백엔드 구성 및 관리

Trident

NetApp
February 02, 2026

This PDF was generated from https://docs.netapp.com/ko-kr/trident/trident-use/backends.html on February 02, 2026. Always check docs.netapp.com for the latest.

목차

백엔드 구성 및 관리 . . . 1
- 백엔드 구성 . . . 1
- Azure NetApp Files . . . 1
  - Azure NetApp Files 백엔드를 구성합니다 . . . 1
  - Azure NetApp Files 백엔드를 구성할 준비를 합니다 . . . 5
  - Azure NetApp Files 백엔드 구성 옵션 및 예 . . . 8
- Google Cloud NetApp 볼륨 . . . 20
  - Google Cloud NetApp 볼륨 백엔드를 구성합니다 . . . 20
  - Google Cloud NetApp 볼륨 백엔드 구성을 준비합니다 . . . 23
  - Google Cloud NetApp 볼륨 백엔드 구성 옵션 및 예 . . . 23
- NetApp HCI 또는 SolidFire 백엔드를 구성합니다 . . . 37
  - 요소 드라이버 세부 정보 . . . 37
  - 시작하기 전에 . . . 38
  - 백엔드 구성 옵션 . . . 38
  - 예 1: 에 대한 백엔드 구성 `solidfire-san` 세 가지 볼륨 유형을 가진 드라이버 . . . 39
  - 예 2: 에 대한 백엔드 및 스토리지 클래스 구성 `solidfire-san` 가상 풀이 있는 드라이버 . . . 40
  - 자세한 내용을 확인하십시오 . . . 42
- ONTAP SAN 드라이버 . . . 42
  - ONTAP SAN 드라이버 개요 . . . 42
  - ONTAP SAN 드라이버를 사용하여 백엔드를 구성할 준비를 합니다 . . . 44
  - ONTAP SAN 구성 옵션 및 예 . . . 52
- ONTAP NAS 드라이버 . . . 71
  - ONTAP NAS 드라이버 개요 . . . 71
  - ONTAP NAS 드라이버를 사용하여 백엔드를 구성할 준비를 합니다 . . . 73
  - ONTAP NAS 구성 옵션 및 예 . . . 85
- NetApp ONTAP용 Amazon FSx . . . 107
  - Trident를 Amazon FSx for NetApp ONTAP와 함께 사용해 보십시오 . . . 107
  - IAM 역할 및 AWS Secret을 생성합니다 . . . 110
  - Trident를 설치합니다 . . . 115
  - 스토리지 백엔드를 구성합니다 . . . 123
  - 저장소 클래스 및 PVC를 구성합니다 . . . 131
  - 샘플 응용 프로그램을 배포합니다 . . . 136
  - EKS 클러스터에서 Trident EKS 애드온을 구성합니다 . . . 137
- kubectl로 백엔드를 만듭니다 . . . 140
  - `TridentBackendConfig` . . . 141
  - 단계 개요 . . . 142
  - 1단계: Kubernetes Secret 생성 . . . 142
  - 2단계: 을 작성합니다 `TridentBackendConfig` 있습니다 . . . 144

3단계: 의 상태를 확인합니다 `TridentBackendConfig` 있습니다 . . . 144
(선택 사항) 4단계: 자세한 내용을 확인하십시오 . . . 145
백엔드 관리 . . . 146
kubeck을 사용하여 백엔드 관리 수행 . . . 147
tridentctl을 사용하여 백엔드 관리를 수행합니다 . . . 148
백엔드 관리 옵션 간 이동 . . . 149

백엔드 구성 및 관리

백엔드 구성

백엔드는 Trident와 스토리지 시스템 간의 관계를 정의합니다. Trident는 해당 스토리지 시스템과 통신하는 방법과 Trident가 스토리지 시스템에서 볼륨을 프로비저닝하는 방법을 알려줍니다.

Trident는 스토리지 클래스에 정의된 요구 사항에 맞는 백 엔드에서 스토리지 풀을 자동으로 제공합니다. 스토리지 시스템에 대한 백엔드를 구성하는 방법에 대해 알아봅니다.

- "Azure NetApp Files 백엔드를 구성합니다"
- "Google Cloud NetApp 볼륨 백엔드를 구성합니다"
- "NetApp HCI 또는 SolidFire 백엔드를 구성합니다"
- "ONTAP 또는 Cloud Volumes ONTAP NAS 드라이버를 사용하여 백엔드를 구성합니다"
- "ONTAP 또는 Cloud Volumes ONTAP SAN 드라이버를 사용하여 백엔드를 구성합니다"
- "Trident를 Amazon FSx for NetApp ONTAP와 함께 사용해 보십시오"

Azure NetApp Files

Azure NetApp Files 백엔드를 구성합니다

Azure NetApp Files를 Trident의 백엔드로 구성할 수 있습니다. Azure NetApp Files 백엔드를 사용하여 NFS 및 SMB 볼륨을 연결할 수 있습니다. 또한 Trident은 Azure Kubernetes Services(AKS) 클러스터에 대해 관리되는 ID를 사용하여 자격 증명 관리를 지원합니다.

Azure NetApp Files 드라이버 세부 정보입니다

Trident은 클러스터와 통신할 수 있도록 다음과 같은 Azure NetApp Files 스토리지 드라이버를 제공합니다. 지원되는 액세스 모드는 *ReadWriteOnce*(RWO), *ReadOnlyMany*(ROX), *ReadWriteMany*(rwx), *ReadWriteOncePod*(RWOP)입니다.

드라이버	프로토콜	볼륨 모드	액세스 모드가 지원됩니다	지원되는 파일 시스템
'Azure-NetApp-파일'	NFS 를 참조하십시오 중소기업	파일 시스템	RWO, ROX, rwx, RWOP	`nfs`, `smb`

고려 사항

- Azure NetApp Files 서비스는 50GiB보다 작은 볼륨을 지원하지 않습니다. 더 작은 볼륨을 요청하는 경우, Trident에서 자동으로 50GiB 볼륨을 생성합니다.
- Trident는 Windows 노드에서만 실행되는 Pod에 마운트된 SMB 볼륨을 지원합니다.

AKS의 관리되는 ID입니다

Trident는 "관리되는 ID입니다"Azure Kubernetes 서비스 클러스터를 지원합니다. 관리되는 ID에서 제공하는 효율적인 자격 증명 관리를 활용하려면 다음을 수행해야 합니다.

- AKS를 사용하여 구축된 Kubernetes 클러스터
- AKS Kubernetes 클러스터에 구성된 관리되는 ID입니다
- 지정할 "Azure" 가 포함된 Trident가 설치되었습니다. cloudProvider

Trident 운영자

Trident 연산자를 사용하여 Trident를 설치하려면 tridentorchestrator_cr.yaml 를 cloudProvider 로 `"Azure"`설정합니다. 예를 들면 다음과 같습니다.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "Azure"
```

헬름

다음 예에서는 환경 변수를 사용하여 Trident 집합을 Azure로 $CP 설치합니다 cloudProvider.

```
helm install trident trident-operator-100.2506.0.tgz --create
-namespace --namespace <trident-namespace> --set cloudProvider=$CP
```

<code>tridentctl</code>

다음 예에서는 Trident를 설치하고 플래그를 로 Azure 설정합니다 cloudProvider.

```
tridentctl install --cloud-provider="Azure" -n trident
```

AKS용 클라우드 ID

클라우드 ID를 사용하면 Kubernetes Pod에서 명시적 Azure 자격 증명을 제공하지 않고 워크로드 ID로 인증하여 Azure 리소스에 액세스할 수 있습니다.

Azure에서 클라우드 ID를 활용하려면 다음이 필요합니다.

- AKS를 사용하여 구축된 Kubernetes 클러스터

- AKS Kubernetes 클러스터에 구성된 워크로드 ID 및 oidc-발급자입니다
- "Azure"`워크로드 ID를 지정하고 `cloudIdentity 지정하는 가 포함된 Trident가 설치되었습니다 cloudProvider

Trident 운영자

Trident 연산자를 사용하여 Trident를 설치하려면 `tridentorchestrator_cr.yaml` 를 `cloudProvider` 로 "Azure" 설정하고 를 로 `cloudIdentity` `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx` 설정합니다.

예를 들면 다음과 같습니다.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "Azure"
  cloudIdentity: 'azure.workload.identity/client-id: xxxxxxxx-xxxx-
xxxx-xxxx-xxxxxxxxxxx' # Edit
```

헬름

다음 환경 변수를 사용하여 * 클라우드 공급자(CP) * 및 * 클라우드 ID(CI) * 플래그의 값을 설정합니다.

```
export CP="Azure"
export CI="'azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxx'"
```

다음 예에서는 Trident를 설치하고 `cloudProvider` 환경 변수를 사용하여 Azure로 `$CP` 설정하고 환경 변수를 사용하여 `$CI` 를 `cloudIdentity` 설정합니다.

```
helm install trident trident-operator-100.6.0.tgz --set
cloudProvider=$CP --set cloudIdentity="$CI"
```

<code>tridentctl</code>

다음 환경 변수를 사용하여 * 클라우드 공급자 * 및 * 클라우드 ID * 플래그의 값을 설정합니다.

```
export CP="Azure"
export CI="azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxx"
```

다음 예에서는 Trident를 설치하고 플래그를 `$CP` , 및 `cloud-identity` 로 설정합니다 `cloud-provider. $CI`

```
tridentctl install --cloud-provider=$CP --cloud-identity="$CI" -n
trident
```

Azure NetApp Files 백엔드를 구성할 준비를 합니다

Azure NetApp Files 백엔드를 구성하기 전에 다음 요구 사항이 충족되는지 확인해야 합니다.

NFS 및 SMB 볼륨의 사전 요구 사항

Azure NetApp Files를 처음 사용하거나 새 위치에서 사용하는 경우 Azure NetApp Files를 설정하고 NFS 볼륨을 생성하려면 몇 가지 초기 구성이 필요합니다. 을 참조하십시오 "Azure: Azure NetApp Files를 설정하고 NFS 볼륨을 생성합니다".

를 구성하고 사용합니다 "Azure NetApp Files" 백엔드, 다음이 필요합니다.

- `subscriptionID`, `tenantID`, `clientID`, `location`, 및 `clientSecret` AKS 클러스터에서 관리되는 ID를 사용하는 경우 선택 사항입니다.
- `tenantID`, `clientID`, 및 `clientSecret` AKS 클러스터에서 클라우드 ID를 사용할 때는 선택 사항입니다.

- 용량 풀입니다. 을 참조하십시오 "Microsoft: Azure NetApp Files에 대한 용량 풀을 생성합니다".
- Azure NetApp Files에 위임된 서브넷. 을 참조하십시오 "Microsoft: Azure NetApp Files에 서브넷을 위임합니다".
- Azure NetApp Files가 활성화된 Azure 구독의 'SubscriptionID'입니다.
- `tenantID`, `clientID`, 및 `clientSecret` 에서 "앱 등록" Azure NetApp Files 서비스에 대한 충분한 권한이 있는 Azure Active Directory에서 앱 등록에서는 다음 중 하나를 사용해야 합니다.
  - 소유자 또는 참가자 역할입니다 "Azure에서 사전 정의".
  - a "사용자 지정 참가자 역할"구독 수준의 (`assignableScopes`경우) Trident에 필요한 권한으로만 제한된 다음 사용 권한이 있습니다. 사용자 지정 역할을 만든 후"Azure 포털을 사용하여 역할을 할당합니다"

사용자 지정 기고자 역할입니다

```
{
  "id": "/subscriptions/<subscription-
id>/providers/Microsoft.Authorization/roleDefinitions/<role-
definition-id>",
  "properties": {
    "roleName": "custom-role-with-limited-perms",
    "description": "custom role providing limited permissions",
    "assignableScopes": [
      "/subscriptions/<subscription-id>"
    ],
    "permissions": [
      {
        "actions": [
          "Microsoft.NetApp/netAppAccounts/capacityPools/read",
          "Microsoft.NetApp/netAppAccounts/capacityPools/write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/read",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/delete",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
read",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
delete",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/MountTarge
ts/read",
          "Microsoft.Network/virtualNetworks/read",
          "Microsoft.Network/virtualNetworks/subnets/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/write",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
```

```
ions/delete",
          "Microsoft.Features/features/read",
          "Microsoft.Features/operations/read",
          "Microsoft.Features/providers/features/read",

"Microsoft.Features/providers/features/register/action",

"Microsoft.Features/providers/features/unregister/action",

"Microsoft.Features/subscriptionFeatureRegistrations/read"
        ],
        "notActions": [],
        "dataActions": [],
        "notDataActions": []
      }
    ]
  }
}
```

- Azure를 선택합니다 `location` 하나 이상의 항목이 포함되어 있습니다 "위임된 서브넷". Trident 22.01부터 `location` 매개 변수는 백엔드 구성 파일의 최상위 수준에 있는 필수 필드입니다. 가상 풀에 지정된 위치 값은 무시됩니다.
- 사용합니다 `Cloud Identity`을(를) 다운로드하십시오 `client ID` 에서 "사용자가 할당한 관리 ID 입니다" 에서 해당 ID를 지정합니다 `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx`.

SMB 볼륨에 대한 추가 요구사항

SMB 볼륨을 생성하려면 다음이 있어야 합니다.

- Active Directory가 구성되어 Azure NetApp Files에 연결되었습니다. 을 참조하십시오 "Microsoft: Azure NetApp Files에 대한 Active Directory 연결을 만들고 관리합니다".
- Linux 컨트롤러 노드 및 Windows Server 2022를 실행하는 Windows 작업자 노드가 있는 Kubernetes 클러스터 Trident는 Windows 노드에서만 실행되는 Pod에 마운트된 SMB 볼륨을 지원합니다.
- Azure NetApp Files가 Active Directory에 인증할 수 있도록 Active Directory 자격 증명을 포함하는 Trident 암호가 하나 이상 있어야 합니다. 비밀 생성하기 `smbcreds`:

```
kubectl create secret generic smbcreds --from-literal username=user
--from-literal password='password'
```

- Windows 서비스로 구성된 CSI 프록시. 를 구성합니다 `csi-proxy`를 참조하십시오 "GitHub:CSI 프록시" 또는 "GitHub: Windows용 CSI 프록시" Windows에서 실행되는 Kubernetes 노드의 경우:

Azure NetApp Files 백엔드 구성 옵션 및 예

Azure NetApp Files에 대한 NFS 및 SMB 백엔드 구성 옵션에 대해 알아보고 구성 예제를 검토합니다.

백엔드 구성 옵션

Trident은 백엔드 구성(서브넷, 가상 네트워크, 서비스 수준 및 위치)을 사용하여 요청된 위치에서 사용 가능하고 요청된 서비스 수준 및 서브넷과 일치하는 용량 풀에 Azure NetApp Files 볼륨을 생성합니다.

Azure NetApp Files 백엔드는 이러한 구성 옵션을 제공합니다.

매개 변수	설명	기본값
'내전'		항상 1
'torageDriverName'입니다	스토리지 드라이버의 이름입니다	"Azure-NetApp-파일"
백엔드이름	사용자 지정 이름 또는 스토리지 백엔드	드라이버 이름 + "_" + 임의 문자
'스크립트 ID'입니다	Azure 구독의 구독 ID입니다 AKS 클러스터에서 관리되는 ID가 설정된 경우 선택 사항입니다.	
텐antID	앱 등록에서 테넌트 ID입니다 관리 ID 또는 클라우드 ID가 AKS 클러스터에서 사용되는 경우 선택 사항입니다.	
'클라이언트 ID'	앱 등록의 클라이언트 ID입니다 관리 ID 또는 클라우드 ID가 AKS 클러스터에서 사용되는 경우 선택 사항입니다.	
'clientSecret	앱 등록에서 클라이언트 암호 관리 ID 또는 클라우드 ID가 AKS 클러스터에서 사용되는 경우 선택 사항입니다.	
'저급'	'초표준', '프리미엄', '울트라' 중 하나	""(임의)
위치	새 볼륨을 생성할 Azure 위치의 이름입니다 AKS 클러스터에서 관리되는 ID가 설정된 경우 선택 사항입니다.	
[재치 단체	검색된 자원을 필터링하기 위한 자원 그룹 목록입니다	"[]"(필터 없음)
'netap계정'	검색된 리소스를 필터링하기 위한 NetApp 계정의 목록입니다	"[]"(필터 없음)

매개 변수	설명	기본값
용량풀	검색된 리소스를 필터링하기 위한 용량 풀 목록입니다	"[]"(필터 없음, 임의)
가상네트워크	위임된 서브넷이 있는 가상 네트워크의 이름입니다	""
'우방'	Microsoft.Netapp/volumes` 에 위임된 서브넷의 이름입니다	""
'네트워크 기능'	볼륨에 대한 VNET 기능 집합은 일 수 있습니다 `Basic` 또는 `Standard`. 일부 지역에서는 네트워크 기능을 사용할 수 없으며 구독에서 활성화해야 할 수도 있습니다. 지정 `networkFeatures` 이 기능을 사용하지 않으면 볼륨 프로비저닝이 실패합니다.	""
nfsMountOptions를 선택합니다	NFS 마운트 옵션에 대한 세밀한 제어 SMB 볼륨에 대해 무시됩니다. NFS 버전 4.1을 사용하여 볼륨을 마운트하려면 을 포함합니다 `nfsvers=4` 쉼표로 구분된 마운트 옵션 목록에서 NFS v4.1을 선택합니다. 스토리지 클래스 정의에 설정된 마운트 옵션은 백엔드 구성에 설정된 마운트 옵션을 재정의합니다.	"nfsvers=3"
LimitVolumeSize	요청된 볼륨 크기가 이 값보다 큰 경우 용량 할당에 실패합니다	""(기본적으로 적용되지 않음)
debugTraceFlags를 선택합니다	문제 해결 시 사용할 디버그 플래그입니다. 예: "\{"api":false, "method":true, "discovery":true}". 문제 해결 중이 아니며 자세한 로그 덤프가 필요한 경우가 아니면 이 방법을 사용하지 마십시오.	null입니다
`nasType`	NFS 또는 SMB 볼륨 생성을 구성합니다. 옵션은 입니다 `nfs`, `smb` 또는 null입니다. Null로 설정하면 기본적으로 NFS 볼륨이 설정됩니다.	`nfs`
`supportedTopologies`	이 백엔드에서 지원하는 영역 및 영역의 목록을 나타냅니다. 자세한 내용은 을 "CSI 토폴로지를 사용합니다"참조하십시오.	
`qosType`	QoS 유형(자동 또는 수동)을 나타냅니다.	자동
`maxThroughput`	허용되는 최대 처리량을 MiB/초 단위로 설정합니다. 수동 QoS 용량 풀에서만 지원됩니다.	`4 MiB/sec`

네트워크 기능에 대한 자세한 내용은 을 참조하십시오 "Azure NetApp Files 볼륨에 대한 네트워크 기능을 구성합니다".

필요한 권한 및 리소스

PVC 생성 시 "No capacity pool found" 오류가 발생하는 경우 앱 등록에 필요한 권한 및 리소스(서브넷, 가상 네트워크, 용량 풀)가 없는 것일 수 있습니다. DEBUG가 활성화된 경우 Trident는 백엔드가 생성될 때 검색된 Azure 리소스를 기록합니다. 적절한 역할이 사용되고 있는지 확인합니다.

의 값 `resourceGroups`, `netappAccounts`, `capacityPools`, `virtualNetwork`, 및 `subnet` 간단한 이름 또는 정규화된 이름을 사용하여 지정할 수 있습니다. 이름이 같은 여러 리소스와 이름이 일치할 수 있으므로 대부분의 경우 정규화된 이름을 사용하는 것이 좋습니다.

vNet이 Azure NetApp Files (ANF) 스토리지 계정과 다른 리소스 그룹에 있는 경우 백엔드에 대한 resourceGroups 목록을 구성하는 동안 가상 네트워크에 대한 리소스 그룹을 지정합니다.

를 클릭합니다 `resourceGroups`, `netappAccounts`, 및 `capacityPools` 값은 검색된 리소스 집합을 이 스토리지 백엔드에서 사용할 수 있는 리소스로 제한하는 필터이며, 이 둘을 조합하여 지정할 수 있습니다. 정규화된 이름은 다음 형식을 따릅니다.

유형	형식
리소스 그룹	리소스 그룹>
NetApp 계정	리소스 그룹>/<NetApp 계정>
용량 풀	리소스 그룹>/<NetApp 계정>/<용량 풀>
가상 네트워크	리소스 그룹>/<가상 네트워크>
서브넷	리소스 그룹>/<가상 네트워크>/<서브넷>

볼륨 프로비저닝

구성 파일의 특수 섹션에서 다음 옵션을 지정하여 기본 볼륨 프로비저닝을 제어할 수 있습니다. 을 참조하십시오 예제 설정 를 참조하십시오.

매개 변수	설명	기본값
엑포트 규칙	새 볼륨에 대한 엑스포트 규칙 `exportRule` CIDR 표기법을 사용하여 IPv4 주소 또는 IPv4 서브넷의 조합을 쉼표로 구분해야 합니다. SMB 볼륨에 대해 무시됩니다.	"0.0.0.0/0"
나프산디렉토리	스냅샷 디렉터리의 표시 여부를 제어합니다	NFSv3의 경우 NFSv4의 경우 "true"입니다
'크기'입니다	새 볼륨의 기본 크기입니다	"100G"
유니크권한	새 볼륨의 UNIX 사용 권한(8진수 4자리) SMB 볼륨에 대해 무시됩니다.	""(미리보기 기능, 가입 시 화이트리스트 필요)

예제 설정

다음 예에서는 대부분의 매개 변수를 기본값으로 두는 기본 구성을 보여 줍니다. 이는 백엔드를 정의하는 가장 쉬운 방법입니다.

최소 구성

이는 절대적인 최소 백엔드 구성입니다. 이 구성을 통해 Trident은 구성된 위치에서 Azure NetApp Files에 위임된 모든 NetApp 계정, 용량 풀 및 서브넷을 검색하고 이러한 풀과 서브넷 중 하나에 무작위로 새 볼륨을 배치합니다. 이 생략되므로 `nasType` `nfs` 기본값이 적용되고 백엔드에서 NFS 볼륨에 대한 프로비저닝이 수행됩니다.

이 구성은 Azure NetApp Files를 시작하여 시험할 때 이상적이지만, 실제로는 프로비저닝한 볼륨에 대해 추가 범위를 제공하고 싶을 것입니다.

```
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
  tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
  clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
  clientSecret: SECRET
  location: eastus
```

AKS의 관리되는 **ID**입니다

이 백엔드 구성은 생략됩니다 `subscriptionID`, `tenantID`, `clientID`, 및 `clientSecret`관리되는 ID를 사용할 경우 선택 사항입니다.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  capacityPools:
    - resource-group-1/netapp-account-1/ultra-pool
  resourceGroups:
    - resource-group-1
  netappAccounts:
    -  resource-group-1/netapp-account-1
  virtualNetwork: resource-group-1/eastus-prod-vnet
  subnet: resource-group-1/eastus-prod-vnet/eastus-anf-subnet
```

AKS용 클라우드 ID

이 백엔드 구성은 생략됩니다 `tenantID`, `clientID`, 및 `clientSecret`클라우드 ID를 사용할 경우 선택 사항입니다.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  capacityPools:
    - ultra-pool
  resourceGroups:
    - aks-ami-eastus-rg
  netappAccounts:
    - smb-na
  virtualNetwork: eastus-prod-vnet
  subnet: eastus-anf-subnet
  location: eastus
  subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
```

용량 풀 필터를 사용한 특정 서비스 수준 구성

이 백엔드 구성은 Azure의 용량 풀 위치에 `Ultra` 볼륨을 `eastus` 배치합니다. Trident은 해당 위치에서 Azure NetApp Files에 위임된 모든 서브넷을 자동으로 검색하여 임의로 새 볼륨을 배치합니다.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
serviceLevel: Ultra
capacityPools:
  - application-group-1/account-1/ultra-1
  - application-group-1/account-1/ultra-2
```

수동 **QoS** 용량 풀을 사용한 백엔드 예제

이 백엔드 구성은 Azure의 볼륨을 배치합니다. `eastus` 수동 QoS 용량 풀이 있는 위치입니다.

```
---
version: 1
storageDriverName: azure-netapp-files
backendName: anf1
location: eastus
labels:
  clusterName: test-cluster-1
  cloud: anf
  nasType: nfs
defaults:
  qosType: Manual
storage:
  - serviceLevel: Ultra
    labels:
      performance: gold
    defaults:
      maxThroughput: 10
  - serviceLevel: Premium
    labels:
      performance: silver
    defaults:
      maxThroughput: 5
  - serviceLevel: Standard
    labels:
      performance: bronze
    defaults:
      maxThroughput: 3
```

고급 구성

이 백엔드 구성은 단일 서브넷에 대한 볼륨 배치 범위를 더욱 줄여주고 일부 볼륨 프로비저닝 기본값도 수정합니다.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
serviceLevel: Ultra
capacityPools:
  - application-group-1/account-1/ultra-1
  - application-group-1/account-1/ultra-2
virtualNetwork: application-group-1/eastus-prod-vnet
subnet: application-group-1/eastus-prod-vnet/my-subnet
networkFeatures: Standard
nfsMountOptions: vers=3,proto=tcp,timeo=600
limitVolumeSize: 500Gi
defaults:
  exportRule: 10.0.0.0/24,10.0.1.0/24,10.0.2.100
  snapshotDir: "true"
  size: 200Gi
  unixPermissions: "0777"
```

이 백엔드 구성은 단일 파일에 여러 스토리지 풀을 정의합니다. 다양한 서비스 수준을 지원하는 여러 용량 풀이 있고 이를 나타내는 Kubernetes의 스토리지 클래스를 생성하려는 경우에 유용합니다. 가상 풀 레이블을 사용하여 에 따라 풀을 구분했습니다 `performance`.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
resourceGroups:
  - application-group-1
networkFeatures: Basic
nfsMountOptions: vers=3,proto=tcp,timeo=600
labels:
  cloud: azure
storage:
  - labels:
      performance: gold
    serviceLevel: Ultra
    capacityPools:
      - application-group-1/netapp-account-1/ultra-1
      - application-group-1/netapp-account-1/ultra-2
    networkFeatures: Standard
  - labels:
      performance: silver
    serviceLevel: Premium
    capacityPools:
      - application-group-1/netapp-account-1/premium-1
  - labels:
      performance: bronze
    serviceLevel: Standard
    capacityPools:
      - application-group-1/netapp-account-1/standard-1
      - application-group-1/netapp-account-1/standard-2
```

지원되는 토폴로지 구성

Trident은 지역 및 가용 영역을 기준으로 워크로드에 대한 볼륨을 손쉽게 프로비저닝할 수 있도록 지원합니다. `supportedTopologies`이 백엔드 구성의 블록은 백엔드당 영역 및 영역 목록을 제공하는 데 사용됩니다. 여기에 지정한 지역 및 영역 값은 각 Kubernetes 클러스터 노드의 레이블에 있는 지역 및 영역 값과 일치해야 합니다. 이러한 영역 및 영역은 스토리지 클래스에서 제공할 수 있는 허용 가능한 값 목록을 나타냅니다. 백엔드에서 제공되는 영역 및 영역의 하위 집합이 포함된 스토리지 클래스의 경우 Trident는 언급한 영역 및 영역에 볼륨을 생성합니다. 자세한 내용은 을 "CSI 토폴로지를 사용합니다"참조하십시오.

```
---
version: 1
storageDriverName: azure-netapp-files
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: eastus
serviceLevel: Ultra
capacityPools:
  - application-group-1/account-1/ultra-1
  - application-group-1/account-1/ultra-2
supportedTopologies:
  - topology.kubernetes.io/region: eastus
    topology.kubernetes.io/zone: eastus-1
  - topology.kubernetes.io/region: eastus
    topology.kubernetes.io/zone: eastus-2
```

스토리지 클래스 정의

다음 사항을 참조하십시오 `StorageClass` 정의는 위의 스토리지 풀을 참조합니다.

을 사용한 정의 예 `parameter.selector` 필드에 입력합니다

사용 `parameter.selector` 각각에 대해 지정할 수 있습니다 `StorageClass` 볼륨을 호스팅하는 데 사용되는 가상 풀입니다. 볼륨은 선택한 풀에 정의된 측면을 갖습니다.

```
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gold
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=gold
allowVolumeExpansion: true

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: silver
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver
allowVolumeExpansion: true

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: bronze
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=bronze
allowVolumeExpansion: true
```

SMB 볼륨에 대한 정의의 예

사용 `nasType`, `node-stage-secret-name`, 및 `node-stage-secret-namespace`, SMB 볼륨을 지정하고 필요한 Active Directory 자격 증명을 제공할 수 있습니다.

기본 네임스페이스에 대한 기본 구성

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
```

네임스페이스별로 다른 암호 사용

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

볼륨별로 다른 암호 사용

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: ${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

`nasType: smb` SMB 볼륨을 지원하는 풀에 대한 필터입니다. `nasType: nfs` 또는 `nasType: null` NFS 풀에 대한 필터입니다.

백엔드를 생성합니다

백엔드 구성 파일을 생성한 후 다음 명령을 실행합니다.

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

백엔드 생성에 실패하면 백엔드 구성에 문제가 있는 것입니다. 다음 명령을 실행하여 로그를 보고 원인을 확인할 수 있습니다.

```
tridentctl logs
```

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

Google Cloud NetApp 볼륨

Google Cloud NetApp 볼륨 백엔드를 구성합니다

이제 Google Cloud NetApp 볼륨을 Trident의 백엔드로 구성할 수 있습니다. Google Cloud NetApp 볼륨 백엔드를 사용하여 NFS 및 SMB 볼륨을 연결할 수 있습니다.

Google Cloud NetApp 볼륨 드라이버 세부 정보입니다

Trident는 `google-cloud-netapp-volumes` 클러스터와 통신할 수 있는 드라이버를 제공합니다. 지원되는 액세스 모드는 *ReadWriteOnce*(RWO), *ReadOnlyMany*(ROX), *ReadWriteMany*(rwx), *ReadWriteOncePod*(RWOP)입니다.

드라이버	프로토콜	볼륨 모드	액세스 모드가 지원됩니다	지원되는 파일 시스템
`google-cloud-netapp-volumes`	NFS 를 참조하십시오 중소기업	파일 시스템	RWO, ROX, rwx, RWOP	`nfs`, `smb`

GKE용 클라우드 ID

클라우드 ID를 사용하면 Kubernetes Pod에서 명시적 Google Cloud 자격 증명을 제공하지 않고 워크로드 ID로 인증하여 Google Cloud 리소스에 액세스할 수 있습니다.

Google Cloud에서 클라우드 ID를 활용하려면 다음이 필요합니다.

- GKE를 사용하여 구축된 Kubernetes 클러스터
- 노드 풀에 구성된 GKE 클러스터 및 GKE 메타데이터 서버에 구성된 워크로드 ID입니다.
- Google Cloud NetApp 볼륨 관리자(역할/NetApp.admin) 역할 또는 사용자 지정 역할이 있는 GCP 서비스 계정

- 새 GCP 서비스 계정을 지정하는 "GCP"와 cloudIdentity를 지정하는 cloudProvider가 포함된 Trident가 설치되었습니다. 예를 들면 다음과 같습니다.

