backup y restauración coherentes con los fallos para equipos virtuales de virtualización OpenShift"
- 11 de marzo de 2025: "Ampliar los patrones de GitOps a la protección de datos de aplicaciones con NetApp Trident"
- 03 de marzo de 2025: "Trident 25,02: Mejorar la experiencia de Red Hat OpenShift con nuevas y emocionantes funciones"
- 15 de enero de 2025:"Presentamos el control de acceso basado en roles de Trident Protect"
- 11 de noviembre de 2024:"Gestión de datos impulsada por Kubernetes: la nueva era con Trident Protect"

Conocimiento y apoyo

Preguntas frecuentes

Encuentre respuestas a las preguntas frecuentes sobre instalación, configuración, actualización y solución de problemas de Trident.

Preguntas generales

¿Con qué frecuencia se publica Trident?

A partir de la versión 24,02, Trident se lanza cada cuatro meses: Febrero, junio y octubre.

¿Trident admite todas las funciones que están publicadas en una versión concreta de Kubernetes?

Trident por lo general no admite 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.

¿Trident tiene alguna dependencia de otros productos de NetApp para su funcionamiento?

Trident no tiene ninguna 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 información completa sobre la configuración de Trident?

Use el `tridentctl get` comando para obtener más información acerca de la configuración de Trident.

¿Puedo obtener métricas acerca de cómo aprovisiona el almacenamiento Trident?

Sí. Extremos de Prometheus que se pueden utilizar para recopilar información acerca de la operación de 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 cuando utiliza 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 Trident si desea utilizar todas las funciones nuevas proporcionadas por las versiones actuales y futuras.

Instale y utilice Trident en un clúster de Kubernetes

¿Trident admite una instalación sin conexión desde un registro privado?

Sí, Trident puede instalarse sin conexión. Consulta "Obtenga información acerca de la instalación de Trident".

¿Puedo instalar Trident Be de forma remota?

Sí. Trident 18,10 y versiones posteriores admiten la capacidad de instalación remota desde cualquier máquina que `kubectl` tenga 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 Trident?

Trident se instala como una puesta en marcha de Kubernetes (ReplicaSet) con una instancia, por lo que tiene HA incorporada. No debería aumentar el número de réplicas en la implementación. Si se pierde el nodo donde está instalado Trident o no se puede acceder al pod de otro modo, Kubernetes vuelve a poner en marcha automáticamente el pod en un nodo del clúster en buen estado. Trident es solo el plano de control, por lo que los pods montados en este momento no se ven afectados si se vuelve a poner en marcha Trident.

¿Trident necesita acceso al espacio de nombres del sistema kube?

Trident lee desde el servidor de la API de Kubernetes para determinar cuándo las aplicaciones solicitan nuevos RVP, por lo que necesita acceder al sistema kube.

¿Cuáles son los roles y Privileges que usa 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. Consulta "Personalice la instalación trimentctl".

¿Puedo generar localmente los archivos de manifiesto exactos que Trident utiliza para la instalación?

Puedes generar y modificar localmente los archivos de manifiesto exactos que Trident usa para la instalación, si lo necesitas. Consulta "Personalice la instalación trimentctl".

¿Puedo compartir la misma SVM back-end de ONTAP para dos instancias de Trident separadas para dos clústeres de Kubernetes separados?

Aunque no se aconseja, puede utilizar la misma SVM de back-end para dos instancias de 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. Esto se asegura de que no se utiliza el mismo FlexVol volume para ambas instancias.

¿Es posible instalar Trident bajo containerLinux (anteriormente CoreOS)?

Trident es simplemente un pod de Kubernetes y se puede instalar dondequiera que se ejecute Kubernetes.

¿Puedo usar Trident con NetApp Cloud Volumes ONTAP?

Sí, Trident es compatible con AWS, Google Cloud y Azure.

Solución de problemas y soporte técnico

¿NetApp admite Trident?

Aunque Trident es de código abierto y se proporciona de forma gratuita, NetApp es totalmente compatible siempre que su backend NetApp sea compatible.

¿Cómo levanto un caso de soporte?

Para levantar un caso de soporte, realice una de las siguientes acciones:

1. Póngase en contacto con su responsable técnico de soporte y obtenga ayuda para emitir una incidencia.
2. Levante un caso de soporte con el contacto "Soporte de NetApp".

¿Cómo se genera un bundle del registro de soporte?

Puede crear un paquete de soporte en ejecución `tridentctl logs -a`. Además de los registros capturados en el paquete, capture el registro kubelet para diagnosticar los problemas de montaje en el lado de Kubernetes. Las instrucciones para obtener el registro de Kubelet varían en función de cómo se instale Kubernetes.

¿Qué debo hacer si necesito solicitar una nueva función?

Crea un problema "Trident Github" y menciona **RFE** en el asunto y la descripción del problema.

¿Dónde puedo elevar un defecto?

Crear un problema en "Trident Github". Asegúrese de incluir toda la información y registros necesarios relacionados con el problema.

¿Qué sucede si tengo preguntas rápidas sobre Trident sobre las que necesito aclaraciones? ¿Hay una comunidad o un foro?

Si tiene alguna pregunta, problema o solicitud, póngase en contacto con nosotros a través de nuestro Trident "Canal de discordia"o GitHub.

La contraseña de mi sistema de almacenamiento ha cambiado y Trident ya no funciona. ¿Cómo puedo recuperarme?

Actualice la contraseña del backend con `tridentctl update backend myBackend -f </path/to_new_backend.json> -n trident`. Sustituya `myBackend` en el ejemplo con su nombre de fondo, y. `` `/path/to_new_backend.json `` con la ruta a la correcta `backend.json` archivo.

Trident no puede encontrar mi nodo de Kubernetes. ¿Cómo se soluciona esto?

Hay dos situaciones probables por las que 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 de red después de instalar Trident, este problema solo se produce con los nuevos nodos de Kubernetes 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 Trident

¿Puedo actualizar directamente desde una versión anterior a una versión nueva (omitiendo algunas versiones)?

NetApp permite actualizar Trident de una versión principal a la siguiente versión principal inmediata. 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 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

¿Necesito definir las LIF de datos y de gestión en un archivo de definición de backend de ONTAP?

El LIF de gestión es obligatorio. La LIF de datos varía:

- SAN de ONTAP: No se especifica para iSCSI. Trident utiliza "Asignación de LUN selectiva de ONTAP" para descubrir las LIF iSCSI necesarias para establecer una sesión de rutas múltiples. Se genera una advertencia si `dataLIF` se define explícitamente. Consulte "Opciones y ejemplos de configuración DE SAN ONTAP" para obtener más información.
- NAS de ONTAP: NetApp recomienda especificar `dataLIF`. Si no se proporciona, Trident recupera las LIF de datos de la SVM. Puede especificar un nombre de dominio completo (FQDN) que se utilice para las operaciones de montaje de NFS, lo que permite crear un DNS por turnos para equilibrar la carga en varias LIF de datos. Consulte "Opciones y ejemplos de configuración NAS de ONTAP"para obtener más información

¿Puede Trident configurar CHAP para back-ends de ONTAP?

Sí. Trident admite CHAP bidireccional para back-ends de ONTAP. Esto requiere configuración `useCHAP=true` en la configuración de backend.

