



管理后端 Trident

NetApp
October 29, 2026

目录

- 管理后端 1
 - 使用 kubectl 管理 Trident 后端 1
 - 删除后端 1
 - 查看现有后端 1
 - 更新后端 1
 - 使用 tridentctl 管理 Trident 后端 2
 - 创建后端 2
 - 删除后端 2
 - 查看现有后端 3
 - 更新后端 3
 - 标识使用后端的存储类 3
- 选择 Trident 后端管理方法 3
 - 管理后端的选项 3
 - 使用 tridentctl`管理后端`TridentBackendConfig 4
 - 使用 TridentBackendConfig`管理后端`tridentctl 8

管理后端

使用 **kubectl** 管理 **Trident** 后端

了解如何使用 `kubectl` 执行后端管理操作。

删除后端

通过删除 `TridentBackendConfig`，您指示 Trident 删除/保留后端（基于 `deletionPolicy`）。要删除后端，请确保将 `deletionPolicy` 设置为 `delete`。要仅删除 `TridentBackendConfig`，请确保将 `deletionPolicy` 设置为 `retain`。这可以确保后端仍然存在，并且可以通过使用 `tridentctl` 进行管理。

运行以下命令：

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

Trident 不会删除正在使用的 Kubernetes Secrets `TridentBackendConfig`。Kubernetes 用户负责清理机密。删除机密时必须谨慎行事。仅当机密未被后端使用时，才应将其删除。

查看现有后端

运行以下命令：

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

您还可以运行 `tridentctl get backend -n trident` 或 `tridentctl get backend -o yaml -n trident` 以获取所有现有后端的列表。此列表还将包括使用 `tridentctl` 创建的后端。

更新后端

更新后端可能有多种原因：

- 存储系统的凭据已更改。要更新凭据，必须更新 `TridentBackendConfig` 对象中使用的 Kubernetes Secret。Trident 将使用提供的最新凭据自动更新后端。运行以下命令以更新 Kubernetes Secret：

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

- 需要更新参数（例如正在使用的 ONTAP SVM 的名称）。
 - 您可以使用以下命令直接通过 Kubernetes 更新 `TridentBackendConfig` 对象：

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

- 或者，您可以使用以下命令更改现有 `TridentBackendConfig` CR：

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



- 如果后端更新失败，则后端继续保持其最后已知的配置。您可以通过运行 ``kubectl get tbc <tbc-name> -o yaml -n trident`` 或 ``kubectl describe tbc <tbc-name> -n trident`` 来查看日志以确定原因。
- 确定并更正配置文件的问题后，您可以重新运行更新命令。

使用 `tridentctl` 管理 Trident 后端

了解如何使用 ``tridentctl`` 执行后端管理操作。

创建后端

创建 "[后端配置文件](#)" 后，运行以下命令：

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

如果后端创建失败，则表示后端配置有问题。您可以通过运行以下命令查看日志以确定原因：

```
tridentctl logs -n trident
```

在识别并更正配置文件的问题后，只需再次运行此 `create` 命令即可。

删除后端

要从 Trident 删除后端，请执行以下操作：

1. 检索后端名称：

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

2. 删除后端：

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



如果 Trident 已从此后端配置仍然存在的卷和快照，则删除后端会阻止其配置新卷。后端将继续处于 "Deleting" 状态。

查看现有后端

要查看 Trident 知道的后端，请执行以下操作：

- 要获取摘要，请运行以下命令：

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

- 要获取所有详细信息，请运行以下命令：

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

更新后端

创建新的后端配置文件后，运行以下命令：

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

如果后端更新失败，则说明后端配置有问题或您尝试的更新无效。您可以通过运行以下命令查看日志以确定原因：

```
tridentctl logs -n trident
```

在识别并更正配置文件的问题后，只需再次运行此 `update` 命令即可。

标识使用后端的存储类

这是一个示例，说明您可以使用 `tridentctl` 为后端对象输出的 JSON 来回答这类问题。这使用了 `jq` 实用程序，您需要安装它。

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

这也适用于使用 `TridentBackendConfig` 创建的后端。

选择 Trident 后端管理方法

了解在 Trident 中管理后端的不同方法。

管理后端的选项

随着 `TridentBackendConfig` 的推出，管理员现在有两种独特的后端管理方式。这就提出了以下问题：

- 使用 `tridentctl` 创建的后端可以通过 `TridentBackendConfig` 进行管理吗？
- 使用 `TridentBackendConfig` 创建的后端可以通过 `tridentctl` 进行管理吗？

使用 `tridentctl` 管理后端 `TridentBackendConfig`

本节介绍了管理通过 `tridentctl` 直接通过 Kubernetes 界面创建 `TridentBackendConfig` 对象而创建的后端所需的步骤。

这适用于以下情形：

- 已存在的后端，没有 `TridentBackendConfig`，因为它们是使用 `tridentctl` 创建的。
- 已使用 `tridentctl` 创建新后端，但存在其他 `TridentBackendConfig` 对象。

在这两种情况下，后端都将继续存在，Trident 调度卷并对其进行操作。管理员在此处有两种选择：

- 继续使用 `tridentctl` 来管理使用它创建的后端。
- 将使用 `tridentctl` 创建的后端绑定到新的 `TridentBackendConfig` 对象。这样做意味着后端将使用 `kubectl` 而不是 `tridentctl` 进行管理。

