Gestionar back-ends

Trident

NetApp
January 14, 2026

Tabla de contenidos

Gestionar back-ends

Realice la gestión del entorno de administración con kubectl

Obtenga información sobre cómo realizar operaciones de gestión de backend mediante `kubectl`.

Eliminar un back-end

Al 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` u `tridentctl get backend -o yaml -n trident` obtener una lista de todos los back-ends existentes. Esta lista también incluirá back-ends creados con `tridentctl`.

Actualizar un back-end

Puede haber varias razones para actualizar un back-end:

- Las credenciales del sistema de almacenamiento han cambiado. Para actualizar las credenciales, se debe actualizar el secreto de Kubernetes utilizado en el `TridentBackendConfig` objeto. 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 `TridentBackendConfig` actualizar objetos directamente a través de Kubernetes mediante el siguiente comando:

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

  - Como alternativa, puede realizar cambios en el CR existente `TridentBackendConfig` mediante el siguiente comando:

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

> - Si falla una actualización de back-end, el back-end continúa en su última configuración conocida. Puede ver los logs para determinar la causa ejecutando `kubectl get tbc <tbc-name> -o yaml -n trident` o `kubectl describe tbc <tbc-name> -n trident`.
> - Después de identificar y corregir el problema con el archivo de configuración, puede volver a ejecutar el comando update.

Realizar la administración de back-end con trimentctl

Obtenga información sobre cómo realizar operaciones de gestión de backend mediante `tridentctl`.

Cree un back-end

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

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

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

```
tridentctl logs -n trident
```

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

Eliminar un back-end

Para 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
```

> 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 back-end continuará existiendo en un estado de “eliminación” y Trident seguirá gestionando esos volúmenes y instantáneas hasta que se eliminen.

Ver los back-ends existentes

Para ver los back-ends que Trident conoce, haga lo siguiente:

- Para obtener un resumen, ejecute el siguiente comando:

```
tridentctl get backend -n trident
```

- Para obtener todos los detalles, ejecute el siguiente comando:

```
tridentctl get backend -o json -n trident
```

Actualizar un back-end

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

```
tridentctl update backend <backend-name> -f <backend-file> -n trident
```

Si falla la actualización del back-end, algo estaba mal con la configuración del back-end o intentó una actualización no válida. Puede ver los registros para determinar la causa ejecutando el siguiente comando:

```
tridentctl logs -n trident
```

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

Identifique las clases de almacenamiento que utilizan un back-end

Este es un ejemplo del tipo de preguntas que puede responder con el JSON que `tridentctl` genera los objetos backend. Esto utiliza la `jq` utilidad, que necesita instalar.

```
tridentctl get backend -o json | jq '[.items[] | {backend: .name,
storageClasses: [.storage[].storageClasses]|unique}]'
```

Esto también se aplica a los back-ends que se crearon mediante el uso `TridentBackendConfig`de .

Pasar entre las opciones de administración del back-end

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 gestionar back-ends. Esto plantea las siguientes preguntas:

- ¿Se pueden crear back-ends mediante `tridentctl` ser gestionados con `TridentBackendConfig`?
- Se pueden crear back-ends mediante la utilización `TridentBackendConfig` de `tridentctl`?

Gestionar `tridentctl` back-ends utilizando `TridentBackendConfig`

Esta sección cubre los pasos necesarios para administrar los back-ends que se crearon `tridentctl` directamente a través de la interfaz de Kubernetes mediante la creación de `TridentBackendConfig` objetos.

Esto se aplica a las siguientes situaciones:

- Back-ends preexistentes, que no tienen un `TridentBackendConfig` porque fueron creados con `tridentctl`.
- Nuevos back-ends creados con `tridentctl`, mientras existen otros `TridentBackendConfig` objetos.

En ambos escenarios, los back-ends seguirán presentes, con Trident programando volúmenes y operando en ellos. A continuación, los administradores tienen una de estas dos opciones:

- Siga `tridentctl` utilizando para gestionar los back-ends creados con él.
- Backend de enlace creado mediante `tridentctl` a un nuevo `TridentBackendConfig` objeto. Hacerlo significaría que los back-ends se gestionarán usando `kubectl` y no `tridentctl`.

Para gestionar un backend preexistente mediante `kubectl`, deberá crear un `TridentBackendConfig` que se vincule al backend existente. A continuación se ofrece una descripción general de cómo funciona:

1. Cree un secreto de Kubernetes. El secreto contiene las credenciales que Trident necesita para comunicarse con el clúster/servicio de almacenamiento.
2. Crear `TridentBackendConfig` un objeto. Este contiene detalles sobre el servicio/clúster de almacenamiento y hace referencia al secreto creado en el paso anterior. Se debe tener cuidado de especificar parámetros de configuración idénticos (`spec.backendName`como , , , `spec.storagePrefix` `spec.storageDriverName` etc.). `spec.backendName` se debe definir en el nombre del backend existente.

Paso 0: Identificar el back-end

Para crear un `TridentBackendConfig` que se vincule a un backend existente, deberá obtener la configuración de backend. En este ejemplo, supongamos que se ha creado un back-end mediante la siguiente definición JSON:

```
tridentctl get backend ontap-nas-backend -n trident
+---------------------+----------------
+--------------------------------------+--------+---------+
|          NAME       | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+---------------------+----------------
+--------------------------------------+--------+---------+
| ontap-nas-backend   | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |      25 |
+---------------------+----------------
+--------------------------------------+--------+---------+

cat ontap-nas-backend.json

{
    "version": 1,
    "storageDriverName": "ontap-nas",
    "managementLIF": "10.10.10.1",
    "dataLIF": "10.10.10.2",
    "backendName": "ontap-nas-backend",
    "svm": "trident_svm",
    "username": "cluster-admin",
    "password": "admin-password",

    "defaults": {
        "spaceReserve": "none",
        "encryption": "false"
    },
    "labels":{"store":"nas_store"},
    "region": "us_east_1",
    "storage": [
        {
            "labels":{"app":"msoffice", "cost":"100"},
            "zone":"us_east_1a",
            "defaults": {
                "spaceReserve": "volume",
                "encryption": "true",
                "unixPermissions": "0755"
            }
        },
        {
```

```
            "labels":{"app":"mysqldb", "cost":"25"},
            "zone":"us_east_1d",
            "defaults": {
                "spaceReserve": "volume",
                "encryption": "false",
                "unixPermissions": "0775"
            }
        }
    ]
}
```

Paso 1: Cree un secreto de Kubernetes

Cree un secreto que contenga las credenciales del back-end, como se muestra en este ejemplo:

```
cat tbc-ontap-nas-backend-secret.yaml

apiVersion: v1
kind: Secret
metadata:
  name: ontap-nas-backend-secret
type: Opaque
stringData:
  username: cluster-admin
  password: admin-password

kubectl create -f tbc-ontap-nas-backend-secret.yaml -n trident
secret/backend-tbc-ontap-san-secret created
```

Paso 2: Crear un `TridentBackendConfig` CR

El siguiente paso consiste en crear un `TridentBackendConfig` CR que se enlazará automáticamente a la preexistente `ontap-nas-backend` (como en este ejemplo). Asegurarse de que se cumplen los siguientes requisitos:

- El mismo nombre de backend se define en `spec.backendName`.
- Los parámetros de configuración son idénticos al backend original.
- Los pools virtuales (si están presentes) deben conservar el mismo orden que en el back-end original.
- Las credenciales se proporcionan a través de un secreto de Kubernetes, pero no en texto sin formato.

En este caso, el `TridentBackendConfig` se verá así:

```
cat backend-tbc-ontap-nas.yaml
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-ontap-nas-backend
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.10.10.1
  dataLIF: 10.10.10.2
  backendName: ontap-nas-backend
  svm: trident_svm
  credentials:
    name: mysecret
  defaults:
    spaceReserve: none
    encryption: 'false'
  labels:
    store: nas_store
  region: us_east_1
  storage:
  - labels:
      app: msoffice
      cost: '100'
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: 'true'
      unixPermissions: '0755'
  - labels:
      app: mysqldb
      cost: '25'
    zone: us_east_1d
    defaults:
      spaceReserve: volume
      encryption: 'false'
      unixPermissions: '0775'

kubectl create -f backend-tbc-ontap-nas.yaml -n trident
tridentbackendconfig.trident.netapp.io/tbc-ontap-nas-backend created
```

Paso 3: Verifique el estado de la `TridentBackendConfig` CR

Una vez creado el `TridentBackendConfig`, su fase debe ser `Bound`. También debería reflejar el mismo nombre de fondo y UUID que el del back-end existente.

```
kubectl get tbc tbc-ontap-nas-backend -n trident
NAME                   BACKEND NAME          BACKEND UUID
PHASE   STATUS
tbc-ontap-nas-backend  ontap-nas-backend     52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7   Bound   Success

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

El backend ahora será completamente administrado usando el `tbc-ontap-nas-backend` `TridentBackendConfig` objeto.

Gestionar `TridentBackendConfig` back-ends utilizando `tridentctl`

```
`tridentctl` se puede utilizar para mostrar los back-ends creados con
`TridentBackendConfig`. Además, los administradores también pueden optar
por administrar completamente dichos back-ends `tridentctl` mediante la
eliminación `TridentBackendConfig` y asegurarse de `spec.deletionPolicy`
que se establece en `retain`.
```

Paso 0: Identificar el back-end

Por ejemplo, supongamos que el siguiente backend se creó usando `TridentBackendConfig`:

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

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

A partir de la salida, se ve que `TridentBackendConfig` se ha creado correctamente y está enlazado a un backend [observe el UUID del backend].

Paso 1: Confirme `deletionPolicy` **que está establecido en** `retain`

Echemos un vistazo al valor de `deletionPolicy`. Se debe establecer en `retain`. Esto 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
```

No continúe con el siguiente paso a menos que `deletionPolicy` esté establecido en `retain`.

Paso 2: Eliminar el `TridentBackendConfig` CR

El paso final es eliminar la `TridentBackendConfig` CR. Después de confirmar que el `deletionPolicy` está definido en `retain`, puede continuar con la eliminación:

```
kubectl delete tbc backend-tbc-ontap-san -n trident
tridentbackendconfig.trident.netapp.io "backend-tbc-ontap-san" deleted

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

Tras la eliminación del `TridentBackendConfig` objeto, Trident simplemente lo elimina sin eliminar realmente el backend.