



백엔드 구성 및 관리

Astra Trident

NetApp
April 18, 2024

목차

백엔드 구성 및 관리	1
백엔드 구성	1
Azure NetApp Files	1
Google Cloud용 Cloud Volumes Service 백엔드를 구성합니다	17
NetApp HCI 또는 SolidFire 백엔드를 구성합니다	33
ONTAP SAN 드라이버	39
ONTAP NAS 드라이버	61
NetApp ONTAP용 Amazon FSx	89
kubectl로 백엔드를 만듭니다	106
백엔드 관리	112

백엔드 구성 및 관리

백엔드 구성

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

Astra Trident는 스토리지 클래스에 정의된 요구 사항과 일치하는 백엔드에서 스토리지 풀을 자동으로 제공합니다. 스토리지 시스템에 대한 백엔드를 구성하는 방법에 대해 알아봅니다.

- ["Azure NetApp Files 백엔드를 구성합니다"](#)
- ["Google Cloud Platform 백엔드에 Cloud Volumes Service를 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"](#)
- ["NetApp ONTAP용 Amazon FSx와 함께 Astra Trident를 사용하십시오"](#)

Azure NetApp Files

Azure NetApp Files 백엔드를 구성합니다

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

Azure NetApp Files 드라이버 세부 정보입니다

Astra Trident는 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 100GB 미만의 볼륨을 지원하지 않습니다. Astra Trident는 더 작은 볼륨을 요청하는 경우 100GiB 볼륨을 자동으로 생성합니다.
- Astra Trident는 Windows 노드에서 실행되는 Pod에만 마운트된 SMB 볼륨을 지원합니다.

AKS의 관리되는 ID입니다

Astra Trident가 지원합니다 **"관리되는 ID입니다"** NetApp 프라이빗 클라우드 서비스 클러스터의 경우 관리되는 ID에서 제공하는 효율적인 자격 증명 관리를 활용하려면 다음을 수행해야 합니다.

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

Trident 운영자

Trident 연산자를 사용하여 Astra Trident를 설치하려면 편집을 진행합니다

tridentorchestrator_cr.yaml 를 눌러 설정합니다 cloudProvider 를 선택합니다 "Azure".

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 Astra Trident 세트를 설치합니다 cloudProvider Azure에 연결할 수 있습니다 \$CP:

```
helm install trident trident-operator-100.2402.0.tgz --create
--namespace --namespace <trident-namespace> --set cloudProvider=$CP
```

<code>tridentctl</code>

다음 예에서는 Astra Trident를 설치하고 를 설정합니다 cloudProvider 에 플래그 지정 Azure:

```
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
- 가 포함된 Astra Trident가 설치되었습니다 cloudProvider 지정합니다 "Azure" 및 cloudIdentity 작업 부하 ID 지정

Trident 운영자

Trident 연산자를 사용하여 Astra Trident를 설치하려면 편집을 진행합니다

tridentorchestrator_cr.yaml 를 눌러 설정합니다 cloudProvider 를 선택합니다 "Azure" 그리고 설정합니다 cloudIdentity 를 선택합니다 azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx.

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-xxxxxxxxxxxxx' *
```

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-xxxxxxxxxxxxx"
```

다음 예에서는 Astra Trident 및 Sets를 설치합니다 cloudProvider Azure에 연결할 수 있습니다 \$CP 및 를 설정합니다 cloudIdentity 환경 변수를 사용합니다 \$CI:

```
helm install trident trident-operator-100.2402.0.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-xxxxxxxxxxxxx"
```

다음 예에서는 Astra Trident를 설치하고 를 설정합니다 cloud-provider 에 플래그 지정 \$CP, 및 cloud-identity 를 선택합니다 \$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 레벨입니다 (assignableScopes) 다음 사용 권한은 Astra Trident에 필요한 권한만 가집니다. 사용자 지정 역할을 만든 후 ["Azure 포털을 사용하여 역할을 할당합니다"](#).

```
{
  "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/MountTargets/read",

          "Microsoft.Network/virtualNetworks/read",

"Microsoft.Network/virtualNetworks/subnets/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrations/read",
        ]
      }
    ]
  }
}
```



```

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrations/write",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrations/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 ID에서 ["사용자가 할당한 관리 ID입니다"](#)에서 해당 ID를 지정합니다 azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx.

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 2019를 실행하는 Windows 작업자 노드가 있는 Kubernetes 클러스터 Astra Trident는 Windows 노드에서 실행되는 Pod에만 마운트된 SMB 볼륨을 지원합니다.
- Azure NetApp Files가 Active Directory에 인증할 수 있도록 Active Directory 자격 증명이 포함된 Astra Trident 암호가 하나 이상 있어야 합니다. 암호를 생성합니다 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

Astra Trident는 백엔드 구성(서브넷, 가상 네트워크, 서비스 수준 및 위치)을 사용하여 요청된 위치에서 사용할 수 있고 요청된 서비스 수준 및 서브넷과 일치하는 용량 풀에 Azure NetApp Files 볼륨을 생성합니다.



Astra Trident는 수동 QoS 용량 풀을 지원하지 않습니다.

Azure NetApp Files 백엔드는 이러한 구성 옵션을 제공합니다.

매개 변수	설명	기본값
'내전'		항상 1
'storageDriverName'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.	
tenantID	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.	

매개 변수	설명	기본값
[재치 단체	검색된 자원을 필터링하기 위한 자원 그룹 목록입니다	"[]"(필터 없음)
'netap계정'	검색된 리소스를 필터링하기 위한 NetApp 계정의 목록입니다	"[]"(필터 없음)
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 심표로 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



네트워크 기능에 대한 자세한 내용은 을 참조하십시오 ["Azure NetApp Files 볼륨에 대한 네트워크 기능을 구성합니다"](#).

필요한 권한 및 리소스

PVC를 생성할 때 "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 Astra 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.

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](#) 을 참조하십시오.

매개 변수	설명	기본값
엑포트 규칙	새 볼륨에 대한 엑스포트 규칙 exportRule CIDR 표기법을 사용하여 IPv4 주소 또는 IPv4 서브넷의 조합을 심표로 구분해야 합니다. SMB 볼륨에 대해 무시됩니다.	"0.0.0.0/0"
나프산디렉토리	스냅샷 디렉터리의 표시 여부를 제어합니다	"거짓"
'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 Astra Trident는 구성된 위치에서 Azure NetApp Files에 위임된 모든 NetApp 계정, 용량 풀 및 서브넷을 검색하고 이러한 풀과 서브넷 중 하나에 무작위로 새 볼륨을 배치합니다. 왜냐하면 `nasType` 생략됩니다 `nfs` 기본값은 NFS 볼륨에 대해 백엔드가 프로비저닝됩니다.

