Gérer les backends

Trident

NetApp
January 15, 2026

Sommaire

Gérer les backends

Effectuez la gestion du backend avec kubectl

Découvrez comment effectuer des opérations de gestion du backend en utilisant `kubectl` .

Supprimer un backend

En supprimant un `TridentBackendConfig` , vous demandez à Trident de supprimer/conserver les backends (en fonction de `deletionPolicy`). Pour supprimer un backend, assurez-vous que `deletionPolicy` est configuré pour être supprimé. Pour supprimer uniquement le `TridentBackendConfig` , assurez-vous que `deletionPolicy` est prévu pour conserver. Cela garantit que le système dorsal est toujours présent et peut être géré à l'aide de `tridentctl` .

Exécutez la commande suivante :

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

Trident ne supprime pas les secrets Kubernetes qui étaient utilisés par `TridentBackendConfig` . L'utilisateur Kubernetes est responsable du nettoyage des secrets. Il convient d'être prudent lors de la suppression de secrets. Vous ne devez supprimer les secrets que s'ils ne sont pas utilisés par les systèmes backend.

Afficher les backends existants

Exécutez la commande suivante :

```
kubectl get tbc -n trident
```

Vous pouvez également courir `tridentctl get backend -n trident` ou `tridentctl get backend -o yaml -n trident` pour obtenir la liste de tous les serveurs backend existants. Cette liste inclura également les backends créés avec `tridentctl` .

Mettre à jour un backend

Il peut exister plusieurs raisons de mettre à jour un backend :

- Les identifiants d'accès au système de stockage ont changé. Pour mettre à jour les informations d'identification, le secret Kubernetes utilisé dans le `TridentBackendConfig` L'objet doit être mis à jour. Trident mettra automatiquement à jour le système dorsal avec les dernières informations d'identification fournies. Exécutez la commande suivante pour mettre à jour le secret Kubernetes :

  ```
  kubectl apply -f <updated-secret-file.yaml> -n trident
  ```

- Les paramètres (tels que le nom de la SVM ONTAP utilisée) doivent être mis à jour.

- Vous pouvez mettre à jour `TridentBackendConfig` objets directement via Kubernetes à l'aide de la commande suivante :

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

- Vous pouvez également apporter des modifications à l'existant `TridentBackendConfig` CR en utilisant la commande suivante :

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

- En cas d'échec d'une mise à jour du système dorsal, celui-ci reste dans sa dernière configuration connue. Vous pouvez consulter les journaux pour déterminer la cause en exécutant la commande suivante : `kubectl get tbc <tbc-name> -o yaml -n trident` ou `kubectl describe tbc <tbc-name> -n trident`.
- Une fois le problème du fichier de configuration identifié et corrigé, vous pouvez relancer la commande de mise à jour.

Effectuez la gestion du backend avec tridentctl

Découvrez comment effectuer des opérations de gestion du backend en utilisant `tridentctl`.

Créer un backend

Après avoir créé un"fichier de configuration du backend" , exécutez la commande suivante :

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

Si la création du backend échoue, c'est qu'il y a un problème avec sa configuration. Vous pouvez consulter les journaux pour déterminer la cause en exécutant la commande suivante :

```
tridentctl logs -n trident
```

Après avoir identifié et corrigé le problème du fichier de configuration, vous pouvez simplement exécuter la commande suivante : `create` commandez à nouveau.

Supprimer un backend

Pour supprimer un backend de Trident, procédez comme suit :

1. Récupérer le nom du serveur :

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

2. Supprimer le backend :

```
tridentctl delete backend <backend-name> -n trident
```

> Si Trident a provisionné des volumes et des instantanés à partir de ce backend qui existent encore, la suppression du backend empêche le provisionnement de nouveaux volumes par celui-ci. Le système dorsal restera dans un état « Suppression ».

Afficher les backends existants

Pour consulter les serveurs backend connus de Trident , procédez comme suit :

- Pour obtenir un résumé, exécutez la commande suivante :

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

- Pour obtenir tous les détails, exécutez la commande suivante :

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

Mettre à jour un backend

Après avoir créé un nouveau fichier de configuration backend, exécutez la commande suivante :

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

Si la mise à jour du serveur échoue, cela signifie qu'il y a un problème avec la configuration du serveur ou que vous avez tenté une mise à jour invalide. Vous pouvez consulter les journaux pour déterminer la cause en exécutant la commande suivante :

```
tridentctl logs -n trident
```

Après avoir identifié et corrigé le problème du fichier de configuration, vous pouvez simplement exécuter la commande suivante : `update` commandez à nouveau.

Identifiez les classes de stockage qui utilisent un backend

Voici un exemple du type de questions auxquelles vous pouvez répondre avec le JSON. `tridentctl` Sorties pour les objets backend. Cela utilise le `jq` utilitaire que vous devez installer.

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

Cela s'applique également aux backends créés à l'aide de `TridentBackendConfig` .

Passer d'une option de gestion du backend à une autre

Découvrez les différentes manières de gérer les backends dans Trident.