Trident 운영자

Trident 연산자를 사용하여 Trident를 설치하려면 `tridentorchestrator_cr.yaml` 를 `cloudProvider` 로 "GCP" 설정하고 를 로 `cloudIdentity` `iam.gke.io/gcp-service-account: cloudvolumes-admin-sa@mygcpproject.iam.gserviceaccount.com` 설정합니다.

예를 들면 다음과 같습니다.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "GCP"
  cloudIdentity: 'iam.gke.io/gcp-service-account: cloudvolumes-
admin-sa@mygcpproject.iam.gserviceaccount.com'
```

헬름

다음 환경 변수를 사용하여 * 클라우드 공급자(CP) * 및 * 클라우드 ID(CI) * 플래그의 값을 설정합니다.

```
export CP="GCP"
export ANNOTATION="'iam.gke.io/gcp-service-account: cloudvolumes-admin-
sa@mygcpproject.iam.gserviceaccount.com'"
```

다음 예에서는 환경 변수를 사용하여 Trident를 설치하고 를 GCP로 `$CP` 설정하고 `cloudProvider` 환경 변수를 사용하여 `$ANNOTATION` 를 `cloudIdentity` 설정합니다.

```
helm install trident trident-operator-100.6.0.tgz --set
cloudProvider=$CP --set cloudIdentity="$ANNOTATION"
```

<code>tridentctl</code>

다음 환경 변수를 사용하여 * 클라우드 공급자 * 및 * 클라우드 ID * 플래그의 값을 설정합니다.

```
export CP="GCP"
export ANNOTATION="'iam.gke.io/gcp-service-account: cloudvolumes-admin-
sa@mygcpproject.iam.gserviceaccount.com'"
```

다음 예에서는 Trident를 설치하고 플래그를 `$CP` , 및 `cloud-identity` 로 설정합니다 `cloud-provider. $ANNOTATION`

```
tridentctl install --cloud-provider=$CP --cloud
-identity="$ANNOTATION" -n trident
```

Google Cloud NetApp 볼륨 백엔드 구성을 준비합니다

Google Cloud NetApp Volumes 백엔드를 구성하기 전에 다음 요구사항이 충족되는지 확인해야 합니다.

NFS 볼륨을 위한 사전 요구사항

처음으로 또는 새 위치에서 Google Cloud NetApp 볼륨을 사용하는 경우 Google Cloud NetApp 볼륨을 설정하고 NFS 볼륨을 생성하려면 초기 구성이 필요합니다. 을 "시작하기 전에"참조하십시오.

Google Cloud NetApp 볼륨 백엔드를 구성하기 전에 다음 사항이 있는지 확인하십시오.

- Google Cloud NetApp Volumes 서비스로 구성된 Google Cloud 계정 을 "Google Cloud NetApp 볼륨 "참조하십시오.
- Google Cloud 계정의 프로젝트 번호입니다. 을 "프로젝트 식별"참조하십시오.
- NetApp 볼륨 관리자) 역할을 가진 Google Cloud 서비스 계정입니다. (`roles/netapp.admin` 을 "ID 및 액세스 관리 역할 및 권한"참조하십시오.
- GCNV 계정에 대한 API 키 파일입니다. 을 참조하십시오 "서비스 계정 키를 생성합니다"
- 스토리지 풀입니다. 을 "스토리지 풀 개요"참조하십시오.

Google Cloud NetApp 볼륨에 대한 액세스를 설정하는 방법에 대한 자세한 내용은 를 참조하십시오 "Google Cloud NetApp 볼륨에 대한 액세스 설정".

Google Cloud NetApp 볼륨 백엔드 구성 옵션 및 예

Google Cloud NetApp 볼륨의 백엔드 구성 옵션에 대해 알아보고 구성 예제를 검토합니다.

백엔드 구성 옵션

각 백엔드는 단일 Google Cloud 지역에 볼륨을 프로비저닝합니다. 다른 영역에 볼륨을 생성하려면 추가 백엔드를 정의할 수 있습니다.

매개 변수	설명	기본값
'내전'		항상 1
'torageDriverName'입니다	스토리지 드라이버의 이름입니다	의 값을 `storageDriverName` "google-cloud-netapp-volumes"로 지정해야 합니다.
백엔드이름	(선택 사항) 스토리지 백엔드의 사용자 지정 이름입니다	드라이버 이름 + "_" + API 키의 일부

매개 변수	설명	기본값
`storagePools`	볼륨 생성을 위한 스토리지 풀을 지정하는 데 사용되는 선택적 매개 변수입니다.	
'프로젝트 번호'	Google Cloud 계정 프로젝트 번호입니다. 이 값은 Google Cloud 포털 홈 페이지에서 확인할 수 있습니다.	
위치	Trident가 GCNV 볼륨을 생성하는 Google Cloud 위치입니다. 교차 지역 Kubernetes 클러스터를 생성할 경우, 에서 생성된 볼륨을 `location` 여러 Google Cloud 지역의 노드에 예약된 워크로드에 사용할 수 있습니다. 지역 간 트래픽에는 추가 비용이 발생합니다.	
아피키	역할이 지정된 Google Cloud 서비스 계정의 API 키입니다. `netapp.admin` 여기에는 Google Cloud 서비스 계정의 개인 키 파일(백엔드 구성 파일에 verbatim 복사)의 JSON 형식 콘텐츠가 포함됩니다. `apiKey`, , , , , 키를 사용하려면 키-값 쌍을 포함해야 합니다 `type` `project_id` `client_email` `client_id` `auth_uri`. `token_uri` `auth_provider_x509_cert_url`, , 및 `client_x509_cert_url`.	
nfsMountOptions를 선택합니다	NFS 마운트 옵션에 대한 세밀한 제어	"nfsvers=3"
LimitVolumeSize	요청된 볼륨 크기가 이 값보다 큰 경우 용량 할당에 실패합니다.	""(기본적으로 적용되지 않음)
'저급'	스토리지 풀 및 해당 볼륨의 서비스 레벨입니다. 값은 `flex`, `standard`, `premium`또는 `extreme`입니다.	
'라벨'	볼륨에 적용할 임의의 JSON 형식 레이블 세트입니다	""
네트워크	GCNV 볼륨에 사용되는 Google Cloud 네트워크입니다.	
debugTraceFlags를 선택합니다	문제 해결 시 사용할 디버그 플래그입니다. 예: `{"api":false, "method":true}` 문제 해결 중이 아니며 자세한 로그 덤프가 필요한 경우가 아니면 이 방법을 사용하지 마십시오.	null입니다
`nasType`	NFS 또는 SMB 볼륨 생성을 구성합니다. 옵션은 입니다 `nfs`, `smb` 또는 null입니다. Null로 설정하면 기본적으로 NFS 볼륨이 설정됩니다.	`nfs`

`supportedTopologies`	이 백엔드에서 지원하는 영역 및 영역의 목록을 나타냅니다. 자세한 내용은 을 "CSI 토폴로지를 사용합니다"참조하십시오. 예를 들면 다음과 같습니다. `supportedTopologies:`
	`- topology.kubernetes.io/region: asia-east1`
	`topology.kubernetes.io/zone: asia-east1-a`	

볼륨 프로비저닝 옵션

에서 기본 볼륨 프로비저닝을 제어할 수 있습니다 `defaults` 구성 파일의 섹션입니다.

매개 변수	설명	기본값
엑포트 규칙	새 볼륨의 내보내기 규칙. IPv4 주소 조합을 쉼표로 구분하여 나열해야 합니다.	"0.0.0.0/0"
나프산디렉토리	'.snapshot' 디렉토리에 액세스합니다	NFSv3의 경우 NFSv4의 경우 "true"입니다
안산예비역	스냅숏용으로 예약된 볼륨의 백분율입니다	""(기본값 0 적용)
유니크권한	새 볼륨의 UNIX 사용 권한(8진수 4자리)	""

예제 설정

다음 예에서는 대부분의 매개 변수를 기본값으로 두는 기본 구성을 보여 줍니다. 이는 백엔드를 정의하는 가장 쉬운 방법입니다.

최소 구성

이는 절대적인 최소 백엔드 구성입니다. 이 구성을 통해 Trident은 Google Cloud NetApp 볼륨에 위임된 모든 스토리지 풀을 구성된 위치에서 검색하고 해당 풀 중 하나에 무작위로 새 볼륨을 배치합니다. 이 생략되므로 `nasType` `nfs` 기본값이 적용되고 백엔드에서 NFS 볼륨에 대한 프로비저닝이 수행됩니다.

이 구성은 Google Cloud NetApp Volumes로 시작한 후 나중에 시험할 때 이상적이지만, 실제로는 프로비저닝한 볼륨에 대한 추가 범위를 제공해야 할 가능성이 높습니다.

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  serviceLevel: premium
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

SMB 볼륨을 위한 구성

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv1
  namespace: trident
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123456789"
  location: asia-east1
  serviceLevel: flex
  nasType: smb
  apiKey:
    type: service_account
    project_id: cloud-native-data
    client_email: trident-sample@cloud-native-
data.iam.gserviceaccount.com
    client_id: "123456789737813416734"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/trident-
sample%40cloud-native-data.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

StoragePools 필터를 사용하여 구성합니다

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  serviceLevel: premium
  storagePools:
    - premium-pool1-europe-west6
    - premium-pool2-europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

가상 풀 구성

이 백엔드 구성은 단일 파일에 여러 개의 가상 풀을 정의합니다. 가상 풀은 섹션에 정의되어 `storage` 있습니다. 서로 다른 서비스 수준을 지원하는 여러 스토리지 풀이 있고 Kubernetes에서 이러한 풀을 나타내는 스토리지 클래스를 생성하려는 경우에 유용합니다. 가상 풀 레이블은 풀을 구분하는 데 사용됩니다. 예를 들어, 아래 예에서는 `performance` 가상 풀을 구분하는 데 레이블 및 `serviceLevel` 유형이 사용됩니다.

일부 기본값을 모든 가상 풀에 적용할 수 있도록 설정하고 개별 가상 풀에 대한 기본값을 덮어쓸 수도 있습니다. 다음 예에서는 `snapshotReserve` 모든 가상 풀에 대해 기본값으로 사용됩니다. `exportRule`

자세한 내용은 을 "가상 풀"참조하십시오.

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
```

```
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
  defaults:
    snapshotReserve: "10"
    exportRule: 10.0.0.0/24
  storage:
    - labels:
        performance: extreme
      serviceLevel: extreme
      defaults:
        snapshotReserve: "5"
        exportRule: 0.0.0.0/0
    - labels:
        performance: premium
      serviceLevel: premium
    - labels:
        performance: standard
      serviceLevel: standard
```

GKE용 클라우드 ID

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcp-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: '012345678901'
  network: gcnv-network
  location: us-west2
  serviceLevel: Premium
  storagePool: pool-premium1
```

지원되는 토폴로지 구성

Trident은 지역 및 가용 영역을 기준으로 워크로드에 대한 볼륨을 손쉽게 프로비저닝할 수 있도록 지원합니다. `supportedTopologies`이 백엔드 구성의 블록은 백엔드당 영역 및 영역 목록을 제공하는 데 사용됩니다. 여기에 지정한 지역 및 영역 값은 각 Kubernetes 클러스터 노드의 레이블에 있는 지역 및 영역 값과 일치해야 합니다. 이러한 영역 및 영역은 스토리지 클래스에서 제공할 수 있는 허용 가능한 값 목록을 나타냅니다. 백엔드에서 제공되는 영역 및 영역의 하위 집합이 포함된 스토리지 클래스의 경우 Trident는 언급한 영역 및 영역에 볼륨을 생성합니다. 자세한 내용은 을 "CSI 토폴로지를 사용합니다"참조하십시오.

```
---
version: 1
storageDriverName: google-cloud-netapp-volumes
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: asia-east1
serviceLevel: flex
supportedTopologies:
  - topology.kubernetes.io/region: asia-east1
    topology.kubernetes.io/zone: asia-east1-a
  - topology.kubernetes.io/region: asia-east1
    topology.kubernetes.io/zone: asia-east1-b
```

다음 단계

백엔드 구성 파일을 생성한 후 다음 명령을 실행합니다.

```
kubectl create -f <backend-file>
```

백엔드가 성공적으로 생성되었는지 확인하려면 다음 명령을 실행합니다.

```
kubectl get tridentbackendconfig

NAME                BACKEND NAME        BACKEND UUID
PHASE    STATUS
backend-tbc-gcnv    backend-tbc-gcnv    b2fd1ff9-b234-477e-88fd-713913294f65
Bound    Success
```

백엔드 생성에 실패하면 백엔드 구성에 문제가 있는 것입니다. 명령을 사용하여 백엔드를 설명하거나 로그를 확인하여 다음 명령을 실행하여 원인을 확인할 수 있습니다 `kubectl get tridentbackendconfig <backend-name>`.

```
tridentctl logs
```

구성 파일의 문제를 확인하고 수정한 후 백엔드를 삭제하고 create 명령을 다시 실행할 수 있습니다.

스토리지 클래스 정의

다음은 위의 백엔드를 참조하는 기본 `StorageClass` 정의입니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-nfs-sc
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
```

- 필드를 사용한 정의 예 `parameter.selector` : *

를 사용하면 `parameter.selector` 볼륨을 호스팅하는 데 사용되는 에 대해 을 지정할 수 `StorageClass` "가상 풀입니다" 있습니다. 볼륨은 선택한 풀에 정의된 측면을 갖습니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: extreme-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=extreme
  backendType: google-cloud-netapp-volumes

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: premium-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=premium
  backendType: google-cloud-netapp-volumes

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: standard-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=standard
  backendType: google-cloud-netapp-volumes
```

스토리지 클래스에 대한 자세한 내용은 을 "스토리지 클래스를 생성합니다"참조하십시오.

SMB 볼륨에 대한 정의의 예

`node-stage-secret-name`, 및 를 사용하여 `nasType` `node-stage-secret-namespace` SMB 볼륨을 지정하고 필요한 Active Directory 자격 증명을 제공할 수 있습니다. 사용 권한이 있거나 없는 모든 Active Directory 사용자/암호는 노드 단계 비밀에 사용할 수 있습니다.

기본 네임스페이스에 대한 기본 구성

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
```

네임스페이스별로 다른 암호 사용

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

볼륨별로 다른 암호 사용

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: ${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

`nasType: smb` SMB 볼륨을 지원하는 풀에 대한 필터입니다. `nasType: nfs` 또는 `nasType: null` NFS 풀에 대한 필터입니다.

PVC 정의 예

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: gcnv-nfs-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 100Gi
  storageClassName: gcnv-nfs-sc
```

PVC가 바인딩되어 있는지 확인하려면 다음 명령을 실행합니다.

```
kubectl get pvc gcnv-nfs-pvc

NAME           STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS AGE
gcnv-nfs-pvc   Bound    pvc-b00f2414-e229-40e6-9b16-ee03eb79a213   100Gi
RWX            gcnv-nfs-sc  1m
```

NetApp HCI 또는 SolidFire 백엔드를 구성합니다

Trident 설치에서 Element 백엔드를 생성하고 사용하는 방법에 대해 알아봅니다.

요소 드라이버 세부 정보

Trident는 `solidfire-san` 클러스터와 통신할 수 있는 스토리지 드라이버를 제공합니다. 지원되는 액세스 모드는 *ReadWriteOnce*(RWO), *ReadOnlyMany*(ROX), *ReadWriteMany*(rwx), *ReadWriteOncePod*(RWOP)입니다.

`solidfire-san`스토리지 드라이버는 _FILE_AND_BLOCK_VOLUME 모드를 지원합니다. 볼륨 모드의 경우 `Filesystem` Trident는 볼륨을 생성하고 파일 시스템을 생성합니다. 파일 시스템 유형은 StorageClass에 의해 지정됩니다.

드라이버	프로토콜	볼륨 모드	액세스 모드가 지원됩니다	지원되는 파일 시스템
'솔더불-산'	iSCSI	블록	RWO, ROX, rwx, RWOP	파일 시스템이 없습니다. 원시 블록 장치.
'솔더불-산'	iSCSI	파일 시스템	RWO, 공화당	xfs, ext3, ext4

시작하기 전에

Element 백엔드를 생성하기 전에 다음이 필요합니다.

- Element 소프트웨어를 실행하는 지원되는 스토리지 시스템
- 볼륨을 관리할 수 있는 NetApp HCI/SolidFire 클러스터 관리자 또는 테넌트 사용자에 대한 자격 증명
- 모든 Kubernetes 작업자 노드에 적절한 iSCSI 툴이 설치되어 있어야 합니다. 을 참조하십시오 "작업자 노드 준비 정보".

백엔드 구성 옵션

백엔드 구성 옵션은 다음 표를 참조하십시오.

매개 변수	설명	기본값
'내전'		항상 1
'torageDriverName'입니다	스토리지 드라이버의 이름입니다	항상 "SolidFire-SAN"
백엔드이름	사용자 지정 이름 또는 스토리지 백엔드	"SolidFire_" + 스토리지(iSCSI) IP 주소입니다
끝점	테넌트 자격 증명이 있는 SolidFire 클러스터의 MVIP입니다	
'VIP'	스토리지(iSCSI) IP 주소 및 포트	
'라벨'	볼륨에 적용할 임의의 JSON 형식 레이블 세트입니다.	""
테난트이름	사용할 테넌트 이름(찾을 수 없는 경우 생성됨)	
이니토IFace	iSCSI 트래픽을 특정 호스트 인터페이스로 제한합니다	"기본값"
'UseCHAP'입니다	CHAP를 사용하여 iSCSI를 인증합니다. Trident는 CHAP를 사용합니다.	참
"액세스 그룹"	사용할 액세스 그룹 ID 목록입니다	"Trident"라는 액세스 그룹의 ID를 찾습니다.
'유형'	QoS 사양	

매개 변수	설명	기본값
LimitVolumeSize	요청된 볼륨 크기가 이 값보다 큰 경우 용량 할당에 실패합니다	""(기본적으로 적용되지 않음)
debugTraceFlags를 선택합니다	문제 해결 시 사용할 디버그 플래그입니다. 예: {"api":false, "method":true}	null입니다

문제 해결 및 자세한 로그 덤프가 필요한 경우가 아니면 debugTraceFlags를 사용하지 마십시오.

예 1: 에 대한 백엔드 구성 `solidfire-san` 세 가지 볼륨 유형을 가진 드라이버

이 예에서는 CHAP 인증을 사용하는 백엔드 파일을 보여 주고 특정 QoS 보장을 포함하는 세 가지 볼륨 유형을 모델링합니다. 그런 다음 "IOPS" 스토리지 클래스 매개 변수를 사용하여 각 스토리지 클래스를 사용할 스토리지 클래스를 정의할 가능성이 높습니다.

```
---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0
SVIP: <svip>:3260
TenantName: <tenant>
labels:
  k8scluster: dev1
  backend: dev1-element-cluster
UseCHAP: true
Types:
  - Type: Bronze
    Qos:
      minIOPS: 1000
      maxIOPS: 2000
      burstIOPS: 4000
  - Type: Silver
    Qos:
      minIOPS: 4000
      maxIOPS: 6000
      burstIOPS: 8000
  - Type: Gold
    Qos:
      minIOPS: 6000
      maxIOPS: 8000
      burstIOPS: 10000
```

예 2: 에 대한 백엔드 및 스토리지 클래스 구성 `solidfire-san` 가상 풀이 있는 드라이버

이 예에서는 가상 풀과 이를 다시 참조하는 StorageClasses와 함께 구성된 백엔드 정의 파일을 보여 줍니다.

Trident는 용량 할당 시 스토리지 풀에 있는 레이블을 백엔드 스토리지 LUN에 복제합니다. 편의를 위해 스토리지 관리자는 가상 풀 및 그룹 볼륨별로 레이블을 레이블별로 정의할 수 있습니다.

아래 표시된 샘플 백엔드 정의 파일에서 특정 기본값은 를 설정하는 모든 스토리지 풀에 대해 설정됩니다 `type` 실버. 가상 풀은 에 정의되어 있습니다 `storage` 섹션을 참조하십시오. 이 예에서는 일부 스토리지 풀이 자체 유형을 설정하고 일부 풀은 위에 설정된 기본값을 재정의합니다.

```
---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0
SVIP: <svip>:3260
TenantName: <tenant>
UseCHAP: true
Types:
  - Type: Bronze
    Qos:
      minIOPS: 1000
      maxIOPS: 2000
      burstIOPS: 4000
  - Type: Silver
    Qos:
      minIOPS: 4000
      maxIOPS: 6000
      burstIOPS: 8000
  - Type: Gold
    Qos:
      minIOPS: 6000
      maxIOPS: 8000
      burstIOPS: 10000
type: Silver
labels:
  store: solidfire
  k8scluster: dev-1-cluster
region: us-east-1
storage:
  - labels:
      performance: gold
      cost: "4"
    zone: us-east-1a
    type: Gold
  - labels:
      performance: silver
```

```
      cost: "3"
    zone: us-east-1b
    type: Silver
  - labels:
      performance: bronze
      cost: "2"
    zone: us-east-1c
    type: Bronze
  - labels:
      performance: silver
      cost: "1"
    zone: us-east-1d
```

다음 StorageClass 정의는 위의 가상 풀을 참조합니다. 를 사용합니다 `parameters.selector` 필드에서 각 StorageClass는 볼륨을 호스팅하는 데 사용할 수 있는 가상 풀을 호출합니다. 선택한 가상 풀에 볼륨이 정의되어 있습니다.

첫 번째 StorageClass(`solidfire-gold-four`)가 첫 번째 가상 풀에 매핑됩니다. 이 수영장은 금색 연주를 제공하는 유일한 수영장입니다. `Volume Type QoS` Last StorageClass(`solidfire-silver`)는 은색 성능을 제공하는 모든 스토리지 풀을 호출합니다. Trident는 어떤 가상 풀이 선택되었는지 결정하고 스토리지 요구 사항이 충족되는지 확인합니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-gold-four
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=gold; cost=4
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-three
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver; cost=3
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-bronze-two
```

```
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=bronze; cost=2
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-one
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver; cost=1
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver
  fsType: ext4
```

자세한 내용을 확인하십시오

- "볼륨 액세스 그룹"

ONTAP SAN 드라이버

ONTAP SAN 드라이버 개요

ONTAP 및 Cloud Volumes ONTAP SAN 드라이버를 사용하여 ONTAP 백엔드를 구성하는 방법에 대해 알아보십시오.

ONTAP SAN 드라이버 세부 정보입니다

Trident는 ONTAP 클러스터와 통신할 수 있도록 다음과 같은 SAN 스토리지 드라이버를 제공합니다. 지원되는 액세스 모드는 *ReadWriteOnce*(RWO), *ReadOnlyMany*(ROX), *ReadWriteMany*(rwx), *ReadWriteOncePod*(RWOP)입니다.

드라이버	프로토콜	볼륨 모드	액세스 모드가 지원됩니다	지원되는 파일 시스템
'ONTAP-SAN'	FC를 통한 iSCSI SCSI	블록	RWO, ROX, rwx, RWOP	파일 시스템이 없습니다. 원시 블록 디바이스입니다

드라이버	프로토콜	볼륨 모드	액세스 모드가 지원됩니다	지원되는 파일 시스템
'ONTAP-SAN'	FC를 통한 iSCSI SCSI	파일 시스템	RWO, 공화당 파일 시스템 볼륨 모드에서는 ROX 및 rwx를 사용할 수 없습니다.	xfs, ext3, ext4
'ONTAP-SAN'	NVMe/TCP 을 참조하십시오 NVMe/TCP에 대한 추가 고려사항.	블록	RWO, ROX, rwx, RWOP	파일 시스템이 없습니다. 원시 블록 디바이스입니다
'ONTAP-SAN'	NVMe/TCP 을 참조하십시오 NVMe/TCP에 대한 추가 고려사항.	파일 시스템	RWO, 공화당 파일 시스템 볼륨 모드에서는 ROX 및 rwx를 사용할 수 없습니다.	xfs, ext3, ext4
ONTAP-SAN-이코노미	iSCSI	블록	RWO, ROX, rwx, RWOP	파일 시스템이 없습니다. 원시 블록 디바이스입니다
ONTAP-SAN-이코노미	iSCSI	파일 시스템	RWO, 공화당 파일 시스템 볼륨 모드에서는 ROX 및 rwx를 사용할 수 없습니다.	xfs, ext3, ext4

- 사용 `ontap-san-economy` 영구 볼륨 사용 수가 보다 높을 것으로 예상되는 경우에만 "지원되는 ONTAP 볼륨 제한".
- 사용 `ontap-nas-economy` 영구 볼륨 사용 수가 보다 높을 것으로 예상되는 경우에만 "지원되는 ONTAP 볼륨 제한" 및 `ontap-san-economy` 드라이버를 사용할 수 없습니다.
- 사용하지 마십시오 `ontap-nas-economy` 데이터 보호, 재해 복구 또는 이동성이 필요할 것으로 예상되는 경우
- NetApp은 ONTAP-SAN을 제외한 모든 ONTAP 드라이버에서 FlexVol 자동 확장을 사용하지 않는 것이 좋습니다. 이 문제를 해결하려면 Trident에서 스냅숏 예비 공간 사용을 지원하고 그에 따라 FlexVol 볼륨의 크기를 조정합니다.

사용자 권한

Trident는 ONTAP 또는 SVM 관리자로 실행해야 하며, 일반적으로 클러스터 사용자 `vsadmin` 또는 SVM 사용자 또는 같은 역할을 가진 다른 이름의 사용자를 사용할 `admin` 것입니다. Amazon FSx for NetApp ONTAP 배포의 경우 Trident은 클러스터 사용자 또는 `vsadmin` SVM 사용자를 사용하여 ONTAP 또는 SVM 관리자로 실행하거나

fsxadmin 동일한 역할을 가진 다른 이름의 사용자를 실행해야 합니다. `fsxadmin`사용자는 클러스터 관리자를 제한적으로 대체합니다.

> limitAggregateUsage`매개 변수를 사용하려면 클러스터 관리 권한이 필요합니다. Trident와 함께 Amazon FSx for NetApp ONTAP를 사용할 때 `limitAggregateUsage 매개 변수는 및 fsxadmin 사용자 계정에서 작동하지 vsadmin 않습니다. 이 매개 변수를 지정하면 구성 작업이 실패합니다.

Trident 드라이버가 사용할 수 있는 더 제한적인 역할을 ONTAP 내에 만들 수 있지만 권장하지 않습니다. Trident의 대부분의 새로운 릴리즈에서는 추가 API를 호출하므로 업그레이드가 어렵고 오류가 발생하기 쉽습니다.

NVMe/TCP에 대한 추가 고려사항

Trident는 다음과 같은 드라이버를 사용하여 비휘발성 메모리 익스프레스(NVMe) 프로토콜을 ontap-san 지원합니다.

- IPv6
- NVMe 볼륨의 스냅샷 및 클론
- NVMe 볼륨 크기 조정
- Trident에서 라이프사이클을 관리할 수 있도록 Trident 외부에서 생성된 NVMe 볼륨을 가져옵니다
- NVMe 네이티브 다중 경로
- K8 노드의 정상 또는 비정상적으로 종료 (24.06)

Trident는 다음을 지원하지 않습니다.

- NVMe에서 기본적으로 지원되는 DH-HMAC-CHAP
- DM(Device Mapper) 경로 다중화
- LUKS 암호화

> NVMe는 ONTAP REST API에서만 지원되며 ONTAPI(ZAPI)에서는 지원되지 않습니다.

ONTAP SAN 드라이버를 사용하여 백엔드를 구성할 준비를 합니다

ONTAP SAN 드라이버를 사용하여 ONTAP 백엔드를 구성하기 위한 요구 사항 및 인증 옵션을 이해합니다.

요구 사항

모든 ONTAP 백엔드의 경우 Trident에서는 최소한 하나의 집계가 SVM에 할당되어야 합니다.

> "ASA r2 시스템"다른 ONTAP 시스템(ASA, AFF, FAS)과 저장 계층 구현 방식이 다릅니다. ASA r2 시스템에서는 집계 대신 스토리지 가용성 영역이 사용됩니다. 참조하다"여기" ASA r2 시스템에서 SVM에 집계를 할당하는 방법에 대한 지식 기반 문서입니다.

또한 둘 이상의 드라이버를 실행하고 둘 중 하나를 가리키는 스토리지 클래스를 생성할 수도 있습니다. 예를 들어, ONTAP-SAN 드라이버와 ONTAP-SAN-이코노미 클래스를 사용하는 '기본 클래스'를 사용하는 'san-dev' 클래스를 구성할 수 있습니다.

모든 Kubernetes 작업자 노드에는 적절한 iSCSI 툴이 설치되어 있어야 합니다. 을 참조하십시오 "작업자 노드를 준비합니다" 를 참조하십시오.

ONTAP 백엔드를 인증합니다

Trident는 ONTAP 백엔드를 인증하는 두 가지 모드를 제공합니다.

- 자격 증명 기반: 필요한 권한이 있는 ONTAP 사용자의 사용자 이름 및 암호입니다. ONTAP 버전과의 호환성을 최대한 보장하기 위해 admin 또는 vsadmin과 같은 미리 정의된 보안 로그인 역할을 사용하는 것이 좋습니다.
- 인증서 기반: Trident는 백엔드에 설치된 인증서를 사용하여 ONTAP 클러스터와 통신할 수도 있습니다. 이 경우 백엔드 정의에는 클라이언트 인증서, 키 및 사용할 경우 신뢰할 수 있는 CA 인증서의 Base64로 인코딩된 값이 있어야 합니다(권장).

자격 증명 기반 방법과 인증서 기반 방법 간에 이동하기 위해 기존 백엔드를 업데이트할 수 있습니다. 그러나 한 번에 하나의 인증 방법만 지원됩니다. 다른 인증 방법으로 전환하려면 백엔드 구성에서 기존 방법을 제거해야 합니다.

> 자격 증명과 인증서 * 를 모두 제공하려고 하면 구성 파일에 둘 이상의 인증 방법이 제공된다는 오류가 발생하여 백엔드 생성이 실패합니다.

자격 증명 기반 인증을 사용합니다

Trident은 ONTAP 백엔드와 통신하기 위해 SVM 범위/클러스터 범위 관리자에 대한 자격 증명이 필요합니다. 또는 `vsadmin` 과 같은 미리 정의된 표준 역할을 사용하는 것이 좋습니다 `admin`. 따라서 향후 Trident 릴리스에서 사용할 기능 API를 노출할 수 있는 향후 ONTAP 릴리즈와의 호환성이 보장됩니다. 사용자 지정 보안 로그인 역할을 만들어 Trident와 함께 사용할 수 있지만 권장하지는 않습니다.

백엔드 정의의 예는 다음과 같습니다.

YAML

```
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: password
```

JSON을 참조하십시오

```
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password"
}
```

백엔드 정의는 자격 증명이 일반 텍스트로 저장되는 유일한 위치라는 점에 유의하십시오. 백엔드가 생성된 후 사용자 이름/암호는 Base64로 인코딩되어 Kubernetes 암호로 저장됩니다. 백엔드의 생성 또는 업데이트는 자격 증명에 대한 지식이 필요한 유일한 단계입니다. 따라서 Kubernetes/스토리지 관리자가 수행할 수 있는 관리 전용 작업입니다.