¿Cómo se gestionan las políticas de exportación con Trident?

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 manera, Trident administra automáticamente la adición y eliminación de reglas para nodos con IP dentro de los CIDR dados.

¿Pueden utilizarse las direcciones IPv6 para las LIF de datos y gestión?

Trident admite definir direcciones IPv6 para:

- `managementLIF` y.. `dataLIF` Para back-ends NAS de ONTAP.
- `managementLIF` Para back-ends DE SAN de ONTAP. No puede especificar `dataLIF` En un entorno de administración SAN de ONTAP.

Trident debe instalarse utilizando el indicador `--use-ipv6` (para `tridentctl` la instalación), `IPv6` (para el operador Trident) o `tridentTPv6` (para la instalación Helm) para que funcione a través de IPv6.

¿Se puede actualizar la LIF de gestión en el back-end?

Sí, es posible actualizar la LIF de gestión del back-end mediante el `tridentctl update backend` comando.

¿Es posible actualizar DataLIF en el backend?

Puede actualizar DataLIF en `ontap-nas` y `ontap-nas-economy` sólo.

¿Puedo crear varios back-ends en Trident para Kubernetes?

Trident puede admitir muchos back-ends simultáneamente, ya sea con el mismo controlador o con controladores diferentes.

¿Cómo almacena Trident las credenciales de backend?

Trident almacena las credenciales de back-end como secretos de Kubernetes.

¿Cómo selecciona Trident un backend específico?

Si los atributos back-end no se pueden utilizar para seleccionar automáticamente los grupos adecuados para una clase, el `storagePools` y.. `additionalStoragePools` los parámetros se usan para seleccionar un conjunto específico de pools.

¿Cómo me aseguro de que Trident no aprovisiona desde un back-end específico?

```
`excludeStoragePools`El parámetro se utiliza para filtrar el conjunto de pools que Trident utiliza para el aprovisionamiento y eliminará los pools que coincidan.
```

Si hay varios back-ends del mismo tipo, ¿cómo selecciona Trident qué backend utilizar?

Si hay varios back-ends configurados del mismo tipo, Trident selecciona el backend adecuado en función de los parámetros presentes en `StorageClass` y `PersistentVolumeClaim`. Por ejemplo, si hay varios back-ends de controlador ONTAP-nas, Trident intenta hacer coincidir los parámetros en el `StorageClass` y `PersistentVolumeClaim` combinado y hacer coincidir un backend que puede entregar los requisitos enumerados en `StorageClass` y `PersistentVolumeClaim`. Si hay varios back-ends que coincidan con la solicitud, Trident selecciona uno de ellos al azar.

¿Trident admite CHAP bidireccional con Element/SolidFire?

Sí.

¿Cómo implementa 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 volume (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 volume que pueda dar servicio al nuevo qtree. Si FlexVol volume no existe y puede reparar el qtree, se crea una nueva FlexVol volume.
```

¿Cómo puedo establecer los permisos de Unix para los volúmenes aprovisionados en NAS de ONTAP?

Puede establecer permisos Unix en el volumen aprovisionado por Trident estableciendo un parámetro en el

archivo de definición de backend.

¿Cómo puedo configurar un conjunto explícito de opciones de montaje NFS de ONTAP al aprovisionar un volumen?

De forma predeterminada, Trident no establece las opciones de montaje como 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 el acceso de hosts adecuados a un volumen, use el `exportPolicy` parámetro configurado en el archivo de definición de backend.

¿Cómo puedo configurar el cifrado de volúmenes mediante 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 Trident con NVE y NAE"

¿Cuál es la mejor forma de implementar la calidad de servicio para ONTAP mediante Trident?

Uso `StorageClasses` Para implementar QoS en ONTAP.

¿Cómo se especifica el aprovisionamiento ligero o grueso mediante Trident?

Los controladores ONTAP admiten thin provisioning o thick. Los controladores ONTAP, de manera predeterminada, son thin provisioning. Si se desea un aprovisionamiento grueso, debe configurar el archivo de definición back-end o el `StorageClass`. Si se configuran ambas, `StorageClass` tiene prioridad. Configure lo siguiente para ONTAP:

1. Encendido `StorageClass`, establezca la `provisioningType` atributo como grueso.
2. En el archivo de definición del back-end, habilite los volúmenes gruesos mediante la configuración `backend spaceReserve parameter` como volumen.

¿Cómo se asegura de que los volúmenes que se están utilizando no se eliminen incluso si se elimina accidentalmente la RVP?

La protección contra RVP se habilita automáticamente en Kubernetes a partir de la versión 1.10.

¿Puedo aumentar los RVP de NFS creados por Trident?

Sí. Puede expandir una RVP creada por 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 sistema de almacenamiento de NetApp no puede cumplir con la configuración de la cuota de Kubernetes debido a la falta de capacidad, Trident intenta aprovisionar, pero se produce un error.

¿Puedo crear snapshots de volumen con Trident?

Sí. La creación de instantáneas de volumen bajo demanda y volúmenes persistentes desde Snapshots son compatibles con Trident. Para crear VP a partir de instantáneas, asegúrese de que `VolumeSnapshotDataSource` se ha activado la puerta de función.

¿Cuáles son los controladores que admiten las snapshots de volumen de Trident?

A partir de hoy, el soporte de instantáneas bajo demanda está disponible para nuestros `ontap-nas` , `ontap-nas-flexgroup` , `ontap-san` , `ontap-san-economy` , `solidfire-san` , y `azure-netapp-files` Controladores de backend.

¿Cómo hago un backup de una copia Snapshot de un volumen aprovisionado por Trident con ONTAP?

Este está disponible en `ontap-nas`, `ontap-san`, y. `ontap-nas-flexgroup` de windows También puede especificar un `snapshotPolicy` para la `ontap-san-economy` Controlador en el nivel FlexVol.

También está disponible en `ontap-nas-economy` los controladores, pero en la granularidad de FlexVol volume, no en la de qtree. Para habilitar la capacidad de los volúmenes Snapshot aprovisionados por Trident, se debe establecer la opción del parámetro backend `snapshotPolicy` en la política Snapshot que se desee tal y como se define en el back-end de ONTAP. Trident no conoce las copias Snapshot tomadas por la controladora de almacenamiento.

¿Puedo establecer un porcentaje de reserva de snapshots para un volumen aprovisionado mediante Trident?

Sí, puede reservar un porcentaje específico de espacio en disco para almacenar las copias snapshot mediante Trident estableciendo el `snapshotReserve` atributo en el archivo de definición de backend. 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, es posible acceder al directorio Snapshot en el volumen aprovisionado por Trident estableciendo el `snapshotDir` parámetro en el archivo de definición de backend.

¿Puedo configurar SnapMirror para volúmenes a través de 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 podrían surgir durante la instalación y el uso de Trident.

NOTA Si necesita ayuda con Trident, cree un paquete de soporte que utilice `tridentctl logs -a -n trident` y envíelo al soporte de NetApp.

Resolución de problemas generales

- Si el pod de Trident no sale correctamente (por ejemplo, cuando el pod de Trident se encuentra atascado en el `ContainerCreating` fase con menos de dos contenedores listos para usar) en ejecución `kubectl -n trident describe deployment trident` y.. `kubectl -n trident describe pod trident-**` puede proporcionar información adicional. Obtención de registros de kubelet (por ejemplo, mediante `journalctl -xeu kubelet`) también puede ser útil.
- Si no hay suficiente información en los registros de Trident, puede intentar habilitar el modo de depuración para Trident aprobando el `-d` marque el parámetro install en función de su opción de instalación.

  A continuación, confirme que la depuración se ha configurado mediante `./tridentctl logs -n trident` y buscando `level=debug msg` en el registro.

Instalado con el operador

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

Así se reiniciarán todos los pods de Trident, que pueden tardar varios segundos. Puede comprobarlo observando la columna "ANTIGÜEDAD" en la salida de `kubectl get pod -n trident`.

Para Trident 20,07 y 20,10, utilice `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 Red Hat Enterprise Linux CoreOS (RHCOS), asegúrese de que `iscsid` está activado en los nodos de trabajo y se inicia de forma predeterminada. Esto se puede hacer usando OpenShift MachineConfigs o modificando las plantillas de ignición.
- Un problema común que podrías encontrar al usar Trident con "Azure NetApp Files" es cuando los secretos de tenant y client vienen de un registro de app con permisos insuficientes. Para ver una lista completa de los requisitos de Trident, consulta "Azure NetApp Files" configuration.
- Si hay problemas con el montaje de un PV en un contenedor, asegúrese de que `rpcbind` está instalado y en ejecución. Utilice el administrador de paquetes necesario para el sistema operativo del host y compruebe si `rpcbind` está en ejecución. Puede comprobar el estado de `rpcbind` ejecutar un `systemctl status rpcbind` o su equivalente.
- Si un back-end de Trident informa que está en la `failed` estado a pesar de haber trabajado antes, es probable que sea por el cambio de las credenciales de SVM/administrador asociadas con el back-end. Actualización de la información del back-end mediante `tridentctl update backend` O rebotando el Pod de Trident se corregirá este problema.
- Si se producen problemas de permisos al instalar Trident con Docker como el tiempo de ejecución de contenedores, intente instalar Trident con el `--in cluster=false` bandera. Esto no utilizará un módulo de instalación y evitará problemas de permisos vistos debido a la `trident-installer` usuario.
- Utilice la `uninstall parameter <Uninstalling Trident>` para limpiar después de una ejecución fallida. De forma predeterminada, la secuencia de comandos no elimina los CRD creados por Trident, por lo que es seguro desinstalar e instalar de nuevo incluso en una implementación en ejecución.
- Si desea degradar a una versión anterior de Trident, ejecute primero el `tridentctl uninstall` Comando para quitar Trident. Descargue el contenido que desee "Versión de Trident" e instálela utilizando

`tridentctl install` comando.

- Después de una instalación correcta, si un PVC está atascado en el `Pending` fase, en marcha `kubectl describe pvc` Puede proporcionar información adicional acerca de por qué Trident no pudo aprovisionar un VP para esta RVP.

Implementación incorrecta de Trident con el operador

Si utiliza el operador, el estado de implementación de Trident `TridentOrchestrator` cambios de `Installing` para `Installed`. Si observa la `Failed` y el operador no puede recuperarse por sí solo, debe comprobar los registros del operador ejecutando el siguiente comando:

```
tridentctl logs -l trident-operator
```

Al dejar atrás los registros del contenedor del operador-trident, puede indicar dónde se encuentra el problema. Por ejemplo, uno de estos problemas podría ser la incapacidad de extraer las imágenes contenedoras necesarias de los registros de entrada en un entorno con conexión aérea.

Para comprender por qué la instalación de Trident no ha tenido éxito, debería 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. Como cada clúster de Kubernetes puede tener una instancia de Trident, el operador se asegura de que en cualquier momento solo exista una activa `TridentOrchestrator` que puede 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 eliminar `TridentOrchestrator`, y cree una nueva con la definición modificada y precisa.

Puesta en marcha de Trident incorrecta mediante `tridentctl`

Para ayudar a averiguar qué fue lo que salió mal, puede ejecutar el instalador de nuevo utilizando el `-d` argumento, que activa el modo de depuración y le ayuda a comprender cuál es el problema:

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

Después de solucionar el problema, puede limpiar la instalación de la siguiente manera y, a continuación, ejecutar el `tridentctl install` comando:

```
./tridentctl uninstall -n trident
INFO Deleted Trident deployment.
INFO Deleted cluster role binding.
INFO Deleted cluster role.
INFO Deleted service account.
INFO Removed Trident user from security context constraint.
INFO Trident uninstallation succeeded.
```

Eliminar completamente Trident y CRD

Puede eliminar completamente Trident y todos los CRD creados y los recursos personalizados asociados.

ADVERTENCIA	Esta acción no se puede deshacer. No haga esto a menos que desee una instalación completamente nueva de Trident. Para desinstalar Trident sin eliminar CRD, consulte "Desinstale Trident".

Operador de Trident

Para desinstalar Trident y eliminar por completo los CRD mediante el operador Trident:

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

Timón

Para desinstalar Trident y eliminar por completo los 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 Trident utilizando `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

Piense en un escenario en el que tenga un espacio de nombres gestionado por 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 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
Añada el `hostNQN` de vuelta al subsistema.
```

Los clientes NFSv4.2 informan "argumento no válido" después de actualizar ONTAP cuando esperaban que "v4.2-xattrs" estuviera habilitado

Después de actualizar ONTAP, los clientes NFSv4.2 pueden informar errores de "argumento inválido" al intentar montar exportaciones NFSv4.2. Este problema ocurre cuando el `v4.2-xattrs` La opción no está habilitada en el SVM. .Solución alternativa Habilite la `v4.2-xattrs` opción en SVM o actualizar a ONTAP 9.12.1 o posterior, donde esta opción está habilitada de forma predeterminada.

Soporte técnico

NetApp ofrece compatibilidad con Trident de distintas formas. 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 Trident

Trident ofrece tres niveles de soporte en función de su versión. Consulte "Compatibilidad con la versión del software NetApp para definiciones".

Soporte completo
: Trident proporciona soporte completo durante doce meses a partir de la fecha de lanzamiento.

Soporte limitado
: Trident ofrece soporte limitado durante los meses 13 - 24 desde la fecha de lanzamiento.

Autosoporte
: La documentación de Trident está disponible durante los meses 25 - 36 desde la fecha de lanzamiento.

Versión	Soporte completo	Soporte limitado	Autosoporte

"26,02"	Febrero de 2026	Febrero de 2027	Febrero de 2028
"25,10"	Octubre de 2026	Octubre de 2027	octubre de 2028
"25,06"	Junio de 2026	Junio de 2027	Junio de 2028
"25,02"	Febrero de 2026	Febrero de 2027	Febrero de 2028
"24,10"	—	Octubre de 2026	Octubre de 2027
"24,06"	—	Junio de 2026	Junio de 2027
"24,02"	—	Febrero de 2026	Febrero de 2027
"23,10"	—	—	Octubre de 2026
"23,07"	—	—	Julio de 2026
"23,04"	—	—	Abril de 2026

Autosoporte

Para obtener una lista completa de artículos sobre solución de problemas, consulta "Base de conocimientos de NetApp (se requiere inicio de sesión)".