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

```
---
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 ID입니다

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: ["ultra-pool"]
  resourceGroups: ["aks-ami-eastus-rg"]
  netappAccounts: ["smb-na"]
  virtualNetwork: eastus-prod-vnet
  subnet: 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 Azure에 볼륨을 배치합니다 eastus 의 위치 Ultra 용량 풀. Astra Trident는 해당 위치에서 Azure NetApp Files에 위임된 모든 서브넷을 자동으로 검색하여 이 중 하나에 무작위로 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
```

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: my-virtual-network
subnet: 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'
```

가상 풀 구성

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:
    - ultra-1
    - ultra-2
    networkFeatures: Standard
- labels:
    performance: silver
    serviceLevel: Premium
    capacityPools:
    - premium-1
- labels:
    performance: bronze
    serviceLevel: Standard
    capacityPools:
    - standard-1
    - standard-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 Cloud용 Cloud Volumes Service 백엔드를 구성합니다

제공된 샘플 구성을 사용하여 Astra Trident 설치를 위한 백엔드로 NetApp Cloud Volumes Service for Google Cloud를 구성하는 방법을 알아보십시오.

Google Cloud 드라이버 세부 정보입니다

Astra Trident가 제공하는 것은 다음과 같습니다 gcp-cvs 클러스터와 통신하는 드라이버입니다. 지원되는 액세스 모드는 *ReadWriteOnce(RWO)*, *ReadOnlyMany(ROX)*, *ReadWriteMany(rwx)*, *ReadWriteOncePod (RWOP)*입니다.

드라이버	프로토콜	볼륨 모드	액세스 모드가 지원됩니다	지원되는 파일 시스템
GCP-CV	NFS 를 참조하십시오	파일 시스템	RWO, ROX, rwx, RWOP	nfs

Cloud Volumes Service for Google Cloud를 위한 Astra Trident 지원에 대해 알아보십시오

Astra Trident는 두 개 중 하나로 Cloud Volumes Service 볼륨을 생성할 수 있습니다 "[서비스 유형](#)":

- * CVS - 성능 *: 기본 Astra Trident 서비스 유형입니다. 이처럼 성능에 최적화된 서비스 유형은 성능을 중요시하는 운영 워크로드에 가장 적합합니다. CVS - 성능 서비스 유형은 최소 100GiB 크기의 볼륨을 지원하는 하드웨어 옵션입니다. 다음 중 하나를 선택할 수 있습니다 "[3가지 서비스 레벨](#)":
 - standard
 - premium
 - extreme
- * CVS *: CVS 서비스 유형은 높은 조널 가용성을 제공하며, 성능은 중간 수준으로 제한됩니다. CVS 서비스 유형은

스토리지 풀을 사용하여 1GiB의 작은 볼륨을 지원하는 소프트웨어 옵션입니다. 스토리지 풀에는 최대 50개의 볼륨이 포함될 수 있으며 이 볼륨에서 풀의 용량과 성능을 공유할 수 있습니다. 다음 중 하나를 선택할 수 있습니다 ["서비스 레벨 2개"](#):

- standardsw
- zoneredundantstandardsw

필요한 것

를 구성하고 사용합니다 ["Google Cloud용 Cloud Volumes Service"](#) 백엔드, 다음이 필요합니다.

- NetApp Cloud Volumes Service로 구성된 Google Cloud 계정
- Google Cloud 계정의 프로젝트 번호입니다
- Google Cloud 서비스 계정에는 netcloudvolumes.admin 역할이 포함되어 있습니다
- Cloud Volumes Service 계정에 대한 API 키 파일입니다

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
'storageDriverName'입니다	스토리지 드라이버의 이름입니다	"GCP-CV"
백엔드이름	사용자 지정 이름 또는 스토리지 백엔드	드라이버 이름 + "_" + API 키의 일부
'원어클래스'	CVS 서비스 유형을 지정하는 데 사용되는 선택적 매개 변수입니다. 사용 software CVS 서비스 유형을 선택합니다. 그렇지 않으면 Astra Trident가 CVS - 성능 서비스 유형을 가정합니다 (hardware)를 클릭합니다.	
storagePools	CVS 서비스 유형만 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.	
hostProjectNumber입니다	공유 VPC 네트워크를 사용하는 경우 필요합니다. 이 시나리오에서는 projectNumber 은(는) 서비스 프로젝트입니다 hostProjectNumber 는 호스트 프로젝트입니다.	
아피지역	Astra Trident가 Cloud Volumes Service 볼륨을 생성하는 Google 클라우드 영역 지역 간 Kubernetes 클러스터를 생성할 때 에서 생성된 볼륨입니다 apiRegion 여러 Google Cloud 지역의 노드에 예약된 워크로드에 사용할 수 있습니다. 지역 간 트래픽에는 추가 비용이 발생합니다.	

매개 변수	설명	기본값
아피키	를 사용하여 Google Cloud 서비스 계정에 대한 API 키입니다 <code>netappcloudvolumes.admin</code> 역할. 여기에는 Google Cloud 서비스 계정의 개인 키 파일 (백엔드 구성 파일에 verbatim 복사)의 JSON 형식 콘텐츠가 포함됩니다.	
proxyURL	프록시 서버가 CVS 계정에 연결해야 하는 경우 프록시 URL입니다. 프록시 서버는 HTTP 프록시 또는 HTTPS 프록시일 수 있습니다. HTTPS 프록시의 경우 프록시 서버에서 자체 서명된 인증서를 사용할 수 있도록 인증서 유효성 검사를 건너뛵니다. 인증이 활성화된 프록시 서버는 지원되지 않습니다.	
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'	새 볼륨에 대한 CVS - 성능 또는 CVS 서비스 수준 CV - 성능 값은 입니다 <code>standard</code> , <code>premium</code> , 또는 <code>extreme</code> . CV 값은 입니다 <code>standardsw</code> 또는 <code>zoneredundantstandardsw</code> .	CV - 성능 기본값은 "표준"입니다. CV 기본값은 "standardsw"입니다.
네트워크	Cloud Volumes Service 볼륨에 사용되는 Google Cloud 네트워크	"기본값"
debugTraceFlags를 선택합니다	문제 해결 시 사용할 디버그 플래그입니다. 예: <code>\{"api":false, "method":true\}</code> . 문제 해결 중이 아니며 자세한 로그 덤프가 필요한 경우가 아니면 이 방법을 사용하지 마십시오.	null입니다
allowedTopologies	지역 간 액세스를 설정하려면 에 대한 <code>StorageClass</code> 정의를 사용합니다 <code>allowedTopologies</code> 모든 지역을 포함해야 합니다. 예를 들면 다음과 같습니다. - <code>key: topology.kubernetes.io/region</code> <code>values:</code> - <code>us-east1</code> - <code>europa-west1</code>	

볼륨 프로비저닝 옵션

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

매개 변수	설명	기본값
엑포트 규칙	새 볼륨의 내보내기 규칙. CIDR 표기법을 사용하여 IPv4 주소 또는 IPv4 서브넷의 조합을 심표로 구분해야 합니다.	"0.0.0.0/0"
나프산디렉토리	'snapshot' 디렉토리에 액세스합니다	"거짓"
안산예비역	스냅숏용으로 예약된 볼륨의 백분율입니다	""(CVS 기본값 0 허용)

매개 변수	설명	기본값
'크기'입니다	새 볼륨의 크기입니다. CVS - 최소 성능은 100GiB입니다. CV 최소값은 1GiB입니다.	CVS - 성능 서비스 유형의 기본값은 "100GiB"입니다. CVS 서비스 유형은 기본값을 설정하지 않지만 최소 1GiB가 필요합니다.

CVS - 성능 서비스 유형의 예

다음 예에서는 CVS - 성능 서비스 유형에 대한 샘플 구성을 제공합니다.

[illegible]

