管理后端

Astra Trident

NetApp
January 14, 2026

This PDF was generated from https://docs.netapp.com/zh-cn/trident-2307/trident-use/backend_ops_kubectl.html on January 14, 2026. Always check docs.netapp.com for the latest.

目录

管理后端 . . . 1
使用 kubectl 执行后端管理 . . . 1
删除后端 . . . 1
查看现有后端 . . . 1
更新后端 . . . 1
使用 tridentctl 执行后端管理 . . . 2
创建后端 . . . 2
删除后端 . . . 2
查看现有后端 . . . 3
更新后端 . . . 3
确定使用后端的存储类 . . . 3
在后端管理选项之间移动 . . . 3
用于管理后端的选项 . . . 4
管理 `tridentctl` 后端使用 `TridentBackendConfig` . . . 4
管理 `TridentBackendConfig` 后端使用 `tridentctl` . . . 8

管理后端

使用 kubectl 执行后端管理

了解如何使用执行后端管理操作 `kubectl`。

删除后端

删除 `TridentBackendConfig`、您可以指示Astra Trident删除/保留后端(基于 `deletionPolicy`）。要删除后端、请确保 `deletionPolicy` 设置为delete。仅删除 `TridentBackendConfig`、请确保 `deletionPolicy` 设置为保留。这样可以确保后端仍然存在、并可使用进行管理 `tridentctl`。

运行以下命令：

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

Astra Trident不会删除正在使用的Kubernetes机密 `TridentBackendConfig`。Kubernetes 用户负责清理密钥。删除机密时必须小心。只有在后端未使用机密时，才应将其删除。

查看现有后端

运行以下命令：

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

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

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、请使用中使用的Kubernetes Secret `TridentBackendConfig` 必须更新对象。Astra 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（例如所使用的 ONTAP SVM 的名称）。
  - 您可以更新 `TridentBackendConfig` 使用以下命令直接通过KubeNet访问对象：

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

- 或者、您也可以对现有进行更改 `TridentBackendConfig` 使用以下命令执行CR：

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

- 如果后端更新失败，则后端仍会保持在其上次已知配置中。您可以通过运行来查看日志以确定发生原因 `kubectl get tbc <tbc-name> -o yaml -n trident` 或 `kubectl describe tbc <tbc-name> -n trident`。
- 确定并更正配置文件中的问题后，您可以重新运行 update 命令。

使用 tridentctl 执行后端管理

了解如何使用执行后端管理操作 `tridentctl`。

创建后端

创建后 "后端配置文件"下，运行以下命令：

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

如果后端创建失败，则后端配置出现问题。您可以运行以下命令来查看日志以确定发生原因：

```
tridentctl logs -n trident
```

确定并更正配置文件中的问题后、您只需运行即可 `create` 命令。

删除后端

要从 Astra Trident 中删除后端，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 Astra Trident 从此后端配置了仍存在的卷和快照，则删除后端将阻止其配置新卷。后端将继续处于 " 删除 " 状态，而 Trident 将继续管理这些卷和快照，直到将其删除为止。

查看现有后端

要查看 Trident 了解的后端，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，则后端配置出现问题或您尝试的更新无效。您可以运行以下命令来查看日志以确定发生原因：

```
tridentctl logs -n trident
```

确定并更正配置文件中的问题后、您只需运行即可 `update` 命令。

确定使用后端的存储类

这是一个示例、说明您可以通过问题解答 与JSON一起提出的问题 `tridentctl` 后端对象的输出。此操作将使用 `jq` 实用程序、您需要安装该实用程序。

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

这也适用于使用创建的后端 `TridentBackendConfig`。

在后端管理选项之间移动

了解如何在 Astra Trident 中管理后端。

用于管理后端的选项

随附 `TridentBackendConfig`现在、管理员可以通过两种独特的方式管理后端。这会提出以下问题：

- 可以使用创建后端 `tridentctl` 通过进行管理 `TridentBackendConfig`?
- 可以使用创建后端 `TridentBackendConfig` 可使用进行管理 `tridentctl`?

管理 `tridentctl` 后端使用 `TridentBackendConfig`

本节介绍管理使用创建的后端所需的步骤 `tridentctl` 通过创建直接通过Kubernetes界面 `TridentBackendConfig` 对象。

这适用于以下情形：

- 已有后端、但没有 `TridentBackendConfig` 因为它们是使用创建的 `tridentctl`。
- 使用创建的新后端 `tridentctl`而其他 `TridentBackendConfig` 对象存在。

在这两种情况下，后端仍会存在，其中 Astra Trident 会计划卷并对其进行操作。管理员可以选择以下两种方式之一：

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

使用管理已有后端 `kubectl`、您需要创建 `TridentBackendConfig` 绑定到现有后端。下面简要介绍了它的工作原理：

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

第 0 步：确定后端

以创建 `TridentBackendConfig` 如果绑定到现有后端、则需要获取后端配置。在此示例中，假设已使用以下 JSON 定义创建了后端：

```
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"
            }
        }
    ]
}
```

第 1 步：创建 Kubernetes 机密

创建一个包含后端凭据的机密，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
```

第2步：创建 `TridentBackendConfig` CR

下一步是创建 `TridentBackendConfig` 将自动绑定到已有的的CR `ontap-nas-backend` (如本示例所示)。确保满足以下要求：

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

在这种情况下、将显示 `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：验证的状态 `TridentBackendConfig` CR

在之后 `TridentBackendConfig` 已创建、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、后端将使用进行完全管理 `tbc-ontap-nas-backend` `TridentBackendConfig` 对象。

管理 `TridentBackendConfig` 后端使用 `tridentctl`

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

第 0 步：确定后端

例如、假设以下后端是使用创建的 `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 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

从输出中可以看出这一点 `TridentBackendConfig` 已成功创建并绑定到后端[观察后端的UUID]。

第1步：确认 `deletionPolicy` 设置为 `retain`

让我们来了解一下的价值 `deletionPolicy`。此值需要设置为 `retain`。这样可以确保在出现时 `TridentBackendConfig` CR已删除、后端定义仍存在、可使用进行管理 `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
```

请勿继续执行下一步、除非 `deletionPolicy` 设置为 `retain`。

第2步：删除 `TridentBackendConfig` CR

最后一步是删除 `TridentBackendConfig` CR.确认后 `deletionPolicy` 设置为 `retain`、您可以继续执行删除操作：

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

删除时 `TridentBackendConfig` 对象、Astra Trident只需将其删除、而不实际删除后端本身。

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。