보안
Trident

NetApp
October 29, 2026

This PDF was generated from https://docs.netapp.com/ko-kr/trident-2606/trident-reco/security-reco.html on October 29, 2026. Always check docs.netapp.com for the latest.

목차

보안 . . . 1
Trident 보안 권장 사항 . . . 1
Trident를 자체 네임스페이스에서 실행합니다 . . . 1
ONTAP SAN 백엔드에서 CHAP 인증을 사용하십시오 . . . 1
NetApp HCI 및 SolidFire 백엔드에서 CHAP 인증을 사용하십시오 . . . 1
NVE 및 NAE와 함께 Trident 사용 . . . 1
Trident Linux Unified Key Setup(LUKS) . . . 2
LUKS 암호화 활성화 . . . 2
LUKS 볼륨 가져오기를 위한 백엔드 구성 . . . 4
LUKS 볼륨 가져오기를 위한 PVC 구성 . . . 4
LUKS 암호 교체 . . . 5
볼륨 확장 활성화 . . . 6
Trident Kerberos 전송 중 암호화 . . . 7
온프레미스 ONTAP 볼륨을 사용하여 전송 중 Kerberos 암호화 구성 . . . 7
Azure NetApp Files 볼륨에서 전송 중 Kerberos 암호화 구성 . . . 11

보안

Trident 보안 권장 사항

여기에 나열된 권장 사항을 사용하여 Trident 설치의 보안을 확보하십시오.

Trident를 자체 네임스페이스에서 실행합니다

안정적인 스토리지를 보장하고 잠재적인 악의적 활동을 차단하려면 애플리케이션, 애플리케이션 관리자, 사용자 및 관리 애플리케이션이 Trident 객체 정의 또는 Pod에 액세스하는 것을 방지하는 것이 중요합니다.

다른 애플리케이션 및 사용자와 Trident를 분리하려면 항상 Trident를 자체 Kubernetes 네임스페이스에 설치하십시오(`trident`. Trident를 자체 네임스페이스에 배치하면 Kubernetes 관리 담당자만 Trident Pod 및 네임스페이스 CRD 객체에 저장된 아티팩트(해당하는 경우 백엔드 및 CHAP 암호)에 액세스할 수 있습니다. 관리자만 Trident 네임스페이스에 액세스하여 `tridentctl` 애플리케이션에 액세스할 수 있도록 해야 합니다.

ONTAP SAN 백엔드에서 CHAP 인증을 사용하십시오

Trident는 ONTAP SAN 워크로드에 대해 CHAP 기반 인증을 지원합니다(`ontap-san` 및 `ontap-san-economy` 드라이버 사용). NetApp은 호스트와 스토리지 백엔드 간의 인증을 위해 Trident와 함께 양방향 CHAP를 사용하는 것을 권장합니다.

SAN 스토리지 드라이버를 사용하는 ONTAP 백엔드의 경우 Trident는 `tridentctl`를 통해 양방향 CHAP를 설정하고 CHAP 사용자 이름과 암호를 관리할 수 있습니다. Trident가 ONTAP 백엔드에서 CHAP를 구성하는 방법을 이해하려면"ONTAP SAN 드라이버를 사용하여 백엔드를 구성할 준비를 하십시오"를 참조하십시오.

NetApp HCI 및 SolidFire 백엔드에서 CHAP 인증을 사용하십시오

NetApp에서는 호스트와 NetApp HCI 및 SolidFire 백엔드 간의 인증을 보장하기 위해 양방향 CHAP 배포를 권장합니다. Trident는 테넌트당 두 개의 CHAP 암호를 포함하는 secret 객체를 사용합니다. Trident가 설치되면 CHAP secret을 관리하고 해당 PV에 대한 `tridentvolume` CR 객체에 저장합니다. PV를 생성하면 Trident는 CHAP secret을 사용하여 iSCSI 세션을 시작하고 CHAP를 통해 NetApp HCI 및 SolidFire 시스템과 통신합니다.