```
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/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
```


예 2: 서비스 수준 구성

이 샘플에서는 서비스 수준 및 볼륨 기본값을 포함한 백엔드 구성 옵션을 보여 줍니다.

```
--  
version: 1  
storageDriverName: gcp-cvs  
projectNumber: '012345678901'  
apiRegion: us-west2  
apiKey:  
  type: service_account  
  project_id: my-gcp-project  
  private_key_id: "<id_value>"  
  private_key: |  
    -----BEGIN PRIVATE KEY-----  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
XsYg6gyxy4zq7OlwWgLwGa==  
    -----END PRIVATE KEY-----  
client_email: cloudvolumes-admin-sa@my-gcp-  
project.iam.gserviceaccount.com  
client_id: '123456789012345678901'  
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/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
proxyURL: http://proxy-server-hostname/
nfsMountOptions: vers=3,proto=tcp,timeo=600
limitVolumeSize: 10Ti
serviceLevel: premium
defaults:
  snapshotDir: 'true'
  snapshotReserve: '5'
  exportRule: 10.0.0.0/24,10.0.1.0/24,10.0.2.100
  size: 5Ti
```

예 3: 가상 풀 구성

이 샘플은 `를` 사용합니다 `storage` 가상 풀 및 `를` 구성합니다 `StorageClasses` 다시 언급한다는 것입니다. 을 참조하십시오 [스토리지 클래스 정의](#) 스토리지 클래스를 정의한 방법을 확인합니다.

이 경우 를 설정하는 모든 가상 풀에 대해 특정 기본값이 설정됩니다 snapshotReserve 5% 및 에서 exportRule 를 0.0.0.0/0으로 설정합니다. 가상 풀은 에 정의되어 있습니다 storage 섹션을 참조하십시오. 각 개별 가상 풀은 고유한 가상 풀을 정의합니다 serviceLevel 그리고 일부 풀은 기본값을 덮어씁니다. 가상 풀 레이블을 사용하여 에 따라 풀을 구분했습니다 performance 및 protection.

[illegible]

```

XsYg6gyxy4zq70lwWgLwGa==
-----END PRIVATE KEY-----
client_email: cloudvolumes-admin-sa@my-gcp-
project.iam.gserviceaccount.com
client_id: '123456789012345678901'
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/cloudvolumes-admin-
sa%40my-gcp-project.iam.gserviceaccount.com
nfsMountOptions: vers=3,proto=tcp,timeo=600
defaults:
  snapshotReserve: '5'
  exportRule: 0.0.0.0/0
labels:
  cloud: gcp
region: us-west2
storage:
- labels:
  performance: extreme
  protection: extra
  serviceLevel: extreme
  defaults:
    snapshotDir: 'true'
    snapshotReserve: '10'
    exportRule: 10.0.0.0/24
- labels:
  performance: extreme
  protection: standard
  serviceLevel: extreme
- labels:
  performance: premium
  protection: extra
  serviceLevel: premium
  defaults:
    snapshotDir: 'true'
    snapshotReserve: '10'
- labels:
  performance: premium
  protection: standard
  serviceLevel: premium
- labels:
  performance: standard
  serviceLevel: standard

```

스토리지 클래스 정의

다음 StorageClass 정의는 가상 풀 구성 예에 적용됩니다. 사용 ``parameters.selector`` 볼륨을 호스팅하는 데 사용되는 가상 풀을 각 StorageClass에 대해 지정할 수 있습니다. 볼륨은 선택한 풀에 정의된 측면을 갖습니다.

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-extra-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=extreme; protection=extra"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extreme-standard-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=premium; protection=standard"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-premium-extra-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=premium; protection=extra"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-premium
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=premium; protection=standard"
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-standard
provisioner: csi.trident.netapp.io
parameters:
  selector: "performance=standard"

```

```
allowVolumeExpansion: true
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: cvs-extra-protection
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=extra"
allowVolumeExpansion: true
```

- 첫 번째 StorageClass입니다 (cvs-extreme-extra-protection)가 첫 번째 가상 풀에 매핑됩니다. 이 풀은 스냅샷 예약 공간이 10%인 최고 성능을 제공하는 유일한 풀입니다.
- 마지막 StorageClass입니다 (cvs-extra-protection) 10%의 스냅샷 예약 공간을 제공하는 스토리지 풀을 호출합니다. Astra Trident는 선택된 가상 풀을 결정하고 스냅샷 예약 요구 사항이 충족되는지 확인합니다.

CVS 서비스 유형 예

다음 예에서는 CVS 서비스 유형에 대한 샘플 구성을 제공합니다.