Soporte de la comunidad

Hay una vibrante comunidad pública de usuarios de contenedores (incluidos los desarrolladores de Trident) en nuestro "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 Trident, cree un paquete de soporte con `tridentctl logs -a -n trident` y envíelo a `NetApp Support <Getting Help>`.

Si quiere más información

- "Recursos de Trident"
- "Kubernetes Hub"

Referencia

Puertos Trident

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

Descripción general

Trident utiliza varios puertos para la comunicación dentro de clústeres de Kubernetes y con storage backends. A continuación tienes un resumen de los puertos clave, sus propósitos y consideraciones de seguridad.

- **Enfoque de salida**: los nodos Kubernetes (controlador y trabajador) inician principalmente el tráfico hacia las LIF/IP de almacenamiento, así que las reglas de iptables deben permitir la salida desde las IP de los nodos hacia IP de almacenamiento específicas en estos puertos. Evita las reglas amplias "any-to-any".
- **Restricciones de entrada**: Limita los puertos internos de Trident al tráfico interno del clúster (por ejemplo, usando CNI como Calico). Sin exposición entrante innecesaria en los cortafuegos del host.
- **Seguridad de protocolo**:
  - Usa TCP siempre que sea posible (más fiable).
  - Habilita CHAP/IPsec para iSCSI si es sensible; TLS/HTTPS para gestión (puerto 443/8443).
  - Para NFSv4 (por defecto en Trident), elimina los puertos UDP/antiguos de NFSv3 (por ejemplo, 4045-4049) si no los necesitas.
  - Restringe a subredes de confianza; monitorea con herramientas como Prometheus (puerto 8001 opcional).

Puertos para nodos controladora

Estos puertos son principalmente para el operador de Trident (gestión de backend). Todos los puertos internos son de nivel pod; permite en los nodos solo si el firewall del host interfiere con CNI.

Puerto/Protocolo	Dirección	Específico	Controlador/Protocolo	Notas de seguridad
TCP 8000	Entrada/Salida (cluster-internal)	Servidor REST de Trident (comunicaciones operator-controller)	Todo	Restringe a los CIDR de los pods; sin exposición externa.
TCP 8443	Entrada/Salida (cluster-internal)	Backchannel HTTPS (API interna segura)	Todo	Cifrado TLS; limita a la malla de servicios de Kubernetes si se usa.
TCP 8001	Entrada (clúster interno, opcional)	Métricas de Prometheus	Todo	Expón solo a herramientas de supervisión (por ejemplo, usando RBAC); desactiva si no se usa.
TCP 443	Salida	HTTPS a ONTAP SVM/cluster mgmt LIF	ONTAP (todos), ANF	Requiere validación de certificado TLS; restringe solo a las IPs de mgmt LIF.

Puerto/Protocolo	Dirección	Específico	Controlador/Protocolo	Notas de seguridad
TCP 8443	Salida	HTTPS a E-Series Web Services Proxy	E-Series (iSCSI)	API de REST por defecto; usa certificados; configurable en backend YAML.

Puertos para nodos worker

Estos puertos son para daemonsets de nodos CSI y montajes de pods. Los puertos de datos son salientes hacia los LIF de datos de almacenamiento; incluye extras de NFSv3 si usas NFSv3 (opcional para NFSv4).

Puerto/Protocolo	Dirección	Específico	Controlador/Protocolo	Notas de seguridad
TCP 17546	Entrada (local a pod)	Sondas de liveness/readiness del nodo CSI	Todo	Configurable (--probe-port); asegúrate de que no haya conflictos de host; solo local.
TCP 8000	Entrada/Salida (cluster-internal)	Servidor REST de Trident	Todo	Como arriba; pod-interno.
TCP 8443	Entrada/Salida (cluster-internal)	HTTPS de canal posterior	Todo	Como arriba.
TCP 8001	Entrada (clúster interno, opcional)	Métricas de Prometheus	Todo	Como arriba.
TCP 443	Salida	HTTPS a ONTAP SVM/cluster mgmt LIF	ONTAP (todos), ANF	Como en el caso anterior; se usa para el descubrimiento.
TCP 8443	Salida	HTTPS a E-Series Web Services Proxy	E-Series (iSCSI)	Como arriba.
TCP/UDP 111	Salida	RPCBIND/portmapper	ONTAP-NAS (NFSv3/v4), ANF (NFS)	Obligatorio para v3; opcional para v4 (firewall offload); restringir si usas solo NFSv4.
TCP/UDP 2049	Salida	Demonio NFS	ONTAP-NAS (NFSv3/v4), ANF (NFS)	Datos básicos; bien conocidos; usa TCP para fiabilidad.
TCP/UDP 635	Salida	Daemon de montaje	ONTAP-NAS (NFSv3/v4), ANF (NFS)	Montaje; devoluciones de llamada bidireccionales posibles (permitir entrante efímero si es necesario).
UDP 4045	Salida	Gestor de bloqueos NFS (nlockmgr)	ONTAP-NAS (NFSv3)	Bloqueo de archivos; saltar para v4 (pNFS handles); UDP-only.

Puerto/Protocolo	Dirección	Específico	Controlador/Protocolo	Notas de seguridad
UDP 4046	Salida	Monitor de estado NFS (statd)	ONTAP-NAS (NFSv3)	Notificaciones; puede necesitar puertos efímeros de entrada (1024-65535) para callbacks.
UDP 4049	Salida	Demonio de cuotas NFS (rquotad)	ONTAP-NAS (NFSv3)	Cuotas; omitir para v4.
TCP 3260	Salida	objetivo iSCSI (discovery/data/CHAP)	ONTAP-SAN (iSCSI), E-Series (iSCSI)	Bien conocido; CHAP auth sobre este puerto; habilita CHAP mutuo para seguridad.
TCP 445	Salida	SMB/CIFS	ONTAP-NAS (SMB), ANF (SMB)	Bien conocido; usa SMB3 con cifrado (Trident annotation netapp.io/smb-encryption=true).
TCP/UDP 88 (opcional)	Salida	Autenticación Kerberos	ONTAP (NFS/SMB/iSCSI con Kerb)	Si usas Kerberos (no por defecto); a servidores AD, no almacenamiento.
TCP/UDP 389 (opcional)	Salida	LDAP	ONTAP (NFS/SMB con LDAP)	Similar; para resolución de nombres/auth; restringir a AD.

NOTA El puerto de la sonda de nivel de gravedad/preparación se puede cambiar durante la instalación utilizando el `--probe-port` bandera. Es importante asegurarse de que este puerto no esté siendo utilizado por otro proceso en los nodos de trabajo.

API de REST de Trident

Si bien "comandos y opciones de trimentctl" son la forma más sencilla de interactuar con la API de REST de Trident, puedes usar el endpoint REST directamente si lo prefieres.

Cuándo utilizar la API DE REST

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

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

Uso de la API DE REST

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

La API funciona de la siguiente manera:

OBTENGA

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

Muestra todos los objetos de ese tipo.

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

Obtiene los detalles del objeto con nombre.