Trident에서 생성한 볼륨은 어떤 볼륨 액세스 그룹과도 연결되지 않습니다.

NVE 및 NAE와 함께 Trident 사용

NetApp ONTAP는 디스크가 도난당하거나 반환되거나 용도가 변경될 경우 중요한 데이터를 보호하기 위해 저장 데이터 암호화를 제공합니다. 자세한 내용은 "NetApp Volume Encryption 구성 개요"을(를) 참조하십시오.

- 백엔드에서 NAE가 활성화된 경우 Trident에서 프로비저닝된 모든 볼륨은 NAE가 활성화됩니다.
  - NVE encryption 플래그를 `""`로 설정하여 NAE 지원 볼륨을 생성할 수 있습니다.
- 백엔드에서 NAE가 활성화되지 않은 경우 백엔드 구성에서 NVE 암호화 플래그가 `false`(기본값)로 설정되지 않는 한 Trident에서 프로비저닝된 모든 볼륨은 NVE가 활성화됩니다.

NAE가 활성화된 백엔드에서 Trident로 생성된 볼륨은 NVE 또는 NAE로 암호화되어야 합니다.

- Trident 백엔드 구성에서 NVE 암호화 플래그를 `true`로 설정하여 NAE 암호화를 재정의하고 볼륨별로 특정 암호화 키를 사용할 수 있습니다.
- NAE가 활성화된 백엔드에서 NVE 암호화 플래그를 `false`로 설정하면 NAE가 활성화된 볼륨이 생성됩니다. NVE 암호화 플래그를 `false`로 설정하여 NAE 암호화를 비활성화할 수는 없습니다.

- Trident에서 NVE 암호화 플래그를 명시적으로 `true`로 설정하여 수동으로 NVE 볼륨을 생성할 수 있습니다.

백엔드 구성 옵션에 대한 자세한 내용은 다음을 참조하십시오.

- "ONTAP SAN 구성 옵션"
- "ONTAP NAS 구성 옵션"

Trident Linux Unified Key Setup(LUKS)

Trident에서 Linux Unified Key Setup(LUKS)을 활성화하여 ONTAP SAN 및 ONTAP SAN ECONOMY 볼륨을 암호화할 수 있습니다. Trident는 LUKS로 암호화된 볼륨에 대해 암호 순환 및 볼륨 확장을 지원합니다.

Trident에서 LUKS로 암호화된 볼륨은 "NIST"에서 권장하는 대로 aes-xts-plain64 암호 및 모드를 사용합니다.

LUKS 암호화는 ASA r2 시스템에서 지원되지 않습니다. ASA r2 시스템에 대한 자세한 내용은 "ASA r2 스토리지 시스템에 대해 알아보세요"를 참조하십시오.

시작하기 전에

- 워커 노드에는 cryptsetup 2.1 이상(단, 3.0 미만)이 설치되어 있어야 합니다. 자세한 내용은 "Gitlab: cryptsetup"을(를) 방문하십시오.
- 성능 향상을 위해 NetApp에서는 워커 노드가 AES-NI(Advanced Encryption Standard New Instructions)를 지원하도록 권장합니다. AES-NI 지원 여부를 확인하려면 다음 명령을 실행하십시오.

```
grep "aes" /proc/cpuinfo
```

아무것도 반환되지 않으면 프로세서가 AES-NI를 지원하지 않는 것입니다. AES-NI에 대한 자세한 내용은 "Intel: Advanced Encryption Standard Instructions(AES-NI)"를 참조하십시오.

LUKS 암호화 활성화

ONTAP SAN 및 ONTAP SAN ECONOMY 볼륨의 경우 Linux Unified Key Setup(LUKS)을 사용하여 볼륨별 호스트 측 암호화를 활성화할 수 있습니다.