인증서 기반 인증 활성화

신규 및 기존 백엔드는 인증서를 사용하여 ONTAP 백엔드와 통신할 수 있습니다. 백엔드 정의에는 세 가지 매개 변수가 필요합니다.

- clientCertificate: Base64로 인코딩된 클라이언트 인증서 값입니다.
- clientPrivateKey: Base64 - 연결된 개인 키의 인코딩된 값입니다.
- TrustedCACertificate: 신뢰할 수 있는 CA 인증서의 Base64 인코딩 값입니다. 신뢰할 수 있는 CA를 사용하는 경우 이 매개 변수를 제공해야 합니다. 신뢰할 수 있는 CA가 사용되지 않으면 이 작업을 무시할 수 있습니다.

일반적인 워크플로에는 다음 단계가 포함됩니다.

단계

1. 클라이언트 인증서 및 키를 생성합니다. 생성 시 CN(일반 이름)을 ONTAP 사용자로 설정하여 인증하십시오.

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=admin"
```

2. 신뢰할 수 있는 CA 인증서를 ONTAP 클러스터에 추가합니다. 이는 스토리지 관리자가 이미 처리한 것일 수 있습니다. 트러스트된 CA가 사용되지 않으면 무시합니다.

```
security certificate install -type server -cert-name <trusted-ca-cert-
name> -vserver <vserver-name>
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled
true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca
<cert-authority>
```

3. ONTAP 클러스터에 클라이언트 인증서 및 키(1단계)를 설치합니다.

```
security certificate install -type client-ca -cert-name <certificate-
name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

> ⓘ 이 명령을 실행하면 ONTAP에서 인증서 입력을 요청합니다. 1단계에서 생성된 `k8senv.pem` 파일의 내용을 붙여넣은 다음 `END`를 입력하여 설치를 완료하십시오.

4. ONTAP 보안 로그인 역할이 인증서 인증 방법을 지원하는지 확인합니다.

```
security login create -user-or-group-name admin -application ontapi
-authentication-method cert
security login create -user-or-group-name admin -application http
-authentication-method cert
```

5. 생성된 인증서를 사용하여 인증을 테스트합니다. ONTAP 관리 LIF> 및 <SVM 이름>을 관리 LIF IP 및 SVM 이름으로 바꿉니다.

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns="http://www.netapp.com/filer/admin" version="1.21"
vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. Base64로 인증서, 키 및 신뢰할 수 있는 CA 인증서를 인코딩합니다.

```
base64 -w 0 k8senv.pem >> cert_base64
base64 -w 0 k8senv.key >> key_base64
base64 -w 0 trustedca.pem >> trustedca_base64
```

7. 이전 단계에서 얻은 값을 사용하여 백엔드를 생성합니다.

```
cat cert-backend.json
{
"version": 1,
"storageDriverName": "ontap-san",
"backendName": "SanBackend",
"managementLIF": "1.2.3.4",
"svm": "vserver_test",
"clientCertificate": "Faaaakkkkeeee...Vaaalllluuuueeee",
"clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
"trustedCACertificate": "QNFinfO...SiqOyN",
"storagePrefix": "myPrefix_"
}

tridentctl create backend -f cert-backend.json -n trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
online |       0 |
+------------+----------------+--------------------------------------
+--------+---------+
```

인증 방법을 업데이트하거나 자격 증명을 회전합니다

다른 인증 방법을 사용하거나 자격 증명을 회전하도록 기존 백엔드를 업데이트할 수 있습니다. 이렇게 하면 사용자 이름/암호를 사용하는 백엔드를 인증서를 사용하도록 업데이트할 수 있고 인증서를 사용하는 백엔드는 사용자 이름/암호 기반으로 업데이트할 수 있습니다. 이렇게 하려면 기존 인증 방법을 제거하고 새 인증 방법을 추가해야 합니다. 그런 다음 필요한 매개 변수가 포함된 업데이트된 backend.json 파일을 사용하여 'tridentctl backend update'를 실행합니다.

```
cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-san",
"backendName": "SanBackend",
"managementLIF": "1.2.3.4",
"svm": "vserver_test",
"username": "vsadmin",
"password": "password",
"storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend SanBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE   | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

암호를 회전할 때 스토리지 관리자는 먼저 ONTAP에서 사용자의 암호를 업데이트해야 합니다. 그 다음에는 백엔드 업데이트가 있습니다. 인증서를 회전할 때 여러 인증서를 사용자에게 추가할 수 있습니다. 그런 다음 백엔드가 업데이트되어 새 인증서를 사용합니다. 그러면 ONTAP 클러스터에서 이전 인증서를 삭제할 수 있습니다.

백엔드를 업데이트해도 이미 생성된 볼륨에 대한 액세스가 중단되거나 이후에 생성된 볼륨 연결에 영향을 미치지 않습니다. 백엔드 업데이트에 성공하면 Trident가 ONTAP 백엔드와 통신하여 향후 볼륨 작업을 처리할 수 있음을 나타냅니다.

Trident에 대한 사용자 지정 ONTAP 역할을 생성합니다

Privileges에서 작업을 수행할 때 ONTAP 관리자 역할을 사용할 필요가 없도록 최소 Trident로 ONTAP 클러스터 역할을 생성할 수 있습니다. Trident 백엔드 구성에 사용자 이름을 포함하면 Trident은 사용자가 생성한 ONTAP 클러스터 역할을 사용하여 작업을 수행합니다.

Trident 사용자 지정 역할 생성에 대한 자세한 내용은 을 "Trident 사용자 지정 역할 생성기"참조하십시오.

ONTAP CLI 사용

1. 다음 명령을 사용하여 새 역할을 생성합니다.

```
security login role create <role_name\> -cmddirname "command" -access all
-vserver <svm_name\>
```

2. Trident 사용자에 대한 사용 이름 만들기:

```
security login create -username <user_name\> -application ontapi
-authmethod <password\> -role <name_of_role_in_step_1\> -vserver
<svm_name\> -comment "user_description"
```

3. 역할을 사용자에게 매핑:

```
security login modify username <user_name\> -vserver <svm_name\> -role
<role_name\> -application ontapi -application console -authmethod
<password\>
```

System Manager 사용

ONTAP System Manager에서 다음 단계를 수행하십시오.

1. * 사용자 지정 역할 생성 *:
   a. 클러스터 레벨에서 사용자 지정 역할을 생성하려면 * 클러스터 > 설정 * 을 선택합니다.

      SVM 레벨에서 사용자 지정 역할을 생성하려면 * 스토리지 > 스토리지 VM >> 설정 > 사용자 및 역할 * 을 선택합니다 `required SVM`.

   b. 사용자 및 역할 * 옆의 화살표 아이콘(*→ *)을 선택합니다.
   c. 역할 * 아래에서 * + 추가 * 를 선택합니다.
   d. 역할에 대한 규칙을 정의하고 * 저장 * 을 클릭합니다.
2. * 역할을 Trident 사용자에게 매핑 *: + * 사용자 및 역할 * 페이지에서 다음 단계를 수행하십시오.
   a. 사용자 * 아래에서 추가 아이콘 * + * 를 선택합니다.
   b. 필요한 사용자 이름을 선택하고 * Role * 에 대한 드롭다운 메뉴에서 역할을 선택합니다.
   c. 저장 * 을 클릭합니다.

자세한 내용은 다음 페이지를 참조하십시오.

- "ONTAP 관리를 위한 사용자 지정 역할" 또는 "사용자 지정 역할을 정의합니다"
- "역할 및 사용자 작업"

양방향 **CHAP**를 사용하여 연결 인증

Trident는 및 `ontap-san-economy` 드라이버에 대해 양방향 CHAP를 사용하여 iSCSI 세션을 인증할 수 `ontap-san` 있습니다. 이를 위해서는 백엔드 정의에서 옵션을 활성화해야 `useCHAP` 합니다. 로 `true` 설정하면 Trident는 SVM의 기본 이니시에이터 보안을 양방향 CHAP로 구성하고 백엔드 파일의 사용자 이름과 암호를 설정합니다. 양방향

CHAP를 사용하여 연결을 인증하는 것이 좋습니다. 다음 샘플 구성을 참조하십시오.

```
---
version: 1
storageDriverName: ontap-san
backendName: ontap_san_chap
managementLIF: 192.168.0.135
svm: ontap_iscsi_svm
useCHAP: true
username: vsadmin
password: password
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
```

useCHAP는 한 번만 설정할 수 있는 Boolean 옵션이다. 기본적으로 false로 설정되어 있습니다. true로 설정한 후에는 false 로 설정할 수 없습니다.

useCHAP=true 외에, chapInitatorSecret, chapTargetInitialatorSecret, chapchTargetUsername, chapUsername 필드가 백엔드 정의에 포함되어야 합니다. tridentctl update를 실행하여 백엔드를 생성한 후 비밀을 변경할 수 있다.

작동 원리

`useCHAP`true로 설정하면 스토리지 관리자가 Trident에 스토리지 백엔드에서 CHAP를 구성하도록 지시합니다. 여기에는 다음이 포함됩니다.

- SVM에서 CHAP 설정:
  - SVM의 기본 이니시에이터 보안 유형이 NONE(기본값 설정) * 이고 * 이미 존재하는 LUN이 볼륨에 없는 경우 Trident는 기본 보안 유형을 로 설정하고 CHAP 이니시에이터 및 타겟 사용자 이름과 암호를 구성합니다. CHAP
  - SVM에 LUN이 포함된 경우 Trident은 SVM에서 CHAP를 사용하도록 설정하지 않습니다. 따라서 SVM에 이미 있는 LUN에 대한 액세스가 제한되지 않습니다.
- CHAP 이니시에이터 및 타겟 사용자 이름과 암호를 구성합니다. 이러한 옵션은 백엔드 구성에 지정해야 합니다(위 참조).

백엔드가 생성된 후 Trident는 해당 tridentbackend CRD를 생성하고 CHAP 암호 및 사용자 이름을 Kubernetes 비밀로 저장합니다. 이 백엔드에서 Trident에 의해 생성된 모든 PVS가 CHAP를 통해 마운트되고 연결됩니다.

자격 증명을 순환하고 백엔드를 업데이트합니다.

backend.json 파일에서 CHAP 파라미터를 업데이트하여 CHAP 자격 증명을 업데이트할 수 있다. 이렇게 하려면 CHAP 암호를 업데이트하고 "tridentctl update" 명령을 사용하여 이러한 변경 사항을 반영해야 합니다.

백엔드의 CHAP 암호를 업데이트할 때 을 사용하여 백엔드를 업데이트해야 `tridentctl` 합니다. Trident은 이러한 변경 사항을 파악할 수 없으므로 ONTAP CLI 또는 ONTAP System Manager를 사용하여 스토리지 클러스터의 자격 증명을 업데이트하지 마십시오.

```
cat backend-san.json
{
    "version": 1,
    "storageDriverName": "ontap-san",
    "backendName": "ontap_san_chap",
    "managementLIF": "192.168.0.135",
    "svm": "ontap_iscsi_svm",
    "useCHAP": true,
    "username": "vsadmin",
    "password": "password",
    "chapInitiatorSecret": "cl9qxUpDaTeD",
    "chapTargetInitiatorSecret": "rqxigXgkeUpDaTeD",
    "chapTargetUsername": "iJF4heBRT0TCwxyz",
    "chapUsername": "uh2aNCLSd6cNwxyz",
}

./tridentctl update backend ontap_san_chap -f backend-san.json -n trident
+----------------+----------------+--------------------------------------
+--------+---------+
|   NAME         | STORAGE DRIVER |                 UUID                 |
STATE   | VOLUMES |
+----------------+----------------+--------------------------------------
+--------+---------+
| ontap_san_chap | ontap-san      | aa458f3b-ad2d-4378-8a33-1a472ffbeb5c |
online |       7 |
+----------------+----------------+--------------------------------------
+--------+---------+
```

기존 연결은 영향을 받지 않고 그대로 유지되며 SVM의 Trident가 자격 증명을 업데이트하는 경우 계속 활성 상태로 유지됩니다. 새 연결은 업데이트된 자격 증명을 사용하며 기존 연결은 계속 활성 상태를 유지합니다. 기존 PVS를 연결 해제하고 다시 연결하면 업데이트된 자격 증명을 사용하게 됩니다.

ONTAP SAN 구성 옵션 및 예

Trident 설치 시 ONTAP SAN 드라이버를 생성하고 사용하는 방법에 대해 알아봅니다. 이 섹션에서는 백엔드 구성 예제 및 Backend를 StorageClasses에 매핑하는 방법에 대한 세부 정보를 제공합니다.

"ASA r2 시스템"다른 ONTAP 시스템(ASA, AFF, FAS)과 저장 계층 구현 방식이 다릅니다. 이러한 변화는 표기된 특정 매개변수의 사용에 영향을 미칩니다. "ASA r2 시스템과 다른 ONTAP 시스템 간의 차이점에 대해 자세히 알아보세요.".

오직 `ontap-san` 드라이버(iSCSI, NVMe/TCP 및 FC 프로토콜 포함)는 ASA r2 시스템에서 지원됩니다.

Trident 백엔드 구성에서는 시스템이 ASA r2라고 지정할 필요가 없습니다. 선택할 때 `ontap-san` 로서 `storageDriverName` Trident ASA r2 또는 기타 ONTAP 시스템을 자동으로 감지합니다. 아래 표에 나와 있는 것처럼 일부 백엔드 구성 매개변수는 ASA r2 시스템에 적용할 수 없습니다.

백엔드 구성 옵션

백엔드 구성 옵션은 다음 표를 참조하십시오.

매개 변수	설명	기본값
'내전'		항상 1
'torageDriverName'입니다	스토리지 드라이버의 이름입니다	`ontap-san` 또는 `ontap-san-economy`
백엔드이름	사용자 지정 이름 또는 스토리지 백엔드	드라이버 이름 + "_" + dataLIF
마나멘타LIF	클러스터 또는 SVM 관리 LIF의 IP 주소입니다. FQDN(정규화된 도메인 이름)을 지정할 수 있습니다. IPv6 플래그를 사용하여 Trident가 설치된 경우 IPv6 주소를 사용하도록 설정할 수 있습니다. IPv6 주소는 과 같이 대괄호로 정의해야 `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` 합니다. 원활한 MetroCluster 전환은 를 MetroCluster 예참조하십시오. "vsadmin" 자격 증명을 사용하는 경우, 은 `managementLIF` SVM의 자격 증명이어야 합니다. "admin" 자격 증명을 사용하는 경우에는 이 클러스터의 자격 증명이어야 `managementLIF` 합니다.	"10.0.0.1","[2001:1234:ABCD::fefe]"
다타LIF	프로토콜 LIF의 IP 주소입니다. IPv6 플래그를 사용하여 Trident가 설치된 경우 IPv6 주소를 사용하도록 설정할 수 있습니다. IPv6 주소는 과 같이 대괄호로 정의해야 `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` 합니다. **iSCSI**에 대해 지정하지 마십시오. Trident는 을 사용하여 "ONTAP 선택적 LUN 맵"다중 경로 세션을 설정하는 데 필요한 iSCSI LIF를 검색합니다. 이 명시적으로 정의된 경우 경고가 `dataLIF` 생성됩니다. * MetroCluster의 경우 생략합니다. * 를 MetroCluster 예참조하십시오.	SVM에서 파생됩니다

매개 변수	설명	기본값
'VM'입니다	사용할 스토리지 가상 머신입니다 *MetroCluster의 경우 생략합니다. * 를 참조하십시오 MetroCluster 예.	SVM 'managementLIF'가 지정된 경우에 파생됩니다
'useCHAP'입니다	CHAP를 사용하여 ONTAP SAN 드라이버에 대한 iSCSI 인증 [Boolean]. 백엔드에 제공된 SVM에 대한 기본 인증으로 양방향 CHAP를 구성하고 사용하려면 Trident에 대해 으로 `true` 설정합니다. 자세한 내용은 을 "ONTAP SAN 드라이버를 사용하여 백엔드를 구성할 준비를 합니다" 참조하십시오. **FCP** 또는 **NVMe/TCP**에서는 지원되지 않습니다.	거짓입니다
챕터시크릿	CHAP 이니시에이터 암호입니다. useCHAP=true인 경우 필수입니다	""
'라벨'	볼륨에 적용할 임의의 JSON 형식 레이블 세트입니다	""
챕터타겟이니터시크릿	CHAP 타겟 이니시에이터 암호입니다. useCHAP=true인 경우 필수입니다	""
'chapUsername'입니다	인바운드 사용자 이름입니다. useCHAP=true인 경우 필수입니다	""
'chapTargetUsername'입니다	대상 사용자 이름입니다. useCHAP=true인 경우 필수입니다	""
'고객증명서'	Base64 - 클라이언트 인증서의 인코딩된 값입니다. 인증서 기반 인증에 사용됩니다	""
'clientPrivateKey'입니다	Base64 - 클라이언트 개인 키의 인코딩된 값입니다. 인증서 기반 인증에 사용됩니다	""
신탁인증서다	Base64 - 신뢰할 수 있는 CA 인증서의 인코딩된 값입니다. 선택 사항. 인증서 기반 인증에 사용됩니다.	""
'사용자 이름'	ONTAP 클러스터와 통신하려면 사용자 이름이 필요합니다. 자격 증명 기반 인증에 사용됩니다. Active Directory 인증에 대해서는 다음을 참조하세요. "Active Directory 자격 증명을 사용하여 백엔드 SVM에 Trident 인증".	""
"암호"	ONTAP 클러스터와 통신하려면 비밀번호가 필요합니다. 자격 증명 기반 인증에 사용됩니다. Active Directory 인증에 대해서는 다음을 참조하세요. "Active Directory 자격 증명을 사용하여 백엔드 SVM에 Trident 인증".	""
'VM'입니다	사용할 스토리지 가상 머신입니다	SVM 'managementLIF'가 지정된 경우에 파생됩니다
'토르agePrefix'	SVM에서 새 볼륨을 프로비저닝할 때 사용되는 접두사 나중에 수정할 수 없습니다. 이 매개 변수를 업데이트하려면 새 백엔드를 생성해야 합니다.	`trident`

매개 변수	설명	기본값
골재	프로비저닝을 위한 애그리게이트(선택 사항, SVM에 셋팅해야 하는 경우) 드라이버의 경우 `ontap-nas-flexgroup` 이 옵션은 무시됩니다. 할당되지 않은 경우 사용 가능한 애그리게이트를 사용하여 FlexGroup 볼륨을 프로비저닝할 수 있습니다. SVM에서 Aggregate를 업데이트하면 Trident 컨트롤러를 다시 시작하지 않고도 SVM을 폴링하여 Trident에서 자동으로 업데이트됩니다. 볼륨을 프로비저닝하기 위해 Trident의 특정 애그리게이트를 구성한 경우, 애그리게이트의 이름을 바꾸거나 SVM에서 이동할 경우 SVM 애그리게이트를 폴링하는 동안 백엔드가 Trident에서 오류 상태로 전환됩니다. Aggregate를 SVM에 있는 Aggregate로 변경하거나 완전히 제거하여 백엔드를 다시 온라인 상태로 전환해야 합니다. **ASA r2** 시스템에 대해서는 지정하지 마세요.	""
제한선태사용법	사용량이 이 비율을 초과하면 프로비저닝이 실패합니다. Amazon FSx for NetApp ONTAP 백엔드를 사용하는 경우 을 지정하지 `limitAggregateUsage`마십시오. 제공된 및 `vsadmin` 에는 `fsxadmin` 애그리게이트 사용량을 검색하고 Trident를 사용하여 제한하는 데 필요한 권한이 포함되어 있지 않습니다. **ASA r2** 시스템에 대해서는 지정하지 마세요.	""(기본적으로 적용되지 않음)
LimitVolumeSize	요청된 볼륨 크기가 이 값보다 큰 경우 용량 할당에 실패합니다. 또한 LUN에 대해 관리하는 볼륨의 최대 크기를 제한합니다.	""(기본적으로 적용되지 않음)
'오만유연한'	FlexVol당 최대 LUN 수는 범위[50, 200]에 있어야 합니다.	`100`
debugTraceFlags를 선택합니다	문제 해결 시 사용할 디버그 플래그입니다. 예: {"api":false, "method":true} 문제 해결 중이지 않고 자세한 로그 덤프가 필요한 경우가 아니면 사용하지 마십시오.	`null`

매개 변수	설명	기본값
'useREST'	ONTAP REST API를 사용하기 위한 부울 매개변수입니다. `useREST`설정 시 `true` , Trident 백엔드와 통신하기 위해 ONTAP REST API를 사용합니다. `false` Trident 백엔드와 통신하기 위해 ONTAPI(ZAPI) 호출을 사용합니다. 이 기능을 사용하려면 ONTAP 9.11.1 이상이 필요합니다. 또한, 사용되는 ONTAP 로그인 역할에는 다음에 대한 액세스 권한이 있어야 합니다. `ontapi` 애플리케이션. 이는 사전 정의된 것에 의해 충족됩니다. `vsadmin` 그리고 `cluster-admin` 역할. Trident 24.06 릴리스 및 ONTAP 9.15.1 이상부터 `useREST` 로 설정됩니다 `true` 기본적으로; 변경 `useREST` 에게 `false` ONTAPI(ZAPI) 호출을 사용합니다. `useREST`NVMe/TCP에 완벽하게 적합합니다. NVMe는 ONTAP REST API에서만 지원되며 ONTAPI(ZAPI)에서는 지원되지 않습니다. *지정된 경우 항상 다음으로 설정됩니다. true ASA r2 시스템*용.	true ONTAP 9.15.1 이상, 그렇지 않은 경우 false.
sanType	iSCSI, nvme NVMe/TCP 또는 fcp FC(SCSI over Fibre Channel)를 선택할 때 iscsi 사용합니다.	iscsi 비어 있는 경우
formatOptions	`formatOptions`볼륨을 포맷할 때마다 적용되는 명령에 대한 명령줄 인수를 지정하는 데 `mkfs` 사용합니다. 이렇게 하면 기본 설정에 따라 볼륨을 포맷할 수 있습니다. 장치 경로를 제외하고 mkfs 명령 옵션과 비슷한 formatOptions를 지정해야 합니다. 예: "-E NODEARD" 지원됨 **ontap-san** 그리고 **ontap-san-economy** **iSCSI** 프로토콜을 사용하는 드라이버. 또한 **iSCSI** 및 **NVMe/TCP** 프로토콜을 사용하는 **ASA r2** 시스템에서도 지원됩니다.	

매개 변수	설명	기본값
`limitVolumePoolSize`	ONTAP-SAN-Economy 백엔드에서 LUN을 사용할 때 요청될 수 있는 최대 FlexVol 크기입니다.	""(기본적으로 적용되지 않음)
`denyNewVolumePools`	백엔드가 LUN을 포함하도록 새 FlexVol 볼륨을 생성하지 못하도록 `ontap-san-economy` 제한합니다. 기존 FlexVol만 새 PVS 프로비저닝에 사용됩니다.	

포맷옵션 사용에 대한 권장 사항

Trident 포맷 과정을 신속하게 진행하기 위해 다음 옵션을 권장합니다.

- **-E nodiscard(ext3, ext4):** mkfs 시점에 블록을 삭제하려고 시도하지 마세요(처음에 블록을 삭제하는 것은 솔리드 스테이트 장치와 스파스/씬 프로비저닝 스토리지에서 유용합니다). 이는 더 이상 사용되지 않는 옵션인 "-K"를 대체하며 ext3, ext4 파일 시스템에 적용할 수 있습니다.
- **-K (xfs):** mkfs 시간에 블록을 버리려고 시도하지 마십시오. 이 옵션은 xfs 파일 시스템에 적용할 수 있습니다.

Active Directory 자격 증명을 사용하여 백엔드 **SVM**에 **Trident** 인증

Active Directory(AD) 자격 증명을 사용하여 백엔드 SVM에 인증하도록 Trident 구성할 수 있습니다. AD 계정이 SVM에 액세스하려면 먼저 클러스터 또는 SVM에 대한 AD 도메인 컨트롤러 액세스를 구성해야 합니다. AD 계정으로 클러스터를 관리하려면 도메인 터널을 만들어야 합니다. 참조하다 "ONTAP 에서 Active Directory 도메인 컨트롤러 액세스 구성" 자세한 내용은.

단계

1. 백엔드 SVM에 대한 DNS(도메인 이름 시스템) 설정을 구성합니다.

```
vserver services dns create -vserver <svm_name> -dns-servers
<dns_server_ip1>,<dns_server_ip2>
```

2. Active Directory에서 SVM에 대한 컴퓨터 계정을 만들려면 다음 명령을 실행하세요.

```
vserver active-directory create -vserver DataSVM -account-name ADSERVER1
-domain demo.netapp.com
```

3. 이 명령을 사용하여 클러스터 또는 SVM을 관리할 AD 사용자 또는 그룹을 만듭니다.

```
security login create -vserver <svm_name> -user-or-group-name
<ad_user_or_group> -application <application> -authentication-method domain
-role vsadmin
```

4. Trident 백엔드 구성 파일에서 다음을 설정합니다. `username` 그리고 `password` 각각 AD 사용자 또는 그룹 이름과 비밀번호에 대한 매개변수입니다.

볼륨 프로비저닝을 위한 백엔드 구성 옵션

에서 이러한 옵션을 사용하여 기본 프로비저닝을 제어할 수 있습니다 `defaults` 섹션을 참조하십시오. 예를 들어, 아래 구성 예제를 참조하십시오.

매개 변수	설명	기본값
'팩시배부	LUN에 대한 공간 할당	"true" *지정된 경우 설정됨 `true` ASA r2 시스템*용.
'예비공간'	공간 예약 모드, "없음"(씬) 또는 "볼륨"(일반). 설정 **none** **ASA r2** 시스템용.	"없음"
냅샷정책	사용할 스냅샷 정책입니다. *설정 `none` ASA r2 시스템* 용.	"없음"
"qosPolicy"	생성된 볼륨에 할당할 QoS 정책 그룹입니다. 스토리지 풀/백엔드에서 qosPolicy 또는 adapativeQosPolicy 중 하나를 선택합니다. Trident에서 QoS 정책 그룹을 사용하려면 ONTAP 9 .8 이상이 필요합니다. 비공유 QoS 정책 그룹을 사용하고 정책 그룹이 각 구성 요소에 개별적으로 적용되도록 해야 합니다. 공유 QoS 정책 그룹은 모든 워크로드의 총 처리량에 대한 제한을 적용합니다.	""
적응성 QosPolicy	생성된 볼륨에 할당할 적응형 QoS 정책 그룹입니다. 스토리지 풀/백엔드에서 qosPolicy 또는 adapativeQosPolicy 중 하나를 선택합니다	""
안산예비역	스냅숏용으로 예약된 볼륨의 비율입니다. **ASA r2** 시스템에 대해서는 지정하지 마세요.	"0"인 경우 `snapshotPolicy` "없음"이고, 그렇지 않으면""입니다.
'plitOnClone'을 선택합니다	생성 시 상위 클론에서 클론을 분할합니다	"거짓"
암호화	새 볼륨에서 NetApp 볼륨 암호화(NVE)를 활성화하고, 기본값은 로 설정합니다. `false` 이 옵션을 사용하려면 NVE 라이센스가 클러스터에서 활성화되어 있어야 합니다. 백엔드에서 NAE가 활성화된 경우 Trident에서 프로비저닝된 모든 볼륨은 NAE가 사용됩니다. 자세한 내용은 다음을 "Trident가 NVE 및 NAE와 작동하는 방법"참조하십시오.	"false" *지정된 경우 설정됨 `true` ASA r2 시스템*용.
`luksEncryption`	LUKS 암호화를 사용합니다. 을 "LUKS(Linux Unified Key Setup) 사용"참조하십시오.	"" *로 설정 `false` ASA r2 시스템*용.
'계층화 정책'	"없음"을 사용하는 계층화 정책 **ASA r2** 시스템에는 지정하지 마세요.	
`nameTemplate`	사용자 지정 볼륨 이름을 생성하는 템플릿입니다.	""

볼륨 프로비저닝의 예

다음은 기본값이 정의된 예입니다.

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: trident_svm
username: admin
password: <password>
labels:
  k8scluster: dev2
  backend: dev2-sanbackend
storagePrefix: alternate-trident
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: standard
  spaceAllocation: 'false'
  snapshotPolicy: default
  snapshotReserve: '10'
```

ⓘ 드라이버를 사용하여 생성된 모든 볼륨의 경우 `ontap-san` Trident는 LUN 메타데이터를 수용하기 위해 FlexVol에 10%의 용량을 추가합니다. LUN은 사용자가 PVC에서 요청하는 정확한 크기로 프로비저닝됩니다. Trident는 FlexVol에 10%를 추가합니다(ONTAP에서 사용 가능한 크기로 표시됨). 이제 사용자가 요청한 가용 용량을 얻을 수 있습니다. 또한 이 변경으로 인해 사용 가능한 공간이 완전히 활용되지 않는 한 LUN이 읽기 전용이 되는 것을 방지할 수 있습니다. ONTAP-SAN-경제에는 적용되지 않습니다.

을 정의하는 백엔드의 경우 `snapshotReserve` Trident는 다음과 같이 볼륨 크기를 계산합니다.

```
Total volume size = [(PVC requested size) / (1 - (snapshotReserve
percentage) / 100)] * 1.1
```

Trident FlexVol 에 추가하는 10%입니다. `snapshotReserve` = 5%, PVC 요청 = 5GiB인 경우 총 볼륨 크기는 5.79GiB이고 사용 가능한 크기는 5.5GiB입니다 . `volume show` 명령을 실행하면 다음 예와 비슷한 결과가 표시됩니다.

현재 기존 볼륨에 대해 새 계산을 사용하는 유일한 방법은 크기 조정입니다.

최소 구성의 예

다음 예에서는 대부분의 매개 변수를 기본값으로 두는 기본 구성을 보여 줍니다. 이는 백엔드를 정의하는 가장 쉬운 방법입니다.

> NetApp ONTAP on Trident와 함께 Amazon FSx를 사용하는 경우, NetApp은 IP 주소 대신 LIF에 대한 DNS 이름을 지정할 것을 권장합니다.

ONTAP SAN의 예

이것은 를 사용하는 기본 구성입니다 `ontap-san` 드라이버.

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
labels:
  k8scluster: test-cluster-1
  backend: testcluster1-sanbackend
username: vsadmin
password: <password>
```

MetroCluster 예

전환 및 전환 중에 백엔드 정의를 수동으로 업데이트할 필요가 없도록 백엔드를 구성할 수 있습니다 "SVM 복제 및 복구".

원활한 스위치오버 및 스위치백의 경우 를 사용하여 SVM을 지정하고 `managementLIF` 매개 변수를 생략합니다. `svm` 예를 들면 다음과 같습니다.

```
version: 1
storageDriverName: ontap-san
managementLIF: 192.168.1.66
username: vsadmin
password: password
```

ONTAP SAN 경제 예

```
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
username: vsadmin
password: <password>
```

인증서 기반 인증의 예

이 기본 구성 예에서 `clientCertificate`, `clientPrivateKey`, 및 `trustedCACertificate` (신뢰할 수 있는 CA를 사용하는 경우 선택 사항)는 에 채워집니다 `backend.json` 그리고 각각 클라이언트 인증서, 개인 키 및 신뢰할 수 있는 CA 인증서의 base64로 인코딩된 값을 사용합니다.