要使用 `kubectl` 管理预先存在的后端，您需要创建一个 `TridentBackendConfig` 绑定到现有后端。以下是其工作原理概述：

1. 创建 Kubernetes Secret。密钥包含 Trident 与存储集群/服务通信所需的凭据。
2. 创建 `TridentBackendConfig` 对象。其中包含有关存储群集/服务的详细信息，并引用了上一步中创建的密码。必须注意指定相同的配置参数（如 `spec.backendName`、`spec.storagePrefix`、`spec.storageDriverName` 等）。`spec.backendName` 必须设置为现有后端的名称。

第 0 步：识别后端

要创建绑定到现有后端的 `TridentBackendConfig`，您需要获取后端配置。在此示例中，假设后端是使用以下 JSON 定义创建的：

```
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"
      }
    }
  ]
}

```

步骤 1: 创建 **Kubernetes Secret**

创建一个包含后端凭据的 Secret，如以下示例所示：

```
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
```

步骤 2: 创建 **TridentBackendConfig CR**

下一步是创建一个 `TridentBackendConfig` CR，该 CR 将自动绑定到预先存在的 `ontap-nas-backend`（如本示例所示）。确保满足以下要求：

- 在 `spec.backendName` 中定义了相同的后端名称。
- 配置参数与原始后端相同。
- 虚拟池（如果存在）必须保持与原始后端相同的顺序。
- 凭据通过 `Kubernetes Secret` 提供，而不是以纯文本形式提供。

在此情况下，`TridentBackendConfig` 将如下所示：

```
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

```

第 3 步：验证 **TridentBackendConfig CR** 的状态

创建 **TridentBackendConfig** 后，其阶段必须为 **Bound**。它还应反映与现有后端相同的后端名称和 UUID。

```

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 |
+-----+-----+
+-----+-----+-----+-----+

```

现在，将使用 `tbc-ontap-nas-backend` `TridentBackendConfig` 对象来完全管理后端。

使用 `TridentBackendConfig` 管理后端 `tridentctl`

`tridentctl` 可用于列出使用 `TridentBackendConfig` 创建的后端。此外，管理员还可以选择通过 `tridentctl` 完全管理此类后端，方法是删除 `TridentBackendConfig` 并确保将 `spec.deletionPolicy` 设置为 `retain`。

第 0 步：识别后端

例如，让我们假设以下后端是使用 `TridentBackendConfig` 创建的：

```

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 |
+-----+-----+
+-----+-----+-----+-----+

```

从输出中可以看出，`TridentBackendConfig` 已成功创建并绑定到后端 [观察后端的 UUID]。

步骤 1: 确认 `deletionPolicy` 设置为 `retain`

让我们来看看 `deletionPolicy` 的价值。这需要设置为 `retain`。这可以确保在删除 `TridentBackendConfig` CR 时, 后端定义仍然存在, 并且可以使用 `tridentctl` 进行管理。

```
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
```



除非 `deletionPolicy` 设置为 `retain`, 否则请勿继续下一步。

步骤 2: 删除 `TridentBackendConfig` CR

最后一步是删除 `TridentBackendConfig` CR。确认 `deletionPolicy` 设置为 `retain` 后, 您可以继续删除:

```
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 |
```

删除 `TridentBackendConfig` 对象后，Trident 只需将其删除，而不会实际删除后端本身。

版权信息

版权所有 © 2026 NetApp, Inc.。保留所有权利。中国印刷。未经版权所有者事先书面许可，本文档中受版权保护的任何部分不得以任何形式或通过任何手段（图片、电子或机械方式，包括影印、录音、录像或存储在电子检索系统中）进行复制。

从受版权保护的 NetApp 资料派生的软件受以下许可和免责声明的约束：

本软件由 NetApp 按“原样”提供，不含任何明示或暗示担保，包括但不限于适销性以及针对特定用途的适用性的隐含担保，特此声明不承担任何责任。在任何情况下，对于因使用本软件而以任何方式造成的任何直接性、间接性、偶然性、特殊性、惩罚性或后果性损失（包括但不限于购买替代商品或服务；使用、数据或利润方面的损失；或者业务中断），无论原因如何以及基于何种责任理论，无论出于合同、严格责任或侵权行为（包括疏忽或其他行为），NetApp 均不承担责任，即使已被告知存在上述损失的可能性。

NetApp 保留在不另行通知的情况下随时对本文档所述的任何产品进行更改的权利。除非 NetApp 以书面形式明确同意，否则 NetApp 不承担因使用本文档所述产品而产生的任何责任或义务。使用或购买本产品不表示获得 NetApp 的任何专利权、商标权或任何其他知识产权许可。

本手册中描述的产品可能受一项或多项美国专利、外国专利或正在申请的专利的保护。

有限权利说明：政府使用、复制或公开本文档受 DFARS 252.227-7013（2014 年 2 月）和 FAR 52.227-19（2007 年 12 月）中“技术数据权利 — 非商用”条款第 (b)(3) 条规定的限制条件的约束。

本文档中所含数据与商业产品和/或商业服务（定义见 FAR 2.101）相关，属于 NetApp, Inc. 的专有信息。根据本协议提供的所有 NetApp 技术数据和计算机软件具有商业性质，并完全由私人出资开发。美国政府对这些数据的使用权具有非排他性、全球性、受限且不可撤销的许可，该许可既不可转让，也不可再许可，但仅限在与交付数据所依据的美国政府合同有关且受合同支持的情况下使用。除本文档规定的情形外，未经 NetApp, Inc. 事先书面批准，不得使用、披露、复制、修改、操作或显示这些数据。美国政府对国防部的授权仅限于 DFARS 的第 252.227-7015(b)（2014 年 2 月）条款中明确的权利。

商标信息

NetApp、NetApp 标识和 <http://www.netapp.com/TM> 上所列的商标是 NetApp, Inc. 的商标。其他公司和产品名称可能是其各自所有者的商标。