



스토리지 클래스 생성 및 관리

Trident

NetApp
March 05, 2026

목차

스토리지 클래스 생성 및 관리	1
스토리지 클래스 생성	1
Kubernetes StorageClass 객체 구성	1
스토리지 클래스 생성	1
스토리지 클래스 관리	3
기존 스토리지 클래스 보기	3
기본 저장소 클래스 설정	4
스토리지 클래스의 백엔드 식별	4
스토리지 클래스 삭제	4

스토리지 클래스 생성 및 관리

스토리지 클래스 생성

Kubernetes StorageClass 객체를 구성하고 Trident 가 볼륨을 프로비저닝하는 방법을 지시하는 스토리지 클래스를 만듭니다.

Kubernetes StorageClass 객체 구성

그만큼 "Kubernetes StorageClass 객체" Trident 해당 클래스에 사용되는 프로비저너로 식별하고 Trident 볼륨을 프로비저닝하는 방법을 지시합니다. 예를 들어:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
mountOptions:
  - nfsvers=3
  - nolock
parameters:
  backendType: "ontap-nas"
  media: "ssd"
allowVolumeExpansion: true
volumeBindingMode: Immediate
```

참조하다 "Kubernetes 및 Trident 객체" 저장소 클래스가 어떻게 상호 작용하는지에 대한 자세한 내용은 다음과 같습니다. PersistentVolumeClaim Trident 가 볼륨을 프로비저닝하는 방식을 제어하기 위한 매개변수입니다.

스토리지 클래스 생성

StorageClass 객체를 만든 후에는 스토리지 클래스를 만들 수 있습니다. [저장 클래스 샘플](#) 사용하거나 수정할 수 있는 몇 가지 기본 샘플을 제공합니다.

단계

1. 이것은 Kubernetes 객체이므로 다음을 사용합니다. `kubectl` Kubernetes에서 생성하세요.

```
kubectl create -f sample-input/storage-class-basic-csi.yaml
```

2. 이제 Kubernetes와 Trident 모두에서 **basic-csi** 스토리지 클래스를 볼 수 있어야 하며, Trident 가 백엔드에서 풀을 검색했어야 합니다.

```
kubectl get sc basic-csi
```

NAME	PROVISIONER	AGE
basic-csi	csi.trident.netapp.io	15h

```
./tridentctl -n trident get storageclass basic-csi -o json
```

```
{
  "items": [
    {
      "Config": {
        "version": "1",
        "name": "basic-csi",
        "attributes": {
          "backendType": "ontap-nas"
        },
        "storagePools": null,
        "additionalStoragePools": null
      },
      "storage": {
        "ontapnas_10.0.0.1": [
          "aggr1",
          "aggr2",
          "aggr3",
          "aggr4"
        ]
      }
    }
  ]
}
```

저장 클래스 샘플

Trident 제공합니다 "특정 백엔드에 대한 간단한 스토리지 클래스 정의".

또는 편집할 수 있습니다 `sample-input/storage-class-csi.yaml.template` 설치 프로그램과 함께 제공되는 파일을 교체합니다. `BACKEND_TYPE` 저장소 드라이버 이름으로.

```

./tridentctl -n trident get backend
+-----+-----+-----+-----+
+-----+-----+
|   NAME   | STORAGE DRIVER |           UUID           |
STATE | VOLUMES |
+-----+-----+-----+-----+
+-----+-----+
| nas-backend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |         0 |
+-----+-----+-----+-----+
+-----+-----+

cp sample-input/storage-class-csi.yaml.template sample-input/storage-class-
basic-csi.yaml

# Modify __BACKEND_TYPE__ with the storage driver field above (e.g.,
ontap-nas)
vi sample-input/storage-class-basic-csi.yaml

```

스토리지 클래스 관리

기존 스토리지 클래스를 보고, 기본 스토리지 클래스를 설정하고, 스토리지 클래스 백엔드를 식별하고, 스토리지 클래스를 삭제할 수 있습니다.

기존 스토리지 클래스 보기

- 기존 Kubernetes 스토리지 클래스를 보려면 다음 명령을 실행하세요.

```
kubectl get storageclass
```

- Kubernetes 스토리지 클래스 세부 정보를 보려면 다음 명령을 실행하세요.

```
kubectl get storageclass <storage-class> -o json
```

- Trident의 동기화된 스토리지 클래스를 보려면 다음 명령을 실행하세요.

```
tridentctl get storageclass
```

- Trident의 동기화된 스토리지 클래스 세부 정보를 보려면 다음 명령을 실행하세요.

```
tridentctl get storageclass <storage-class> -o json
```

기본 저장소 클래스 설정

Kubernetes 1.6에서는 기본 스토리지 클래스를 설정하는 기능이 추가되었습니다. 이는 사용자가 PVC(영구 볼륨 클레임)에서 영구 볼륨을 지정하지 않은 경우 영구 볼륨을 프로비저닝하는 데 사용되는 스토리지 클래스입니다.

- 주석을 설정하여 기본 저장 클래스를 정의합니다. `storageclass.kubernetes.io/is-default-class` 저장 클래스 정의에서 `true`로 설정합니다. 사양에 따르면, 다른 값이나 주석이 없는 경우 거짓으로 해석됩니다.
- 다음 명령을 사용하여 기존 스토리지 클래스를 기본 스토리지 클래스로 구성할 수 있습니다.