예 1: 최소 구성

을 사용하는 최소 백엔드 구성입니다 `storageClass` CVS 서비스 유형과 기본값을 지정합니다 `standardsw` 서비스 레벨:

```
--  
version: 1  
storageDriverName: gcp-cvs  
projectNumber: '012345678901'  
storageClass: software  
apiRegion: us-east4  
apiKey:  
  type: service_account  
  project_id: my-gcp-project  
  private_key_id: "<id_value>"  
  private_key: |  
    -----BEGIN PRIVATE KEY-----  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    znHczZsr rtHisIsAbOguSaPIKeyAZNchRAGz lzZE4jK3bl /qp8B4Kws8zX5ojY9m  
    XsYg6gyxy4zq7OlwWgLwGa==  
    -----END PRIVATE KEY-----  
  client_email: cloudvolumes-admin-sa@my-gcp-  
project.iam.gserviceaccount.com
```



```
client_id: '123456789012345678901'  
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/cloudvolumes-admin-  
sa%40my-gcp-project.iam.gserviceaccount.com  
serviceLevel: standardsw
```

예 2: 스토리지 풀 구성

이 백엔드 구성은 를 사용합니다 storagePools 스토리지 풀을 구성하려면 다음을 수행합니다.

```
---
version: 1
storageDriverName: gcp-cvs
backendName: gcp-std-so-with-pool
projectNumber: '531265380079'
apiRegion: europe-west1
apiKey:
  type: service_account
  project_id: cloud-native-data
  private_key_id: "<id_value>"
  private_key: |-
    -----BEGIN PRIVATE KEY-----
    MIIEvAIBADANBgkqhkiG9w0BAQEFAASCBKYwggSiAgEAAoIBAQDaT+Oui9FBAw19
    L1AGEkrYU5xd9K5NlO5jMkIFND5wCD+Nv+jd1GvtFRLaLK5RvXyF5wzvztmODNS+
    qtScpQ+5cFpQkuGtv9U9+N6qtuVYYO3b504Kp5CtqVPJCgMJaK2j8pZTIqUiMum/
    5/Y9oTbZrjAHSMgJm2nHzFq2X0rqVMAHghI6ATm4DOuWx8XGWKTGIPlc0qPqJlqS
    LLaWOH4VIZQZCAyW5IU9p9CAmwqHgdG0uhFNfCgMmED6PBUvVLsLvcq86X+QSWR9k
    ETqElj/sGCenPF7ti1DhGBFafd9hPnxg9PZY29ArEZwY9G/ZjZQX7WPgs0VvxiNR
    DxZRC3GXAgMBAAECggEACn5c59bG/qnVEVI1CwMAalM5M2z09JFh1L1ljKwntNPj
    Vilw2eTW2+UE7HbJru/S7KQgA5Dnn9kvCraEahPRuddUMrD0vG4kTl/IODV6uFuk
    Y0sZfbqd4jMUQ21smvGsqFzwloYWS5qzO1W83ivXH/HW/iqkmY2eW+EPRS/hwSSu
    SscR+SojI7PB0BWSJhlV4yqYf3vcD/D95e12CVHfRCkL85DKumeZ+yHENpiXGZAE
    t8xSs4a50OPm6NHhevCw2a/UQ95/foXNUR450HtbjieJo5o+FF6EYZQGfU2ZHZO8
    37FBKuaJkdGW5xqaI9TL7aqkGkFMF4F2qvOZM+vy8QKBgQD4oVuOkJD1hkTHP86W
    esFlw1kpWyJR9ZA7LI0g/rVpslnX+XdDq0WQf4umDLNau5hYEH9LU6ZSGs1Xk3/B
    NHwR6OXFuqEKNiu83d0zSlHhTy7PZpOzdj5a/vVvQfPDMz7OvsqLRd7YCAbdzuQ0
    +Ahq0Ztwwg0HQ64hdW0ukpYRRwKBgQDgyHj98oqswoYuIa+pPlyS0pPwLmjwKyNm
    /HayzCp+Qjiyy7Tzg8AUqlH1Ou83XbV428jvg7kDhO7PCCKFq+mMmfqHmTpb0Maq
    KpKnZg4ipsqPlyHNNEOrmcailXbwIhCLewMqMrggUiLOmCw4PscL5nK+4GKu2XE1
    jLqjWAZFMQKBgFhkQ9XXRAJ1kR3XpGHOgn890pZOkCVSrqu6aUef/5KY1Fct8ew
    F/+aIxM2iQSVmWQYOvVCnhuY/F2GfAQ7d0om3decuwIOCX/xy7PjHMkLXa2uaZs4
    WR17sLduj62RqXRLX0c0QkwBiNFyHbRcpdkZJQujbYMhBa+7j7SxT4BtAoGAWMWT
    UucocRXZm/pdvz9wteNH3YDwnJLMxm1KC06qMXbBoYrliY4sm3ywJWMC+iCd/H8A
    Gecxd/xVu5mA2L2N3KMq18Zh8Th0G5DwKyDRJgOQ0Q46yuNXOoYEjlo4Wjyk8Me
    +t1Q8iK98E0UmZnhTgfSpSNElbz2AqnzQ3MN9uECgYAqdvVPnKGfvdT2ZDjyMoJ
    E89UIC41WjjJGmHsd8W65+3X0RwMzKMT6aZc5tK9J5dHvmWIETnbM+lTImdBbFga
    NWOC6f3r2xbGXHhaWSl+nobpTuvlo56ZRJVvVk7lFMsiddzMuHH8pxfgNJemwA4P
    ThDHCEjv035NNV6KyoO0tA==
    -----END PRIVATE KEY-----
  client_email: cloudvolumes-admin-sa@cloud-native-
  data.iam.gserviceaccount.com
  client_id: '107071413297115343396'
```

```

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/cloudvolumes-admin-
sa%40cloud-native-data.iam.gserviceaccount.com
storageClass: software
zone: europe-west1-b
network: default
storagePools:
- 1bc7f380-3314-6005-45e9-c7dc8c2d7509
serviceLevel: Standardsw

```

다음 단계

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.

NetApp HCI 또는 SolidFire 백엔드를 구성합니다

Astra Trident 설치 시 Element 백엔드를 생성하고 사용하는 방법을 알아보십시오.

요소 드라이버 세부 정보

Astra Trident가 제공하는 것은 다음과 같습니다 solidfire-san 클러스터와 통신하는 스토리지 드라이버입니다. 지원되는 액세스 모드는 *ReadWriteOnce(RWO)*, *ReadOnlyMany(ROX)*, *ReadWriteMany(rwx)*, *ReadWriteOncePod(RWOP)*입니다.