단계

1. 백엔드 구성에서 LUKS 암호화 속성을 정의합니다. ONTAP SAN의 백엔드 구성 옵션에 대한 자세한 내용은 "ONTAP SAN 구성 옵션"을 참조하십시오.

```
{
  "storage": [
```

```
    {
      "labels": {
        "luks": "true"
      },
      "zone": "us_east_1a",
      "defaults": {
        "luksEncryption": "true"
      }
    },
    {
      "labels": {
        "luks": "false"
      },
      "zone": "us_east_1a",
      "defaults": {
        "luksEncryption": "false"
      }
    }
  ]
}
```

2. `parameters.selector`를 사용하여 LUKS 암호화를 사용하는 스토리지 풀을 정의합니다. 예를 들면 다음과 같습니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: csi.trident.netapp.io
parameters:
  selector: "luks=true"
  csi.storage.k8s.io/node-stage-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

3. LUKS 암호를 포함하는 비밀 키를 생성합니다. 예를 들면 다음과 같습니다.

```
kubectl -n trident create -f luks-pvc1.yaml
apiVersion: v1
kind: Secret
metadata:
  name: luks-pvc1
stringData:
  luks-passphrase-name: A
  luks-passphrase: secretA
```

제한 사항

LUKS로 암호화된 볼륨은 ONTAP 중복 제거 및 압축을 활용할 수 없습니다.

LUKS 볼륨 가져오기를 위한 백엔드 구성

LUKS 볼륨을 가져오려면 백엔드에서 `luksEncryption`을(`true`로 설정해야 합니다. `luksEncryption` 옵션은 다음 예와 같이 볼륨이 LUKS 규격을 준수하는지(`true` 또는 LUKS 규격을 준수하지 않는지(`false`를 Trident에 알려줍니다.

```
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: trident_svm
username: admin
password: password
defaults:
  luksEncryption: 'true'
  spaceAllocation: 'false'
  snapshotPolicy: default
  snapshotReserve: '10'
```

LUKS 볼륨 가져오기를 위한 PVC 구성

LUKS 볼륨을 동적으로 가져오려면 annotation `trident.netapp.io/luksEncryption`을 `true`로 설정하고 이 예제와 같이 PVC에 LUKS 지원 스토리지 클래스를 포함하십시오.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: luks-pvc
  namespace: trident
  annotations:
    trident.netapp.io/luksEncryption: "true"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: luks-sc
```

LUKS 암호 교체

LUKS 암호를 교체하고 교체를 확인할 수 있습니다.

> 볼륨, 스냅샷 또는 시크릿에서 더 이상 참조되지 않는다는 것을 확인하기 전까지는 암호를 잊지 마십시오. 참조된 암호를 분실하면 볼륨을 마운트할 수 없으며 데이터는 암호화된 상태로 유지되어 액세스할 수 없게 됩니다.

이 작업 정보

LUKS 암호 순환은 새 LUKS 암호가 지정된 후 해당 볼륨을 마운트하는 Pod가 생성될 때 발생합니다. 새 Pod가 생성될 때 Trident는 볼륨의 LUKS 암호를 시크릿에 있는 활성 암호와 비교합니다.

- 볼륨의 암호가 비밀의 활성 암호와 일치하지 않으면 순환이 발생합니다.
- 볼륨의 암호가 비밀 키에 있는 활성 암호와 일치하면 `previous-luks-passphrase` 매개변수는 무시됩니다.

단계

1. `node-publish-secret-name` 및 `node-publish-secret-namespace` StorageClass 매개변수를 추가합니다. 예를 들면 다음과 같습니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-san
provisioner: csi.trident.netapp.io
parameters:
  trident.netapp.io/backendType: "ontap-san"
  csi.storage.k8s.io/node-stage-secret-name: luks
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
  csi.storage.k8s.io/node-publish-secret-name: luks
  csi.storage.k8s.io/node-publish-secret-namespace: ${pvc.namespace}
```

2. 볼륨 또는 스냅샷에 있는 기존 암호를 확인합니다.