```
kubectl patch storageclass <storage-class-name> -p '{"metadata":{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

- 마찬가지로 다음 명령을 사용하여 기본 저장소 클래스 주석을 제거할 수 있습니다.

```
kubectl patch storageclass <storage-class-name> -p '{"metadata":{"annotations":{"storageclass.kubernetes.io/is-default-class":"false"}}}'
```

Trident 설치 프로그램 번들에는 이 주석이 포함된 예도 있습니다.



클러스터에는 한 번에 하나의 기본 스토리지 클래스만 있어야 합니다. Kubernetes는 기술적으로 두 개 이상을 갖는 것을 막지는 않지만, 기본 스토리지 클래스가 전혀 없는 것처럼 동작합니다.

스토리지 클래스의 백엔드 식별

이는 JSON으로 답할 수 있는 질문 종류의 예입니다. `tridentctl` Trident 백엔드 객체에 대한 출력. 이것은 다음을 사용합니다 `jq` 먼저 설치해야 할 유틸리티가 있습니다.

```
tridentctl get storageclass -o json | jq '[.items[] | {storageClass: .Config.name, backends: [.storage]|unique}]'
```

스토리지 클래스 삭제

Kubernetes에서 스토리지 클래스를 삭제하려면 다음 명령을 실행하세요.

```
kubectl delete storageclass <storage-class>
```

`<storage-class>` 저장 클래스로 대체해야 합니다.

이 스토리지 클래스를 통해 생성된 모든 영구 볼륨은 변경되지 않으며 Trident 계속해서 이를 관리합니다.



Trident 공백을 강화합니다. fsType 그것이 만들어내는 볼륨 때문이에요. iSCSI 백엔드의 경우 다음을 적용하는 것이 좋습니다. `parameters.fsType` StorageClass에서. 기존 StorageClass를 삭제하고 다시 생성해야 합니다. `parameters.fsType` 지정된.

저작권 정보

Copyright © 2026 NetApp, Inc. All Rights Reserved. 미국에서 인쇄된 본 문서의 어떠한 부분도 저작권 소유자의 사전 서면 승인 없이는 어떠한 형식이나 수단(복사, 녹음, 녹화 또는 전자 검색 시스템에 저장하는 것을 비롯한 그래픽, 전자적 또는 기계적 방법)으로도 복제될 수 없습니다.

NetApp이 저작권을 가진 자료에 있는 소프트웨어에는 아래의 라이선스와 고지사항이 적용됩니다.

본 소프트웨어는 NetApp에 의해 '있는 그대로' 제공되며 상품성 및 특정 목적에의 적합성에 대한 명시적 또는 묵시적 보증을 포함하여(이에 제한되지 않음) 어떠한 보증도 하지 않습니다. NetApp은 대체품 또는 대체 서비스의 조달, 사용 불능, 데이터 손실, 이익 손실, 영업 중단을 포함하여(이에 국한되지 않음), 이 소프트웨어의 사용으로 인해 발생하는 모든 직접 및 간접 손해, 우발적 손해, 특별 손해, 징벌적 손해, 결과적 손해의 발생에 대하여 그 발생 이유, 책임론, 계약 여부, 엄격한 책임, 불법 행위(과실 또는 그렇지 않은 경우)와 관계없이 어떠한 책임도 지지 않으며, 이와 같은 손실의 발생 가능성이 통지되었다 하더라도 마찬가지입니다.

NetApp은 본 문서에 설명된 제품을 언제든지 예고 없이 변경할 권리를 보유합니다. NetApp은 NetApp의 명시적인 서면 동의를 받은 경우를 제외하고 본 문서에 설명된 제품을 사용하여 발생하는 어떠한 문제에도 책임을 지지 않습니다. 본 제품의 사용 또는 구매의 경우 NetApp에서는 어떠한 특허권, 상표권 또는 기타 지적 재산권이 적용되는 라이선스도 제공하지 않습니다.

본 설명서에 설명된 제품은 하나 이상의 미국 특허, 해외 특허 또는 출원 중인 특허로 보호됩니다.

제한적 권리 표시: 정부에 의한 사용, 복제 또는 공개에는 DFARS 252.227-7013(2014년 2월) 및 FAR 52.227-19(2007년 12월)의 기술 데이터-비상업적 품목에 대한 권리(Rights in Technical Data -Noncommercial Items) 조항의 하위 조항 (b)(3)에 설명된 제한사항이 적용됩니다.

여기에 포함된 데이터는 상업용 제품 및/또는 상업용 서비스(FAR 2.101에 정의)에 해당하며 NetApp, Inc.의 독점 자산입니다. 본 계약에 따라 제공되는 모든 NetApp 기술 데이터 및 컴퓨터 소프트웨어는 본질적으로 상업용이며 개인 비용만으로 개발되었습니다. 미국 정부는 데이터가 제공된 미국 계약과 관련하여 해당 계약을 지원하는 데에만 데이터에 대한 전 세계적으로 비독점적이고 양도할 수 없으며 재사용이 불가능하며 취소 불가능한 라이선스를 제한적으로 가집니다. 여기에 제공된 경우를 제외하고 NetApp, Inc.의 사전 서면 승인 없이는 이 데이터를 사용, 공개, 재생산, 수정, 수행 또는 표시할 수 없습니다. 미국 국방부에 대한 정부 라이선스는 DFARS 조항 252.227-7015(b)(2014년 2월)에 명시된 권한으로 제한됩니다.

상표 정보

NETAPP, NETAPP 로고 및 <http://www.netapp.com/TM>에 나열된 마크는 NetApp, Inc.의 상표입니다. 기타 회사 및 제품 이름은 해당 소유자의 상표일 수 있습니다.