를 클릭합니다 solidfire-san 스토리지 드라이버는 `_FILE_AND_BLOCK_VOLUME` 모드를 지원합니다. 의 경우 Filesystem 볼륨 코드, Astra Trident가 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 사용자에게 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
'storageDriverName'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.	"{}"
테넌트이름	사용할 테넌트 이름(찾을 수 없는 경우 생성됨)	
이니토IFace	iSCSI 트래픽을 특정 호스트 인터페이스로 제한합니다	"기본값"
'UseCHAP'입니다	CHAP를 사용하여 iSCSI를 인증합니다. Astra Trident는 CHAP를 사용합니다.	참
"액세스 그룹"	사용할 액세스 그룹 ID 목록입니다	"트리덴트"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.

Astra Trident는 스토리지 풀에 있는 레이블을 프로비저닝할 때 백엔드 스토리지 LUN에 복사합니다. 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 StorageClass입니다 (`solidfire-gold-four`)는 첫 번째 가상 풀에 매핑합니다. 이 수영장은 골드 성능을 제공하는 유일한 수영장입니다 `Volume Type QoS` 않습니다. 마지막 StorageClass입니다 (`solidfire-silver`)은 뛰어난 성능을 제공하는 스토리지 풀을 호출합니다. Astra Trident가 선택한 가상 풀을 결정하고 스토리지 요구 사항을 충족시킵니다.

```

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

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



보호, 복구 및 이동성을 위해 Astra Control을 사용하는 경우 를 참조하십시오 [Astra Control 드라이버 호환성](#).

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

드라이버	프로토콜	볼륨 모드	액세스 모드가 지원됩니다	지원되는 파일 시스템
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

Astra Control 드라이버 호환성

Astra Control은 로 생성한 볼륨을 위해 원활한 보호, 재해 복구, 이동성(Kubernetes 클러스터 간에 볼륨 이동)을 제공합니다 `ontap-nas`, `ontap-nas-flexgroup`, 및 `ontap-san` 드라이버. 을 참조하십시오 ["Astra Control 복제 사전 요구 사항"](#) 를 참조하십시오.



- 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

사용자 권한

Astra Trident는 일반적으로 "admin" 클러스터 사용자 또는 "vsadmin" SVM 사용자를 사용하거나 동일한 역할을 가진 다른 이름을 가진 사용자를 사용하여 ONTAP 또는 SVM 관리자로 실행될 것으로 예상합니다. NetApp ONTAP 구축을 위한 Amazon FSx의 경우 Astra Trident는 클러스터 'fsxadmin' 사용자 또는 'vsadmin' SVM 사용자 또는 동일한 역할을 가진 다른 이름의 사용자를 사용하여 ONTAP 또는 SVM 관리자로 실행될 것으로 예상합니다. 'fsxadmin' 사용자는 클러스터 관리자 사용자에게 제한된 교체품입니다.



'limitAggregateUsage' 매개 변수를 사용하는 경우 클러스터 관리자 권한이 필요합니다. Astra Trident와 함께 NetApp ONTAP용 Amazon FSx를 사용하는 경우 "limitAggregateUsage" 매개 변수는 "vsadmin" 및 "fsxadmin" 사용자 계정과 작동하지 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

Astra Trident는 를 사용하여 NVMe(비휘발성 메모리 익스프레스) 프로토콜을 지원합니다 `ontap-san` 다음을 포함한 드라이버:

- IPv6
- NVMe 볼륨의 스냅샷 및 클론
- NVMe 볼륨 크기 조정

- Astra Trident 외부에서 생성된 NVMe 볼륨을 가져와 Astra Trident로 라이프사이클을 관리할 수 있음
- NVMe 네이티브 다중 경로
- K8 노드의 정상 또는 비정상적으로 종료 (24.02)

Astra Trident는 다음을 지원하지 않습니다.

- NVMe에서 기본적으로 지원하는 DH-HMAC-CHAP입니다
- DM(Device Mapper) 경로 다중화
- LUKS 암호화

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, Astra Trident는 SVM에 하나 이상의 Aggregate가 할당되어 있어야 합니다.

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

Astra 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: Astra 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

Astra Trident는 SVM 범위/클러스터 범위 관리자에게 ONTAP 백엔드와 통신하기 위한 자격 증명을 요구합니다. admin 또는 vsadmin과 같이 미리 정의된 표준 역할을 사용하는 것이 좋습니다. 이를 통해 향후 Astra Trident 릴리스에서 사용할 기능 API를 노출할 수 있는 향후 ONTAP 릴리스와 향후 호환성이 보장됩니다. 사용자 지정 보안 로그인 역할은 Astra Trident와 함께 생성 및 사용할 수 있지만 권장되지 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 증명에 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 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=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
```

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...Vaaallllluuuueeee",
  "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
SanBackend	ontap-san	586b1cd5-8cf8-428d-a76c-2872713612c1



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 업데이트가 성공적이면 Astra Trident가 ONTAP 백엔드와 통신하고 향후 볼륨 작업을 처리할 수 있음을 나타냅니다.

양방향 **CHAP**를 사용하여 연결을 인증합니다

Astra Trident는 양방향 CHAP를 사용하여 iSCSI 세션을 인증할 수 있습니다. `ontap-san` 및 `ontap-san-economy` 드라이버. 이를 위해서는 `useCHAP` 백엔드 정의에서 선택할 수 있습니다. `로`로 설정한 경우 `true` Astra Trident는 SVM의 기본 이니시에이터 보안을 양방향 CHAP로 구성하고 백엔드 파일에서 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 false로 설정할 수 없습니다.

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

작동 방식

스토리지 관리자는 useCHAP를 true로 설정하여 스토리지 백엔드에서 CHAP를 구성하도록 Astra Trident에 지시합니다. 여기에는 다음이 포함됩니다.

- SVM에서 CHAP 설정:
 - SVM의 기본 이니시에이터 보안 유형이 none(기본적으로 설정) * 이고 * 기존 LUN이 볼륨에 이미 있으면 Astra Trident가 기본 보안 유형을 로 설정합니다 CHAP CHAP 이니시에이터와 타겟 사용자 이름 및 암호 구성 을 진행합니다.
 - SVM에 LUN이 포함된 경우 Astra Trident는 SVM에서 CHAP를 활성화하지 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 Astra Trident는 해당 'tridentbackend' CRD를 생성하고 CHAP 비밀과 사용자 이름을 Kubernetes 비밀로 저장합니다. 이 백엔드에서 Astra Trident에 의해 생성된 모든 PVS는 CHAP를 통해 마운트되고 연결됩니다.

