Gerenciar back-ends

Trident

NetApp
January 15, 2026

Índice

Gerenciar back-ends

Realize o gerenciamento de backend com kubectl

Aprenda como executar operações de gerenciamento de back-end usando `kubectl` .

Excluir um backend

Ao excluir um `TridentBackendConfig` , você instrui o Trident a excluir/reter backends (com base em `deletionPolicy`). Para excluir um backend, certifique-se de que `deletionPolicy` Está configurado para excluir. Para excluir apenas o `TridentBackendConfig` , assegure-se de que `deletionPolicy` está definido para manter. Isso garante que o backend ainda esteja presente e possa ser gerenciado usando `tridentctl` .

Execute o seguinte comando:

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

O Trident não exclui os segredos do Kubernetes que estavam em uso pelo `TridentBackendConfig` . O usuário do Kubernetes é responsável por limpar os segredos. É preciso ter cuidado ao apagar segredos. Você deve excluir segredos somente se eles não estiverem sendo usados pelos servidores de backend.

Veja os backends existentes

Execute o seguinte comando:

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

Você também pode executar `tridentctl get backend -n trident` ou `tridentctl get backend -o yaml -n trident` Para obter uma lista de todos os backends existentes. Esta lista também incluirá backends que foram criados com `tridentctl` .

Atualizar um backend

Existem diversos motivos para atualizar um backend:

- As credenciais de acesso ao sistema de armazenamento foram alteradas. Para atualizar as credenciais, o segredo do Kubernetes que é usado no `TridentBackendConfig` O objeto precisa ser atualizado. O Trident atualizará automaticamente o backend com as credenciais mais recentes fornecidas. Execute o seguinte comando para atualizar o segredo do Kubernetes:

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

- É necessário atualizar os parâmetros (como o nome da SVM do ONTAP que está sendo usada).
  - Você pode atualizar `TridentBackendConfig` objetos diretamente através do Kubernetes usando o seguinte comando:

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

- Alternativamente, você pode fazer alterações no existente. `TridentBackendConfig` CR usando o seguinte comando:

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

- Se uma atualização do backend falhar, o backend permanecerá em sua última configuração conhecida. Você pode visualizar os registros para determinar a causa executando o seguinte comando: `kubectl get tbc <tbc-name> -o yaml -n trident` ou `kubectl describe tbc <tbc-name> -n trident`.
- Após identificar e corrigir o problema com o arquivo de configuração, você pode executar o comando de atualização novamente.

Realize a gestão de backend com o tridentctl

Aprenda como executar operações de gerenciamento de back-end usando `tridentctl`.

Crie um backend

Depois de criar um"arquivo de configuração do backend" Execute o seguinte comando:

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

Se a criação do backend falhar, algo estava errado com a configuração do backend. Você pode visualizar os registros para determinar a causa executando o seguinte comando:

```
tridentctl logs -n trident
```

Após identificar e corrigir o problema com o arquivo de configuração, você pode simplesmente executar o `create` Comande novamente.

Excluir um backend

Para excluir um backend do Trident, faça o seguinte:

1. Recuperar o nome do backend:

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

2. Exclua o backend:

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

Se o Trident tiver provisionado volumes e snapshots desse backend que ainda existam, a exclusão do backend impede que novos volumes sejam provisionados por ele. O sistema de backend continuará existindo em um estado de "Exclusão".

Veja os backends existentes

Para visualizar os backends que o Trident conhece, faça o seguinte:

- Para obter um resumo, execute o seguinte comando:

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

- Para obter todos os detalhes, execute o seguinte comando:

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

Atualizar um backend

Após criar um novo arquivo de configuração de backend, execute o seguinte comando:

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

Se a atualização do servidor falhar, algo estava errado com a configuração do servidor ou você tentou uma atualização inválida. Você pode visualizar os registros para determinar a causa executando o seguinte comando:

```
tridentctl logs -n trident
```

Após identificar e corrigir o problema com o arquivo de configuração, você pode simplesmente executar o `update` Comande novamente.

Identifique as classes de armazenamento que utilizam um backend.

Este é um exemplo do tipo de pergunta que você pode responder com o JSON que `tridentctl` Saídas para objetos de backend. Isso usa o `jq` utilitário, que você precisa instalar.

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

Isso também se aplica a backends que foram criados usando `TridentBackendConfig` .

Alternar entre opções de gerenciamento de back-end

Aprenda sobre as diferentes maneiras de gerenciar back-ends no Trident.

Opções para gerenciar back-ends

Com a introdução de `TridentBackendConfig` , os administradores agora têm duas maneiras exclusivas de gerenciar os sistemas de back-end. Isso levanta as seguintes questões:

- É possível criar back-ends usando `tridentctl` ser gerenciado com `TridentBackendConfig` ?
- É possível criar back-ends usando `TridentBackendConfig` ser gerenciado usando `tridentctl` ?