```
---
version: 1
storageDriverName: ontap-san
backendName: DefaultSANBackend
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
```

양방향 **CHAP** 예

이 예에서는 를 사용하여 백엔드를 생성합니다 `useCHAP` 를 로 설정합니다 `true`.

ONTAP SAN CHAP의 예

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
labels:
  k8scluster: test-cluster-1
  backend: testcluster1-sanbackend
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
username: vsadmin
password: <password>
```

ONTAP SAN 이코노미 **CHAP**의 예

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
username: vsadmin
password: <password>
```

NVMe/TCP 예

ONTAP 백엔드에서 NVMe로 구성된 SVM이 있어야 합니다. NVMe/TCP에 대한 기본 백엔드 구성입니다.

```
---
version: 1
backendName: NVMeBackend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_nvme
username: vsadmin
password: password
sanType: nvme
useREST: true
```

FCP(SCSI over FC) 예

ONTAP 백엔드에서 FC로 SVM을 구성해야 합니다. FC에 대한 기본 백엔드 구성입니다.

```
---
version: 1
backendName: fcp-backend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_fc
username: vsadmin
password: password
sanType: fcp
useREST: true
```

nameTemplate이 포함된 백엔드 구성 예

```
---
version: 1
storageDriverName: ontap-san
backendName: ontap-san-backend
managementLIF: <ip address>
svm: svm0
username: <admin>
password: <password>
defaults:
  nameTemplate:
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.vo\
    lume.RequestName}}"
labels:
  cluster: ClusterA
  PVC: "{{.volume.Namespace}}_{{.volume.RequestName}}"
```

ONTAP-SAN-이코노미 드라이버에 대한 옵션 예

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: ""
svm: svm1
username: ""
password: "!"
storagePrefix: whelk_
debugTraceFlags:
  method: true
  api: true
defaults:
  formatOptions: -E nodiscard
```

가상 풀의 백엔드 예

이러한 백엔드 정의 파일 샘플에서는 와 같은 모든 스토리지 풀에 대해 특정 기본값이 설정됩니다 `spaceReserve` 없음, `spaceAllocation` 거짓일 경우, 및 `encryption` 거짓일 때. 가상 풀은 스토리지 섹션에 정의됩니다.

Trident는 "Comments" 필드에 프로비저닝 레이블을 설정합니다. FlexVol volume Trident에 주석이 설정됨 용량 할당 시 가상 풀에 있는 모든 레이블을 스토리지 볼륨으로 복제합니다. 편의를 위해 스토리지 관리자는 가상 풀 및 그룹 볼륨별로 레이블을 레이블별로 정의할 수 있습니다.

이 예에서는 일부 스토리지 풀이 자체적으로 설정됩니다 `spaceReserve`, `spaceAllocation`, 및 `encryption` 일부 풀은 기본값을 재정의합니다.

ONTAP SAN의 예

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
username: vsadmin
password: <password>
defaults:
  spaceAllocation: "false"
  encryption: "false"
  qosPolicy: standard
labels:
  store: san_store
  kubernetes-cluster: prod-cluster-1
region: us_east_1
storage:
  - labels:
      protection: gold
      creditpoints: "40000"
    zone: us_east_1a
    defaults:
      spaceAllocation: "true"
      encryption: "true"
      adaptiveQosPolicy: adaptive-extreme
  - labels:
      protection: silver
      creditpoints: "20000"
    zone: us_east_1b
    defaults:
      spaceAllocation: "false"
      encryption: "true"
      qosPolicy: premium
  - labels:
      protection: bronze
      creditpoints: "5000"
    zone: us_east_1c
    defaults:
      spaceAllocation: "true"
      encryption: "false"
```

ONTAP SAN 경제 예

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
username: vsadmin
password: <password>
defaults:
  spaceAllocation: "false"
  encryption: "false"
labels:
  store: san_economy_store
region: us_east_1
storage:
  - labels:
      app: oracledb
      cost: "30"
    zone: us_east_1a
    defaults:
      spaceAllocation: "true"
      encryption: "true"
  - labels:
      app: postgresdb
      cost: "20"
    zone: us_east_1b
    defaults:
      spaceAllocation: "false"
      encryption: "true"
  - labels:
      app: mysqldb
      cost: "10"
    zone: us_east_1c
    defaults:
      spaceAllocation: "true"
      encryption: "false"
  - labels:
      department: legal
      creditpoints: "5000"
```

```
    zone: us_east_1c
    defaults:
      spaceAllocation: "true"
      encryption: "false"
```

NVMe/TCP 예

```
---
version: 1
storageDriverName: ontap-san
sanType: nvme
managementLIF: 10.0.0.1
svm: nvme_svm
username: vsadmin
password: <password>
useREST: true
defaults:
  spaceAllocation: "false"
  encryption: "true"
storage:
  - labels:
      app: testApp
      cost: "20"
    defaults:
      spaceAllocation: "false"
      encryption: "false"
```

백엔드를 **StorageClasses**에 매핑합니다

다음 StorageClass 정의는 을 참조하십시오 가상 풀의 백엔드 예. 를 사용합니다 `parameters.selector` 필드에서 각 StorageClass는 볼륨을 호스팅하는 데 사용할 수 있는 가상 풀을 호출합니다. 선택한 가상 풀에 볼륨이 정의되어 있습니다.

- 를 클릭합니다 `protection-gold` StorageClass는 의 첫 번째 가상 풀에 매핑됩니다 `ontap-san` 백엔드. 골드 레벨 보호 기능을 제공하는 유일한 풀입니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=gold"
  fsType: "ext4"
```

- 를 클릭합니다 `protection-not-gold` StorageClass는 의 두 번째 및 세 번째 가상 풀에 매핑됩니다 `ontap-san` 백엔드. 금 이외의 보호 수준을 제공하는 유일한 풀입니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
```

- 를 클릭합니다 `app-mysqldb` StorageClass는 의 세 번째 가상 풀에 매핑됩니다 `ontap-san-economy` 백엔드. mysqldb 유형 앱에 대한 스토리지 풀 구성을 제공하는 유일한 풀입니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
```

- 를 클릭합니다 `protection-silver-creditpoints-20k` StorageClass는 의 두 번째 가상 풀에 매핑됩니다 `ontap-san` 백엔드. 실버 레벨 보호 및 20,000포인트 적립을 제공하는 유일한 풀입니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
```

- 를 클릭합니다 `creditpoints-5k` StorageClass는 의 세 번째 가상 풀에 매핑됩니다 `ontap-san` 에 있는 백엔드 및 네 번째 가상 풀입니다 `ontap-san-economy` 백엔드. 5000 크레딧 포인트를 보유한 유일한 풀 서비스입니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: csi.trident.netapp.io
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"
```

- 를 클릭합니다 `my-test-app-sc` StorageClass 가 에 매핑됩니다 `testAPP` 의 가상 풀입니다 `ontap-san` 를 사용하여 운전합니다 `sanType: nvme`. 이것은 유일한 풀 제안입니다 `testApp`.

```
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: my-test-app-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=testApp"
  fsType: "ext4"
```

Trident는 어떤 가상 풀이 선택되었는지 결정하고 스토리지 요구 사항이 충족되는지 확인합니다.

ONTAP NAS 드라이버

ONTAP NAS 드라이버 개요

ONTAP 및 Cloud Volumes ONTAP NAS 드라이버를 사용하여 ONTAP 백엔드를 구성하는

방법에 대해 알아보십시오.

ONTAP NAS 드라이버 세부 정보입니다

Trident는 ONTAP 클러스터와 통신할 수 있도록 다음과 같은 NAS 스토리지 드라이버를 제공합니다. 지원되는 액세스 모드는 *ReadWriteOnce*(RWO), *ReadOnlyMany*(ROX), *ReadWriteMany*(rwx), *ReadWriteOncePod*(RWOP)입니다.

드라이버	프로토콜	볼륨 모드	액세스 모드가 지원됩니다	지원되는 파일 시스템
'ONTAP-NAS'	NFS 를 참조하십시오 중소기업	파일 시스템	RWO, ROX, rwx, RWOP	"", `nfs`, `smb`
ONTAP-NAS-이코노미	NFS 를 참조하십시오 중소기업	파일 시스템	RWO, ROX, rwx, RWOP	"", `nfs`, `smb`
'ONTAP-NAS-Flexgroup'	NFS 를 참조하십시오 중소기업	파일 시스템	RWO, ROX, rwx, RWOP	"", `nfs`, `smb`

- 사용 `ontap-san-economy` 영구 볼륨 사용 수가 보다 높을 것으로 예상되는 경우에만 "지원되는 ONTAP 볼륨 제한".
- 사용 `ontap-nas-economy` 영구 볼륨 사용 수가 보다 높을 것으로 예상되는 경우에만 "지원되는 ONTAP 볼륨 제한" 및 `ontap-san-economy` 드라이버를 사용할 수 없습니다.
- 사용하지 마십시오 `ontap-nas-economy` 데이터 보호, 재해 복구 또는 이동성이 필요할 것으로 예상되는 경우
- NetApp은 ONTAP-SAN을 제외한 모든 ONTAP 드라이버에서 FlexVol 자동 확장을 사용하지 않는 것이 좋습니다. 이 문제를 해결하려면 Trident에서 스냅숏 예비 공간 사용을 지원하고 그에 따라 FlexVol 볼륨의 크기를 조정합니다.

사용자 권한

Trident는 ONTAP 또는 SVM 관리자로 실행해야 하며, 일반적으로 클러스터 사용자 `vsadmin` 또는 SVM 사용자 또는 같은 역할을 가진 다른 이름의 사용자를 사용할 `admin` 것입니다.

Amazon FSx for NetApp ONTAP 배포의 경우 Trident은 클러스터 사용자 또는 `vsadmin` SVM 사용자를 사용하여 ONTAP 또는 SVM 관리자로 실행하거나 `fsxadmin` 동일한 역할을 가진 다른 이름의 사용자를 실행해야 합니다. `fsxadmin`사용자는 클러스터 관리자를 제한적으로 대체합니다.

`limitAggregateUsage`매개 변수를 사용하려면 클러스터 관리 권한이 필요합니다. Trident와 함께 `Amazon FSx for NetApp ONTAP`를 사용할 때 `limitAggregateUsage` 매개 변수는 및 `fsxadmin` 사용자 계정에서 작동하지 `vsadmin` 않습니다. 이 매개 변수를 지정하면 구성 작업이 실패합니다.

Trident 드라이버가 사용할 수 있는 더 제한적인 역할을 ONTAP 내에 만들 수 있지만 권장하지 않습니다. Trident의 대부분의 새로운 릴리즈에서는 추가 API를 호출하므로 업그레이드가 어렵고 오류가 발생하기 쉽습니다.

ONTAP NAS 드라이버를 사용하여 백엔드를 구성할 준비를 합니다

ONTAP NAS 드라이버를 사용하여 ONTAP 백엔드를 구성하기 위한 요구 사항, 인증 옵션 및 엑스포트 정책을 이해합니다.

25.10 릴리스부터 NetApp Trident 다음을 지원합니다."NetApp AFX 스토리지 시스템" . NetApp AFX 스토리지 시스템은 스토리지 계층 구현 측면에서 다른 ONTAP 시스템(ASA, AFF, FAS)과 다릅니다.

> 오직 `ontap-nas` 드라이버(NFS 프로토콜 포함)는 AFX 시스템에서 지원됩니다. SMB 프로토콜은 지원되지 않습니다.

Trident 백엔드 구성에서는 시스템이 AFX라고 지정할 필요가 없습니다. 선택할 때 `ontap-nas` 로서 `storageDriverName` Trident AFX 시스템을 자동으로 감지합니다.

요구 사항

- 모든 ONTAP 백엔드의 경우 Trident에서는 최소한 하나의 집계가 SVM에 할당되어야 합니다.
- 둘 이상의 드라이버를 실행하고 둘 중 하나를 가리키는 스토리지 클래스를 생성할 수 있습니다. 예를 들어, 을 사용하는 Gold 클래스를 구성할 수 있습니다 `ontap-nas` 드라이버 및 를 사용하는 Bronze 클래스 `ontap-nas-economy` 1개.
- 모든 Kubernetes 작업자 노드에 적절한 NFS 툴이 설치되어 있어야 합니다. 을 참조하십시오 "여기" 를 참조하십시오.
- Trident는 Windows 노드에서만 실행되는 Pod에 마운트된 SMB 볼륨을 지원합니다. 자세한 내용은 을 SMB 볼륨 프로비저닝을 위한 준비 참조하십시오.

ONTAP 백엔드를 인증합니다

Trident는 ONTAP 백엔드를 인증하는 두 가지 모드를 제공합니다.

- 자격 증명 기반: 이 모드에서는 ONTAP 백엔드에 대한 충분한 권한이 필요합니다. 과 같이 미리 정의된 보안 로그인 역할과 연결된 계정을 사용하는 것이 좋습니다 `admin` 또는 `vsadmin` ONTAP 버전과의 호환성을 최대한 보장하기 위해
- 인증서 기반: 이 모드에서는 Trident가 ONTAP 클러스터와 통신하기 위해 백엔드에 인증서가 설치되어 있어야 합니다. 이 경우 백엔드 정의에는 클라이언트 인증서, 키 및 사용할 경우 신뢰할 수 있는 CA 인증서의 Base64로 인코딩된 값이 있어야 합니다(권장).

자격 증명 기반 방법과 인증서 기반 방법 간에 이동하기 위해 기존 백엔드를 업데이트할 수 있습니다. 그러나 한 번에 하나의 인증 방법만 지원됩니다. 다른 인증 방법으로 전환하려면 백엔드 구성에서 기존 방법을 제거해야 합니다.

> 자격 증명과 인증서 * 를 모두 제공하려고 하면 구성 파일에 둘 이상의 인증 방법이 제공된다는 오류가 발생하여 백엔드 생성이 실패합니다.

자격 증명 기반 인증을 사용합니다

Trident은 ONTAP 백엔드와 통신하기 위해 SVM 범위/클러스터 범위 관리자에 대한 자격 증명이 필요합니다. 또는 `vsadmin` 과 같은 미리 정의된 표준 역할을 사용하는 것이 좋습니다 `admin`. 따라서 향후 Trident 릴리스에서 사용할 기능 API를 노출할 수 있는 향후 ONTAP 릴리즈와의 호환성이 보장됩니다. 사용자 지정 보안 로그인 역할을 만들어 Trident와 함께 사용할 수 있지만 권장하지는 않습니다.

백엔드 정의의 예는 다음과 같습니다.

YAML

```
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
credentials:
  name: secret-backend-creds
```

JSON을 참조하십시오

```
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "credentials": {
        "name": "secret-backend-creds"
    }
}
```

백엔드 정의는 자격 증명이 일반 텍스트로 저장되는 유일한 위치라는 점에 유의하십시오. 백엔드가 생성된 후 사용자 이름/암호는 Base64로 인코딩되어 Kubernetes 암호로 저장됩니다. 백엔드의 생성/업딩은 자격 증명에 대한 지식이 필요한 유일한 단계입니다. 따라서 Kubernetes/스토리지 관리자가 수행할 수 있는 관리 전용 작업입니다.

인증서 기반 인증을 사용합니다

신규 및 기존 백엔드는 인증서를 사용하여 ONTAP 백엔드와 통신할 수 있습니다. 백엔드 정의에는 세 가지 매개 변수가 필요합니다.

- clientCertificate: Base64로 인코딩된 클라이언트 인증서 값입니다.
- clientPrivateKey: Base64 - 연결된 개인 키의 인코딩된 값입니다.
- TrustedCACertificate: 신뢰할 수 있는 CA 인증서의 Base64 인코딩 값입니다. 신뢰할 수 있는 CA를 사용하는 경우 이 매개 변수를 제공해야 합니다. 신뢰할 수 있는 CA가 사용되지 않으면 이 작업을 무시할 수 있습니다.

일반적인 워크플로에는 다음 단계가 포함됩니다.

단계

1. 클라이언트 인증서 및 키를 생성합니다. 생성 시 CN(일반 이름)을 ONTAP 사용자로 설정하여 인증하십시오.

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=vsadmin"
```

2. 신뢰할 수 있는 CA 인증서를 ONTAP 클러스터에 추가합니다. 이는 스토리지 관리자가 이미 처리한 것일 수 있습니다. 트러스트된 CA가 사용되지 않으면 무시합니다.

```
security certificate install -type server -cert-name <trusted-ca-cert-
name> -vserver <vserver-name>
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled
true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca
<cert-authority>
```

3. ONTAP 클러스터에 클라이언트 인증서 및 키(1단계)를 설치합니다.

```
security certificate install -type client-ca -cert-name <certificate-
name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

4. ONTAP 보안 로그인 역할이 인증서 인증 방법을 지원하는지 확인합니다.

```
security login create -user-or-group-name vsadmin -application ontapi
-authentication-method cert -vserver <vserver-name>
security login create -user-or-group-name vsadmin -application http
-authentication-method cert -vserver <vserver-name>
```

5. 생성된 인증서를 사용하여 인증을 테스트합니다. ONTAP 관리 LIF> 및 <SVM 이름>을 관리 LIF IP 및 SVM 이름으로 바꿉니다. LIF의 서비스 정책이 'default-data-management'로 설정되어 있는지 확인해야 합니다.

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns="http://www.netapp.com/filer/admin" version="1.21"
vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. Base64로 인증서, 키 및 신뢰할 수 있는 CA 인증서를 인코딩합니다.

```
base64 -w 0 k8senv.pem >> cert_base64
base64 -w 0 k8senv.key >> key_base64
base64 -w 0 trustedca.pem >> trustedca_base64
```

7. 이전 단계에서 얻은 값을 사용하여 백엔드를 생성합니다.

```
cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "NasBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"clientCertificate": "Faaaakkkkeeee...Vaaalllluuuueeee",
"clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
"storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

인증 방법을 업데이트하거나 자격 증명을 회전합니다

다른 인증 방법을 사용하거나 자격 증명을 회전하도록 기존 백엔드를 업데이트할 수 있습니다. 이렇게 하면 사용자 이름/암호를 사용하는 백엔드를 인증서를 사용하도록 업데이트할 수 있고 인증서를 사용하는 백엔드는 사용자 이름/암호 기반으로 업데이트할 수 있습니다. 이렇게 하려면 기존 인증 방법을 제거하고 새 인증 방법을 추가해야 합니다. 그런 다음 실행할 필수 매개 변수가 포함된 업데이트된 backend.json 파일을 사용합니다 `tridentctl update backend`.

```
cat cert-backend-updated.json
```

```
{
"version": 1,
"storageDriverName": "ontap-nas",
"backendName": "NasBackend",
"managementLIF": "1.2.3.4",
"dataLIF": "1.2.3.8",
"svm": "vserver_test",
"username": "vsadmin",
"password": "password",
"storagePrefix": "myPrefix_"
}
```

```
#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
+------------+----------------+--------------------------------------
+--------+---------+
|    NAME    | STORAGE DRIVER |                 UUID                 |
STATE   | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

암호를 회전할 때 스토리지 관리자는 먼저 ONTAP에서 사용자의 암호를 업데이트해야 합니다. 그 다음에는 백엔드 업데이트가 있습니다. 인증서를 회전할 때 여러 인증서를 사용자에게 추가할 수 있습니다. 그런 다음 백엔드가 업데이트되어 새 인증서를 사용합니다. 그러면 ONTAP 클러스터에서 이전 인증서를 삭제할 수 있습니다.

백엔드를 업데이트해도 이미 생성된 볼륨에 대한 액세스가 중단되거나 이후에 생성된 볼륨 연결에 영향을 미치지 않습니다. 백엔드 업데이트에 성공하면 Trident가 ONTAP 백엔드와 통신하여 향후 볼륨 작업을 처리할 수 있음을 나타냅니다.

Trident에 대한 사용자 지정 ONTAP 역할을 생성합니다

Privileges에서 작업을 수행할 때 ONTAP 관리자 역할을 사용할 필요가 없도록 최소 Trident로 ONTAP 클러스터 역할을 생성할 수 있습니다. Trident 백엔드 구성에 사용자 이름을 포함하면 Trident은 사용자가 생성한 ONTAP 클러스터 역할을 사용하여 작업을 수행합니다.

Trident 사용자 지정 역할 생성에 대한 자세한 내용은 을 "Trident 사용자 지정 역할 생성기"참조하십시오.

ONTAP CLI 사용

1. 다음 명령을 사용하여 새 역할을 생성합니다.

```
security login role create <role_name\> -cmddirname "command" -access all
-vserver <svm_name\>
```

2. Trident 사용자에 대한 사용 이름 만들기:

```
security login create -username <user_name\> -application ontapi
-authmethod <password\> -role <name_of_role_in_step_1\> -vserver
<svm_name\> -comment "user_description"
```

3. 역할을 사용자에게 매핑:

```
security login modify username <user_name\> -vserver <svm_name\> -role
<role_name\> -application ontapi -application console -authmethod
<password\>
```

System Manager 사용

ONTAP System Manager에서 다음 단계를 수행하십시오.

1. * 사용자 지정 역할 생성 *:
   a. 클러스터 레벨에서 사용자 지정 역할을 생성하려면 * 클러스터 > 설정 * 을 선택합니다.

      SVM 레벨에서 사용자 지정 역할을 생성하려면 * 스토리지 > 스토리지 VM >> 설정 > 사용자 및 역할 * 을 선택합니다 `required SVM`.

   b. 사용자 및 역할 * 옆의 화살표 아이콘(*→ *)을 선택합니다.
   c. 역할 * 아래에서 * + 추가 * 를 선택합니다.
   d. 역할에 대한 규칙을 정의하고 * 저장 * 을 클릭합니다.
2. * 역할을 Trident 사용자에게 매핑 *: + * 사용자 및 역할 * 페이지에서 다음 단계를 수행하십시오.
   a. 사용자 * 아래에서 추가 아이콘 * + * 를 선택합니다.
   b. 필요한 사용자 이름을 선택하고 * Role * 에 대한 드롭다운 메뉴에서 역할을 선택합니다.
   c. 저장 * 을 클릭합니다.

자세한 내용은 다음 페이지를 참조하십시오.

- "ONTAP 관리를 위한 사용자 지정 역할" 또는 "사용자 지정 역할을 정의합니다"
- "역할 및 사용자 작업"

NFS 엑스포트 정책을 관리합니다

Trident는 NFS 엑스포트 정책을 사용하여 프로비저닝한 볼륨에 대한 액세스를 제어합니다.

Trident는 내보내기 정책을 사용할 때 두 가지 옵션을 제공합니다.

- Trident는 엑스포트 정책 자체를 동적으로 관리할 수 있습니다. 이 운영 모드에서 스토리지 관리자는 허용되는 IP 주소를 나타내는 CIDR 블록의 목록을 지정합니다. Trident는 이러한 범위에 속하는 적용 가능한 노드 IP를 게시 시 자동으로 내보내기 정책에 추가합니다. 또는 CIDR을 지정하지 않으면 게시되는 볼륨이 있는 노드에서 찾은 모든 글로벌 범위 유니캐스트 IP가 엑스포트 정책에 추가됩니다.
- 스토리지 관리자는 엑스포트 정책을 생성하고 규칙을 수동으로 추가할 수 있습니다. 구성에 다른 엑스포트 정책 이름을 지정하지 않는 한 Trident는 기본 엑스포트 정책을 사용합니다.

엑스포트 정책을 동적으로 관리

Trident는 ONTAP 백엔드에 대한 엑스포트 정책을 동적으로 관리하는 기능을 제공합니다. 따라서 스토리지 관리자는 명시적 규칙을 수동으로 정의하는 대신 작업자 노드 IP에 허용되는 주소 공간을 지정할 수 있습니다. 엑스포트 정책 관리를 크게 간소화하므로, 엑스포트 정책을 수정하면 더 이상 스토리지 클러스터에 대한 수동 작업이 필요하지 않습니다. 또한 이렇게 하면 볼륨을 마운트하고 지정된 범위 내에서 IP를 갖는 작업자 노드만 스토리지 클러스터에 대한 액세스를 제한하여 세분화된 자동 관리를 지원합니다.

> ⓘ 동적 내보내기 정책을 사용할 때는 NAT(Network Address Translation)를 사용하지 마십시오. NAT를 사용하면 스토리지 컨트롤러는 실제 IP 호스트 주소가 아니라 프런트엔드 NAT 주소를 인식하므로 내보내기 규칙에 일치하는 항목이 없으면 액세스가 거부됩니다.

예

두 가지 구성 옵션을 사용해야 합니다. 다음은 백엔드 정의의 예입니다.

```
---
version: 1
storageDriverName: ontap-nas-economy
backendName: ontap_nas_auto_export
managementLIF: 192.168.0.135
svm: svm1
username: vsadmin
password: password
autoExportCIDRs:
  - 192.168.0.0/24
autoExportPolicy: true
```

> ⓘ 이 기능을 사용할 때는 SVM의 루트 교차점에 노드 CIDR 블록(예: 기본 엑스포트 정책)을 허용하는 엑스포트 규칙과 함께 이전에 생성된 엑스포트 정책이 있는지 확인해야 합니다. Trident 전용 SVM을 사용하려면 항상 NetApp 권장 모범 사례를 따르십시오.

다음은 위의 예를 사용하여 이 기능이 작동하는 방식에 대한 설명입니다.

- `autoExportPolicy` 가 로 설정되어 `true` 있습니다. 이는 Trident이 SVM에 대해 이 백엔드로 프로비저닝된 각 볼륨에 대한 엑스포트 정책을 `svm1` 생성하고 주소 블록을 사용하여 규칙 추가 및 삭제를 `autoexportCIDRs` 처리합니다. 볼륨이 노드에 연결될 때까지 볼륨은 규칙 없이 빈 엑스포트 정책을 사용하여 볼륨에 대한 원치 않는 액세스를 차단합니다. 볼륨이 노드에 게시되면 Trident에서 지정된 CIDR 블록 내에 노드 IP를 포함하는 기본 qtree와 같은 이름의 엑스포트 정책을 생성합니다. 이러한 IP는 상위 FlexVol volume에서 사용하는 내보내기 정책에도 추가됩니다

◦ 예를 들면 다음과 같습니다.

- 백엔드 UUID 403b5326-8482-40dB-96d0-d83fb3f4daec
- `autoExportPolicy` 로 설정합니다 `true`
- 스토리지 접두사입니다 `trident`
- PVC UUID a79bcf5f-7b6d-4a40-9876-e2551f159c1c
- 이름이 Trident_PVC_a79bcf5f_7b6d_4a40_9876_e2551f159c1c인 FlexVol qtree에 대한 엑스포트 정책, 이름이 인 qtree에 대한 `trident-403b5326-8482-40db96d0-d83fb3f4daec` 엑스포트 정책, `trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c` SVM에 명명된 빈 엑스포트 정책을 `trident_empty` 생성합니다. FlexVol 엑스포트 정책 규칙은 qtree 엑스포트 정책에 포함된 모든 규칙의 상위 집합이 됩니다. 빈 내보내기 정책은 연결되지 않은 모든 볼륨에서 다시 사용됩니다.

- `autoExportCIDRs` 주소 블록 목록을 포함합니다. 이 필드는 선택 사항이며 기본적으로 ["0.0.0.0/0", ":/0"] 입니다. 정의되지 않은 경우 Trident는 작업자 노드에 있는 모든 전역 범위의 유니캐스트 주소를 게시물과 함께 추가합니다.

이 예에서는 `192.168.0.0/24` 주소 공간이 제공됩니다. 이는 발행물이 있는 이 주소 범위에 속하는 Kubernetes 노드 IP가 Trident에서 생성하는 엑스포트 정책에 추가된다는 것을 나타냅니다. Trident는 실행되는 노드를 등록할 때 노드의 IP 주소를 검색하여 에서 제공하는 주소 블록과 대조하여 확인합니다 `autoExportCIDRs`. 게시 시 IP를 필터링한 후 Trident는 게시 대상 노드의 클라이언트 IP에 대한 내보내기 정책 규칙을 만듭니다.

백엔드를 생성한 후 백엔드에 대한 자동 내보내기 정책 및 자동 내보내기 CIDR을 업데이트할 수 있습니다. 기존 CIDR을 자동으로 관리하거나 삭제하는 백엔드에 새 CIDR을 추가할 수 있습니다. CIDR을 삭제할 때는 기존 연결이 끊어지지 않도록 주의해야 합니다. 백엔드에 대해 'autoExportPolicy'를 사용하지 않도록 설정하고 수동으로 생성된 내보내기 정책으로 돌아갈 수도 있습니다. 이렇게 하려면 백엔드 구성에서 'exportPolicy' 매개 변수를 설정해야 합니다.

Trident에서 백엔드를 생성하거나 업데이트한 후 또는 해당 `tridentbackend` CRD를 사용하여 백엔드를 확인할 수 `tridentctl` 있습니다.

```
./tridentctl get backends ontap_nas_auto_export -n trident -o yaml
items:
- backendUUID: 403b5326-8482-40db-96d0-d83fb3f4daec
  config:
    aggregate: ""
    autoExportCIDRs:
    - 192.168.0.0/24
    autoExportPolicy: true
    backendName: ontap_nas_auto_export
    chapInitiatorSecret: ""
    chapTargetInitiatorSecret: ""
    chapTargetUsername: ""
    chapUsername: ""
    dataLIF: 192.168.0.135
    debug: false
    debugTraceFlags: null
    defaults:
      encryption: "false"
      exportPolicy: <automatic>
      fileSystemType: ext4
```

노드가 제거되면 Trident는 모든 엑스포트 정책을 확인하여 노드에 해당하는 액세스 규칙을 제거합니다. Trident는 관리되는 백엔드의 내보내기 정책에서 이 노드 IP를 제거하여 클러스터의 새 노드에서 이 IP를 재사용하지 않는 한 불량 마운트를 방지합니다.

기존 백엔드의 경우 백엔드를 로 업데이트하면 `tridentctl update backend` Trident에서 엑스포트 정책을 자동으로 관리할 수 있습니다. 이렇게 하면 필요한 경우 백엔드의 UUID 및 qtree 이름을 따서 명명된 두 개의 새 엑스포트 정책이 생성됩니다. 백엔드에 있는 볼륨은 마운트 해제했다가 다시 마운트하면 새로 생성된 엑스포트 정책을 사용합니다.

> 자동 관리되는 내보내기 정책이 있는 백엔드를 삭제하면 동적으로 생성된 내보내기 정책이 삭제됩니다. 백엔드가 다시 생성되면 백엔드가 새 백엔드로 처리되어 새 엑스포트 정책이 생성됩니다.

라이브 노드의 IP 주소가 업데이트되면 노드에서 Trident Pod를 다시 시작해야 합니다. 그런 다음 Trident는 이 IP 변경 사항을 반영하도록 관리하는 백엔드에 대한 내보내기 정책을 업데이트합니다.

SMB 볼륨 프로비저닝을 위한 준비

준비를 조금만 더 하면 를 사용하여 SMB 볼륨을 프로비저닝할 수 있습니다 `ontap-nas` 드라이버.

> ONTAP 온프레미스 클러스터를 위한 SMB 볼륨을 생성하려면 SVM에서 NFS 및 SMB/CIFS 프로토콜을 모두 구성해야 `ontap-nas-economy` 합니다. 이 두 프로토콜 중 하나를 구성하지 않으면 SMB 볼륨 생성에 실패합니다.

> `autoExportPolicy` SMB 볼륨에는 가 지원되지 않습니다.

시작하기 전에

SMB 볼륨을 프로비저닝하려면 먼저 다음 항목이 있어야 합니다.

- Linux 컨트롤러 노드 및 Windows Server 2022를 실행하는 Windows 작업자 노드가 있는 Kubernetes 클러스터 Trident는 Windows 노드에서만 실행되는 Pod에 마운트된 SMB 볼륨을 지원합니다.
- Active Directory 자격 증명이 포함된 Trident 암호가 하나 이상 있습니다. 비밀 생성하기 `smbcreds`:

```
kubectl create secret generic smbcreds --from-literal username=user
--from-literal password='password'
```

- Windows 서비스로 구성된 CSI 프록시. 를 구성합니다 `csi-proxy`를 참조하십시오 "GitHub:CSI 프록시" 또는 "GitHub: Windows용 CSI 프록시" Windows에서 실행되는 Kubernetes 노드의 경우:

단계

1. 온프레미스 ONTAP의 경우 선택적으로 SMB 공유를 생성하거나 Trident에서 공유를 생성할 수 있습니다.

   ⓘ ONTAP용 Amazon FSx에는 SMB 공유가 필요합니다.

   다음 두 가지 방법 중 하나로 SMB 관리자 공유를 생성할 수 있습니다 "Microsoft 관리 콘솔" 공유 폴더 스냅인 또는 ONTAP CLI 사용 ONTAP CLI를 사용하여 SMB 공유를 생성하려면 다음을 따르십시오.

   a. 필요한 경우 공유에 대한 디렉토리 경로 구조를 생성합니다.

      를 클릭합니다 `vserver cifs share create` 명령은 공유를 생성하는 동안 -path 옵션에 지정된 경로를 확인합니다. 지정한 경로가 없으면 명령이 실패합니다.

   b. 지정된 SVM과 연결된 SMB 공유를 생성합니다.