자격 증명을 회전하고 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 "tridentctl"을 사용하여 백엔드를 업데이트해야 합니다. Astra Trident에서 변경 사항을 선택할 수 없으므로 CLI/ONTAP UI를 통해 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-4378-8a33-1a472ffbe5c |
online | 7 |
+-----+-----+-----+-----+
+-----+-----+
```

기존 연결은 영향을 받지 않습니다. SVM에서 Astra Trident가 자격 증명을 업데이트하면 활성 상태로 유지됩니다. 새 연결은 업데이트된 자격 증명을 사용하며 기존 연결은 계속 활성 상태로 유지됩니다. 기존 PVS를 연결 해제하고 다시 연결하면 업데이트된 자격 증명을 사용하게 됩니다.

ONTAP SAN 구성 옵션 및 예

Astra Trident 설치에서 ONTAP SAN 드라이버를 생성하고 사용하는 방법을 알아보십시오. 이 섹션에서는 백엔드 구성 예제 및 Backend를 StorageClasses에 매핑하는 방법에 대한 세부 정보를 제공합니다.

백엔드 구성 옵션

백엔드 구성 옵션은 다음 표를 참조하십시오.

매개 변수	설명	기본값
'내전'		항상 1

매개 변수	설명	기본값
'storageDriverName'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 주소입니다. FQDN(정규화된 도메인 이름)을 지정할 수 있습니다. IPv6 플래그를 사용하여 Astra Trident를 설치한 경우 IPv6 주소를 사용하도록 설정할 수 있습니다. IPv6 주소는 과 같이 대괄호로 정의해야 합니다 [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]. 원활한 MetroCluster 전환은 를 참조하십시오 MetroCluster 예 .	"10.0.0.1", "[2001:1234:ABCD::fee]"
다타LIF	프로토콜 LIF의 IP 주소입니다. * iSCSI에 대해서는 지정하지 마십시오. * Astra Trident가 사용합니다 "ONTAP 선택적 LUN 맵" 다중 경로 세션을 설정하는 데 필요한 iSCSI LIF를 검색하려면 경고 발생 시 dataLIF 명시적으로 정의됩니다. *MetroCluster의 경우 생략합니다. * 를 참조하십시오 MetroCluster 예 .	SVM에서 파생됩니다
'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]. 를 로 설정합니다 true Astra Trident에서 백엔드에 제공된 SVM에 대한 기본 인증으로 양방향 CHAP를 구성하고 사용합니다. 을 참조하십시오 "ONTAP SAN 드라이버를 사용하여 백엔드를 구성할 준비를 합니다" 를 참조하십시오.	거짓입니다
챗터시크릿	CHAP 이니시에이터 암호입니다. useCHAP=true인 경우 필수입니다	""
'라벨'	볼륨에 적용할 임의의 JSON 형식 레이블 세트입니다	""
챗터타겟이니터시크릿	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	""

매개 변수	설명	기본값
'고객증명서'	Base64 - 클라이언트 인증서의 인코딩된 값입니다. 인증서 기반 인증에 사용됩니다	""
'clientPrivateKey'입니다	Base64 - 클라이언트 개인 키의 인코딩된 값입니다. 인증서 기반 인증에 사용됩니다	""
신택인증서다	Base64 - 신뢰할 수 있는 CA 인증서의 인코딩된 값입니다. 선택 사항. 인증서 기반 인증에 사용됩니다.	""
'사용자 이름'	ONTAP 클러스터와 통신하는 데 필요한 사용자 이름입니다. 자격 증명 기반 인증에 사용됩니다.	""
"암호"	ONTAP 클러스터와 통신하는 데 필요한 암호입니다. 자격 증명 기반 인증에 사용됩니다.	""
'VM'입니다	사용할 스토리지 가상 머신입니다	SVM 'managementLIF'가 지정된 경우에 파생됩니다
'트oragePrefix'	SVM에서 새 볼륨을 프로비저닝할 때 사용되는 접두사 나중 수정할 수 없습니다. 이 매개 변수를 업데이트하려면 새 백엔드를 생성해야 합니다.	trident
제한선택사용법	사용량이 이 비율을 초과하면 프로비저닝이 실패합니다. NetApp ONTAP 백엔드에 Amazon FSx를 사용하는 경우를 지정하지 마십시오 limitAggregateUsage. 제공 fsxadmin 및 vsadmin 애그리게이트 사용을 검색하고 Astra Trident를 사용하여 제한하는 데 필요한 권한이 포함되어 있지 않습니다.	""(기본적으로 적용되지 않음)
LimitVolumeSize	요청된 볼륨 크기가 이 값보다 큰 경우 용량 할당에 실패합니다. 또한 qtree 및 LUN에 대해 관리하는 볼륨의 최대 크기도 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
'useREST'	ONTAP REST API를 사용하는 부울 매개 변수입니다. * 기술 미리 보기 * useREST 프로덕션 작업 부하가 아닌 테스트 환경에 권장되는 기술 미리 보기로 제공됩니다. 를 로 설정한 경우 true, Astra Trident는 ONTAP REST API를 사용하여 백엔드와 통신합니다. 이 기능을 사용하려면 ONTAP 9.11.1 이상이 필요합니다. 또한 사용되는 ONTAP 로그인 역할에 에 대한 액세스 권한이 있어야 합니다 ontap 응용 프로그램. 이는 사전 정의된 에 의해 충족됩니다 vsadmin 및 cluster-admin 역할. useREST 는 MetroCluster에서 지원되지 않습니다. useREST NVMe/TCP에 대해 완전한 자격을 갖추고 있음	거짓입니다

매개 변수	설명	기본값
sanType	를 사용하여 선택합니다 iscsi iSCSI 또는 의 경우 nvme NVMe/TCP의 경우	iscsi 비어 있는 경우

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"
'예약공간'	공간 예약 모드, "없음"(싌) 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. Astra Trident와 함께 QoS 정책 그룹을 사용하려면 ONTAP 9.8 이상이 필요합니다. 비공유 QoS 정책 그룹을 사용하고 정책 그룹이 각 구성요소별로 적용되도록 하는 것이 좋습니다. 공유 QoS 정책 그룹은 모든 워크로드의 총 처리량에 대해 상한을 적용합니다.	""
적응성 QosPolicy	생성된 볼륨에 할당할 적응형 QoS 정책 그룹입니다. 스토리지 풀/백엔드에서 qosPolicy 또는 adapativeQosPolicy 중 하나를 선택합니다	""
안산예비역	스냅샷용으로 예약된 볼륨의 백분율입니다	"0"인 경우 snapshotPolicy "없음"이고, 그렇지 않으면""입니다.
'plitOnClone'을 선택합니다	생성 시 상위 클론에서 클론을 분할합니다	"거짓"
암호화	새 볼륨에 NVE(NetApp Volume Encryption)를 사용하도록 설정하고 기본값은 'false'입니다. 이 옵션을 사용하려면 NVE 라이선스가 클러스터에서 활성화되어 있어야 합니다. 백엔드에서 NAE가 활성화된 경우 Astra Trident에 프로비저닝된 모든 볼륨은 NAE가 활성화됩니다. 자세한 내용은 다음을 참조하십시오. " Astra Trident가 NVE 및 NAE와 연동되는 방식 ".	"거짓"
luksEncryption	LUKS 암호화를 사용합니다. 을 참조하십시오 " LUKS(Linux Unified Key Setup) 사용 ". NVMe/TCP에 대해서는 LUKS 암호화가 지원되지 않습니다.	""
'생태성 스타일'을 참조하십시오	새로운 볼륨에 대한 보안 스타일	unix
'계층화 정책'	"없음"을 사용하는 계층화 정책	ONTAP 9.5 SVM-DR 이전 구성의 경우 "스냅샷 전용"

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'
```



