Gestione y supervise Trident

Trident

NetApp
February 02, 2026

Tabla de contenidos

Gestione y supervise Trident . . . 1
- Actualice Trident . . . 1
  - Actualice Trident . . . 1
  - Actualizar con el operador . . . 2
  - Actualice con trimentctl . . . 7
- Gestione Trident con tridentctl . . . 8
  - Comandos e indicadores globales . . . 8
  - Opciones de comando y indicadores . . . 10
  - Compatibilidad con complementos . . . 16
- Supervisar Trident . . . 16
  - Descripción general . . . 16
  - Paso 1: Definir un objetivo Prometheus . . . 16
  - Paso 2: Cree un Prometheus ServiceMonitor . . . 17
  - Paso 3: Consulte las métricas de Trident con PromQL . . . 19
  - Más información sobre la telemetría de Trident AutoSupport . . . 20
  - Deshabilitar las métricas de Trident . . . 21
- Desinstale Trident . . . 21
  - Determine el método de instalación original . . . 21
  - Desinstale una instalación del operador Trident . . . 21
  - Desinstale un `tridentctl` instalación . . . 22

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.

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

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

- "Actualice con el operador Trident"

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 "Ventajas del uso del operador Trident"muchos que instalan y actualizan Trident es la gestión automática de objetos Trident y Kubernetes sin interrumpir los volúmenes montados existentes. De esta forma, Trident puede admitir renovaciones sin tiempos de inactividad o "*actualizaciones sucesivas*". En concreto, el operador Trident se comunica con el clúster de Kubernetes para:

- Elimine y vuelva a crear la implementación de Trident Controller y DaemonSet de nodos.
- Sustituya el pod de la controladora de Trident y los pods de nodos de Trident por nuevas versiones.

- Si no se actualiza un nodo, no impide que se actualicen los nodos restantes.
- Solo los nodos con un nodo de Trident en ejecución pueden montar volúmenes.

Para obtener más información sobre la arquitectura de 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

Puede actualizar Trident mediante el operador Trident manualmente o mediante Helm. Puede actualizar de una instalación del operador de Trident a otra instalación del operador de Trident o actualizar de una `tridentctl` instalación a una versión del operador de Trident. Revise "Seleccione un método de actualización" antes de actualizar una instalación del operador Trident.

Actualizar una instalación manual

Puede actualizar desde una instalación de 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.

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

Antes de empezar

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

Pasos

1. Compruebe su versión de 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.

> 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"utiliza `helm upgrade trident netapp-trident/trident-operator --version 100.2506.0` para actualizar en un solo paso. Si no ha añadido el repositorio Helm o no puede utilizarlo para actualizar:
   a. Descargue la versión más reciente de Trident en "La sección *Assets* de GitHub".
   b. Utilice el `helm upgrade` comando donde `trident-operator-25.10.0.tgz` refleja la versión a la que desea actualizar.

```
helm upgrade <name> trident-operator-25.10.0.tgz
```

> 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.

> Antes de cambiar entre los métodos de instalación, revise "Moverse entre los métodos de instalación".

Pasos

1. Descargue la versión más reciente de Trident.

```
# Download the release required [25.10.0]
mkdir 25.10.0
cd 25.10.0
wget
https://github.com/NetApp/trident/releases/download/v25.10.0/trident-
installer-25.10.0.tar.gz
tar -xf trident-installer-25.10.0.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:                v25.10.0
```

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. Vuelva a instalar Trident. Consulte "Instale Trident usando tridentctl".

> 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.

> 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.

> 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.

> 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

> 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. Incluye una "CLUF" que no es aplicable a la propia imagen del contenedor de Trident. Puede obtener más información sobre el compromiso de NetApp con la seguridad y la confianza de los datos "aquí".

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>
```