```
vserver cifs share create -vserver vserver_name -share-name
share_name -path path [-share-properties share_properties,...]
[other_attributes] [-comment text]
```

   c. 공유가 생성되었는지 확인합니다.

```
vserver cifs share show -share-name share_name
```

   ⓘ 을 참조하십시오 "SMB 공유를 생성합니다" 를 참조하십시오.

2. 백엔드를 생성할 때 SMB 볼륨을 지정하려면 다음을 구성해야 합니다. 모든 ONTAP 백엔드 구성 옵션에 대한 자세한 내용은 을 참조하십시오 "ONTAP 구성 옵션 및 예제용 FSX".

매개 변수	설명	예
smbShare	Microsoft 관리 콘솔 또는 ONTAP CLI를 사용하여 생성된 SMB 공유의 이름, Trident에서 SMB 공유를 생성할 수 있는 이름, 볼륨에 대한 일반적인 공유 액세스를 방지하기 위해 매개 변수를 비워 둘 수 있습니다. 이 매개 변수는 사내 ONTAP의 경우 선택 사항입니다. 이 매개변수는 ONTAP 백엔드에 대한 아마존 FSx에 필요하며 비워둘 수 없습니다.	smb-share
nasType	* 를 로 설정해야 합니다 smb. * null인 경우 기본값은 로 설정됩니다 nfs.	smb
'생태성 스타일'을 참조하십시오	새로운 볼륨에 대한 보안 스타일 * 를 로 설정해야 합니다 ntfs 또는 mixed SMB 볼륨용. *	ntfs 또는 mixed SMB 볼륨용
유니크권한	모드를 선택합니다. SMB 볼륨에 대해서는 * 를 비워 두어야 합니다. *	""

보안 **SMB** 활성화

25.06 릴리스부터 NetApp Trident는 다음을 사용하여 생성된 SMB 볼륨의 보안 프로비저닝을 지원합니다. ontap-nas 그리고 ontap-nas-economy 백엔드. 보안 SMB가 활성화되면 액세스 제어 목록(ACL)을 사용하여 Active Directory(AD) 사용자 및 사용자 그룹의 SMB 공유에 대한 제어된 액세스를 제공할 수 있습니다.

기억해야 할 사항

- 수입 ontap-nas-economy 볼륨은 지원되지 않습니다.
- 읽기 전용 복제본만 지원됩니다. ontap-nas-economy 볼륨.
- Secure SMB가 활성화된 경우 Trident는 백엔드에 언급된 SMB 공유를 무시합니다.
- PVC 주석, 스토리지 클래스 주석 및 백엔드 필드를 업데이트해도 SMB 공유 ACL은 업데이트되지 않습니다.
- 복제 PVC의 주석에 지정된 SMB 공유 ACL은 소스 PVC의 ACL보다 우선합니다.
- 보안 SMB를 활성화하는 동안 유효한 AD 사용자를 제공해야 합니다. 유효하지 않은 사용자는 ACL에 추가되지 않습니다.
- 백엔드, 스토리지 클래스, PVC에서 동일한 AD 사용자에게 서로 다른 권한을 제공하는 경우 권한 우선순위는 PVC, 스토리지 클래스, 백엔드 순입니다.
- 보안 SMB가 지원됩니다. ontap-nas 관리되는 볼륨 가져오기에는 적용되며 관리되지 않는 볼륨 가져오기에는 적용되지 않습니다.

단계

1. 다음 예와 같이 TridentBackendConfig에 adAdminUser를 지정합니다.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.193.176.x
  svm: svm0
  useREST: true
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

2. 저장 클래스에 주석을 추가합니다.

   추가하다 `trident.netapp.io/smbShareAdUser` 보안 SMB를 오류 없이 사용할 수 있도록 스토리지 클래스에 주석을 추가합니다. 주석에 지정된 사용자 값 `trident.netapp.io/smbShareAdUser` 사용자 이름에 지정된 것과 동일해야 합니다. `smbcreds` 비밀입니다. 다음 중 하나를 선택할 수 있습니다. `smbShareAdUserPermission` : `full_control` , `change` , 또는 `read` . `full_control` .

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

1. PVC을 생성합니다.

   다음 예제에서는 PVC를 생성합니다.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/snapshotDirectory: "true"
    trident.netapp.io/smbShareAccessControl: |
      read:
        - tridentADtest
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

ONTAP NAS 구성 옵션 및 예

Trident 설치 시 ONTAP NAS 드라이버를 생성하고 사용하는 방법에 대해 알아봅니다. 이 섹션에서는 백엔드 구성 예제 및 Backend를 StorageClasses에 매핑하는 방법에 대한 세부 정보를 제공합니다.

25.10 릴리스부터 NetApp Trident 다음을 지원합니다."NetApp AFX 스토리지 시스템" . NetApp AFX 스토리지 시스템은 스토리지 계층 구현 측면에서 다른 ONTAP 기반 시스템(ASA, AFF, FAS)과 다릅니다.

> 오직 `ontap-nas` 드라이버(NFS 프로토콜 포함)는 NetApp AFX 시스템에서 지원됩니다. SMB 프로토콜은 지원되지 않습니다.

Trident 백엔드 구성에서는 시스템이 NetApp AFX 스토리지 시스템임을 지정할 필요가 없습니다. 선택할 때 `ontap-nas` 로서 `storageDriverName` Trident AFX 저장 시스템을 자동으로 감지합니다. 아래 표에 나와 있는 것처럼 일부 백엔드 구성 매개변수는 AFX 스토리지 시스템에 적용할 수 없습니다.

백엔드 구성 옵션

백엔드 구성 옵션은 다음 표를 참조하십시오.

매개 변수	설명	기본값
'내전'		항상 1
'torageDriverName'입니다	스토리지 드라이버의 이름입니다 NetApp AFX 시스템의 경우에만 `ontap-nas` 지원됩니다.	`ontap-nas` `ontap-nas-economy`, 또는 `ontap-nas-flexgroup`

매개 변수	설명	기본값
백엔드이름	사용자 지정 이름 또는 스토리지 백엔드	드라이버 이름 + "_" + dataLIF
마나멘타LIF	클러스터 또는 SVM 관리 LIF의 IP 주소 정규화된 도메인 이름(FQDN)을 지정할 수 있습니다. IPv6 플래그를 사용하여 Trident가 설치된 경우 IPv6 주소를 사용하도록 설정할 수 있습니다. IPv6 주소는 과 같이 대괄호로 정의해야 `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` 합니다. 원활한 MetroCluster 전환은 를 MetroCluster 예참조하십시오.	"10.0.0.1"," [2001:1234:ABCD::fefe]"
다타LIF	프로토콜 LIF의 IP 주소입니다. NetApp에서는 을 지정할 것을 `dataLIF` 권장합니다. 제공되지 않는 경우 Trident는 SVM에서 데이터 LIF를 가져옵니다. NFS 마운트 작업에 사용할 FQDN(정규화된 도메인 이름)을 지정할 수 있습니다. 이렇게 하면 라운드 로빈 DNS를 생성하여 여러 데이터 LIF의 로드 밸런싱을 수행할 수 있습니다. 초기 설정 후에 변경할 수 있습니다. 을 참조하십시오. IPv6 플래그를 사용하여 Trident가 설치된 경우 IPv6 주소를 사용하도록 설정할 수 있습니다. IPv6 주소는 과 같이 대괄호로 정의해야 `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]` 합니다. * MetroCluster의 경우 생략합니다. * 를 MetroCluster 예참조하십시오.	지정되지 않은 경우 SVM에서 지정 주소 또는 파생(권장하지 않음)
'VM'입니다	사용할 스토리지 가상 머신입니다 *MetroCluster의 경우 생략합니다. * 를 참조하십시오 MetroCluster 예.	SVM 'managementLIF'가 지정된 경우에 파생됩니다
자동 내보내기 정책	자동 엑스포트 정책 생성 및 업데이트 [Boolean] 활성화 Trident는 및 `autoExportCIDRs` 옵션을 사용하여 `autoExportPolicy` 엑스포트 정책을 자동으로 관리할 수 있습니다.	거짓
자동 내보내기	이 설정된 경우에 대해 Kubernetes 노드 IP를 필터링하는 CIDR `autoExportPolicy` 목록입니다. Trident는 및 `autoExportCIDRs` 옵션을 사용하여 `autoExportPolicy` 엑스포트 정책을 자동으로 관리할 수 있습니다.	["0.0.0.0/0",":/0"]'
'라벨'	볼륨에 적용할 임의의 JSON 형식 레이블 세트입니다	""
'고객증명서'	Base64 - 클라이언트 인증서의 인코딩된 값입니다. 인증서 기반 인증에 사용됩니다	""
'clientPrivateKey' 입니다	Base64 - 클라이언트 개인 키의 인코딩된 값입니다. 인증서 기반 인증에 사용됩니다	""
신탁인증서다	Base64 - 신뢰할 수 있는 CA 인증서의 인코딩된 값입니다. 선택 사항. 인증서 기반 인증에 사용됩니다	""
'사용자 이름'	클러스터/SVM에 연결할 사용자 이름입니다. 자격 증명 기반 인증에 사용됩니다. Active Directory 인증에 대해서는 다음을 참조하세요. "Active Directory 자격 증명을 사용하여 백엔드 SVM에 Trident 인증".	

매개 변수	설명	기본값
"암호"	클러스터/SVM에 연결하기 위한 비밀번호입니다. 자격 증명 기반 인증에 사용됩니다. Active Directory 인증에 대해서는 다음을 참조하세요. "Active Directory 자격 증명을 사용하여 백엔드 SVM에 Trident 인증".	
'토르agePrefix'	SVM에서 새 볼륨을 프로비저닝할 때 사용되는 접두사 설정한 후에는 업데이트할 수 없습니다 24자 이상인 ONTAP-nas-Economy와 storagePrefix를 사용할 경우 볼륨 이름에 포함되기는 하지만 qtree에 스토리지 접두사가 포함되지 않습니다.	"트리덴트"
골재	프로비저닝을 위한 애그리게이트(선택 사항, SVM에 셋팅해야 하는 경우) 드라이버의 경우 `ontap-nas-flexgroup` 이 옵션은 무시됩니다. 할당되지 않은 경우 사용 가능한 애그리게이트를 사용하여 FlexGroup 볼륨을 프로비저닝할 수 있습니다. SVM에서 Aggregate를 업데이트하면 Trident 컨트롤러를 다시 시작하지 않고도 SVM을 폴링하여 Trident에서 자동으로 업데이트됩니다. 볼륨을 프로비저닝하기 위해 Trident의 특정 애그리게이트를 구성한 경우, 애그리게이트의 이름을 바꾸거나 SVM에서 이동할 경우 SVM 애그리게이트를 폴링하는 동안 백엔드가 Trident에서 오류 상태로 전환됩니다. Aggregate를 SVM에 있는 Aggregate로 변경하거나 완전히 제거하여 백엔드를 다시 온라인 상태로 전환해야 합니다. **AFX** 저장 시스템에 대해서는 지정하지 마세요.	""
제한선태사용법	사용량이 이 백분율을 초과하면 프로비저닝에 실패합니다. * Amazon FSx for ONTAP 에는 적용되지 않습니다*. **AFX** 저장 시스템에 대해서는 지정하지 마세요.	""(기본적으로 적용되지 않음)

매개 변수	설명	기본값
flexgroupAggregateList 를 참조하십시오	프로비저닝을 위한 애그리게이트 목록(선택 사항, SVM에 할당되어야 함) SVM에 할당된 모든 애그리게이트는 FlexGroup 볼륨을 프로비저닝하는 데 사용됩니다. ONTAP-NAS-FlexGroup * 스토리지 드라이버에 대해 지원됩니다. SVM에서 애그리게이트 목록이 업데이트되면 Trident 컨트롤러를 다시 시작하지 않고도 SVM을 폴링하여 Trident에서 목록이 자동으로 업데이트됩니다. 볼륨을 프로비저닝하도록 Trident의 특정 애그리게이트 목록을 구성한 경우, 애그리게이트 목록의 이름을 바꾸거나 SVM에서 이동하면 SVM 애그리게이트를 폴링하는 동안 백엔드가 Trident에서 오류 상태로 전환됩니다. 애그리게이트 목록을 SVM에 있는 목록으로 변경하거나 완전히 제거하여 백엔드를 다시 온라인 상태로 전환해야 합니다.	""
LimitVolumeSize	요청된 볼륨 크기가 이 값보다 크면 프로비저닝에 실패합니다.	""(기본적으로 적용되지 않음)
debugTraceFlags를 선택합니다	문제 해결 시 사용할 디버그 플래그입니다. 예: {"api":false, "method":true} 사용하지 마십시오 `debugTraceFlags` 문제 해결 및 자세한 로그 덤프가 필요한 경우를 제외하고	null입니다
`nasType`	NFS 또는 SMB 볼륨 생성을 구성합니다. 옵션은 다음과 같습니다 `nfs` , `smb` 또는 null. null로 설정하면 기본적으로 NFS 볼륨이 사용됩니다. *지정된 경우 항상 다음으로 설정됩니다. `nfs` AFX 저장 시스템*용.	`nfs`
nfsMountOptions를 선택합니다	쉼표로 구분된 NFS 마운트 옵션 목록입니다. Kubernetes 영구 볼륨의 마운트 옵션은 일반적으로 스토리지 클래스에 지정되어 있지만, 스토리지 클래스에 마운트 옵션이 지정되지 않은 경우 Trident는 스토리지 백엔드의 구성 파일에 지정된 마운트 옵션을 사용하도록 폴백합니다. 스토리지 클래스 또는 구성 파일에 마운트 옵션이 지정되지 않은 경우 Trident는 연결된 영구 볼륨에 마운트 옵션을 설정하지 않습니다.	""
"케트리스퍼플랙스볼륨"	FlexVol당 최대 qtree, 범위 [50, 300]에 있어야 함	"200"

매개 변수	설명	기본값
smbShare	Microsoft 관리 콘솔 또는 ONTAP CLI를 사용하여 생성된 SMB 공유의 이름, Trident에서 SMB 공유를 생성할 수 있는 이름, 볼륨에 대한 일반적인 공유 액세스를 방지하기 위해 매개 변수를 비워 둘 수 있습니다. 이 매개 변수는 사내 ONTAP의 경우 선택 사항입니다. 이 매개변수는 ONTAP 백엔드에 대한 아마존 FSx에 필요하며 비워둘 수 없습니다.	smb-share
'useREST'	ONTAP REST API를 사용하기 위한 부울 매개변수입니다. useREST`설정 시 `true , Trident 백엔드와 통신하기 위해 ONTAP REST API를 사용합니다. false Trident 백엔드와 통신하기 위해 ONTAPI(ZAPI) 호출을 사용합니다. 이 기능을 사용하려면 ONTAP 9.11.1 이상이 필요합니다. 또한, 사용되는 ONTAP 로그인 역할에는 다음에 대한 액세스 권한이 있어야 합니다. ontapi 애플리케이션. 이는 사전 정의된 것에 의해 충족됩니다. vsadmin 그리고 cluster-admin 역할. Trident 24.06 릴리스 및 ONTAP 9.15.1 이상부터 useREST 로 설정됩니다 true 기본적으로; 변경 useREST 에게 false ONTAPI(ZAPI) 호출을 사용합니다. *지정된 경우 항상 다음으로 설정됩니다. true AFX 저장 시스템*용.	true ONTAP 9.15.1 이상, 그렇지 않은 경우 false.
limitVolumePoolSize	ONTAP-NAS-이코노미 백엔드에서 Qtree를 사용할 때 가장 필요한 최대 FlexVol 크기입니다.	""(기본적으로 적용되지 않음)
denyNewVolumePools	못하도록백 엔드가 새 FlexVol 볼륨을 생성하지 포함하도록 ontap-nas-economy 해당 qtree를 제한합니다. 기존 FlexVol만 새 PVS 프로비저닝에 사용됩니다.	
adAdminUser	SMB 공유에 대한 전체 액세스 권한을 가진 Active Directory 관리자 사용자 또는 사용자 그룹입니다. 이 매개변수를 사용하여 SMB 공유에 대한 모든 권한을 가진 관리자 권한을 부여할 수 있습니다.	

볼륨 프로비저닝을 위한 백엔드 구성 옵션

에서 이러한 옵션을 사용하여 기본 프로비저닝을 제어할 수 있습니다 defaults 섹션을 참조하십시오. 예를 들어, 아래 구성 예제를 참조하십시오.

매개 변수	설명	기본값
'팩시배부	Qtree에 공간 할당	"참"
'예비공간'	공간 예약 모드, "없음"(씬) 또는 "볼륨"(일반)	"없음"
냅샷정책	사용할 스냅샷 정책입니다	"없음"
"qosPolicy"	생성된 볼륨에 할당할 QoS 정책 그룹입니다. 스토리지 풀/백엔드에서 qosPolicy 또는 adapativeQosPolicy 중 하나를 선택합니다	""

매개 변수	설명	기본값
적응성 QosPolicy	생성된 볼륨에 할당할 적응형 QoS 정책 그룹입니다. 스토리지 풀/백엔드에서 qosPolicy 또는 adapativeQosPolicy 중 하나를 선택합니다. ONTAP에서 지원되지 않음 - NAS - 이코노미	""
안산예비역	스냅숏용으로 예약된 볼륨의 백분율입니다	"0"인 경우 `snapshotPolicy` "없음"이고, 그렇지 않으면""입니다.
'plitOnClone'을 선택합니다	생성 시 상위 클론에서 클론을 분할합니다	"거짓"
암호화	새 볼륨에서 NetApp 볼륨 암호화(NVE)를 활성화하고, 기본값은 로 설정합니다. `false` 이 옵션을 사용하려면 NVE 라이센스가 클러스터에서 활성화되어 있어야 합니다. 백엔드에서 NAE가 활성화된 경우 Trident에서 프로비저닝된 모든 볼륨은 NAE가 사용됩니다. 자세한 내용은 다음을 "Trident가 NVE 및 NAE와 작동하는 방법"참조하십시오.	"거짓"
'계층화 정책'	"없음"을 사용하는 계층화 정책	
유니크권한	모드를 선택합니다	NFS 볼륨의 경우 "777", SMB 볼륨의 경우 비어 있음(해당 없음
나프산디렉토리	에 액세스를 제어합니다 `.snapshot` 디렉토리	NFSv3의 경우 NFSv4의 경우 "true"입니다
엑포트정책	사용할 엑스포트 정책	"기본값"
'생태성 스타일'을 참조하십시오	새로운 볼륨에 대한 보안 스타일 NFS를 지원합니다 `mixed` 및 `unix` 보안 스타일. SMB 지원 `mixed` 및 `ntfs` 보안 스타일.	NFS 기본값은 입니다 `unix`. SMB 기본값은 입니다 `ntfs`.
`nameTemplate`	사용자 지정 볼륨 이름을 생성하는 템플릿입니다.	""

Trident에서 QoS 정책 그룹을 사용하려면 ONTAP 9 .8 이상이 필요합니다. 비공유 QoS 정책 그룹을 사용하고 정책 그룹이 각 구성 요소에 개별적으로 적용되도록 해야 합니다. 공유 QoS 정책 그룹은 모든 워크로드의 총 처리량에 대한 제한을 적용합니다.

볼륨 프로비저닝의 예

다음은 기본값이 정의된 예입니다.

```
---
version: 1
storageDriverName: ontap-nas
backendName: customBackendName
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
labels:
  k8scluster: dev1
  backend: dev1-nasbackend
svm: trident_svm
username: cluster-admin
password: <password>
limitAggregateUsage: 80%
limitVolumeSize: 50Gi
nfsMountOptions: nfsvers=4
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: premium
  exportPolicy: myk8scluster
  snapshotPolicy: default
  snapshotReserve: "10"
```

을 위한 `ontap-nas` 그리고 `ontap-nas-flexgroups` Trident 이제 FlexVol snapshotReserve 백분율과 PVC에 맞게 올바르게 조정되도록 새로운 계산을 사용합니다. 사용자가 PVC를 요청하면 Trident 새로운 계산을 사용하여 더 많은 공간을 가진 원래 FlexVol 생성합니다. 이 계산은 사용자가 PVC에서 요청한 쓰기 가능 공간을 받고, 요청한 것보다 적은 공간을 받지 않도록 보장합니다. v21.07 이전에는 사용자가 스냅샷 예약 비율을 50%로 설정한 PVC(예: 5GiB)를 요청하면 2.5GiB의 쓰기 가능 공간만 확보되었습니다. 이는 사용자가 요청한 것이 전체 볼륨이기 때문입니다. `snapshotReserve` 그 중 일부입니다. Trident 21.07을 사용하면 사용자가 요청하는 것은 쓰기 가능한 공간이며 Trident 이를 정의합니다. `snapshotReserve` 전체 볼륨에 대한 백분율로 나타낸 숫자입니다. 이것은 적용되지 않습니다 `ontap-nas-economy` . 작동 방식을 알아보려면 다음 예를 참조하세요.

계산은 다음과 같습니다.

```
Total volume size = <PVC requested size> / (1 - (<snapshotReserve
percentage> / 100))
```

에서 요청한 5GiB입니다. `volume show` 명령을 실행하면 다음 예와 비슷한 결과가 표시됩니다.

이전 설치의 기존 백엔드는 Trident 업그레이드 시 위에서 설명한 대로 볼륨을 프로비저닝합니다. 업그레이드 전에 생성한 볼륨의 경우, 변경 사항을 적용하려면 볼륨 크기를 조정해야 합니다. 예를 들어, 2GiB PVC의 경우 `snapshotReserve=50` 이전에는 1GiB의 쓰기 가능 공간을 제공하는 볼륨이 생성되었습니다. 예를 들어 볼륨 크기를 3GiB로 조정하면 애플리케이션은 6GiB 볼륨에서 3GiB의 쓰기 가능 공간을 확보하게 됩니다.

최소 구성의 예

다음 예에서는 대부분의 매개 변수를 기본값으로 두는 기본 구성을 보여 줍니다. 이는 백엔드를 정의하는 가장 쉬운 방법입니다.

> Trident가 있는 NetApp ONTAP에서 Amazon FSx를 사용하는 경우 IP 주소 대신 LIF에 대한 DNS 이름을 지정하는 것이 좋습니다.

ONTAP NAS 경제도 예

```
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

ONTAP NAS FlexGroup 예

```
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

MetroCluster 예

전환 및 전환 중에 백엔드 정의를 수동으로 업데이트할 필요가 없도록 백엔드를 구성할 수 있습니다 "SVM 복제 및 복구".

원활한 스위치오버 및 스위치백의 경우 를 사용하여 SVM을 지정합니다 `managementLIF` 를 생략합니다 `dataLIF` 및 `svm` 매개 변수. 예를 들면 다음과 같습니다.

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 192.168.1.66
username: vsadmin
password: password
```

SMB 볼륨의 예입니다

```
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
nasType: smb
securityStyle: ntfs
unixPermissions: ""
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password
```

인증서 기반 인증의 예

이는 최소 백엔드 구성의 예입니다. `clientCertificate`, `clientPrivateKey`, 및 `trustedCACertificate` (신뢰할 수 있는 CA를 사용하는 경우 선택 사항)는 에 채워집니다 `backend.json` 그리고 각각 클라이언트 인증서, 개인 키 및 신뢰할 수 있는 CA 인증서의 base64로 인코딩된 값을 사용합니다.

```
---
version: 1
backendName: DefaultNASBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.15
svm: nfs_svm
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

자동 엑스포트 정책의 예

이 예에서는 Trident에서 동적 엑스포트 정책을 사용하여 엑스포트 정책을 자동으로 생성 및 관리하도록 하는 방법을 보여 줍니다. 및 `ontap-nas-flexgroup` 드라이버에도 동일하게 `ontap-nas-economy` 작동합니다.

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
labels:
  k8scluster: test-cluster-east-1a
  backend: test1-nasbackend
autoExportPolicy: true
autoExportCIDRs:
- 10.0.0.0/24
username: admin
password: password
nfsMountOptions: nfsvers=4
```

IPv6 주소 예

이 예에서는 를 보여 줍니다 `managementLIF` IPv6 주소 사용.

```
---
version: 1
storageDriverName: ontap-nas
backendName: nas_ipv6_backend
managementLIF: "[5c5d:5edf:8f:7657:bef8:109b:1b41:d491]"
labels:
  k8scluster: test-cluster-east-1a
  backend: test1-ontap-ipv6
svm: nas_ipv6_svm
username: vsadmin
password: password
```

SMB 볼륨을 사용하는 ONTAP용 Amazon FSx의 예

를 클릭합니다 `smbShare` SMB 볼륨을 사용하는 ONTAP용 FSx에 매개 변수가 필요합니다.

```
---
version: 1
backendName: SMBBackend
storageDriverName: ontap-nas
managementLIF: example.mgmt.fqdn.aws.com
nasType: smb
dataLIF: 10.0.0.15
svm: nfs_svm
smbShare: smb-share
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

nameTemplate이 포함된 백엔드 구성 예

```
---
version: 1
storageDriverName: ontap-nas
backendName: ontap-nas-backend
managementLIF: <ip address>
svm: svm0
username: <admin>
password: <password>
defaults:
  nameTemplate:
"{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.vo\
    lume.RequestName}}"
labels:
  cluster: ClusterA
  PVC: "{{.volume.Namespace}}_{{.volume.RequestName}}"
```

가상 풀의 백엔드 예

아래 표시된 샘플 백엔드 정의 파일에서 와 같은 모든 스토리지 풀에 대한 특정 기본값이 설정됩니다 `spaceReserve` 없음, `spaceAllocation` 거짓일 경우, 및 `encryption` 거짓일 때. 가상 풀은 스토리지 섹션에 정의됩니다.

Trident는 "Comments" 필드에 프로비저닝 레이블을 설정합니다. 설명은 의 FlexVol `ontap-nas` 또는 의 FlexGroup에 `ontap-nas-flexgroup` 설정됩니다. Trident는 프로비저닝 시 가상 풀에 있는 모든 레이블을 스토리지 볼륨에 복제합니다. 편의를 위해 스토리지 관리자는 가상 풀 및 그룹 볼륨별로 레이블을 레이블별로 정의할 수 있습니다.

이 예에서는 일부 스토리지 풀이 자체적으로 설정됩니다 `spaceReserve`, `spaceAllocation`, 및 `encryption` 일부 풀은 기본값을 재정의합니다.

ONTAP NAS의 예

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
svm: svm_nfs
username: admin
password: <password>
nfsMountOptions: nfsvers=4
defaults:
  spaceReserve: none
  encryption: "false"
  qosPolicy: standard
labels:
  store: nas_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
  - labels:
      app: msoffice
      cost: "100"
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: "true"
      unixPermissions: "0755"
      adaptiveQosPolicy: adaptive-premium
  - labels:
      app: slack
      cost: "75"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      department: legal
      creditpoints: "5000"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
```

```
      app: wordpress
      cost: "50"
    zone: us_east_1c
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0775"
  - labels:
      app: mysqldb
      cost: "25"
    zone: us_east_1d
    defaults:
      spaceReserve: volume
      encryption: "false"
      unixPermissions: "0775"
```

ONTAP NAS FlexGroup의 예

```
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: <password>
defaults:
  spaceReserve: none
  encryption: "false"
labels:
  store: flexgroup_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
  - labels:
      protection: gold
      creditpoints: "50000"
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      protection: gold
      creditpoints: "30000"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      protection: silver
      creditpoints: "20000"
    zone: us_east_1c
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0775"
  - labels:
      protection: bronze
      creditpoints: "10000"
    zone: us_east_1d
```

```
    defaults:
      spaceReserve: volume
      encryption: "false"
      unixPermissions: "0775"
```

ONTAP NAS 경제도 예

```
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: <password>
defaults:
  spaceReserve: none
  encryption: "false"
labels:
  store: nas_economy_store
region: us_east_1
storage:
  - labels:
      department: finance
      creditpoints: "6000"
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      protection: bronze
      creditpoints: "5000"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
      department: engineering
      creditpoints: "3000"
    zone: us_east_1c
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0775"
  - labels:
      department: humanresource
      creditpoints: "2000"
    zone: us_east_1d
    defaults:
```

```
      spaceReserve: volume
      encryption: "false"
      unixPermissions: "0775"
```

백엔드를 **StorageClasses**에 매핑합니다

다음 StorageClass 정의는 을 참조하십시오 가상 풀의 백엔드 예. 를 사용합니다 `parameters.selector` 필드에서 각 StorageClass는 볼륨을 호스팅하는 데 사용할 수 있는 가상 풀을 호출합니다. 선택한 가상 풀에 볼륨이 정의되어 있습니다.