볼륨

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames:["A"]
```

스냅샷

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>
```

```
...luksPassphraseNames:["A"]
```

3. 볼륨의 LUKS 암호를 업데이트하여 새 암호와 이전 암호를 지정하십시오. `previous-luke-passphrase-name`와 `previous-luks-passphrase`가 이전 암호와 일치하는지 확인하십시오.

```
apiVersion: v1
kind: Secret
metadata:
  name: luks-pvc1
stringData:
  luks-passphrase-name: B
  luks-passphrase: secretB
  previous-luks-passphrase-name: A
  previous-luks-passphrase: secretA
```

4. 볼륨을 마운트하는 새 Pod를 생성합니다. 이는 로테이션을 시작하는 데 필요합니다.
5. 암호가 교체되었는지 확인하십시오.

볼륨

```
tridentctl -d get volume luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>

...luksPassphraseNames:["B"]
```

스냅샷

```
tridentctl -d get snapshot luks-pvc1
GET http://127.0.0.1:8000/trident/v1/volume/<volumeID>/<snapshotID>

...luksPassphraseNames:["B"]
```

결과

볼륨 및 스냅샷에 새 암호만 반환되면 암호가 교체된 것입니다.

> 예를 들어 두 개의 암호문이 반환되면 `luksPassphraseNames: ["B", "A"]` 로테이션이 완료되지 않은 것입니다. 로테이션을 완료하기 위해 새 Pod를 트리거할 수 있습니다.

볼륨 확장 활성화

LUKS로 암호화된 볼륨에서 볼륨 확장을 활성화할 수 있습니다.

단계

1. `CSINodeExpandSecret` 기능 게이트를 활성화합니다(베타 1.25+). 자세한 내용은 "Kubernetes 1.25: 노드

기반 CSI 볼륨 확장을 위한 시크릿 사용"을(를) 참조하십시오.