Gerenciar `tridentctl` backends usando `TridentBackendConfig`

Esta seção aborda os passos necessários para gerenciar backends criados usando `tridentctl` diretamente através da interface do Kubernetes, criando `TridentBackendConfig` objetos.

Isso se aplica aos seguintes cenários:

- Backends pré-existentes que não possuem um `TridentBackendConfig` porque foram criados com `tridentctl` .
- Novos backends que foram criados com `tridentctl` , enquanto outros `TridentBackendConfig` Os objetos existem.

Em ambos os cenários, os servidores de backend continuarão presentes, com o Trident agendando volumes e operando sobre eles. Os administradores têm duas opções aqui:

- Continuar usando `tridentctl` para gerenciar os backends que foram criados usando-o.
- Vincule os backends criados usando `tridentctl` para um novo `TridentBackendConfig` objeto. Fazer isso significaria que os back-ends seriam gerenciados usando `kubectl` e não `tridentctl` .

Para gerenciar um backend pré-existente usando `kubectl` , você precisará criar um `TridentBackendConfig` que se integra ao backend existente. Segue abaixo uma visão geral de como isso funciona:

1. Criar um segredo do Kubernetes. O segredo contém as credenciais que o Trident precisa para se comunicar com o cluster/serviço de armazenamento.
2. Criar um `TridentBackendConfig` objeto. Esta seção contém detalhes específicos sobre o cluster/serviço de armazenamento e faz referência ao segredo criado na etapa anterior. É preciso ter cuidado para especificar parâmetros de configuração idênticos (como, por exemplo, `spec.backendName` , `spec.storagePrefix` , `spec.storageDriverName` , e assim por diante). `spec.backendName` Deve ser definido com o nome do backend existente.

Etapa 0: Identificar o backend

Para criar um `TridentBackendConfig` Para se conectar a um backend existente, você precisará obter a configuração do backend. Neste exemplo, vamos supor que um backend foi criado usando a seguinte definição 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"
      }
    }
  ]
}
```

Passo 1: Crie um segredo do Kubernetes

Crie um segredo que contenha as credenciais para o backend, conforme mostrado neste exemplo:

```
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
```

Passo 2: Crie um `TridentBackendConfig` CR

O próximo passo é criar um `TridentBackendConfig` CR que se vinculará automaticamente ao preexistente `ontap-nas-backend` (como neste exemplo). Assegure-se de que os seguintes requisitos sejam atendidos:

- O mesmo nome de backend está definido em `spec.backendName` .
- Os parâmetros de configuração são idênticos aos do backend original.
- Os pools virtuais (se presentes) devem manter a mesma ordem que no backend original.
- As credenciais são fornecidas por meio de um segredo do Kubernetes e não em texto simples.

Neste caso, o `TridentBackendConfig` Vai ficar assim:

```
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
```

Etapa 3: Verifique o status do `TridentBackendConfig` CR

Depois do `TridentBackendConfig` foi criada, sua fase deve ser `Bound` . Deve também refletir o mesmo nome de backend e UUID do backend 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 |
+---------------------+----------------
+--------------------------------------+--------+---------+
```

O backend agora será totalmente gerenciado usando o `tbc-ontap-nas-backend TridentBackendConfig` objeto.

Gerenciar `TridentBackendConfig` backends usando `tridentctl`

```
`tridentctl`pode ser usado para listar backends que foram criados usando
`TridentBackendConfig` .  Além disso, os administradores também podem
optar por gerenciar completamente esses back-ends por meio de `tridentctl`
ao excluir `TridentBackendConfig` e garantindo `spec.deletionPolicy` está
definido para `retain` .
```

Etapa 0: Identificar o backend

Por exemplo, vamos supor que o seguinte backend foi criado 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 dos resultados, observa-se que `TridentBackendConfig` Foi criado com sucesso e está vinculado a um backend [observe o UUID do backend].

Passo 1: Confirmar `deletionPolicy` está definido para `retain`

Vamos analisar o valor de `deletionPolicy` . Isso precisa ser configurado para `retain` . Isso garante que quando um `TridentBackendConfig` O CR é excluído, mas a definição do backend ainda estará presente e poderá ser gerenciada com `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
```

Não prossiga para a próxima etapa a menos que `deletionPolicy` está definido para `retain` .

Passo 2: Exclua o `TridentBackendConfig` CR

O passo final é excluir o `TridentBackendConfig` CR. Após confirmar o `deletionPolicy` está definido para `retain` Você pode prosseguir com a exclusão:

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

Após a exclusão do `TridentBackendConfig` O objeto, o Trident simplesmente o remove sem realmente excluir o próprio backend.

Informações sobre direitos autorais

Informações sobre marcas comerciais