PUBLICAR

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

Crea un objeto del tipo especificado.

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

ELIMINAR

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

Suprime el recurso con nombre.

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

Opciones de línea de comandos

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

Registro

`-debug`

Activa la salida de depuración.

`-loglevel <level>`

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

Kubernetes

`-k8s_pod`

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

-k8s_api_server <insecure-address:insecure-port>

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

Docker

-volume_driver <name>

Nombre del controlador utilizado al registrar el plugin de Docker. De forma predeterminada 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. Uso `-address ""` Para hacer que la interfaz DE REST sea accesible desde la dirección IP del pod.

PRECAUCIÓN	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` solicitando un nuevo `PersistentVolume` De un tamaño concreto de un Kubernetes `StorageClass` previamente configurado por el administrador.
2. Kubernetes `StorageClass` Identifica a Trident como su aprovisionador y incluye los parámetros que indican a Trident cómo aprovisionar un volumen para la clase solicitada.
3. Trident analiza sus propios recursos `StorageClass` con el mismo nombre que identifica la coincidencia `Backends` y.. `StoragePools` que puede usar para aprovisionar volúmenes para la clase.
4. Trident aprovisiona el almacenamiento en un back-end coincidente y crea dos objetos: Un `PersistentVolume` En Kubernetes, donde se indica cómo encontrar, montar y tratar el volumen, y un volumen en Trident que conserva la relación entre `PersistentVolume` y el almacenamiento real.
5. Kubernetes enlaza con el `PersistentVolumeClaim` a los nuevos `PersistentVolume`. Pods que incluyen `PersistentVolumeClaim` monte ese volumen persistente en cualquier host en el que se ejecute.
6. Un usuario crea un `VolumeSnapshot` De un PVC existente, utilizando un `VolumeSnapshotClass` Eso es lo que apunta a Trident.
7. Trident identifica el volumen asociado con la RVP y crea una copia Snapshot del volumen en su back-end. También crea un `VolumeSnapshotContent` Esto indica a Kubernetes cómo identificar la snapshot.
8. Un usuario puede crear un `PersistentVolumeClaim` uso `VolumeSnapshot` como origen.
9. Trident identifica la instantánea necesaria y realiza el mismo conjunto de pasos involucrados en la creación de un `PersistentVolume` y un `Volume`.

CONSEJO	Para obtener más información sobre los objetos de Kubernetes, recomendamos encarecidamente que lea la "Volúmenes persistentes" De la documentación de Kubernetes.

Kubernetes `PersistentVolumeClaim` objetos

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

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

Anotación	Opción de volumen	Controladores compatibles
trident.netapp.io/fileSystem	Sistema de archivos	ontap-san, solidfire-san, ontap-san-economy
trident.netapp.io/cloneFromPVC	ClonSourceVolume	ontap-nas, ontap-san, solidfire-san, azure-netapp-files, ontap-san-economy
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

Anotación	Opción de volumen	Controladores compatibles
trident.netapp.io/snapshotPolicy	Política de copias Snapshot	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup y ontap-san
trident.netapp.io/snapshotReserve	Reserva de copias Snapshot	ontap-nas, ontap-nas-flexgroup, ontap-san
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
trident.netapp.io/skipRecoveryQueue	omitir la cola de recuperación	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economy

Si el VP creado tiene el `Delete` Reclamar política, Trident elimina el VP y el volumen de respaldo cuando se libera el VP (es decir, cuando el usuario elimina la RVP). Si la acción de eliminación falla, Trident Marca el VP como tal y reintenta periódicamente la operación hasta que esta se complete o se elimine manualmente el VP. Si el VP utiliza `Retain` Política, Trident ignora la operación y asume que el administrador la limpiará desde Kubernetes y el back-end, lo que permitirá realizar un backup o la inspección del volumen antes de su eliminación. Tenga en cuenta que al eliminar el VP, Trident no eliminará el volumen de backup. Debe quitarlo usando la API DE REST (`tridentctl`).

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

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

A continuación se muestra un ejemplo: Si un usuario ya tiene una RVP llamada `mysql`, El usuario puede crear un nuevo PVC llamado `mysqlclone` mediante la anotación, por ejemplo `trident.netapp.io/cloneFromPVC: mysql`. Con este conjunto de anotaciones, Trident clona el volumen correspondiente a la RVP de mysql, en lugar de aprovisionar un volumen desde cero.

Considere los siguientes puntos:

- NetApp recomienda clonar un volumen inactivo.
- Una RVP y su clon deben estar en el mismo espacio de nombres de Kubernetes y tener el mismo tipo de almacenamiento.
- Con la `ontap-nas` y.. `ontap-san` Controladores, es posible que sea conveniente establecer la anotación de PVC `trident.netapp.io/splitOnClone` en conjunto con `trident.netapp.io/cloneFromPVC`. Con `trident.netapp.io/splitOnClone` establezca en `true`, Trident divide el volumen clonado del volumen principal y, por lo tanto, separa completamente el ciclo de vida del volumen clonado de su principal a costa de perder alguna eficiencia de almacenamiento. No está configurado `trident.netapp.io/splitOnClone` o establecerlo en `false` provoca una reducción del consumo de espacio en el back-end a costa de crear dependencias entre los volúmenes principal y clonado, de modo que no se pueda eliminar el volumen principal, a menos que el clon se

elimine primero. Una situación en la que dividir el clon tiene sentido es clonar un volumen de base de datos vacío donde se espera que tanto el volumen como su clon desvíen enormemente y no se beneficien de las eficiencias del almacenamiento ofrecidas por ONTAP.

La `sample-input` el directorio contiene ejemplos de definiciones de PVC para utilizarlas con Trident. Consulte Para obtener una descripción completa de los parámetros y la configuración asociados con los volúmenes de Trident.

Kubernetes `PersistentVolume` objetos

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

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

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

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

Kubernetes `StorageClass` objetos

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

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

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

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

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

Los parámetros de la clase de almacenamiento son:

Atributo	Tipo	Obligatorio	Descripción
atributos	map[string]string	no	Consulte la sección atributos a continuación
Pools de almacenamiento	Map[string]StringList	no	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, 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
clones	bool	verdadero, falso	Pool admite el clonado de volúmenes	Volumen con clones habilitados	ontap-nas, ontap-san, solidfire-san
cifrado	bool	verdadero, falso	El pool admite volúmenes cifrados	Volumen con cifrado habilitado	ontap-nas, ontap-nas-economy, ontap-nas-flexgroups, ontap-san
IOPS	int	entero positivo	El pool es capaz de garantizar IOPS en este rango	El volumen garantizado de estas IOPS	solidfire-san

Esta versión 1: No es compatible con sistemas ONTAP Select

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

Lo ideal es que pueda usar `attributes` solo para modelar las cualidades del almacenamiento que necesita para satisfacer las necesidades de una clase particular. Trident detecta y selecciona automáticamente pools de almacenamiento que coincidan *all* del `attributes` que especifique.

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

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

Puede utilizar el `additionalStoragePools` Parámetro para ampliar el conjunto de pools que Trident utiliza para el aprovisionamiento, independientemente de cualquier pool que seleccione `attributes` y.. `storagePools` parámetros.