'ONTAP-SAN' 드라이버를 사용하여 생성된 모든 볼륨의 경우, Astra Trident가 FlexVol에 10%의 용량을 추가하여 LUN 메타데이터를 수용합니다. LUN은 사용자가 PVC에서 요청하는 정확한 크기로 프로비저닝됩니다. Astra Trident가 FlexVol에 10%를 더합니다(ONTAP에서 사용 가능한 크기로 표시). 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.

'스냅샷 보존'을 정의하는 백엔드의 경우 Astra Trident는 다음과 같이 볼륨의 크기를 계산합니다.

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

1.1은 LUN 메타데이터를 수용하도록 FlexVol에 추가된 10%의 Astra Trident입니다. 나프산예비공간 = 5%, PVC 요청 = 5GiB의 경우 총 용적 크기는 5.79GiB이고 사용 가능한 크기는 5.5GiB입니다. 'volume show' 명령은 다음 예와 유사한 결과를 표시합니다.

Vserver	Volume	Aggregate	State	Type	Size	Available	Used%
		_pvc_89f1c156_3801_4de4_9f9d_034d54c395f4	online	RW	10GB	5.00GB	0%
		_pvc_e42ec6fe_3baa_4af6_996d_134adbbb8e6d	online	RW	5.79GB	5.50GB	0%
		_pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba	online	RW	1GB	511.8MB	0%

3 entries were displayed.

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 ONTAP에서 Astra Trident와 함께 Amazon FSx를 사용하는 경우 IP 주소 대신 LIF에 대한 DNS 이름을 지정하는 것이 좋습니다.

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>
```

ONTAP SAN 경제 예

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
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-san
managementLIF: 192.168.1.66
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
```

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 CHAP의 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.

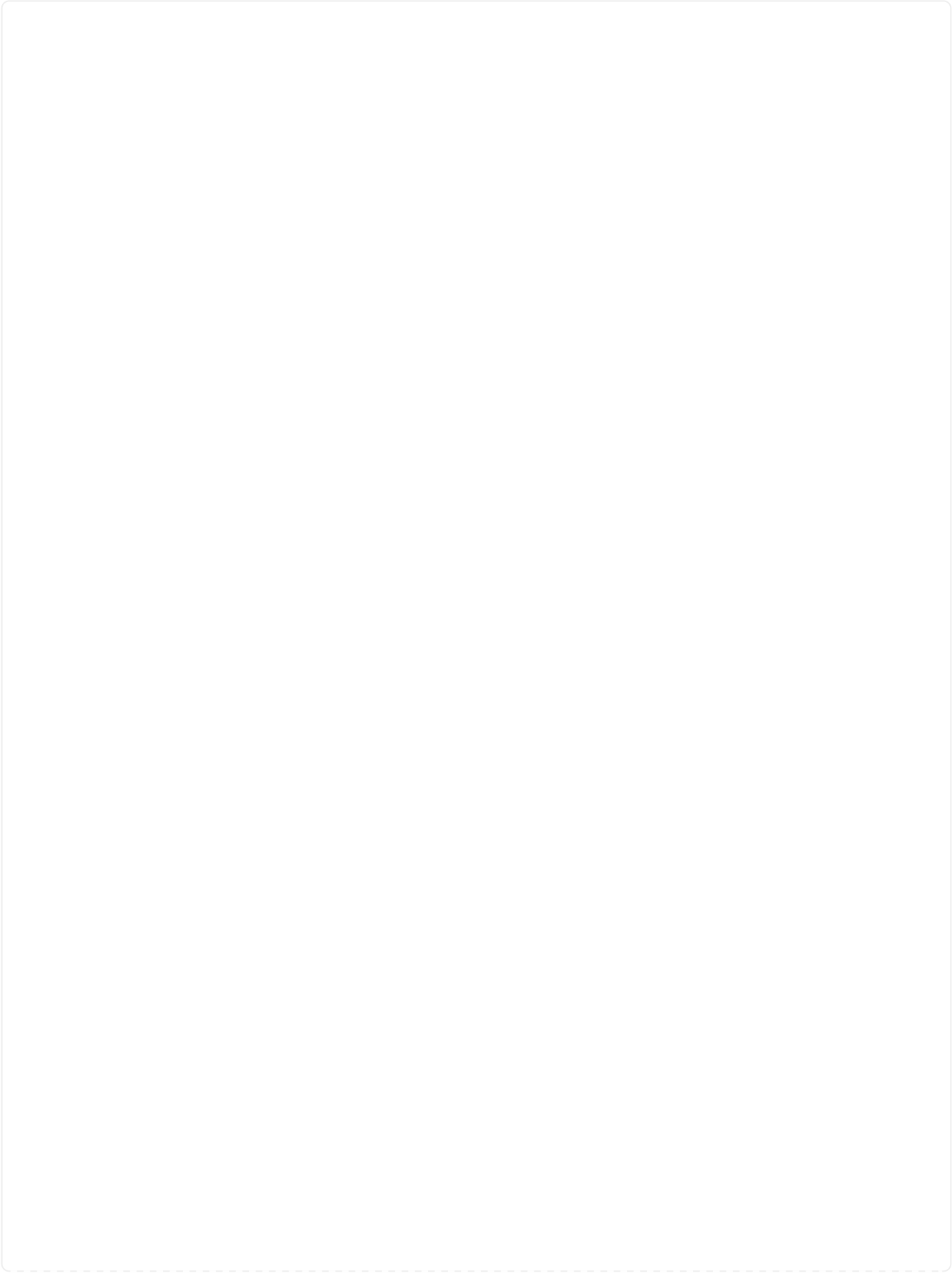
```
---
version: 1
backendName: NVMeBackend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_nvme
username: vsadmin
password: password
sanType: nvme
useREST: true
```

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.