- 를 클릭합니다 `protection-gold` StorageClass는 의 첫 번째 및 두 번째 가상 풀에 매핑됩니다 `ontap-nas-flexgroup` 백엔드. 골드 레벨 보호 기능을 제공하는 유일한 풀입니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=gold"
  fsType: "ext4"
```

- 를 클릭합니다 `protection-not-gold` StorageClass는 의 세 번째 및 네 번째 가상 풀에 매핑됩니다 `ontap-nas-flexgroup` 백엔드. 금 이외의 보호 수준을 제공하는 유일한 풀입니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
```

- 를 클릭합니다 `app-mysqldb` StorageClass는 의 네 번째 가상 풀에 매핑됩니다 `ontap-nas` 백엔드. mysqldb 유형 앱에 대한 스토리지 풀 구성을 제공하는 유일한 풀입니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
```

- 를 누릅니다 `protection-silver-creditpoints-20k` StorageClass는 의 세 번째 가상 풀에 매핑됩니다 `ontap-nas-flexgroup` 백엔드. 실버 레벨 보호 및 20,000포인트 적립을 제공하는 유일한 풀입니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
```

- 를 클릭합니다 `creditpoints-5k` StorageClass는 의 세 번째 가상 풀에 매핑됩니다 `ontap-nas` 의 백엔드 및 두 번째 가상 풀입니다 `ontap-nas-economy` 백엔드. 5000 크레딧 포인트를 보유한 유일한 풀 서비스입니다.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: csi.trident.netapp.io
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"
```

Trident는 어떤 가상 풀이 선택되었는지 결정하고 스토리지 요구 사항이 충족되는지 확인합니다.

업데이트 `dataLIF` 초기 구성 후

초기 구성 후 다음 명령을 실행하여 새 백엔드 JSON 파일에 업데이트된 데이터 LIF를 제공할 수 있습니다.

```
tridentctl update backend <backend-name> -f <path-to-backend-json-file-
with-updated-dataLIF>
```

PVC가 하나 또는 여러 개의 Pod에 연결된 경우, 새로운 데이터 LIF가 적용되려면 해당하는 모든 Pod를 종료한 다음 다시 불러와야 합니다.

보안 SMB 예시

ontap-nas 드라이버를 사용한 백엔드 구성

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

ontap-nas-economy 드라이버를 사용한 백엔드 구성

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas-economy
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

스토리지 풀을 사용한 백엔드 구성

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm0
  useREST: false
  storage:
  - labels:
      app: msoffice
    defaults:
      adAdminUser: tridentADuser
  nasType: smb
  credentials:
    name: backend-tbc-ontap-invest-secret
```

ontap-nas 드라이버를 사용한 스토리지 클래스 예제

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADtest
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

추가했는지 확인하세요 `annotations` 보안 SMB를 활성화합니다. 보안 SMB는 백엔드 또는 PVC에 설정된 구성과 관계없이 주석 없이는 작동하지 않습니다.

ontap-nas-economy 드라이버를 사용한 스토리지 클래스 예제

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser3
parameters:
  backendType: ontap-nas-economy
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

단일 **AD** 사용자가 있는 **PVC** 예

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      change:
        - tridentADtest
      read:
        - tridentADuser
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

여러 **AD** 사용자가 있는 **PVC** 예

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-test-pvc
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      full_control:
        - tridentTestuser
        - tridentuser
        - tridentTestuser1
        - tridentuser1
      change:
        - tridentADuser
        - tridentADuser1
        - tridentADuser4
        - tridentTestuser2
      read:
        - tridentTestuser2
        - tridentTestuser3
        - tridentADuser2
        - tridentADuser3
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

NetApp ONTAP용 Amazon FSx

Trident를 Amazon FSx for NetApp ONTAP와 함께 사용해 보십시오

"NetApp ONTAP용 Amazon FSx" NetApp ONTAP 스토리지 운영 체제가 제공하는 파일 시스템을 실행하고 실행할 수 있도록 완벽하게 관리되는 AWS 서비스입니다. ONTAP용 FSX를 사용하면 익숙한 NetApp 기능, 성능 및 관리 기능을 활용하는 동시에, AWS에 데이터를 저장하는 데 따른 단순성, 민첩성, 보안, 확장성을 활용할 수 있습니다. ONTAP용 FSX는 ONTAP 파일 시스템 기능 및 관리 API를 지원합니다.

Amazon FSx for NetApp ONTAP 파일 시스템을 Trident와 통합하여 Amazon EKS(Elastic Kubernetes Service)에서 실행되는 Kubernetes 클러스터가 ONTAP에서 지원하는 블록 및 파일 영구 볼륨을 프로비저닝할 수 있도록 할 수 있습니다.

파일 시스템은 Amazon FSx의 주요 리소스이며, 이는 사내 ONTAP 클러스터와 유사합니다. 각 SVM 내에서 파일 시스템에 파일과 폴더를 저장하는 데이터 컨테이너인 하나 이상의 볼륨을 생성할 수 있습니다. Amazon FSx for NetApp ONTAP은 클라우드에서 관리형 파일 시스템으로 제공됩니다. 새로운 파일 시스템 유형을 * NetApp ONTAP *

라고 합니다.

Trident를 Amazon FSx for NetApp ONTAP와 함께 사용하면 Amazon EKS(Elastic Kubernetes Service)에서 실행되는 Kubernetes 클러스터가 ONTAP에서 지원하는 블록 및 파일 영구 볼륨을 프로비저닝할 수 있습니다.

요구 사항

"Trident 요구사항"FSx for ONTAP를 Trident와 통합하려면 다음이 필요합니다.

- kubbtl이 설치된 기존 Amazon EKS 클러스터 또는 자체 관리 Kubernetes 클러스터
- 클러스터의 작업자 노드에서 연결할 수 있는 NetApp ONTAP 파일 시스템용 기존 Amazon FSx 및 SVM(Storage Virtual Machine).
- 에 대해 준비된 작업자 노드입니다 "NFS 또는 iSCSI".

> Amazon Linux 및 Ubuntu에 필요한 노드 준비 단계를 따라야 합니다 "Amazon Machine Images(아마존 머신 이미지)" (AMI) EKS AMI 유형에 따라 다릅니다.

고려 사항

- SMB 볼륨:
  - SMB 볼륨은 를 사용하여 지원됩니다 `ontap-nas` 드라이버만 해당.
  - Trident EKS 애드온에서는 SMB 볼륨이 지원되지 않습니다.
  - Trident는 Windows 노드에서만 실행되는 Pod에 마운트된 SMB 볼륨을 지원합니다. 자세한 내용은 을 "SMB 볼륨 프로비저닝을 위한 준비" 참조하십시오.
- Trident 24.02 이전에는 자동 백업이 활성화된 Amazon FSx 파일 시스템에서 생성된 볼륨을 Trident에서 삭제할 수 없었습니다. Trident 24.02 이상에서 이 문제를 방지하려면 `fsxFilesystemID` AWS FSx for ONTAP의 백엔드 구성 파일에, AWS, `apikey` AWS `apiRegion` 및 AWS를 `secretKey` 지정합니다.

> Trident에 IAM 역할을 지정하는 경우, `apiKey` 및 `secretKey` 필드를 명시적으로 Trident에 지정하지 않아도 됩니다 `apiRegion`. 자세한 내용은 을 "ONTAP 구성 옵션 및 예제용 FSX"참조하십시오.

Trident SAN/iSCSI 및 EBS-CSI 드라이버 동시 사용

AWS(EKS, ROSA, EC2 또는 기타 인스턴스)에서 ontap-san 드라이버(예: iSCSI)를 사용하려는 경우 노드에 필요한 다중 경로 구성이 Amazon Elastic Block Store(EBS) CSI 드라이버와 충돌할 수 있습니다. 동일한 노드에 있는 EBS 디스크를 방해하지 않고 멀티패스 기능을 보장하려면 멀티패스 설정에서 EBS를 제외해야 합니다. 이 예에서는 다음을 보여줍니다. `multipath.conf` EBS 디스크를 다중 경로에서 제외하면서 필수 Trident 설정을 포함하는 파일:

```
defaults {
    find_multipaths no
}
blacklist {
    device {
        vendor "NVME"
        product "Amazon Elastic Block Store"
    }
}
```

인증

Trident는 두 가지 인증 모드를 제공합니다.

- 자격 증명 기반(권장): 자격 증명을 AWS Secrets Manager에 안전하게 저장합니다. 파일 시스템 또는 SVM에 구성된 사용자를 사용할 수 있습니다 `fsxadmin` `vsadmin` .

> Trident은 SVM 사용자로 실행하거나 동일한 역할을 가진 다른 이름의 사용자로 실행해야 `vsadmin` 합니다. Amazon FSx for NetApp ONTAP에는 `fsxadmin` ONTAP 클러스터 사용자를 제한적으로 대체하는 사용자가 `admin` 있습니다. Trident와 함께 를 사용하는 `vsadmin` 것이 좋습니다.

- 인증서 기반: Trident은 SVM에 설치된 인증서를 사용하여 FSx 파일 시스템의 SVM과 통신합니다.

인증 활성화에 대한 자세한 내용은 드라이버 유형에 대한 인증을 참조하십시오.

- "ONTAP NAS 인증"
- "ONTAP SAN 인증"

테스트된 아마존 머신 이미지**(AMI)**

EKS 클러스터는 다양한 운영 체제를 지원하지만 AWS는 컨테이너 및 EKS에 대해 특정 AMI(Amazon Machine Images)를 최적화했습니다. 다음 AMI는 NetApp Trident 25.02에서 테스트되었습니다.

아미	NAS	NAS - 경제형	iSCSI	iSCSI 경제
AL2023_x86_64_STANDARD를 참조하십시오	예	예	예	예
AL2_x86_64를 참조하십시오	예	예	예 *	예 *
BOTTLEROCKET_x86_64를 참조하십시오	예**	예	해당 없음	해당 없음
AL2023_ARM_64_STANDARD를 참조하십시오	예	예	예	예

AL2_ARM_64를 참조하십시오	예	예	예 *	예 *
BOTTLEROCKET_ARM_64를 참조하십시오	예**	예	해당 없음	해당 없음

- * 노드를 재시작하지 않고는 PV를 삭제할 수 없습니다.
- ** Trident 버전 25.02에서는 NFSv3와 작동하지 않습니다.

원하는 AMI가 여기에 나열되지 않은 경우 지원되지 않는다는 의미는 아닙니다. 단순히 테스트를 거치지 않았음을 의미합니다. 이 목록은 AMI가 작동하는 방법에 대한 가이드 역할을 합니다.

- 다음을 사용하여 수행한 테스트:
- EKS 버전: 1.32
- 설치 방법: Helm 25.06 및 AWS 추가 기능 25.06
- NAS의 경우 NFSv3과 NFSv4.1이 모두 테스트되었습니다.
- SAN 전용 iSCSI는 테스트되었으며 NVMe-oF는 테스트되지 않았습니다.
- 수행된 테스트 *:
- 생성 : 저장 클래스, PVC, POD
- 삭제: Pod, PVC(일반, qtree/LUN – 경제성, NAS와 AWS 백업)

자세한 내용을 확인하십시오

- "NetApp ONTAP용 Amazon FSx 문서"
- "NetApp ONTAP용 Amazon FSx 블로그 게시물"

IAM 역할 및 AWS Secret을 생성합니다

AWS 자격 증명을 명시적으로 제공하는 대신 AWS IAM 역할로 인증하여 Kubernetes Pod를 구성하여 AWS 리소스에 액세스할 수 있습니다.

AWS IAM 역할을 사용하여 인증하려면 EKS를 사용하여 Kubernetes 클러스터를 구축해야 합니다.

AWS Secrets Manager 암호를 생성합니다

Trident는 사용자를 위해 스토리지 관리를 위해 FSx 가상 서버에 대해 API를 발행하므로 이를 위해 자격 증명이 필요합니다. 이러한 자격 증명을 전달하는 안전한 방법은 AWS Secrets Manager 암호를 사용하는 것입니다. 따라서 아직 계정이 없는 경우 vsadmin 계정의 자격 증명이 포함된 AWS Secrets Manager 암호를 생성해야 합니다.

이 예에서는 Trident CSI 자격 증명을 저장하기 위한 AWS Secrets Manager 암호를 생성합니다.

```
aws secretsmanager create-secret --name trident-secret --description
"Trident CSI credentials"\
  --secret-string
"{\"username\":\"vsadmin\",\"password\":\"<svmpassword>\"}"
```

IAM 정책을 생성합니다

Trident를 올바르게 실행하려면 AWS 권한도 필요합니다. 따라서 Trident에 필요한 사용 권한을 부여하는 정책을 만들어야 합니다.

다음 예에서는 AWS CLI를 사용하여 IAM 정책을 생성합니다.

```
aws iam create-policy --policy-name AmazonFSxNCSIDriverPolicy --policy
-document file://policy.json
  --description "This policy grants access to Trident CSI to FSxN and
Secrets manager"
```

- 정책 JSON 예 *:

```
{
  "Statement": [
    {
      "Action": [
        "fsx:DescribeFileSystems",
        "fsx:DescribeVolumes",
        "fsx:CreateVolume",
        "fsx:RestoreVolumeFromSnapshot",
        "fsx:DescribeStorageVirtualMachines",
        "fsx:UntagResource",
        "fsx:UpdateVolume",
        "fsx:TagResource",
        "fsx:DeleteVolume"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Action": "secretsmanager:GetSecretValue",
      "Effect": "Allow",
      "Resource": "arn:aws:secretsmanager:<aws-region>:<aws-account-
id>:secret:<aws-secret-manager-name>*"
    }
  ],
  "Version": "2012-10-17"
}
```

서비스 계정 연결**(IRSA)**을 위한 **Pod ID** 또는 **IAM** 역할 생성

Kubernetes 서비스 계정이 EKS Pod Identity를 사용하는 AWS Identity and Access Management(IAM) 역할 또는 서비스 계정 연결(IRSA)을 위한 IAM 역할을 맡도록 구성할 수 있습니다. 서비스 계정을 사용하도록 구성된 모든 Pod는 해당 역할에 액세스 권한이 있는 모든 AWS 서비스에 액세스할 수 있습니다.

포드 아이덴티티

Amazon EKS Pod Identity 연결은 Amazon EC2 인스턴스 프로필이 Amazon EC2 인스턴스에 자격 증명을 제공하는 방식과 유사하게 애플리케이션의 자격 증명을 관리하는 기능을 제공합니다.

EKS 클러스터에 **Pod Identity** 설치:

AWS 콘솔을 통해 또는 다음 AWS CLI 명령을 사용하여 Pod ID를 생성할 수 있습니다.

```
aws eks create-addon --cluster-name <EKS_CLUSTER_NAME> --addon-name
eks-pod-identity-agent
```

자세한 내용은 다음을 참조하세요. "Amazon EKS Pod Identity Agent 설정" .

trust-relationship.json을 생성합니다:

EKS 서비스 주체가 Pod Identity에 대한 이 역할을 수행할 수 있도록 trust-relationship.json 파일을 생성하세요. 그런 다음 다음 신뢰 정책을 사용하여 역할을 생성하세요.

```
aws iam create-role \
  --role-name fsxn-csi-role --assume-role-policy-document file://trust-
relationship.json \
  --description "fsxn csi pod identity role"
```

trust-relationship.json 파일:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "pods.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

IAM 역할에 역할 정책 첨부:

이전 단계의 역할 정책을 생성된 IAM 역할에 연결합니다.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:111122223333:policy/fsxn-csi-policy \
  --role-name fsxn-csi-role
```

포드 **ID** 연결 생성:

IAM 역할과 Trident 서비스 계정(trident-controller) 간에 Pod ID 연결을 생성합니다.

```
aws eks create-pod-identity-association \
  --cluster-name <EKS_CLUSTER_NAME> \
  --role-arn arn:aws:iam::111122223333:role/fsxn-csi-role \
  --namespace trident --service-account trident-controller
```

서비스 계정 연결**(IRSA)**을 위한 **IAM** 역할

AWS CLI 사용:

```
aws iam create-role --role-name AmazonEKS_FSxN_CSI_DriverRole \
  --assume-role-policy-document file://trust-relationship.json
```

- trust-relationship.json 파일: *

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::<account_id>:oidc-
provider/<oidc_provider>"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "<oidc_provider>:aud": "sts.amazonaws.com",
          "<oidc_provider>:sub":
"system:serviceaccount:trident:trident-controller"
        }
      }
    }
  ]
}
```

파일에서 다음 값을 `trust-relationship.json` 업데이트합니다.

- * <account_id> * - AWS 계정 ID
- * <oidc_provider> * - EKS 클러스터의 OIDC. 다음을 실행하여 oidc_provider를 가져올 수 있습니다.

```
aws eks describe-cluster --name my-cluster --query
"cluster.identity.oidc.issuer"\
  --output text | sed -e "s/^https:\/\///"
```

- IAM 정책에 IAM 역할 연결 *:

역할이 생성되면 다음 명령을 사용하여 정책(위 단계에서 만든 정책)을 역할에 연결합니다.

```
aws iam attach-role-policy --role-name my-role --policy-arn <IAM policy
ARN>
```

- OICD 공급자가 연결되었는지 확인 *:

OIDC 공급자가 클러스터와 연결되어 있는지 확인합니다. 다음 명령을 사용하여 확인할 수 있습니다.

```
aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
```

출력이 비어 있는 경우 다음 명령을 사용하여 IAM OIDC를 클러스터에 연결합니다.

```
eksctl utils associate-iam-oidc-provider --cluster $cluster_name
--approve
```

eksctl을 사용하는 경우 다음 예를 사용하여 EKS의 서비스 계정에 대한 IAM 역할을 생성하세요.

```
eksctl create iamserviceaccount --name trident-controller --namespace
trident \
  --cluster <my-cluster> --role-name AmazonEKS_FSxN_CSI_DriverRole
--role-only \
  --attach-policy-arn <IAM-Policy ARN> --approve
```

Trident를 설치합니다

Trident은 Kubernetes에서 Amazon FSx for NetApp ONTAP 스토리지 관리를 간소화하여 개발자와 관리자가 애플리케이션 구축에 집중할 수 있도록 지원합니다.

다음 방법 중 하나를 사용하여 Trident를 설치할 수 있습니다.

- 헬름
- EKS 추가 기능

스냅샷 기능을 사용하려면 CSI 스냅샷 컨트롤러 애드온을 설치하십시오. 자세한 내용은 을 "CSI 볼륨에 대해 스냅샷 기능을 활성화합니다" 참조하십시오.

Helm을 통해 **Trident**를 설치합니다

포드 아이덴티티

1. Trident Helm 저장소 추가:

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
```

2. 다음 예를 사용하여 Trident를 설치하세요.

```
helm install trident-operator netapp-trident/trident-operator --version 100.2502.1 --namespace trident --create-namespace
```

명령을 사용하여 이름, 네임스페이스, 차트, 상태, 앱 버전 및 수정 번호와 같은 설치 세부 정보를 검토할 수 `helm list` 있습니다.

```
helm list -n trident
```

```
NAME                NAMESPACE   REVISION    UPDATED
STATUS       CHART                           APP VERSION

trident-operator    trident     1           2024-10-14
14:31:22.463122 +0300 IDT    deployed     trident-operator-
100.2502.0    25.02.0
```

서비스 계정 연결**(IRSA)**

1. Trident Helm 저장소 추가:

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
```

2. 클라우드 공급자 및 *클라우드 ID*에 대한 값을 설정합니다.

```
helm install trident-operator netapp-trident/trident-operator
--version 100.2502.1  \
--set cloudProvider="AWS" \
--set cloudIdentity="'eks.amazonaws.com/role-arn:
arn:aws:iam::<accountID>:role/<AmazonEKS_FSxN_CSI_DriverRole>'" \
--namespace trident \
--create-namespace
```

명령을 사용하여 이름, 네임스페이스, 차트, 상태, 앱 버전 및 수정 번호와 같은 설치 세부 정보를 검토할 수 `helm list` 있습니다.

```
helm list -n trident
```

```
NAME                NAMESPACE   REVISION    UPDATED
STATUS       CHART                          APP VERSION

trident-operator    trident     1           2024-10-14
14:31:22.463122 +0300 IDT    deployed     trident-operator-
100.2510.0    25.10.0
```

iSCSI를 사용하려면 클라이언트 머신에서 iSCSI가 활성화되어 있는지 확인하세요. AL2023 Worker 노드 OS를 사용하는 경우, helm 설치 시 node prep 매개변수를 추가하여 iSCSI 클라이언트 설치를 자동화할 수 있습니다.

```
helm install trident-operator netapp-trident/trident-operator
--version 100.2502.1 --namespace trident --create-namespace --
set nodePrep={iscsi}
```

EKS 애드온을 통해 **Trident**를 설치합니다

Trident EKS 애드온에는 최신 보안 패치 및 버그 수정이 포함되어 있으며 AWS에서 Amazon EKS와 함께 사용할 수 있다는 것이 검증되었습니다. EKS 애드온을 사용하면 Amazon EKS 클러스터의 보안과 안정성을 지속적으로 보장하고 애드온을 설치, 구성 및 업데이트하는 데 필요한 작업량을 줄일 수 있습니다.

필수 구성 요소

AWS EKS용 Trident 애드온을 구성하기 전에 다음 사항이 있는지 확인하십시오.

- 애드온 가입이 있는 Amazon EKS 클러스터 계정입니다
- AWS 마켓플레이스에 대한 AWS 권한:
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- AMI 유형: Amazon Linux 2 (AL2_x86_64) 또는 Amazon Linux 2 Arm (AL2_ARM_64)
- 노드 유형: AMD 또는 ARM
- 기존 Amazon FSx for NetApp ONTAP 파일 시스템

AWS에 대해 **Trident** 애드온을 활성화합니다

관리 콘솔과 직접 연결되어 있습니다

1. 에서 Amazon EKS 콘솔을 엽니다 https://console.aws.amazon.com/eks/home#/clusters.
2. 왼쪽 탐색 창에서 * 클러스터 * 를 선택합니다.
3. NetApp Trident CSI 추가 기능을 구성할 클러스터의 이름을 선택합니다.
4. Add-ons * 를 선택한 다음 * Get more add-ons * 를 선택합니다.
5. 추가 기능을 선택하려면 다음 단계를 따르세요.
   a. **AWS Marketplace** 추가 기능 섹션까지 아래로 스크롤하여 검색 상자에 *"Trident"*를 입력합니다.
   b. Trident by NetApp 상자의 오른쪽 상단에 있는 확인란을 선택하세요.
   c. 다음 * 을 선택합니다.
6. 선택한 추가 기능 구성 * 설정 페이지에서 다음을 수행합니다.

   Pod Identity 연결을 사용하는 경우 이 단계를 건너뜁니다.

   a. 사용할 * 버전 * 을 선택합니다.
   b. IRSA 인증을 사용하는 경우 선택적 구성 설정에서 사용 가능한 구성 값을 설정해야 합니다.
      - 사용할 * 버전 * 을 선택합니다.
      - 추가 기능 구성 스키마*를 따르고 *구성 값 섹션의 **configurationValues** 매개변수를 이전 단계에서 만든 role-arn으로 설정합니다(값은 다음 형식이어야 함).

```
{

  "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'",
  "cloudProvider": "AWS"

}
```

+
충돌 해결 방법으로 재정의 를 선택한 경우 기존 애드온에 대한 하나 이상의 설정을 Amazon EKS 애드온 설정으로 덮어쓸 수 있습니다. 이 옵션을 사용하지 않고 기존 설정과 충돌하는 경우 작업이 실패합니다. 결과 오류 메시지를 사용하여 충돌 문제를 해결할 수 있습니다. 이 옵션을 선택하기 전에 Amazon EKS 추가 기능이 자체 관리해야 하는 설정을 관리하지 않는지 확인하십시오.

7. 다음 * 을 선택합니다.
8. 검토 및 추가 * 페이지에서 * 만들기 * 를 선택합니다.

   추가 기능 설치가 완료되면 설치된 추가 기능이 표시됩니다.

AWS CLI를 참조하십시오

- 1. 생성하다 `add-on.json` 파일*:

Pod Identity의 경우 다음 형식을 사용하세요:

다음을 사용하세요

```
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v25.6.0-eksbuild.1",
}
```

IRSA 인증의 경우 다음 형식을 사용하세요:

```
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v25.6.0-eksbuild.1",
  "serviceAccountRoleArn": "<role ARN>",
  "configurationValues": {
    "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'",
    "cloudProvider": "AWS"
  }
}
```

`<role ARN>`이전 단계에서 생성한 역할의 ARN으로 바꿉니다.

- 2. Trident EKS 애드온을 설치하세요.*

```
aws eks create-addon --cli-input-json file://add-on.json
```

eksctl입니다

다음 명령 예에서는 Trident EKS 추가 기능을 설치합니다.

```
eksctl create addon --name netapp_trident-operator --cluster
<cluster_name> --force
```

Trident EKS 추가 기능을 업데이트합니다

관리 콘솔과 직접 연결되어 있습니다

1. Amazon EKS 콘솔을 https://console.aws.amazon.com/eks/home#/clusters엽니다.
2. 왼쪽 탐색 창에서 * 클러스터 * 를 선택합니다.
3. NetApp Trident CSI 애드온을 업데이트할 클러스터의 이름을 선택합니다.
4. Add-ons * 탭을 선택합니다.
5. Trident by NetApp * 를 선택한 다음 * 편집 * 을 선택합니다.
6. Trident by NetApp * 구성 페이지에서 다음을 수행합니다.
   a. 사용할 * 버전 * 을 선택합니다.
   b. 선택적 구성 설정 * 을 확장하고 필요에 따라 수정합니다.
   c. 변경 내용 저장 * 을 선택합니다.

AWS CLI를 참조하십시오

다음 예에서는 EKS 추가 기능을 업데이트합니다.

```
aws eks update-addon --cluster-name <eks_cluster_name> --addon-name
netapp_trident-operator --addon-version v25.6.0-eksbuild.1 \
  --service-account-role-arn <role-ARN> --resolve-conflict preserve \
  --configuration-values "{\"cloudIdentity\":
\"'eks.amazonaws.com/role-arn: <role ARN>'\"}"
```

eksctl입니다

- FSxN Trident CSI 추가 기능의 현재 버전을 확인합니다. 클러스터 이름으로 교체합니다 `my-cluster`.

```
eksctl get addon --name netapp_trident-operator --cluster my-cluster
```

- 출력 예: *

```
NAME                          VERSION              STATUS    ISSUES
IAMROLE    UPDATE AVAILABLE    CONFIGURATION VALUES
netapp_trident-operator    v25.6.0-eksbuild.1    ACTIVE    0
{"cloudIdentity":"'eks.amazonaws.com/role-arn:
arn:aws:iam::139763910815:role/AmazonEKS_FSXN_CSI_DriverRole'"}
```

- 이전 단계의 출력에서 사용할 수 있는 업데이트 아래에 반환된 버전으로 추가 기능을 업데이트합니다.

```
eksctl update addon --name netapp_trident-operator --version
v25.6.0-eksbuild.1 --cluster my-cluster --force
```

옵션을 제거하고 Amazon EKS 추가 기능 설정이 기존 설정과 충돌하는 경우 `--force` Amazon EKS 추가 기능 업데이트가 실패하고 충돌 문제를 해결하는 데 도움이 되는 오류 메시지가 표시됩니다. 이 옵션을 지정하기 전에 Amazon EKS 애드온이 관리해야 하는 설정을 관리하지 않는지 확인하십시오. 이러한 설정은 이 옵션으로 덮어써지기 때문입니다. 이 설정의 다른 옵션에 대한 자세한 내용은 을 참조하십시오 "추가 기능". Amazon EKS Kubernetes 필드 관리에 대한 자세한 내용은 를 참조하십시오 "Kubernetes 현장 관리".

Trident EKS 추가 기능을 제거/제거합니다

Amazon EKS 애드온을 제거하는 두 가지 옵션이 있습니다.

- * 클러스터에 애드온 소프트웨어 유지 * – 이 옵션은 모든 설정의 Amazon EKS 관리를 제거합니다. 또한 업데이트를 시작한 후 Amazon EKS에서 업데이트를 알리고 Amazon EKS 애드온을 자동으로 업데이트하는 기능도 제거합니다. 하지만 클러스터에 애드온 소프트웨어가 보존됩니다. 이 옵션을 사용하면 Amazon EKS 애드온이 아닌 자가 관리형 설치가 됩니다. 이 옵션을 사용하면 애드온에 대한 다운타임이 없습니다. `--preserve`명령의 옵션을 유지하여 추가 기능을 유지합니다.
- * 클러스터에서 애드온 소프트웨어 완전히 제거 * – NetApp는 클러스터에 종속된 리소스가 없는 경우에만 클러스터에서 Amazon EKS 애드온을 제거할 것을 권장합니다. `--preserve`추가 기능을 제거하려면 명령에서 옵션을 `delete` 제거하십시오.

ⓘ 애드온에 IAM 계정이 연결되어 있으면 IAM 계정이 제거되지 않습니다.

관리 콘솔과 직접 연결되어 있습니다

1. 에서 Amazon EKS 콘솔을 엽니다 https://console.aws.amazon.com/eks/home#/clusters.
2. 왼쪽 탐색 창에서 * 클러스터 * 를 선택합니다.
3. NetApp Trident CSI 추가 기능을 제거할 클러스터의 이름을 선택합니다.
4. 추가 기능 * 탭을 선택한 다음 * Trident by NetApp *. * 를 선택합니다
5. 제거 * 를 선택합니다.
6. Remove netapp_trident-operator confirmation * 대화 상자에서 다음을 수행합니다.
   a. Amazon EKS가 애드온에 대한 설정 관리를 중지하도록 하려면 * 클러스터에서 유지 * 를 선택합니다. 추가 기능의 모든 설정을 직접 관리할 수 있도록 클러스터에 추가 소프트웨어를 유지하려는 경우 이 작업을 수행합니다.
   b. netapp_trident-operator * 를 입력합니다.
   c. 제거 * 를 선택합니다.

AWS CLI를 참조하십시오

클러스터 이름으로 바꾸고 `my-cluster` 다음 명령을 실행합니다.

```
aws eks delete-addon --cluster-name my-cluster --addon-name
netapp_trident-operator --preserve
```

eksctl입니다

다음 명령을 실행하면 Trident EKS 추가 기능이 제거됩니다.

```
eksctl delete addon --cluster K8s-arm --name netapp_trident-operator
```

스토리지 백엔드를 구성합니다

ONTAP SAN 및 **NAS** 드라이버 통합

스토리지 백엔드를 생성하려면 JSON 또는 YAML 형식으로 구성 파일을 만들어야 합니다. 파일에서 원하는 스토리지 유형(NAS 또는 SAN), 파일 시스템, SVM을 가져와 인증 방법을 지정해야 합니다. 다음 예제는 NAS 기반 스토리지를 정의하고 AWS 암호를 사용하여 사용하려는 SVM에 자격 증명을 저장하는 방법을 보여줍니다.

YAML

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  backendName: tbc-ontap-nas
  svm: svm-name
  aws:
    fsxFilesystemID: fs-xxxxxxxxxx
  credentials:
    name: "arn:aws:secretsmanager:us-west-2:xxxxxxxx:secret:secret-
name"
    type: awsarn
```

JSON을 참조하십시오

```
{
  "apiVersion": "trident.netapp.io/v1",
  "kind": "TridentBackendConfig",
  "metadata": {
    "name": "backend-tbc-ontap-nas"
    "namespace": "trident"
  },
  "spec": {
    "version": 1,
    "storageDriverName": "ontap-nas",
    "backendName": "tbc-ontap-nas",
    "svm": "svm-name",
    "aws": {
      "fsxFilesystemID": "fs-xxxxxxxxxx"
    },
    "managementLIF": null,
    "credentials": {
      "name": "arn:aws:secretsmanager:us-west-2:xxxxxxxx:secret:secret-
name",
      "type": "awsarn"
    }
  }
}
```

다음 명령을 실행하여 Trident TBC(백엔드 구성)를 생성하고 검증합니다.

- YAML 파일에서 TBC(Trident 백엔드 구성)를 생성하고 다음 명령을 실행합니다.