Puede utilizar el `excludeStoragePools` Parámetro para filtrar el conjunto de pools que Trident utiliza para el aprovisionamiento. Cuando se usa este parámetro, se quitan todos los pools que coinciden.

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

Atributos de Kubernetes

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

Atributo	Tipo	Valores	Descripción	Controladores relevantes	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, 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

CONSEJO

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

  Para los controladores de almacenamiento NFS, ya existe un sistema de archivos como parte de la exportación NFS. Para utilizar `fsGroup` la clase de almacenamiento aún debe especificar un `fsType`. Puede configurarlo en `nfs` o cualquier valor que no sea nulo.
- Consulte "Expanda los volúmenes" para obtener más información sobre la expansión de volumen.
- El paquete de instalación de Trident proporciona varias definiciones de clase de almacenamiento de ejemplo para usar con Trident en `sample-input/storage-class-*.yaml`. Al eliminar una clase de almacenamiento Kubernetes, también se elimina el tipo de almacenamiento Trident correspondiente.

Kubernetes `VolumeSnapshotClass` objetos

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

1. `VolumeSnapshotClass` debe ser definido por un administrador para crear snapshots. Una clase de snapshot de volumen se crea con la siguiente definición:

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

La `driver` Especifica a Kubernetes que solicitudes de snapshots de volumen del `csi-snapclass` Trident gestiona la clase. La `deletionPolicy` especifica la acción que se debe realizar cuando se debe eliminar una instantánea. Cuando `deletionPolicy` se establece en `Delete`, los objetos de instantánea del volumen, así como la instantánea subyacente en el clúster de almacenamiento, se eliminan cuando se elimina una instantánea. Como alternativa, establecerlo en `Retain` significa eso `VolumeSnapshotContent` y se conserva la snapshot física.

Kubernetes `VolumeSnapshot` objetos

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

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

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

Kubernetes `VolumeSnapshotContent` objetos

Un Kubernetes `VolumeSnapshotContent` object representa una snapshot tomada de un volumen ya aprovisionado. Es similar a un `PersistentVolume` y significa una instantánea aprovisionada en el clúster de almacenamiento. Similar a. `PersistentVolumeClaim` y.. `PersistentVolume` los objetos, cuando se crea una snapshot, el `VolumeSnapshotContent` object mantiene una asignación de uno a uno `VolumeSnapshot` objeto, que solicitó la creación de la snapshot.

La `VolumeSnapshotContent` el objeto contiene detalles que identifican de manera única la instantánea, como la `snapshotHandle`. Este `snapshotHandle` Es una combinación única del nombre del PV y el nombre del `VolumeSnapshotContent` objeto.

Cuando llega una solicitud de Snapshot, Trident crea la snapshot en el back-end. Una vez creada la copia de Snapshot, Trident configura un `VolumeSnapshotContent` Objeto y, por lo tanto, expone la snapshot a la API de Kubernetes.

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

`VolumeGroupSnapshotClass`Objetos de Kubernetes

Los objetos de Kubernetes `VolumeGroupSnapshotClass` son análogos a `VolumeSnapshotClass`. Ayudan a definir múltiples clases de almacenamiento y las instantáneas de grupos de volúmenes las referencian para asociarlas con la clase de instantánea requerida. Cada instantánea de grupo de volúmenes está asociada a una única clase de instantánea de grupo de volúmenes.