Options pour la gestion des backends

Avec l'introduction de `TridentBackendConfig` Les administrateurs disposent désormais de deux méthodes uniques pour gérer les systèmes d'arrière-plan. Cela soulève les questions suivantes :

- Les backends peuvent-ils être créés à l'aide de `tridentctl` être géré avec `TridentBackendConfig` ?
- Les backends peuvent-ils être créés à l'aide de `TridentBackendConfig` être géré à l'aide de `tridentctl` ?

Gérer `tridentctl` backends utilisant `TridentBackendConfig`

Cette section décrit les étapes nécessaires à la gestion des backends créés à l'aide de `tridentctl` directement via l'interface Kubernetes en créant `TridentBackendConfig` objets.

Cela s'appliquera aux scénarios suivants :

- Les systèmes backend préexistants, qui n'ont pas de `TridentBackendConfig` parce qu'ils ont été créés avec `tridentctl` .
- De nouveaux backends créés avec `tridentctl` , tandis que d'autres `TridentBackendConfig` Les objets existent.

Dans les deux cas, les serveurs d'arrière-plan resteront en place, Trident assurant la planification et le traitement des volumes. Les administrateurs ont ici l'un des deux choix suivants :

- Continuer à utiliser `tridentctl` pour gérer les backends créés à l'aide de celui-ci.
- Liaison des backends créés à l'aide de `tridentctl` à un nouveau `TridentBackendConfig` objet. Cela impliquerait que les backends seraient gérés à l'aide de `kubectl` et non `tridentctl` .

Pour gérer un backend préexistant en utilisant `kubectl` , vous devrez créer un `TridentBackendConfig` qui se lie au système dorsal existant. Voici un aperçu de son fonctionnement :

1. Créer un secret Kubernetes. Ce secret contient les identifiants dont Trident a besoin pour communiquer avec le cluster/service de stockage.
2. Créer un `TridentBackendConfig` objet. Ce fichier contient des informations spécifiques sur le cluster/service de stockage et fait référence au secret créé à l'étape précédente. Il convient de veiller à spécifier des paramètres de configuration identiques (tels que `spec.backendName` , `spec.storagePrefix` , `spec.storageDriverName` , et ainsi de suite). `spec.backendName` doit être défini sur le nom du backend existant.

Étape 0 : Identifier le backend

Pour créer un `TridentBackendConfig` Pour que cette fonctionnalité se lie à un système dorsal existant, vous devrez obtenir la configuration de ce système dorsal. Dans cet exemple, supposons qu'un backend ait été créé à l'aide de la définition JSON suivante :

```
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"
      }
    }
  ]
}
```

Étape 1 : Créer un secret Kubernetes

Créez un secret contenant les identifiants du serveur, comme indiqué dans cet exemple :

```
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
```

Étape 2 : Créer un `TridentBackendConfig` CR

L'étape suivante consiste à créer un `TridentBackendConfig` CR qui se liera automatiquement à la CR préexistante `ontap-nas-backend` (comme dans cet exemple). Assurez-vous que les exigences suivantes sont respectées :

- Le même nom de backend est défini dans `spec.backendName` .
- Les paramètres de configuration sont identiques à ceux du système dorsal d'origine.
- Les pools virtuels (le cas échéant) doivent conserver le même ordre que dans le backend d'origine.
- Les informations d'identification sont fournies via un secret Kubernetes et non en clair.

Dans ce cas, le `TridentBackendConfig` Cela ressemblera à ceci :

```
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
```

Étape 3 : Vérifier l'état de `TridentBackendConfig` CR

Après le `TridentBackendConfig` a été créée, sa phase doit être `Bound` . Il doit également refléter le même nom de backend et le même UUID que ceux du backend existant.

```
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 |
+---------------------+----------------
+--------------------------------------+--------+---------+
```

Le backend sera désormais entièrement géré à l'aide de `tbc-ontap-nas-backend TridentBackendConfig` objet.

Gérer `TridentBackendConfig` backends utilisant `tridentctl`

```
`tridentctl`peut être utilisé pour lister les backends qui ont été créés à
l'aide de `TridentBackendConfig` .  De plus, les administrateurs peuvent
également choisir de gérer entièrement ces backends via `tridentctl` en
supprimant `TridentBackendConfig` et en veillant à `spec.deletionPolicy`
est réglé sur `retain` .
```

Étape 0 : Identifier le backend

Par exemple, supposons que le backend suivant ait été créé à l'aide de `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 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

Les résultats montrent que `TridentBackendConfig` a été créé avec succès et est lié à un backend [observer l'UUID du backend].

Étape 1 : Confirmer `deletionPolicy` **est réglé sur** `retain`

Examinons la valeur de `deletionPolicy` . Il faut régler cela sur `retain` . Cela garantit que lorsqu'un `TridentBackendConfig` La CR est supprimée, mais la définition du backend reste présente et peut être gérée avec `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
```

Ne passez pas à l'étape suivante sauf si `deletionPolicy` est réglé sur `retain` .

Étape 2 : Supprimer le `TridentBackendConfig` CR

La dernière étape consiste à supprimer le `TridentBackendConfig` CR. Après avoir confirmé le `deletionPolicy` est réglé sur `retain` , vous pouvez procéder à la suppression :

```
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 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

Suite à la suppression de `TridentBackendConfig` Trident supprime simplement l'objet sans supprimer réellement le backend lui-même.

Informations sur le copyright

Informations sur les marques commerciales