2. `node-expand-secret-name` 및 `node-expand-secret-namespace` StorageClass 매개변수를 추가합니다. 예를 들면 다음과 같습니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: luks
provisioner: csi.trident.netapp.io
parameters:
  selector: "luks=true"
  csi.storage.k8s.io/node-stage-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
  csi.storage.k8s.io/node-expand-secret-name: luks-${pvc.name}
  csi.storage.k8s.io/node-expand-secret-namespace: ${pvc.namespace}
allowVolumeExpansion: true
```

결과

온라인 스토리지 확장을 시작하면 kubelet이 적절한 자격 증명을 드라이버에 전달합니다.

Trident Kerberos 전송 중 암호화

Kerberos 전송 중 암호화를 사용하면 관리형 클러스터와 스토리지 백엔드 간의 트래픽에 대한 암호화를 활성화하여 데이터 액세스 보안을 개선할 수 있습니다.

Trident는 스토리지 백엔드로 ONTAP에 대한 Kerberos 암호화를 지원합니다.

- 온프레미스 **ONTAP** - Trident는 Red Hat OpenShift 및 업스트림 Kubernetes 클러스터에서 온프레미스 ONTAP 볼륨으로의 NFSv3 및 NFSv4 연결을 통해 Kerberos 암호화를 지원합니다.

NFS 암호화를 사용하는 볼륨을 생성, 삭제, 크기 조정, 스냅샷, 클론 복제, 읽기 전용 클론 복제 및 가져오기를 수행할 수 있습니다.

온프레미스 **ONTAP** 볼륨을 사용하여 전송 중 **Kerberos** 암호화 구성

관리형 클러스터와 온프레미스 ONTAP 스토리지 백엔드 간의 스토리지 트래픽에 Kerberos 암호화를 활성화할 수 있습니다.

> ⓘ 온프레미스 ONTAP 스토리지 백엔드를 사용하는 NFS 트래픽에 대한 Kerberos 암호화는 `ontap-nas` 스토리지 드라이버를 통해서만 지원됩니다.

시작하기 전에

- `tridentctl` 유틸리티에 액세스할 수 있는지 확인하십시오.
- ONTAP 스토리지 백엔드에 대한 관리자 액세스 권한이 있는지 확인하십시오.
- ONTAP 스토리지 백엔드에서 공유할 볼륨의 이름을 알고 있는지 확인하십시오.

- NFS 볼륨에 대한 Kerberos 암호화를 지원하도록 ONTAP 스토리지 VM을 준비했는지 확인하십시오. 지침은 "dataLIF에서 Kerberos를 활성화합니다"을(를) 참조하십시오.
- Kerberos 암호화와 함께 사용하는 모든 NFSv4 볼륨이 올바르게 구성되었는지 확인하십시오. "NetApp NFSv4 향상 기능 및 모범 사례 가이드"의 NetApp NFSv4 도메인 구성 섹션(13페이지)을 참조하십시오.

ONTAP 내보내기 정책을 추가 또는 수정합니다

기존 ONTAP 내보내기 정책에 규칙을 추가하거나 ONTAP 스토리지 VM 루트 볼륨과 업스트림 Kubernetes 클러스터와 공유되는 모든 ONTAP 볼륨에 대해 Kerberos 암호화를 지원하는 새 내보내기 정책을 생성해야 합니다. 추가하는 내보내기 정책 규칙 또는 새로 생성하는 내보내기 정책은 다음 액세스 프로토콜 및 액세스 권한을 지원해야 합니다.

액세스 프로토콜

NFS, NFSv3 및 NFSv4 액세스 프로토콜을 사용하여 엑스포트 정책을 구성합니다.

액세스 세부 정보

볼륨에 대한 요구 사항에 따라 세 가지 버전의 Kerberos 암호화 중 하나를 구성할 수 있습니다.

- **Kerberos 5** - (인증 및 암호화)
- **Kerberos 5i** - (ID 보호 기능을 갖춘 인증 및 암호화)
- **Kerberos 5p** - (신원 및 개인정보 보호를 통한 인증 및 암호화)

적절한 액세스 권한으로 ONTAP 엑스포트 정책 규칙을 구성합니다. 예를 들어, 클러스터가 Kerberos 5i와 Kerberos 5p 암호화를 혼합하여 NFS 볼륨을 마운트하는 경우 다음 액세스 설정을 사용하십시오.

유형	읽기 전용 액세스	읽기/쓰기 액세스	슈퍼유저 액세스
UNIX	활성화됨	활성화됨	활성화됨
Kerberos 5i	활성화됨	활성화됨	활성화됨
Kerberos 5p	활성화됨	활성화됨	활성화됨

ONTAP 내보내기 정책 및 내보내기 정책 규칙을 생성하는 방법은 다음 문서를 참조하십시오.

- "엑스포트 정책을 생성합니다"
- "엑스포트 정책에 규칙 추가"

스토리지 백엔드를 생성합니다

Kerberos 암호화 기능을 포함하는 Trident 스토리지 백엔드 구성을 생성할 수 있습니다.

이 작업 정보

Kerberos 암호화를 구성하는 스토리지 백엔드 구성 파일을 생성할 때 `spec.nfsMountOptions` 매개변수를 사용하여 세 가지 Kerberos 암호화 버전 중 하나를 지정할 수 있습니다.

- `spec.nfsMountOptions: sec=krb5` (인증 및 암호화)
- `spec.nfsMountOptions: sec=krb5i` (ID 보호를 통한 인증 및 암호화)
- `spec.nfsMountOptions: sec=krb5p` (ID 및 개인 정보 보호를 통한 인증 및 암호화)

Kerberos 수준을 하나만 지정하십시오. 매개변수 목록에 두 개 이상의 Kerberos 암호화 수준을 지정하면 첫 번째 옵션만 사용됩니다.

단계

1. 관리형 클러스터에서 다음 예제를 사용하여 스토리지 백엔드 구성 파일을 생성합니다. 괄호 <> 안의 값은 사용자 환경에 맞는 정보로 바꾸십시오.

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-ontap-nas-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-ontap-nas
spec:
  version: 1
  storageDriverName: "ontap-nas"
  managementLIF: <STORAGE_VM_MGMT_LIF_IP_ADDRESS>
  dataLIF: <PROTOCOL_LIF_FQDN_OR_IP_ADDRESS>
  svm: <STORAGE_VM_NAME>
  username: <STORAGE_VM_USERNAME_CREDENTIAL>
  password: <STORAGE_VM_PASSWORD_CREDENTIAL>
  nasType: nfs
  nfsMountOptions: ["sec=krb5i"] #can be krb5, krb5i, or krb5p
  qtreesPerFlexvol:
  credentials:
    name: backend-ontap-nas-secret
```