A `VolumeGroupSnapshotClass` Un administrador debe definir la clase de instantánea para crear un grupo de volúmenes. La clase de instantánea de un grupo de volúmenes se crea con la siguiente definición:

```
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshotClass
metadata:
  name: csi-group-snap-class
  annotations:
    kubernetes.io/description: "Trident group snapshot class"
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

clases son manejadas por Trident. `deletionPolicy` especifica la acción que se debe tomar cuando se debe eliminar una instantánea de grupo. Cuando `deletionPolicy` está configurado para `Delete` , los objetos de instantánea del grupo de volúmenes, así como la instantánea subyacente en el clúster de almacenamiento, se eliminan cuando se elimina una instantánea. Como alternativa, si se configura en `Retain`, `VolumeGroupSnapshotContent` se conservan la instantánea física y la física.

`VolumeGroupSnapshot`Objetos de Kubernetes

Un Kubernetes `VolumeGroupSnapshot` Un objeto es una solicitud para crear una instantánea de varios volúmenes. Al igual que una PVC representa una solicitud de volumen realizada por un usuario, una instantánea de grupo de volúmenes es una solicitud de un usuario para crear una instantánea de una PVC existente.

Cuando llega una solicitud de instantánea de un grupo de volúmenes, Trident administra automáticamente la creación de la instantánea del grupo para los volúmenes en el backend y expone la instantánea creando una instantánea única. `VolumeGroupSnapshotContent` objeto. Puede crear instantáneas a partir de EVs existentes y utilizar las instantáneas como DataSource al crear nuevas CVP.

NOTA	El ciclo de vida de una instantánea de grupo de volúmenes es independiente del PVC de origen: la 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. La instantánea del grupo de volúmenes se elimina cuando se eliminan todas las instantáneas asociadas.

`VolumeGroupSnapshotContent`Objetos de Kubernetes

Un Kubernetes `VolumeGroupSnapshotContent` El objeto representa una instantánea de grupo 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 `VolumeGroupSnapshotContent` El objeto contiene detalles que identifican el grupo de instantáneas, como el `volumeGroupSnapshotHandle` y volumeSnapshotHandles individuales existentes en el sistema de almacenamiento.

Cuando se recibe una solicitud de instantánea, Trident crea la instantánea del grupo de volúmenes en el backend. Una vez creada, Trident configura un `VolumeGroupSnapshotContent` objeto y, por lo tanto, expone la instantánea a la API de Kubernetes.

Kubernetes `CustomResourceDefinition` objetos

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

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

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

Los multos son una estructura de Kubernetes. Trident crea los objetos de los recursos definidos anteriormente. Como ejemplo simple, cuando se crea un back-end usando `tridentctl`, a correspondiente `tridentbackends` El objeto CRD se crea para el consumo por parte de Kubernetes.

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

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

Objetos deTrident `StorageClass`

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

NOTA Con Kubernetes, estos objetos se crean automáticamente cuando se crea un Kubernetes `StorageClass` Que usa Trident como aprovisionador está registrado.

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

Puede crear configuraciones de clase de almacenamiento para definir clases de almacenamiento directamente mediante la API DE REST. Sin embargo, en el caso de las puestas en marcha de Kubernetes, esperamos que se creen al registrar el nuevo Kubernetes `StorageClass` objetos.

Objetos de back-end de Trident

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

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

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

Objetos deTrident `StoragePool`

Los grupos de almacenamiento representan las distintas ubicaciones disponibles para el aprovisionamiento en cada backend. Para ONTAP, estos corresponden a agregados en SVM. Para NetApp HCI/ SolidFire, estos corresponden a las bandas QoS especificadas por el administrador. Cada grupo de almacenamiento tiene un conjunto de atributos de almacenamiento distintos, que definen sus características de rendimiento y de protección de datos.

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

Objetos deTrident `Volume`

Los volúmenes son la unidad básica de aprovisionamiento, compuestos por extremos back-end, como recursos compartidos NFS y LUN iSCSI y FC. 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.

NOTA

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

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

Atributo	Tipo	Obligatorio	Descripción
versión	cadena	no	Versión de la API de Trident ("1")
nombre	cadena	sí	Nombre del volumen que se va a crear
Clase de almacenamiento	cadena	sí	Clase de almacenamiento que se utilizará al aprovisionar el volumen
tamaño	cadena	sí	El tamaño del volumen que se va a aprovisionar en bytes
protocolo	cadena	no	Tipo de protocolo que se va a utilizar; "archivo" o "bloque"
InternalName	cadena	no	Nombre del objeto en el sistema de almacenamiento, generado por Trident

Atributo	Tipo	Obligatorio	Descripción
ClonSourceVolume	cadena	no	ONTAP (nas, san) y SolidFire-*: Nombre del volumen desde el que se va a clonar
SplitOnClone	cadena	no	ONTAP (nas, san): Divida el clon entre su primario
Política de copias Snapshot	cadena	no	ONTAP-*: Política de instantánea a utilizar
Reserva de copias Snapshot	cadena	no	ONTAP-*: Porcentaje del volumen reservado para instantáneas
Política de exportoPolicy	cadena	no	ontap-nas*: Política de exportación que se va a utilizar
Snapshot shotDirectory	bool	no	ontap-nas*: Si el directorio de instantáneas está visible
Permisos univalados	cadena	no	ontap-nas*: Permisos iniciales de UNIX
Tamaño del bloque	cadena	no	SolidFire-*: Tamaño de bloque/sector
Sistema de archivos	cadena	no	Tipo de sistema de archivos
omitir la cola de recuperación	cadena	no	Durante la eliminación de un volumen, omita la cola de recuperación en el almacenamiento y elimine el volumen inmediatamente.

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

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

Objetos deTrident `Snapshot`

Las Snapshot son una copia de un momento específico de los volúmenes, que se pueden usar para aprovisionar nuevos volúmenes o restaurar el estado. En Kubernetes, se corresponden directamente con

`VolumeSnapshotContent` objetos. Cada copia de Snapshot se asocia con un volumen, que es el origen de los datos de la copia de Snapshot.

Cada uno `Snapshot` object incluye las propiedades que se enumeran a continuación:

Atributo	Tipo	Obligatorio	Descripción
versión	Cadena	Sí	Versión de la API de Trident ("1")
nombre	Cadena	Sí	Nombre del objeto Snapshot de Trident
InternalName	Cadena	Sí	Nombre del objeto Snapshot de Trident en el sistema de almacenamiento
Nombre de volumen	Cadena	Sí	Nombre del volumen persistente para el que se crea la snapshot
VolumeInternalName	Cadena	Sí	Nombre del objeto de volumen de Trident asociado en el sistema de almacenamiento

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

Cuando un Kubernetes `VolumeSnapshot` Se crea la solicitud del objeto, Trident funciona mediante la creación de un objeto Snapshot en el sistema de almacenamiento que realiza backups. La `internalName` de este objeto snapshot se genera combinando el prefijo `snapshot-` con la `UID` de la `VolumeSnapshot` objeto (por ejemplo, `snapshot-e8d8a0ca-9826-11e9-9807-525400f3f660`). `volumeName` y.. `volumeInternalName` para rellenar, obtendrá los detalles del volumen de backup.

ObjetoTrident `ResourceQuota`

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

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

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

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

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

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

Limpie `ResourceQuota` si la instalación falla

En el raro caso en que la instalación falle después del `ResourceQuota` se crea el objeto, primero se intenta "desinstalando" y, a continuación, vuelva a instalar.

Si esto no funciona, quite manualmente la `ResourceQuota` objeto.

Quitar `ResourceQuota`

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

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

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

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

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

Contexto de Kubernetes Security y campos relacionados necesarios

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

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

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

Estándares de seguridad para POD (PSS)

Etiqueta	Descripción	Predeterminado

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

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

Directivas de seguridad de POD (PSP)

Campo	Descripción	Predeterminado
`allowPrivilegeEscalation`	Los contenedores con privilegios deben permitir la escala de privilegios.	`true`