```
kubectl create -f backendconfig.yaml -n trident
```

```
tridentbackendconfig.trident.netapp.io/backend-tbc-ontap-nas created
```

- Trident 백엔드 구성(TBC)이 성공적으로 생성되었는지 확인:

```
Kubectl get tbc -n trident
```

```
NAME                          BACKEND NAME          BACKEND UUID
PHASE    STATUS

backend-tbc-ontap-nas         tbc-ontap-nas         933e0071-66ce-4324-
b9ff-f96d916ac5e9    Bound    Success
```

FSx for ONTAP 드라이버 세부 정보

다음 드라이버를 사용하여 Trident를 Amazon FSx for NetApp ONTAP와 통합할 수 있습니다.

- `ontap-san`: 프로비저닝된 각 PV는 자체 Amazon FSx for NetApp ONTAP 볼륨 내의 LUN입니다. 블록 스토리지에 권장됩니다.
- `ontap-nas`: 프로비저닝된 각 PV는 전체 Amazon FSx for NetApp ONTAP 볼륨입니다. NFS 및 SMB에 권장됩니다.
- 'ONTAP-SAN-이코노미': 프로비저닝되는 각 PV는 NetApp ONTAP 볼륨에 대해 Amazon FSx당 구성 가능한 LUN 수를 가진 LUN입니다.
- 'ONTAP-NAS-E경제적인': 각 PV 프로비저닝은 qtree이며, NetApp ONTAP 볼륨에 대해 Amazon FSx당 qtree를 구성할 수 있습니다.
- 'ONTAP-NAS-flexgroup': 프로비저닝되는 각 PV는 NetApp ONTAP FlexGroup 볼륨에 대한 전체 Amazon FSx입니다.

드라이버에 대한 자세한 내용은 을 참조하십시오 "NAS 드라이버" 및 "SAN 드라이버".

구성 파일이 생성되면 다음 명령을 실행하여 EKS 내에 구성 파일을 생성합니다.

```
kubectl create -f configuration_file
```

상태를 확인하려면 다음 명령을 실행합니다.

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

```
NAME                        BACKEND NAME            BACKEND UUID
PHASE   STATUS
backend-fsx-ontap-nas   backend-fsx-ontap-nas   7a551921-997c-4c37-a1d1-
f2f4c87fa629   Bound   Success
```

백엔드 고급 구성 및 예

백엔드 구성 옵션은 다음 표를 참조하십시오.

매개 변수	설명	예
'내전'		항상 1
'torageDriverName'입니다	스토리지 드라이버의 이름입니다	`ontap-nas`, `ontap-nas-economy`, `ontap-nas-flexgroup`, `ontap-san`, `ontap-san-economy`
백엔드이름	사용자 지정 이름 또는 스토리지 백엔드	드라이버 이름 + "_" + dataLIF
마나멘타LIF	클러스터 또는 SVM 관리 LIF의 IP 주소 정규화된 도메인 이름(FQDN)을 지정할 수 있습니다. IPv6 플래그를 사용하여 Trident가 설치된 경우 IPv6 주소를 사용하도록 설정할 수 있습니다. IPv6 주소는 대괄호로 묶어야 합니다(예: [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]). `aws`현장에서 을 제공하는 경우 `fsxFilesystemID`, Trident이 AWS에서 SVM 정보를 검색하기 때문에 를 제공할 필요가 없습니다 `managementLIF`. `managementLIF` 따라서 SVM에서 사용자에 대한 자격 증명(예: vsadmin)을 제공해야 하며 사용자에게 역할이 있어야 합니다. `vsadmin`	"10.0.0.1"," [2001:1234:ABCD::fefe]"

매개 변수	설명	예
다타LIF	프로토콜 LIF의 IP 주소입니다. * ONTAP NAS 드라이버 *: NetApp는 dataLIF를 지정할 것을 권장합니다. 제공되지 않는 경우 Trident는 SVM에서 데이터 LIF를 가져옵니다. NFS 마운트 작업에 사용할 FQDN(정규화된 도메인 이름)을 지정할 수 있습니다. 이렇게 하면 라운드 로빈 DNS를 생성하여 여러 데이터 LIF의 로드 밸런싱을 수행할 수 있습니다. 초기 설정 후에 변경할 수 있습니다. 을 참조하십시오. * ONTAP SAN 드라이버 *: iSCSI에 대해 지정하지 마십시오. Trident는 ONTAP 선택적 LUN 맵을 사용하여 다중 경로 세션을 설정하는 데 필요한 iSCI LIF를 검색합니다. 데이터 LIF가 명시적으로 정의되어 있으면 경고가 생성됩니다. IPv6 플래그를 사용하여 Trident가 설치된 경우 IPv6 주소를 사용하도록 설정할 수 있습니다. IPv6 주소는 대괄호로 묶어야 합니다(예: [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]).	
자동 내보내기 정책	자동 엑스포트 정책 생성 및 업데이트 [Boolean] 활성화 Trident는 및 `autoExportCIDRs` 옵션을 사용하여 `autoExportPolicy` 엑스포트 정책을 자동으로 관리할 수 있습니다.	거짓입니다
자동 내보내기	이 설정된 경우에 대해 Kubernetes 노드 IP를 필터링하는 CIDR `autoExportPolicy` 목록입니다. Trident는 및 `autoExportCIDRs` 옵션을 사용하여 `autoExportPolicy` 엑스포트 정책을 자동으로 관리할 수 있습니다.	"["0.0.0.0/0",":/0"]"
'라벨'	볼륨에 적용할 임의의 JSON 형식 레이블 세트입니다	""
'고객증명서'	Base64 - 클라이언트 인증서의 인코딩된 값입니다. 인증서 기반 인증에 사용됩니다	""
'clientPrivateKey’입니다	Base64 - 클라이언트 개인 키의 인코딩된 값입니다. 인증서 기반 인증에 사용됩니다	""
신탁인증서다	Base64 - 신뢰할 수 있는 CA 인증서의 인코딩된 값입니다. 선택 사항. 인증서 기반 인증에 사용됩니다.	""

매개 변수	설명	예
'사용자 이름'	클러스터 또는 SVM에 연결할 사용자 이름입니다. 자격 증명 기반 인증에 사용됩니다. 예: vsadmin.	
"암호"	클러스터 또는 SVM에 연결하는 암호 자격 증명 기반 인증에 사용됩니다.	
'VM'입니다	사용할 스토리지 가상 머신입니다	SVM 관리 LIF가 지정된 경우에 파생됩니다.
'토르agePrefix'	SVM에서 새 볼륨을 프로비저닝할 때 사용되는 접두사 생성 후에는 수정할 수 없습니다. 이 매개 변수를 업데이트하려면 새 백엔드를 생성해야 합니다.	`trident`
제한선태사용법	* NetApp ONTAP * 용 아마존 FSx에 대해서는 지정하지 마십시오 제공된 및 `vsadmin` 에는 `fsxadmin` 애그리게이트 사용량을 검색하고 Trident를 사용하여 제한하는 데 필요한 권한이 포함되어 있지 않습니다.	사용하지 마십시오.
LimitVolumeSize	요청된 볼륨 크기가 이 값보다 큰 경우 용량 할당에 실패합니다. 또한 qtree 및 LUN에서 관리하는 볼륨의 최대 크기를 제한하고, `qtreesPerFlexvol` 옵션을 통해 FlexVol volume당 최대 qtree 수를 사용자 지정할 수 있습니다	""(기본적으로 적용되지 않음)
'오만유연한'	FlexVol volume당 최대 LUN은 [50, 200] 범위에 있어야 합니다. SAN만 해당.	"'100'"
debugTraceFlags를 선택합니다	문제 해결 시 사용할 디버그 플래그입니다. 예: {"api":false, "method":true} 사용하지 마십시오 `debugTraceFlags` 문제 해결 및 자세한 로그 덤프가 필요한 경우를 제외하고	null입니다

매개 변수	설명	예
nfsMountOptions를 선택합니다	쉼표로 구분된 NFS 마운트 옵션 목록입니다. Kubernetes 영구 볼륨의 마운트 옵션은 일반적으로 스토리지 클래스에 지정되어 있지만, 스토리지 클래스에 마운트 옵션이 지정되지 않은 경우 Trident는 스토리지 백엔드의 구성 파일에 지정된 마운트 옵션을 사용하도록 폴백합니다. 스토리지 클래스 또는 구성 파일에 마운트 옵션이 지정되지 않은 경우 Trident는 연결된 영구 볼륨에 마운트 옵션을 설정하지 않습니다.	""
`nasType`	NFS 또는 SMB 볼륨 생성을 구성합니다. 옵션은 입니다 `nfs`, `smb`또는 `null`입니다. `*` 를 로 설정해야 합니다 `smb` SMB 볼륨의 경우. * null로 설정하면 기본적으로 NFS 볼륨이 설정됩니다.	`nfs`
"케트리스퍼플랙스볼륨"	FlexVol volume당 최대 qtree, 범위 [50, 300] 내에 있어야 함	`"200"`
`smbShare`	Microsoft 관리 콘솔 또는 ONTAP CLI를 사용하여 생성된 SMB 공유의 이름 또는 Trident에서 SMB 공유를 생성하도록 허용하는 이름을 지정할 수 있습니다. 이 매개변수는 ONTAP 백엔드에 대한 Amazon FSx에 필요합니다.	`smb-share`
'useREST'	ONTAP REST API를 사용하는 부울 매개 변수입니다. 로 `true` 설정하면 Trident에서 ONTAP REST API를 사용하여 백엔드와 통신합니다. 이 기능을 사용하려면 ONTAP 9.11.1 이상이 필요합니다. 또한 사용되는 ONTAP 로그인 역할에는 애플리케이션에 대한 액세스 권한이 있어야 `ontap` 합니다. 이는 미리 정의된 역할과 역할에 의해 충족됩니다. `vsadmin` `cluster-admin`	거짓입니다
`aws`	AWS FSx for ONTAP의 구성 파일에서 다음을 지정할 수 있습니다. - `fsxFilesystemID`: AWS FSx 파일 시스템의 ID를 지정합니다. - `apiRegion`: AWS API 지역 이름입니다. - `apikey`: AWS API 키입니다. - `secretKey`: AWS 비밀 키입니다.	"" "" ""

매개 변수	설명	예
`credentials`	AWS Secrets Manager에 저장할 FSx SVM 자격 증명을 지정합니다. - `name`: SVM의 자격 증명이 포함된 비밀의 ARN(Amazon Resource Name). `type`-: 로 `awsarn`설정합니다. 자세한 내용은 을 "AWS Secrets Manager 암호를 생성합니다" 참조하십시오.	

볼륨 프로비저닝을 위한 백엔드 구성 옵션

에서 이러한 옵션을 사용하여 기본 프로비저닝을 제어할 수 있습니다 `defaults` 섹션을 참조하십시오. 예를 들어, 아래 구성 예제를 참조하십시오.

매개 변수	설명	기본값
'팩시배부	LUN에 대한 공간 할당	"참"입니다
'예비공간'	공간 예약 모드, "없음"(씬) 또는 " 볼륨"(일반)	"없음"
냅샷정책	사용할 스냅샷 정책입니다	"없음"
"qosPolicy"	생성된 볼륨에 할당할 QoS 정책 그룹입니다. 스토리지 풀 또는 백엔드에서 qosPolicy 또는 adapativeQosPolicy 중 하나를 선택합니다. Trident에서 QoS 정책 그룹을 사용하려면 ONTAP 9 .8 이상이 필요합니다. 비공유 QoS 정책 그룹을 사용하고 정책 그룹이 각 구성 요소에 개별적으로 적용되도록 해야 합니다. 공유 QoS 정책 그룹은 모든 워크로드의 총 처리량에 대한 제한을 적용합니다.	""
적응성 QosPolicy	생성된 볼륨에 할당할 적응형 QoS 정책 그룹입니다. 스토리지 풀 또는 백엔드에서 qosPolicy 또는 adapativeQosPolicy 중 하나를 선택합니다. ONTAP에서 지원되지 않음 - NAS - 이코노미	""
안산예비역	스냅샷 "0"에 대해 예약된 볼륨의 백분율	`snapshotPolicy` `none`있다면, `else`""
'plitOnClone'을 선택합니다	생성 시 상위 클론에서 클론을 분할합니다	거짓입니다

매개 변수	설명	기본값
암호화	새 볼륨에서 NetApp 볼륨 암호화(NVE)를 활성화하고, 기본값은 로 설정합니다. `false` 이 옵션을 사용하려면 NVE 라이센스가 클러스터에서 활성화되어 있어야 합니다. 백엔드에서 NAE가 활성화된 경우 Trident에서 프로비저닝된 모든 볼륨은 NAE가 사용됩니다. 자세한 내용은 다음을 "Trident가 NVE 및 NAE와 작동하는 방법"참조하십시오.	거짓입니다
`luksEncryption`	LUKS 암호화를 사용합니다. 을 참조하십시오 "LUKS(Linux Unified Key Setup) 사용". SAN만 해당.	""
'계층화 정책'	사용할 계층화 정책 `none`	
유니크권한	모드를 선택합니다. * SMB 볼륨의 경우 비워 둡니다. *	""
'생태성 스타일'을 참조하십시오	새로운 볼륨에 대한 보안 스타일 NFS를 지원합니다 `mixed` 및 `unix` 보안 스타일. SMB 지원 `mixed` 및 `ntfs` 보안 스타일.	NFS 기본값은 입니다 `unix`. SMB 기본값은 입니다 `ntfs`.

SMB 볼륨 제공

다음을 사용하여 SMB 볼륨을 프로비저닝할 수 있습니다. `ontap-nas` 운전사. 완료하기 전에 ONTAP SAN 및 NAS 드라이버 통합 다음 단계를 완료하세요. "SMB 볼륨 프로비저닝을 위한 준비".

저장소 클래스 및 PVC를 구성합니다

Kubernetes StorageClass 개체를 구성하고 스토리지 클래스를 생성하여 Trident에 볼륨 프로비저닝 방법을 지시합니다. 구성된 Kubernetes StorageClass를 사용하여 PV에 대한 액세스를 요청하는 PersistentVolumeClaim(PVC)을 생성합니다. 그런 다음 PV를 포드에 장착할 수 있습니다.

스토리지 클래스를 생성합니다

Kubernetes StorageClass 개체를 구성합니다

그만큼 "Kubernetes StorageClass 객체" 객체는 Trident 해당 클래스에 사용되는 프로비저너로 식별하고 Trident 볼륨을 프로비저닝하는 방법을 지시합니다. NFS를 사용하여 볼륨에 대한 Storageclass를 설정하려면 이 예제를 사용하세요(전체 속성 목록은 아래의 Trident 속성 섹션을 참조하세요).

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  provisioningType: "thin"
  snapshots: "true"
```

iSCSI를 사용하여 볼륨에 대한 Storageclass를 설정하려면 다음 예를 사용하세요.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
  provisioningType: "thin"
  snapshots: "true"
```

AWS Bottlerocket에서 NFSv3 볼륨을 프로비저닝하려면 필요한 를 `mountOptions` 스토리지 클래스에 추가합니다.

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

Trident에서 볼륨을 프로비저닝하는 방법을 제어하는 데 사용되는 및 매개 변수와 스토리지 클래스가 상호 작용하는 방법에 대한 자세한 `PersistentVolumeClaim` 내용은 을 "Kubernetes 및 Trident 오브젝트"참조하십시오.

스토리지 클래스를 생성합니다

단계

1. Kubernetes 오브젝트이므로 를 사용하십시오 `kubectl` Kubernetes에서 생성해야 합니다.

```
kubectl create -f storage-class-ontapnas.yaml
```

2. 이제 Kubernetes와 Trident 모두에서 * BASIC-CSI * 스토리지 클래스가 표시되고 Trident는 백엔드에서 풀을 검색해야 합니다.

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

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

PVC를 작성합니다

A "*PersistentVolumeClaim*"(PVC)는 클러스터의 PersistentVolume에 대한 액세스 요청입니다.

PVC는 특정 크기 또는 액세스 모드의 저장을 요청하도록 구성할 수 있습니다. 클러스터 관리자는 연결된 StorageClass를 사용하여 PersistentVolume 크기 및 액세스 모드(예: 성능 또는 서비스 수준)를 제어할 수 있습니다.

PVC를 생성한 후 포드에 볼륨을 장착할 수 있습니다.

샘플 매니페스트

PersistentVolumeClaim 샘플 매니페스트

이러한 예는 기본적인 PVC 구성 옵션을 보여줍니다.

RWX 액세스 **PVC**

이 예에서는 이름이 인 StorageClass와 연결된 rwx 액세스 권한이 있는 기본 PVC를 보여 `basic-csi`줍니다.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-storage
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-gold
```

iSCSI를 사용한 **PVC** 예제

이 예에서는 RWO 액세스가 있는 iSCSI용 기본 PVC를 보여줍니다. 이 PVC는 StorageClass와 연결되어 있습니다. `protection-gold` .

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: pvc-san
spec:
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 1Gi
storageClassName: protection-gold
```

PVC 생성

단계

1. PVC를 작성합니다.

```
kubectl create -f pvc.yaml
```

2. PVC 상태를 확인합니다.

```
kubectl get pvc
```

```
NAME          STATUS VOLUME     CAPACITY ACCESS MODES STORAGECLASS AGE
pvc-storage Bound  pv-name 2Gi       RWO                           5m
```

Trident에서 볼륨을 프로비저닝하는 방법을 제어하는 데 사용되는 및 매개 변수와 스토리지 클래스가 상호 작용하는 방법에 대한 자세한 `PersistentVolumeClaim` 내용은 을 "Kubernetes 및 Trident 오브젝트"참조하십시오.

Trident 특성

이러한 매개 변수는 지정된 유형의 볼륨을 프로비저닝하는 데 사용해야 하는 Trident 관리 스토리지 풀을 결정합니다.

속성	유형	값	제공합니다	요청하십시오	에 의해 지원됩니다
미디어 [1]	문자열	HDD, 하이브리드, SSD	풀에는 이 유형의 미디어가 포함되어 있으며, 하이브리드는 둘 모두를 의미합니다	지정된 미디어 유형입니다	ONTAP-NAS, ONTAP-NAS-이코노미, ONTAP-NAS-Flexgroup, ONTAP-SAN, solidfire-SAN
프로비저닝 유형	문자열	얇고 두껍습니다	풀은 이 프로비저닝 방법을 지원합니다	프로비저닝 방법이 지정되었습니다	Thick: All ONTAP; Thin: All ONTAP & solidfire-SAN
백엔드 유형	문자열	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, solidfire-san, azure-netapp-files, ontap-san-economy	풀이 이 백엔드 유형에 속합니다	백엔드가 지정되었습니다	모든 드라이버
스냅샷 수	불입니다	참, 거짓	풀은 스냅샷이 있는 볼륨을 지원합니다	스냅샷이 활성화된 볼륨	온탑나스, 온탑씨, 솔리드파이어씨
복제	불입니다	참, 거짓	풀은 볼륨 클론을 지원합니다	클론이 활성화된 볼륨	온탑나스, 온탑씨, 솔리드파이어씨

속성	유형	값	제공합니다	요청하십시오	에 의해 지원됩니다
암호화	불입니다	참, 거짓	풀은 암호화된 볼륨을 지원합니다	암호화가 활성화된 볼륨입니다	ONTAP-NAS, ONTAP-NAS-이코노미, ONTAP-NAS-Flexgroups, ONTAP-SAN
IOPS	내부	양의 정수입니다	풀은 이 범위에서 IOPS를 보장할 수 있습니다	볼륨은 이러한 IOPS를 보장합니다	solidfire-SAN

[1]: ONTAP Select 시스템에서 지원되지 않습니다

샘플 응용 프로그램을 배포합니다

저장 클래스 및 PVC가 생성되면 PV를 포드에 장착할 수 있습니다. 이 섹션에는 PV를 Pod에 연결하기 위한 명령 및 구성의 예가 나와 있습니다.

단계

1. 볼륨을 Pod에 마운트합니다.

```
kubectl create -f pv-pod.yaml
```

다음 예는 PVC를 포드에 부착하기 위한 기본 구성을 보여줍니다. * 기본 구성 *:

```
kind: Pod
apiVersion: v1
metadata:
  name: pv-pod
spec:
  volumes:
    - name: pv-storage
      persistentVolumeClaim:
       claimName: basic
  containers:
    - name: pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: pv-storage
```

을 사용하여 진행 상황을 모니터링할 수 있습니다 `kubectl get pod --watch`.

2. 볼륨이 에 마운트되어 있는지 확인합니다 `/my/mount/path`.

```
kubectl exec -it pv-pod -- df -h /my/mount/path
```

```
Filesystem                                                      Size
Used  Avail Use%  Mounted on
192.168.188.78:/trident_pvc_ae45ed05_3ace_4e7c_9080_d2a83ae03d06  1.1G
320K  1.0G   1%   /my/mount/path
```

이제 Pod를 삭제할 수 있습니다. Pod 응용 프로그램은 더 이상 존재하지 않지만 볼륨은 유지됩니다.

```
kubectl delete pod pv-pod
```

EKS 클러스터에서 Trident EKS 애드온을 구성합니다

NetApp Trident은 Kubernetes에서 Amazon FSx for NetApp ONTAP 스토리지 관리를 간소화하여 개발자와 관리자가 애플리케이션 구축에 집중할 수 있도록 지원합니다. NetApp Trident EKS 애드온에는 최신 보안 패치 및 버그 수정이 포함되어 있으며 AWS에서 Amazon EKS와 함께 사용할 수 있다는 것이 검증되었습니다. EKS 애드온을 사용하면 Amazon EKS 클러스터의 보안과 안정성을 지속적으로 보장하고 애드온을 설치, 구성 및 업데이트하는 데 필요한 작업량을 줄일 수 있습니다.

필수 구성 요소

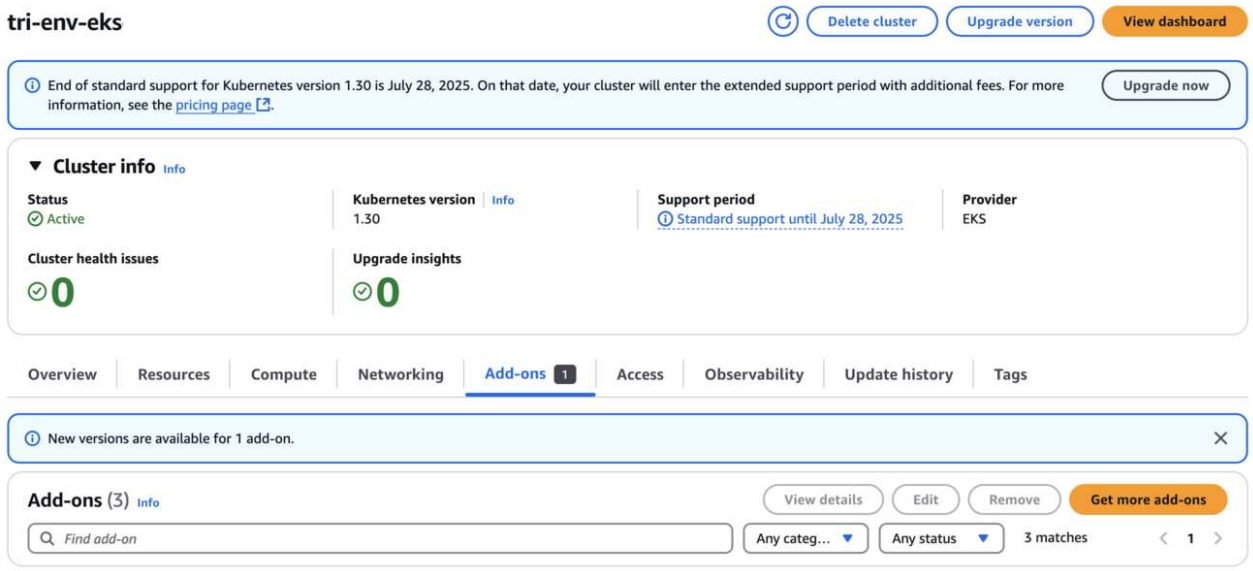
AWS EKS용 Trident 애드온을 구성하기 전에 다음 사항이 있는지 확인하십시오.

- 추가 기능을 사용할 수 있는 권한이 있는 Amazon EKS 클러스터 계정입니다. 을 "Amazon EKS 애드온 "참조하십시오.
- AWS 마켓플레이스에 대한 AWS 권한:
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- AMI 유형: Amazon Linux 2 (AL2_x86_64) 또는 Amazon Linux 2 Arm (AL2_ARM_64)
- 노드 유형: AMD 또는 ARM
- 기존 Amazon FSx for NetApp ONTAP 파일 시스템

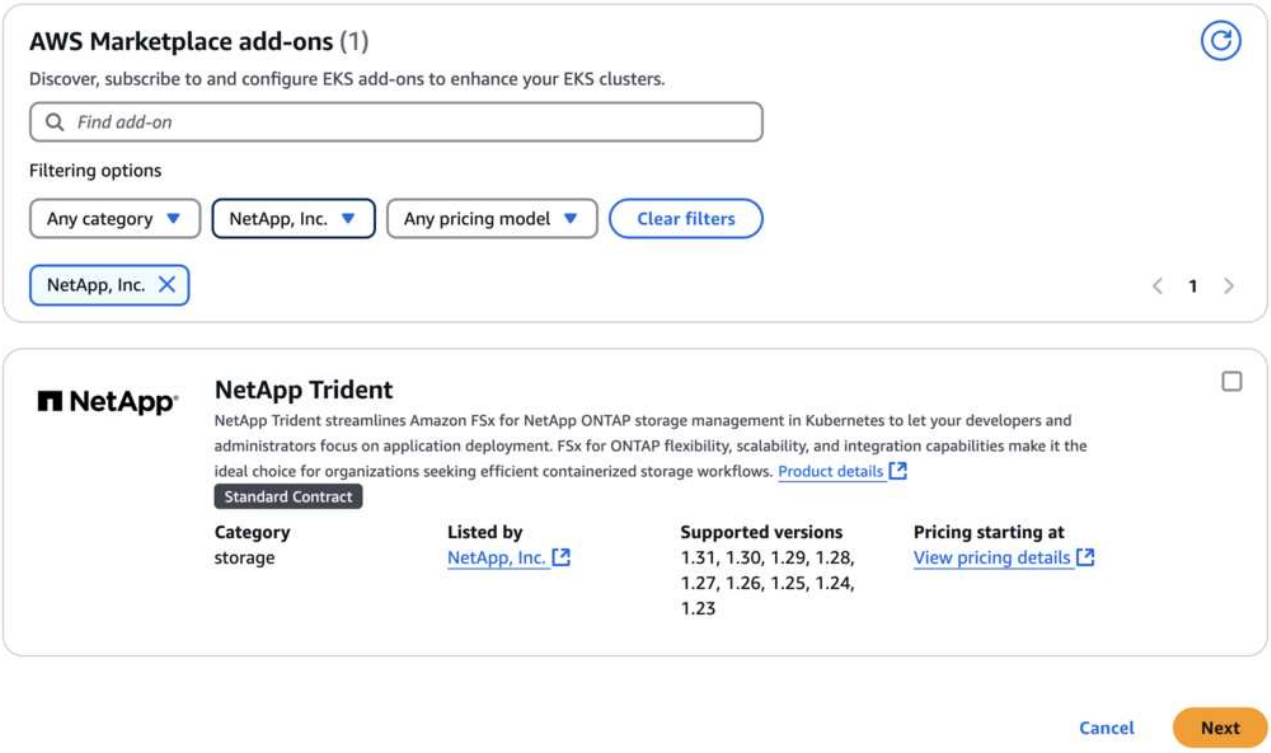
단계

1. EKS Pod에서 AWS 리소스에 액세스할 수 있도록 IAM 역할 및 AWS 암호를 생성하십시오. 자세한 내용은 을 "IAM 역할 및 AWS Secret을 생성합니다"참조하십시오.

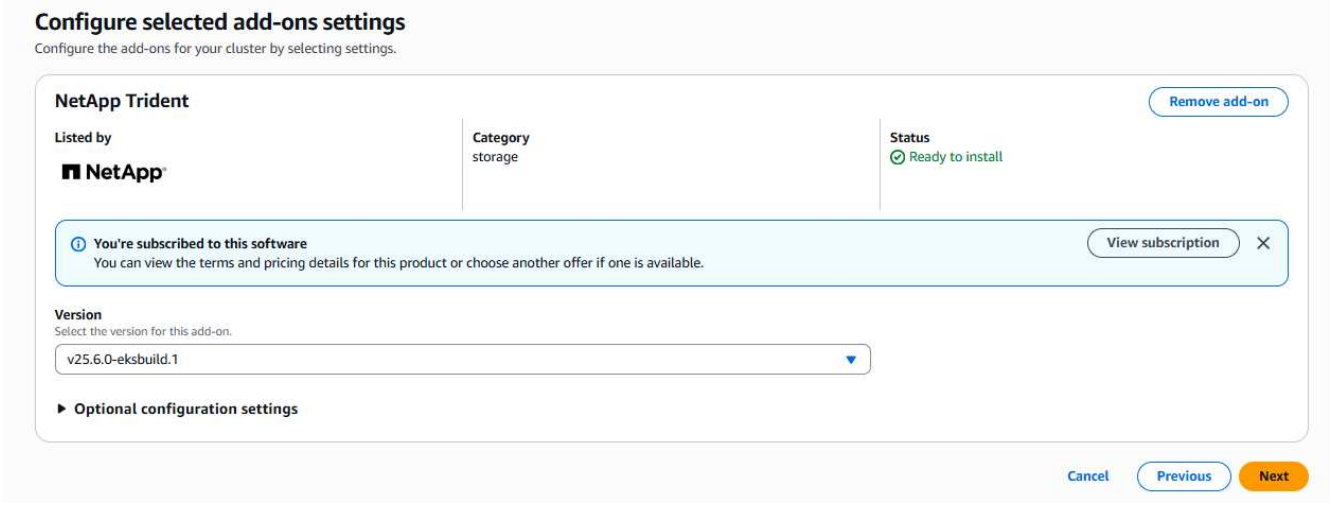
2. EKS Kubernetes 클러스터에서 * Add-ons * 탭으로 이동합니다.



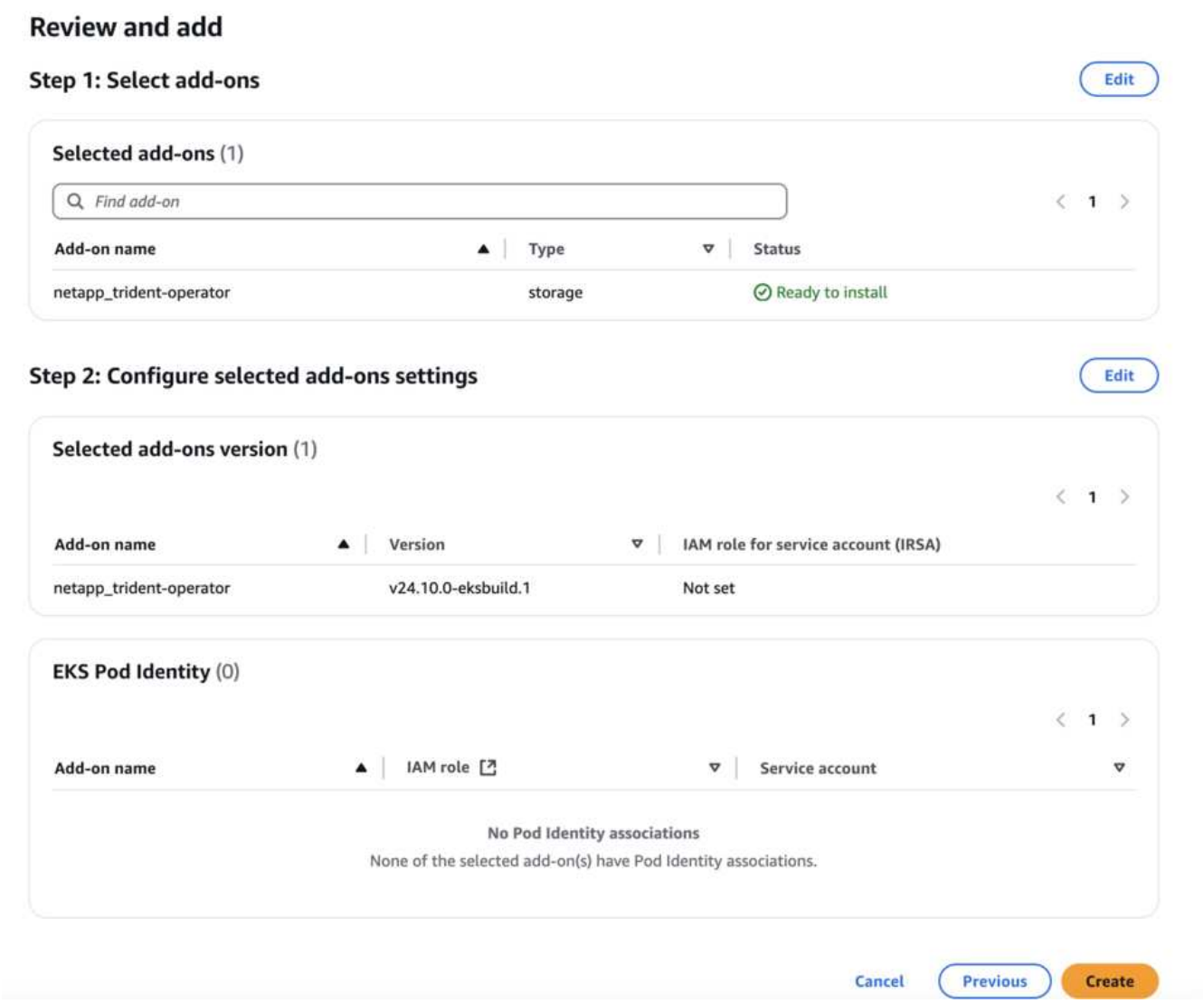
3. AWS Marketplace 애드온 * 으로 이동하여 _STORAGE_CATEGORY를 선택합니다.



4. NetApp Trident * 를 찾아 Trident 애드온의 확인란을 선택하고 * 다음 * 을 클릭합니다.
5. 원하는 추가 기능 버전을 선택합니다.

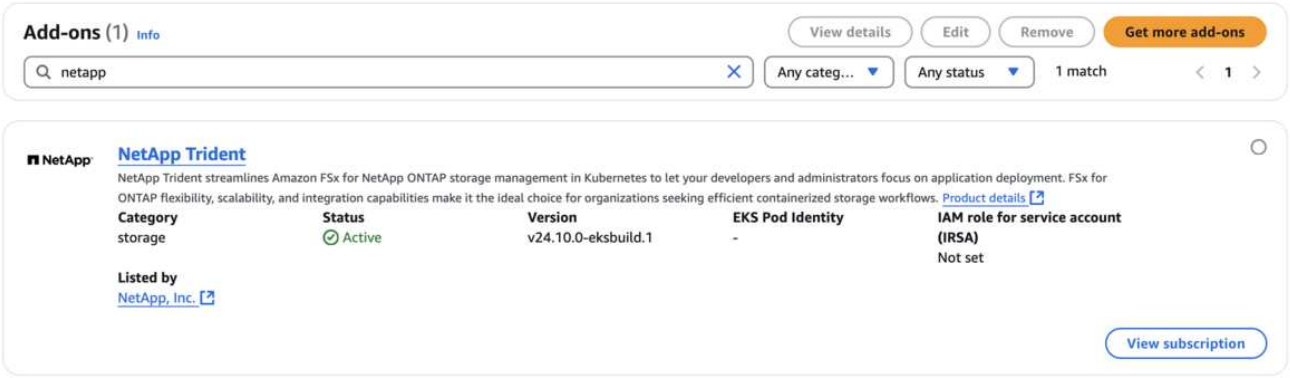


6. 필요한 추가 기능 설정을 구성합니다.



7. IRSA(서비스 계정에 대한 IAM 역할)를 사용하는 경우 추가 구성 단계를 참조하세요."여기" .
8. Create * 를 선택합니다.

9. 애드온의 상태가 _Active_인지 확인합니다.



10. 다음 명령을 실행하여 Trident가 클러스터에 올바르게 설치되어 있는지 확인합니다.