2. 이전 단계에서 생성한 구성 파일을 사용하여 백엔드를 생성하십시오.

```
tridentctl create backend -f <backend-configuration-file>
```

백엔드 생성에 실패하면 백엔드 구성에 문제가 있는 것입니다. 다음 명령을 실행하여 로그를 확인하고 원인을 파악할 수 있습니다.

```
tridentctl logs
```

구성 파일의 문제를 식별하고 수정한 후 create 명령을 다시 실행할 수 있습니다.

스토리지 클래스를 생성합니다

Kerberos 암호화를 사용하여 볼륨을 프로비저닝하는 스토리지 클래스를 생성할 수 있습니다.

이 작업 정보

스토리지 클래스 객체를 생성할 때 `mountOptions` 매개변수를 사용하여 세 가지 Kerberos 암호화 버전 중 하나를 지정할 수 있습니다.

- `mountOptions: sec=krb5` (인증 및 암호화)
- `mountOptions: sec=krb5i` (ID 보호를 통한 인증 및 암호화)
- `mountOptions: sec=krb5p` (ID 및 개인 정보 보호를 통한 인증 및 암호화)

Kerberos 수준을 하나만 지정하십시오. 매개변수 목록에 두 개 이상의 Kerberos 암호화 수준을 지정하면 첫 번째 옵션만 사용됩니다. 스토리지 백엔드 구성에서 지정한 암호화 수준이 스토리지 클래스 객체에서 지정한 수준과 다른 경우 스토리지 클래스 객체가 우선합니다.

단계

1. 다음 예제를 사용하여 StorageClass Kubernetes 객체를 생성합니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-nas-sc
provisioner: csi.trident.netapp.io
mountOptions:
  - sec=krb5i #can be krb5, krb5i, or krb5p
parameters:
  backendType: ontap-nas
  storagePools: ontapnas_pool
  trident.netapp.io/nasType: nfs
allowVolumeExpansion: true
```

2. 스토리지 클래스를 생성합니다.

```
kubectl create -f sample-input/storage-class-ontap-nas-sc.yaml
```

3. 스토리지 클래스가 생성되었는지 확인하십시오.

```
kubectl get sc ontap-nas-sc
```

다음과 유사한 출력이 표시됩니다.