`allowedCSIDrivers`	Trident no utiliza volúmenes efímeros de CSI en línea.	Vacío
`allowedCapabilities`	Los contenedores Trident no con privilegios no requieren más funcionalidades de las que se establece de forma predeterminada y se conceden todas las funcionalidades posibles a los contenedores con privilegios.	Vacío
`allowedFlexVolumes`	Trident no utiliza "Controlador FlexVolume", por lo tanto, no se incluyen en la lista de volúmenes permitidos.	Vacío
`allowedHostPaths`	El pod del nodo Trident monta el sistema de archivos raíz del nodo, por lo que no hay ninguna ventaja para configurar esta lista.	Vacío
`allowedProcMountTypes`	Trident no utiliza ninguna `ProcMountTypes`.	Vacío
`allowedUnsafeSysctls`	Trident no requiere que no sea seguro `sysctls`.	Vacío
`defaultAddCapabilities`	No es necesario añadir capacidades a contenedores con privilegios.	Vacío
`defaultAllowPrivilegeEscalation`	En cada POD de Trident, se permite el escalado de privilegios.	`false`

Campo	Descripción	Predeterminado
`forbiddenSysctls`	No `sysctls` se permiten.	Vacío
`fsGroup`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`hostIPC`	El montaje de volúmenes NFS requiere que el IPC del host se comunique con `nfsd`	`true`
`hostNetwork`	Iscsiadm requiere que la red del host se comunique con el demonio iSCSI.	`true`
`hostPID`	Se requiere el PID del host para comprobar si `rpc-statd` está ejecutándose en el nodo.	`true`
`hostPorts`	Trident no utiliza puertos de host.	Vacío
`privileged`	Los pods de nodo Trident deben ejecutar un contenedor privilegiado para montar volúmenes.	`true`
`readOnlyRootFilesystem`	Los contenedores de nodos Trident deben escribir en el sistema de archivos del nodo.	`false`
`requiredDropCapabilities`	Los pods de nodo de Trident ejecutan un contenedor privilegiado y no pueden soltar las funcionalidades.	`none`
`runAsGroup`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`runAsUser`	Los contenedores Trident se ejecutan como raíz.	`runAsAny`
`runtimeClass`	Trident no utiliza `RuntimeClasses`.	Vacío
`seLinux`	Trident no está configurado `seLinuxOptions` Debido a que actualmente existen diferencias en el modo en que los tiempos de ejecución de contenedores y las distribuciones de Kubernetes se encargan de SELinux.	Vacío
`supplementalGroups`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`volumes`	Los pods de Trident requieren estos complementos de volumen.	`hostPath, projected, emptyDir`

Restricciones de contexto de seguridad (SCC)

Etiquetas	Descripción	Predeterminado
`allowHostDirVolumePlugin`	Los contenedores de nodos Trident montan el sistema de archivos raíz del nodo.	`true`
`allowHostIPC`	El montaje de volúmenes NFS requiere que el IPC del host se comunique con `nfsd`.	`true`
`allowHostNetwork`	Iscsiadm requiere que la red del host se comunique con el demonio iSCSI.	`true`
`allowHostPID`	Se requiere el PID del host para comprobar si `rpc-statd` está ejecutándose en el nodo.	`true`
`allowHostPorts`	Trident no utiliza puertos de host.	`false`
`allowPrivilegeEscalation`	Los contenedores con privilegios deben permitir la escala de privilegios.	`true`
`allowPrivilegedContainer`	Los pods de nodo Trident deben ejecutar un contenedor privilegiado para montar volúmenes.	`true`
`allowedUnsafeSysctls`	Trident no requiere que no sea seguro `sysctls`.	`none`
`allowedCapabilities`	Los contenedores Trident no con privilegios no requieren más funcionalidades de las que se establece de forma predeterminada y se conceden todas las funcionalidades posibles a los contenedores con privilegios.	Vacío
`defaultAddCapabilities`	No es necesario añadir capacidades a contenedores con privilegios.	Vacío
`fsGroup`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`groups`	Este SCC es específico de Trident y está vinculado a su usuario.	Vacío
`readOnlyRootFilesystem`	Los contenedores de nodos Trident deben escribir en el sistema de archivos del nodo.	`false`
`requiredDropCapabilities`	Los pods de nodo de Trident ejecutan un contenedor privilegiado y no pueden soltar las funcionalidades.	`none`

Etiquetas	Descripción	Predeterminado
`runAsUser`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`seLinuxContext`	Trident no está configurado `seLinuxOptions` Debido a que actualmente existen diferencias en el modo en que los tiempos de ejecución de contenedores y las distribuciones de Kubernetes se encargan de SELinux.	Vacío
`seccompProfiles`	Los contenedores privilegiados siempre funcionan "sin confinar".	Vacío
`supplementalGroups`	Los contenedores Trident se ejecutan como raíz.	`RunAsAny`
`users`	Se proporciona una entrada para vincular este SCC al usuario Trident en el espacio de nombres Trident.	n.a.
`volumes`	Los pods de Trident requieren estos complementos de volumen.	`hostPath, downwardAPI, projected, emptyDir`

Avisos legales

Los avisos legales proporcionan acceso a las declaraciones de copyright, marcas comerciales, patentes y mucho más.

Derechos de autor

"https://www.netapp.com/company/legal/copyright/"

Marcas comerciales

NETAPP, el logotipo de NETAPP y las marcas enumeradas en la página de marcas comerciales de NetApp son marcas comerciales de NetApp, Inc. Los demás nombres de empresas y productos son marcas comerciales de sus respectivos propietarios.

"https://www.netapp.com/company/legal/trademarks/"

Estadounidenses

Puede encontrar una lista actual de las patentes propiedad de NetApp en:

https://www.netapp.com/pdf.html?item=/media/11887-patentspage.pdf

Política de privacidad

"https://www.netapp.com/company/legal/privacy-policy/"

Código abierto

Puede revisar los derechos de autor y las licencias de terceros utilizados en el software de NetApp para Trident en el archivo de avisos de cada versión en https://github.com/NetApp/trident/.

Información de copyright

Copyright © 2026 NetApp, Inc. Todos los derechos reservados. Imprimido en EE. UU. No se puede reproducir este documento protegido por copyright ni parte del mismo de ninguna forma ni por ningún medio (gráfico, electrónico o mecánico, incluidas fotocopias, grabaciones o almacenamiento en un sistema de recuperación electrónico) sin la autorización previa y por escrito del propietario del copyright.

El software derivado del material de NetApp con copyright está sujeto a la siguiente licencia y exención de responsabilidad:

ESTE SOFTWARE LO PROPORCIONA NETAPP «TAL CUAL» Y SIN NINGUNA GARANTÍA EXPRESA O IMPLÍCITA, INCLUYENDO, SIN LIMITAR, LAS GARANTÍAS IMPLÍCITAS DE COMERCIALIZACIÓN O IDONEIDAD PARA UN FIN CONCRETO, CUYA RESPONSABILIDAD QUEDA EXIMIDA POR EL PRESENTE DOCUMENTO. EN NINGÚN CASO NETAPP SERÁ RESPONSABLE DE NINGÚN DAÑO DIRECTO, INDIRECTO, ESPECIAL, EJEMPLAR O RESULTANTE (INCLUYENDO, ENTRE OTROS, LA OBTENCIÓN DE BIENES O SERVICIOS SUSTITUTIVOS, PÉRDIDA DE USO, DE DATOS O DE BENEFICIOS, O INTERRUPCIÓN DE LA ACTIVIDAD EMPRESARIAL) CUALQUIERA SEA EL MODO EN EL QUE SE PRODUJERON Y LA TEORÍA DE RESPONSABILIDAD QUE SE APLIQUE, YA SEA EN CONTRATO, RESPONSABILIDAD OBJETIVA O AGRAVIO (INCLUIDA LA NEGLIGENCIA U OTRO TIPO), QUE SURJAN DE ALGÚN MODO DEL USO DE ESTE SOFTWARE, INCLUSO SI HUBIEREN SIDO ADVERTIDOS DE LA POSIBILIDAD DE TALES DAÑOS.

NetApp se reserva el derecho de modificar cualquiera de los productos aquí descritos en cualquier momento y sin aviso previo. NetApp no asume ningún tipo de responsabilidad que surja del uso de los productos aquí descritos, excepto aquello expresamente acordado por escrito por parte de NetApp. El uso o adquisición de este producto no lleva implícita ninguna licencia con derechos de patente, de marcas comerciales o cualquier otro derecho de propiedad intelectual de NetApp.

Es posible que el producto que se describe en este manual esté protegido por una o más patentes de EE. UU., patentes extranjeras o solicitudes pendientes.

LEYENDA DE DERECHOS LIMITADOS: el uso, la copia o la divulgación por parte del gobierno están sujetos a las restricciones establecidas en el subpárrafo (b)(3) de los derechos de datos técnicos y productos no comerciales de DFARS 252.227-7013 (FEB de 2014) y FAR 52.227-19 (DIC de 2007).

Los datos aquí contenidos pertenecen a un producto comercial o servicio comercial (como se define en FAR 2.101) y son propiedad de NetApp, Inc. Todos los datos técnicos y el software informático de NetApp que se proporcionan en este Acuerdo tienen una naturaleza comercial y se han desarrollado exclusivamente con fondos privados. El Gobierno de EE. UU. tiene una licencia limitada, irrevocable, no exclusiva, no transferible, no sublicenciable y de alcance mundial para utilizar los Datos en relación con el contrato del Gobierno de los Estados Unidos bajo el cual se proporcionaron los Datos. Excepto que aquí se disponga lo contrario, los Datos no se pueden utilizar, desvelar, reproducir, modificar, interpretar o mostrar sin la previa aprobación por escrito de NetApp, Inc. Los derechos de licencia del Gobierno de los Estados Unidos de América y su Departamento de Defensa se limitan a los derechos identificados en la cláusula 252.227-7015(b) de la sección DFARS (FEB de 2014).

Información de la marca comercial

NETAPP, el logotipo de NETAPP y las marcas que constan en http://www.netapp.com/TM son marcas comerciales de NetApp, Inc. El resto de nombres de empresa y de producto pueden ser marcas comerciales de sus respectivos propietarios.