```
kubectl get pods -n trident
```

11. 설치를 계속하고 스토리지 백엔드를 구성합니다. 자세한 내용은 을 "스토리지 백엔드를 구성합니다"참조하십시오.

CLI를 사용하여 **Trident EKS** 애드온을 설치/제거합니다

CLI를 사용하여 **NetApp Trident EKS** 추가 기능을 설치합니다**.**

다음 예제 명령은 Trident EKS 추가 기능을 설치합니다.

`eksctl create addon --cluster clusterName --name netapp_trident-operator --version v25.6.0-eksbuild.1` (전용 버전 포함)

다음 예시 명령은 Trident EKS 애드온 버전 25.6.1을 설치합니다:

`eksctl create addon --cluster clusterName --name netapp_trident-operator --version v25.6.1-eksbuild.1` (전용 버전 포함)

다음 예시 명령은 Trident EKS 애드온 버전 25.6.2를 설치합니다:

`eksctl create addon --cluster clusterName --name netapp_trident-operator --version v25.6.2-eksbuild.1` (전용 버전 포함)

CLI를 사용하여 **NetApp Trident EKS** 애드온을 제거합니다**.**

다음 명령을 실행하면 Trident EKS 추가 기능이 제거됩니다.

```
eksctl delete addon --cluster K8s-arm --name netapp_trident-operator
```

kubectl로 백엔드를 만듭니다

백엔드는 Trident와 스토리지 시스템 간의 관계를 정의합니다. Trident는 해당 스토리지 시스템과 통신하는 방법과 Trident가 스토리지 시스템에서 볼륨을 프로비저닝하는 방법을 알려줍니다. Trident를 설치한 후 다음 단계는 백엔드를 생성하는 것입니다.

`TridentBackendConfig`CRD(사용자 지정 리소스 정의)를 사용하면 Kubernetes

인터페이스를 통해 Trident 백엔드를 직접 생성하고 관리할 수 있습니다. 이 작업은 또는 Kubernetes 배포에 해당하는 CLI 툴을 사용하여 수행할 수 있습니다 `kubectl.

TridentBackendConfig

TridentBackendConfig(tbc tbconfig tbackendconfig, ,)는 을 사용하여 Trident 백엔드를 관리할 수 있는 프런트 엔드 이름 기반 CRD `kubectl`입니다. 이제 Kubernetes 및 스토리지 관리자는 전용 명령줄 유틸리티 없이 Kubernetes CLI를 통해 직접 백엔드를 만들고 관리할 수 (`tridentctl`있습니다.

'트리펜엔드구성' 객체가 생성되면 다음과 같은 현상이 발생합니다.

- 백엔드는 사용자가 제공하는 구성에 따라 Trident에 의해 자동으로 생성됩니다. 이 값은 내부적으로 a TridentBackend (tbe, tridentbackend) CR로 표시됩니다.
- 는 TridentBackendConfig Trident에서 만든 에 고유하게 바인딩됩니다. TridentBackend

각각의 트리젠백엔드구성은 트리젠백엔드(트리젠백엔드)를 통해 일대일 매핑을 유지합니다. 전자는 백엔드를 설계 및 구성하기 위해 사용자에게 제공되는 인터페이스이며, 후자는 Trident가 실제 백엔드 객체를 나타내는 방법입니다.

> TridentBackend CRS는 Trident에 의해 자동으로 생성됩니다. 수정할 수 없습니다. 백엔드를 업데이트하려면 객체를 수정하여 이 작업을 TridentBackendConfig 수행합니다.

'트리엔백엔드구성' CR의 형식은 다음 예를 참조하십시오.

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret
```

의 예를 살펴볼 수도 있습니다 "Trident - 장착 공구" 원하는 스토리지 플랫폼/서비스의 샘플 구성을 위한 디렉토리입니다.

를 클릭합니다 spec 백엔드 관련 구성 매개 변수를 사용합니다. 이 예에서는 백엔드에서 를 사용합니다 ontap-san 여기에 표로 제공된 구성 매개 변수를 사용하여 스토리지 드라이버를 다운로드합니다. 원하는 스토리지 드라이버에 대한 구성 옵션 목록은 을 참조하십시오 "스토리지 드라이버에 대한 백엔드 구성 정보입니다".

이번 시기에는 트리젠백엔드Config CR에 새로 도입된 자격 증명과 delitionPolicy 필드가 포함되어 있습니다.

- "credentials": 이 매개변수는 필수 필드이며 스토리지 시스템/서비스를 인증하는 데 사용되는 자격 증명을 포함합니다. 사용자 생성 Kubernetes Secret으로 설정됩니다. 자격 증명을 일반 텍스트로 전달할 수 없으며 오류가 발생합니다.
- "설명 정책": 이 필드는 트리젠BackendConfig가 삭제될 때 발생하는 동작을 정의합니다. 다음 두 가지 값 중 하나를 사용할 수 있습니다.
  - 삭제(Delete): 이 경우 트리엔백엔드Config CR과 관련 백엔드가 모두 삭제됩니다. 이 값이 기본값입니다.
  - [Tain]: 트리덴트Config CR이 삭제되면 백엔드 정의가 계속 존재하고 tridentctl로 관리될 수 있다. 삭제 정책을 "보존"으로 설정하면 사용자는 이전 릴리스(21.04 이전)로 다운그레이드하고 생성된 백엔드를 유지할 수 있습니다. 이 필드의 값은 '트리엔백엔드구성'이 생성된 후에 업데이트할 수 있습니다.

백엔드 이름은 'pec.backendName'을 사용하여 설정됩니다. 지정하지 않으면 백엔드 이름이 '트리엔백엔드구성' 객체(metadata.name 지정합니다. 'pec.backendName'을 사용하여 백엔드 이름을 명시적으로 설정하는 것이 좋습니다.

로 만든 백엔드에는 `tridentctl` 연결된 `TridentBackendConfig` 개체가 없습니다. CR을 만들어 `TridentBackendConfig` 이러한 백엔드를 관리하도록 선택할 수 `kubectl` 있습니다. 동일한 구성 매개 변수(`spec.storagePrefix`,, `spec.storageDriverName` 등)를 지정할 때 주의를 기울여야 `spec.backendName` 합니다. Trident는 새로 생성된 를 기존 백엔드와 자동으로 바인딩합니다. `TridentBackendConfig`

단계 개요

kubbeck을 사용하여 새 백엔드를 생성하려면 다음을 수행해야 합니다.

1. 를 "쿠버네티스 비밀"생성합니다. 비밀에는 Trident가 스토리지 클러스터/서비스와 통신하는 데 필요한 자격 증명이 포함되어 있습니다.
2. '트리멘백엔드구성' 객체를 만듭니다. 스토리지 클러스터/서비스에 대한 자세한 내용과 이전 단계에서 생성한 암호를 참조하십시오.

백엔드를 생성한 후 "kubbtl get tbc<tbc-name>-n<trident-namespace>"를 사용하여 해당 상태를 관찰하고 추가 세부 정보를 수집할 수 있습니다.

1단계: Kubernetes Secret 생성

백엔드에 대한 액세스 자격 증명이 포함된 암호를 생성합니다. 이는 각 스토리지 서비스/플랫폼마다 다릅니다. 예를 들면 다음과 같습니다.

```
kubectl -n trident create -f backend-tbc-ontap-san-secret.yaml
```

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-san-secret
type: Opaque
stringData:
  username: cluster-admin
  password: password
```

이 표에는 각 스토리지 플랫폼의 비밀에 포함되어야 하는 필드가 요약되어 있습니다.

스토리지 플랫폼 암호 필드 설명입니다	비밀	필드 설명입니다
Azure NetApp Files	클라이언트 ID입니다	앱 등록에서 클라이언트 ID
요소(NetApp HCI/SolidFire)	엔드포인트	테넌트 자격 증명이 있는 SolidFire 클러스터의 MVIP입니다
ONTAP	사용자 이름	클러스터/SVM에 연결할 사용자 이름입니다. 자격 증명 기반 인증에 사용됩니다
ONTAP	암호	클러스터/SVM에 연결하는 암호 자격 증명 기반 인증에 사용됩니다
ONTAP	clientPrivateKey를 선택합니다	Base64 - 클라이언트 개인 키의 인코딩된 값입니다. 인증서 기반 인증에 사용됩니다
ONTAP	챕터 사용자 이름	인바운드 사용자 이름입니다. useCHAP = TRUE인 경우 필수입니다. ONTAP-SAN과 ONTAP-SAN 경제입니다
ONTAP	챕터시토시크릿	CHAP 이니시에이터 암호입니다. useCHAP = TRUE인 경우 필수입니다. ONTAP-SAN과 ONTAP-SAN 경제입니다
ONTAP	chapTargetUsername 을 선택합니다	대상 사용자 이름입니다. useCHAP = TRUE인 경우 필수입니다. ONTAP-SAN과 ONTAP-SAN 경제입니다

스토리지 플랫폼 암호 필드 설명입니다	비밀	필드 설명입니다
ONTAP	챕터타겟이니터시크릿	CHAP 타겟 이니시에이터 암호입니다. useCHAP = TRUE인 경우 필수입니다. ONTAP-SAN과 ONTAP-SAN 경제입니다

이 단계에서 만든 암호는 다음 단계에서 만든 트리젠백엔드Config 개체의 '증명서' 필드에 참조됩니다.

2단계: 을 작성합니다 `TridentBackendConfig` 있습니다

이제 '트리엔백구성' CR을 만들 준비가 되었습니다. 이 예에서 'ONTAP-SAN' 드라이버를 사용하는 백엔드는 아래에 나와 있는 ' TridentBackendConfig ' 객체를 사용하여 생성합니다.

```
kubectl -n trident create -f backend-tbc-ontap-san.yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret
```

3단계: 의 상태를 확인합니다 `TridentBackendConfig` 있습니다

이제 '트리펜엔드구성' CR을 생성했으므로 상태를 확인할 수 있습니다. 다음 예를 참조하십시오.

```
kubectl -n trident get tbc backend-tbc-ontap-san
NAME                       BACKEND NAME          BACKEND UUID
PHASE   STATUS
backend-tbc-ontap-san   ontap-san-backend     8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8   Bound   Success
```

백엔드가 성공적으로 생성되어 '트리엔백엔드구성' CR에 바인딩되었습니다.

위상은 다음 값 중 하나를 사용할 수 있습니다.

- `Bound`: `TridentBackendConfig` CR은 백엔드에 연결되어 있으며 해당 백엔드에는 가 포함되어 있습니다 `configRef` 로 설정합니다 `TridentBackendConfig` Cr'uid(CR'uid)
- 'Unbound': "로 표현됨. 트리젠백엔드Config 객체가 백엔드에 바인딩되지 않습니다. 새로 만든 트리젠백엔드Config CRS는 기본적으로 이 단계에 있습니다. 단계가 변경된 후에는 다시 바인딩되지 않은 상태로 되돌릴 수 없습니다.
- `Deleting`: `TridentBackendConfig` CR의 `deletionPolicy` 이(가) 삭제되도록 설정되었습니다. 를 누릅니다 `TridentBackendConfig` CR이 삭제되어 삭제 상태로 전환됩니다.
  - 백엔드에 영구 볼륨 클레임(PVC)이 없는 경우 을 `TridentBackendConfig` 삭제하면 Trident이 백엔드와 `TridentBackendConfig` CR을 삭제합니다.
  - 백엔드에 PVC가 하나 이상 있는 경우 삭제 상태로 전환됩니다. 이후 트리젠백엔드Config CR도 삭제 단계로 진입한다. 모든 PVC가 삭제된 후에만 백엔드 및 트리젠백엔드구성이 삭제됩니다.
- 손실: 트리젠백엔드Config CR과 관련된 백엔드가 실수로 또는 고의적으로 삭제되었고, 트리젠백엔드Config CR에는 삭제된 백엔드에 대한 참조가 여전히 있습니다. 이 경우에도 '항목 정책' 값에 관계없이 '트리멘백엔드구성' CR은 삭제할 수 있습니다.
- `Unknown`: Trident가 CR과 연결된 백엔드의 상태 또는 존재를 확인할 수 `TridentBackendConfig` 없습니다. 예를 들어, API 서버가 응답하지 않거나 CRD가 누락된 경우 `tridentbackends.trident.netapp.io` 이 경우 개입이 필요할 수 있습니다.

이 단계에서는 백엔드가 성공적으로 생성됩니다! 다음과 같은 몇 가지 작업을 추가로 처리할 수 있습니다 "백엔드 업데이트 및 백엔드 삭제".

(선택 사항) 4단계: 자세한 내용을 확인하십시오

다음 명령을 실행하여 백엔드에 대한 자세한 정보를 얻을 수 있습니다.

```
kubectl -n trident get tbc backend-tbc-ontap-san -o wide
```

```
NAME                      BACKEND NAME         BACKEND UUID
PHASE    STATUS     STORAGE DRIVER    DELETION POLICY
backend-tbc-ontap-san    ontap-san-backend    8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8    Bound    Success    ontap-san         delete
```

또한 '트리엔백구성'의 YAML/JSON 덤프를 얻을 수도 있습니다.

```
kubectl -n trident get tbc backend-tbc-ontap-san -o yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  creationTimestamp: 2021-04-21T20:45:11Z
  finalizers:
    - trident.netapp.io
  generation: 1
  name: backend-tbc-ontap-san
  namespace: trident
  resourceVersion: "947143"
  uid: 35b9d777-109f-43d5-8077-c74a4559d09c
spec:
  backendName: ontap-san-backend
  credentials:
    name: backend-tbc-ontap-san-secret
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  storageDriverName: ontap-san
  svm: trident_svm
  version: 1
status:
  backendInfo:
    backendName: ontap-san-backend
    backendUUID: 8d24fce7-6f60-4d4a-8ef6-bab2699e6ab8
  deletionPolicy: delete
  lastOperationStatus: Success
  message: Backend 'ontap-san-backend' created
  phase: Bound
```

`backendInfo` CR에 대한 응답으로 생성된 백엔드의 `TridentBackendConfig` 및 가 `backendUUID` 포함되어 `backendName` 있습니다. 이 `lastOperationStatus` 필드는 CR의 마지막 작업 상태를 나타냅니다. 이 상태는 `TridentBackendConfig` 사용자가 트리거하거나(예: 사용자가 변경한 내용) Trident에 의해 트리거될 수 있습니다 (예: `spec` Trident 재시작 중). 성공 또는 실패일 수 있습니다. `phase` CR과 백엔드 간의 관계 상태를 `TridentBackendConfig` 나타냅니다. 위의 예에서 에는 `phase` 값이 바인딩되어 있습니다. 즉, CR이 백엔드와 연결되어 있음을 `TridentBackendConfig` 의미합니다.

"kubbctl -n trident tbc <tbc-cr-name>" 명령을 실행하여 이벤트 로그의 세부 정보를 확인할 수 있습니다.

tridentctl을 사용하여 연결된 'TrientBackendConfig' 객체가 포함된 백엔드는 업데이트하거나 삭제할 수 없습니다. tridentctl과 트리멘BackendConfig의 전환 단계를 이해하려면 "여기 를 참조하십시오".

백엔드 관리

kubeck을 사용하여 백엔드 관리 수행

kubbtl을 사용하여 백엔드 관리 작업을 수행하는 방법에 대해 알아보십시오.

백엔드를 삭제합니다

를 삭제하면 `TridentBackendConfig` Trident가 백엔드를 삭제/유지하도록 지시합니다(기준 `deletionPolicy`). 백엔드를 삭제하려면 가 `deletionPolicy` DELETE로 설정되어 있는지 확인합니다. 만 삭제하려면 `TridentBackendConfig` 가 유지 로 설정되어 있는지 `deletionPolicy` 확인합니다. 이렇게 하면 백엔드가 계속 존재하고 을 사용하여 관리할 수 `tridentctl` 있습니다.

다음 명령을 실행합니다.

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

Trident은 에서 사용 중인 Kubernetes 비밀을 삭제하지 `TridentBackendConfig` 않습니다. Kubernetes 사용자는 기밀을 정해야 합니다. 비밀 정보를 삭제할 때는 주의해야 합니다. 암호는 백엔드에서 사용하지 않는 경우에만 삭제해야 합니다.

기존 백엔드를 봅니다

다음 명령을 실행합니다.

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

또한 'tridentctl get backend-n trident' 또는 'tridentctl get backend-o YAML-n trident'를 실행하여 존재하는 모든 백엔드 목록을 확인할 수 있습니다. 이 목록에는 tridentctl로 만든 백엔드 또한 포함됩니다.

백엔드를 업데이트합니다

백엔드를 업데이트해야 하는 이유는 여러 가지가 있을 수 있습니다.

- 스토리지 시스템에 대한 자격 증명이 변경되었습니다. 자격 증명을 업데이트하려면 개체에서 사용되는 Kubernetes 암호를 `TridentBackendConfig` 업데이트해야 합니다. Trident는 제공된 최신 자격 증명으로 백엔드를 자동으로 업데이트합니다. 다음 명령을 실행하여 Kubernetes Secret를 업데이트하십시오.

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

- 매개 변수(예: 사용 중인 ONTAP SVM의 이름)를 업데이트해야 합니다.
  - 업데이트할 수 있습니다 `TridentBackendConfig` 다음 명령을 사용하여 Kubernetes를 통해 직접 오브젝트를 탐색합니다.

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

  - 또는 기존 을 변경할 수 있습니다 `TridentBackendConfig` 다음 명령을 사용하는 CR:

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

- 백엔드 업데이트에 실패하면 백엔드는 마지막으로 알려진 구성으로 계속 유지됩니다. 'kubtl get tbc<tbc-name>-o YAML-n trident' 또는 'kubtl t설명해 tbc<tbc-name>-n trident'를 실행하여 로그를 보고 원인을 확인할 수 있습니다.
- 구성 파일의 문제를 확인하고 수정한 후 update 명령을 다시 실행할 수 있습니다.

tridentctl을 사용하여 백엔드 관리를 수행합니다

tridentctl을 사용하여 백엔드 관리 작업을 수행하는 방법에 대해 알아보십시오.

백엔드를 생성합니다

을 만든 후 "백엔드 구성 파일"에서 다음 명령을 실행합니다.

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

백엔드 생성에 실패하면 백엔드 구성에 문제가 있는 것입니다. 다음 명령을 실행하여 로그를 보고 원인을 확인할 수 있습니다.

```
tridentctl logs -n trident
```

구성 파일의 문제를 확인하고 수정한 후에는 간단히 'create' 명령을 다시 실행할 수 있습니다.

백엔드를 삭제합니다

Trident에서 백엔드를 삭제하려면 다음을 수행합니다.

1. 백엔드 이름 검색:

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

2. 백엔드를 삭제합니다.

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

Trident가 이 백엔드에서 아직 존재하는 볼륨 및 스냅샷을 프로비저닝한 경우 백엔드를 삭제하면 새 볼륨이 프로비저닝되지 않습니다. 백엔드는 계속해서 "삭제 중" 상태로 유지됩니다.

기존 백엔드를 봅니다

Trident가 알고 있는 백엔드를 보려면 다음을 실행합니다.

- 요약을 보려면 다음 명령을 실행합니다.

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

- 모든 세부 정보를 보려면 다음 명령을 실행합니다.

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

백엔드를 업데이트합니다

새 백엔드 구성 파일을 생성한 후 다음 명령을 실행합니다.

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

백엔드 업데이트에 실패하면 백엔드 구성에 문제가 있거나 잘못된 업데이트를 시도했습니다. 다음 명령을 실행하여 로그를 보고 원인을 확인할 수 있습니다.

```
tridentctl logs -n trident
```

구성 파일의 문제를 확인하고 수정한 후에는 간단히 'update' 명령을 다시 실행할 수 있습니다.

백엔드를 사용하는 스토리지 클래스를 식별합니다

이것은 JSON으로 백엔드 객체에 대해 tridentctl이 출력하는 질문의 예입니다. 이 유틸리티는 설치해야 하는 JQ 유틸리티를 사용합니다.

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

이는 ' 재젠백엔드구성'을 사용하여 만든 백엔드에도 적용됩니다.

백엔드 관리 옵션 간 이동

Trident에서 백엔드를 관리하는 다양한 방법에 대해 알아보십시오.

백엔드 관리 옵션

을 소개합니다 `TridentBackendConfig`이제 관리자는 두 가지 고유한 방식으로 백엔드를 관리할 수 있습니다. 이 질문은 다음과 같습니다.

- tridentctl을 사용하여 만든 백엔드는 트리엔백엔드구성을 사용하여 관리할 수 있습니까?
- 트리덴ctl을 사용하여 트리엔디Config를 사용하여 만든 백엔드를 관리할 수 있습니까?

관리 `tridentctl` 을 사용하여 백엔드를 만듭니다 `TridentBackendConfig`

이 섹션에서는 'Tridentctl' 객체를 만들어 Kubernetes 인터페이스를 통해 직접 생성된 백엔드를 관리하는 데 필요한 단계에 대해 설명합니다.

이 내용은 다음 시나리오에 적용됩니다.

- 가 없는 기존 백엔드가 있습니다 `TridentBackendConfig` 을 사용하여 생성되었기 때문입니다 `tridentctl`.
- tridentctl로 만든 새 백엔드와 다른 트리젠BackendConfig 개체가 있습니다.

두 시나리오 모두에서 Trident의 볼륨 예약 및 운영 기능과 함께 백엔드가 계속 표시됩니다. 관리자는 다음 두 가지 옵션 중 하나를 선택할 수 있습니다.

- tridentctl을 사용하여 만든 백엔드를 관리하려면 계속 사용합니다.
- tridentctl을 사용하여 만든 백엔드를 새 트리젠BackendConfig 개체에 바인딩합니다. 이를 통해 뒷골은 트리덴틀이 아니라 쿠벤틀로 관리된다는 뜻이다.

kubbeck을 사용하여 기존 백엔드를 관리하려면 기존 백엔드에 바인딩되는 '트리젠백엔드구성'을 생성해야 합니다. 작동 방식에 대한 개요는 다음과 같습니다.

1. Kubernetes 암호를 생성하십시오. 비밀에는 Trident가 스토리지 클러스터/서비스와 통신하는 데 필요한 자격 증명이 포함되어 있습니다.
2. '트리멘백엔드구성' 객체를 만듭니다. 스토리지 클러스터/서비스에 대한 자세한 내용과 이전 단계에서 생성한 암호를 참조하십시오. 동일한 구성 매개 변수(예: 'pec.backendName', 'pec.storagePrefix', 'pec.storageDriverName' 등)를 지정할 때는 주의해야 합니다. '현재 백엔드 이름'을 설정해야 합니다.

단계 **0:** 백엔드를 식별합니다

을(를) 생성합니다 `TridentBackendConfig` 이 경우 기존 백엔드에 바인딩되므로 백엔드 구성을 확보해야 합니다. 이 예에서는 다음과 같은 JSON 정의를 사용하여 백엔드를 생성했다고 가정합니다.

```
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"
      }
    }
  ]
}
```

1단계: **Kubernetes Secret** 생성

이 예에 표시된 것처럼 백엔드에 대한 자격 증명이 포함된 암호를 생성합니다.

```
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
```

2단계: **A**를 작성합니다 `TridentBackendConfig` 있습니다

다음 단계는 기존 ONTAP-NAS-백엔드에 자동으로 바인딩되는 '트리젠백엔드구성' CR을 생성하는 것입니다(예:). 다음 요구 사항이 충족되는지 확인합니다.

- 같은 백엔드 이름은 'sepec.backendName'에 정의되어 있습니다.
- 구성 매개 변수는 원래 백엔드와 동일합니다.
- 가상 풀(있는 경우)은 원래 백엔드와 동일한 순서를 유지해야 합니다.
- 자격 증명은 일반 텍스트가 아닌 Kubernetes Secret을 통해 제공됩니다.

이 경우 트리젠백엔드구성은 다음과 같습니다.

```
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
```

3단계: 의 상태를 확인합니다 `TridentBackendConfig` 있습니다

트리젠백엔드구성이 만들어지면 그 단계는 반드시 '바운드'되어야 한다. 또한 기존 백엔드의 백엔드 이름과 UUID도 동일하게 반영되어야 합니다.

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

이제 백엔드는 'tbc-ONTAP-nas-backend' 트리펜엔드구성 객체를 사용하여 완벽하게 관리됩니다.

관리 `TridentBackendConfig` 을 사용하여 백엔드를 만듭니다 `tridentctl`

트리덴ctl은 트리엔백구성(TrientBackendConfig)을 사용하여 만든 백엔드를 나열하는 데 사용할 수 있습니다. 또한 관리자는 트리텐틀Config를 삭제하고 pec.deletionPolicy` 가 "Stain"으로 설정되어 있는지 확인하여 tridentctl을 통해 이러한 백엔드를 완벽하게 관리할 수도 있습니다.

단계 **0:** 백엔드를 식별합니다

예를 들어, 다음 백엔드가 ``트리엔백구성”을 사용하여 생성되었다고 가정해 보겠습니다.

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

출력에서 해당 결과가 표시됩니다 `TridentBackendConfig` 성공적으로 생성되었으며 백엔드에 바인딩되었습니다 [백엔드의 UUID 관찰].

1단계: 확인 `deletionPolicy` 가 로 설정되어 있습니다 `retain`

의 가치를 `deletionPolicy` 살펴보겠습니다. 이 설정은 로 `retain`**설정해야 합니다. 이렇게 하면** `CR`**이 삭제되어도** `TridentBackendConfig` 백엔드 정의가 계속 존재하며 로 관리할 수 `tridentctl` 있습니다.

```
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
```

'정책'이 '유지'로 설정되어 있지 않으면 다음 단계로 진행하지 마십시오.

2단계: 를 삭제합니다 `TridentBackendConfig` 있습니다

마지막 단계는 트리엔디Config CR을 삭제하는 것이다. '정책'이 '유지'로 설정되어 있는지 확인한 후 삭제를 계속 수행할 수 있습니다.

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

오브젝트가 삭제되면 `TridentBackendConfig` Trident은 백엔드 자체를 실제로 삭제하지 않고 기존 오브젝트를 단순히 제거합니다.

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.