```
NAME          PROVISIONER             AGE
```

```
ontap-nas-sc    csi.trident.netapp.io   15h
```

볼륨 프로비저닝

스토리지 백엔드와 스토리지 클래스를 생성한 후에는 볼륨을 프로비저닝할 수 있습니다. 자세한 내용은 "볼륨 프로비저닝"을 참조하십시오.

Azure NetApp Files 볼륨에서 전송 중 Kerberos 암호화 구성

관리형 클러스터와 단일 Azure NetApp Files 스토리지 백엔드 또는 Azure NetApp Files 스토리지 백엔드의 가상 풀 간의 스토리지 트래픽에 대해 Kerberos 암호화를 활성화할 수 있습니다.

시작하기 전에

- 관리형 Red Hat OpenShift 클러스터에서 Trident가 활성화되어 있는지 확인하십시오.
- `tridentctl` 유틸리티에 액세스할 수 있는지 확인하십시오.
- 요구 사항을 확인하고 "Azure NetApp Files 설명서"의 지침에 따라 Kerberos 암호화를 위한 Azure NetApp Files 스토리지 백엔드를 준비했는지 확인하십시오.
- Kerberos 암호화와 함께 사용하는 모든 NFSv4 볼륨이 올바르게 구성되었는지 확인하십시오. "NetApp NFSv4 향상 기능 및 모범 사례 가이드"의 NetApp NFSv4 도메인 구성 섹션(13페이지)을 참조하십시오.

스토리지 백엔드를 생성합니다

Kerberos 암호화 기능을 포함하는 Azure NetApp Files 스토리지 백엔드 구성을 만들 수 있습니다.

이 작업 정보

Kerberos 암호화를 구성하는 스토리지 백엔드 구성 파일을 생성할 때 다음 두 가지 수준 중 하나에 적용되도록 정의할 수 있습니다.

- `spec.kerberos` 필드를 사용하는 스토리지 백엔드 레벨
- `spec.storage.kerberos` 필드를 사용한 가상 풀 레벨

가상 풀 수준에서 구성을 정의할 때 스토리지 클래스의 레이블을 사용하여 풀이 선택됩니다.

어느 수준에서든 세 가지 버전의 Kerberos 암호화 중 하나를 지정할 수 있습니다.

- `kerberos: sec=krb5` (인증 및 암호화)
- `kerberos: sec=krb5i` (ID 보호를 통한 인증 및 암호화)
- `kerberos: sec=krb5p` (ID 및 개인 정보 보호를 통한 인증 및 암호화)

단계

1. 관리형 클러스터에서 스토리지 백엔드를 정의해야 하는 위치(스토리지 백엔드 수준 또는 가상 풀 수준)에 따라 다음 예제 중 하나를 사용하여 스토리지 백엔드 구성 파일을 생성합니다. 괄호 <> 안의 값은 사용자 환경에 맞는 정보로 바꾸십시오.

스토리지 백엔드 수준 예

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-secret
```

가상 풀 수준 예

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
```

```
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  storage:
    - labels:
        type: encryption
      kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-secret
```

2. 이전 단계에서 생성한 구성 파일을 사용하여 백엔드를 생성하십시오.

```
tridentctl create backend -f <backend-configuration-file>
```

백엔드 생성에 실패하면 백엔드 구성에 문제가 있는 것입니다. 다음 명령을 실행하여 로그를 확인하고 원인을 파악할 수 있습니다.

```
tridentctl logs
```

구성 파일의 문제를 식별하고 수정한 후 create 명령을 다시 실행할 수 있습니다.

스토리지 클래스를 생성합니다

Kerberos 암호화를 사용하여 볼륨을 프로비저닝하는 스토리지 클래스를 생성할 수 있습니다.

단계

1. 다음 예제를 사용하여 StorageClass Kubernetes 객체를 생성합니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-nfs
provisioner: csi.trident.netapp.io
parameters:
  backendType: azure-netapp-files
  trident.netapp.io/nasType: nfs
  selector: type=encryption
```

2. 스토리지 클래스를 생성합니다.

```
kubectl create -f sample-input/storage-class-sc-nfs.yaml
```

3. 스토리지 클래스가 생성되었는지 확인하십시오.

```
kubectl get sc -sc-nfs
```

다음과 유사한 출력이 표시됩니다.

```
NAME          PROVISIONER             AGE
sc-nfs        csi.trident.netapp.io   15h
```

볼륨 프로비저닝

스토리지 백엔드와 스토리지 클래스를 생성한 후에는 볼륨을 프로비저닝할 수 있습니다. 자세한 내용은 "볼륨 프로비저닝"을 참조하십시오.

저작권 정보

Copyright © 2026 NetApp, Inc. All Rights Reserved. 미국에서 인쇄됨 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 라이센스와 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 라이센스도 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 라이센스를 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 라이센스는 DFARS 조항 252.227-7015(b)(2014년 2월)에 명시된 권한으로 제한됩니다.

상표 정보

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