Astra Trident가 "Comments" 필드에 프로비저닝 레이블을 설정합니다. FlexVol에 주석이 설정됩니다. Astra Trident는 프로비저닝할 때 가상 풀에 있는 모든 레이블을 스토리지 볼륨에 복사합니다. 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.



```

---
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'

```

```

---
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"

```

Astra Trident가 선택한 가상 풀을 결정하고 스토리지 요구 사항을 충족시킵니다.

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

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



보호, 복구 및 이동성을 위해 Astra Control을 사용하는 경우 를 참조하십시오 [Astra Control 드라이버 호환성](#).

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

Astra Control 드라이버 호환성

Astra Control은 로 생성한 볼륨을 위해 원활한 보호, 재해 복구, 이동성(Kubernetes 클러스터 간에 볼륨 이동)을 제공합니다 *ontap-nas*, *ontap-nas-flexgroup*, 및 *ontap-san* 드라이버. 을 참조하십시오 ["Astra Control 복제 사전 요구 사항"](#) 를 참조하십시오.



- 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

사용자 권한

Astra Trident는 일반적으로 를 사용하여 ONTAP 또는 SVM 관리자로 실행될 것으로 예상합니다 *admin* 클러스터 사용자 또는 *vsadmin* SVM 사용자 또는 동일한 역할을 가진 다른 이름의 사용자

NetApp ONTAP 구축을 위한 Amazon FSx의 경우, Astra Trident는 클러스터를 사용하여 ONTAP 또는 SVM 관리자로 실행될 것으로 예상합니다 *fsxadmin* 사용자 또는 *a vsadmin* SVM 사용자 또는 동일한 역할을 가진 다른 이름의 사용자 를 클릭합니다 *fsxadmin* 사용자는 클러스터 관리자를 제한적으로 대체합니다.



'limitAggregateUsage' 매개 변수를 사용하는 경우 클러스터 관리자 권한이 필요합니다. Astra Trident와 함께 NetApp ONTAP용 Amazon FSx를 사용하는 경우 "limitAggregateUsage" 매개 변수는 "vsadmin" 및 "fsxadmin" 사용자 계정과 작동하지 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 익스포트 정책을 이해합니다.

요구 사항

- 모든 ONTAP 백엔드의 경우, Astra Trident는 SVM에 하나 이상의 Aggregate가 할당되어 있어야 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 톨이 설치되어 있어야 합니다. 을 참조하십시오 ["여기"](#) 를 참조하십시오.
- Astra Trident는 Windows 노드에서 실행되는 Pod에만 마운트된 SMB 볼륨을 지원합니다. 을 참조하십시오 [SMB 볼륨 프로비저닝을 위한 준비](#) 를 참조하십시오.

ONTAP 백엔드를 인증합니다

Astra 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 모드를 사용하려면 Astra Trident가 ONTAP 클러스터와 통신하려면 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

Astra Trident는 SVM 범위/클러스터 범위 관리자에게 ONTAP 백엔드와 통신하기 위한 자격 증명을 요구합니다. `admin` 또는 `vsadmin`과 같이 미리 정의된 표준 역할을 사용하는 것이 좋습니다. 이를 통해 향후 Astra Trident 릴리스에서 사용할 기능 API를 노출할 수 있는 향후 ONTAP 릴리스와 향후 호환성이 보장됩니다. 사용자 지정 보안 로그인 역할은 Astra Trident와 함께 생성 및 사용할 수 있지만 권장되지 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
username: vsadmin
password: password
```

JSON을 참조하십시오

```
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password"
}
```

백엔드 정의는 자격 증명에 일반 텍스트로 저장되는 유일한 위치라는 점에 유의하십시오. 백엔드가 생성된 후 사용자 이름/암호는 Base64로 인코딩되어 Kubernetes 암호로 저장됩니다. 백엔드의 생성/업데이트는 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...Vaaalllluuuuueeee",
  "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 업데이트가 성공적이면 Astra Trident가 ONTAP 백엔드와 통신하고 향후 볼륨 작업을 처리할 수 있음을 나타냅니다.

NFS 익스포트 정책을 관리합니다

Astra Trident는 NFS 익스포트 정책을 사용하여 프로비저닝하는 볼륨에 대한 액세스를 제어합니다.

Astra Trident는 익스포트 정책을 사용할 때 다음 두 가지 옵션을 제공합니다.

- Astra Trident는 익스포트 정책 자체를 동적으로 관리할 수 있습니다. 이 운영 모드에서 스토리지 관리자는 허용할 수 있는 IP 주소를 나타내는 CIDR 블록 목록을 지정합니다. Astra Trident는 이러한 범위에 속하는 노드 IP를 익스포트 정책에 자동으로 추가합니다. 또는 CIDR을 지정하지 않으면 노드에서 발견된 글로벌 범위의 유니캐스트 IP가 내보내기 정책에 추가됩니다.

- 스토리지 관리자는 익스포트 정책을 생성하고 규칙을 수동으로 추가할 수 있습니다. Astra Trident는 구성에 다른 익스포트 정책 이름을 지정하지 않는 한 기본 익스포트 정책을 사용합니다.

익스포트 정책을 동적으로 관리

Astra Trident를 사용하면 ONTAP 백엔드의 익스포트 정책을 동적으로 관리할 수 있습니다. 따라서 스토리지 관리자는 명시적 규칙을 수동으로 정의하는 대신 작업자 노드 IP에 허용되는 주소 공간을 지정할 수 있습니다. 익스포트 정책 관리를 크게 간소화하므로, 익스포트 정책을 수정하면 더 이상 스토리지 클러스터에 대한 수동 작업이 필요하지 않습니다. 또한 스토리지 클러스터에 대한 액세스를 지정된 범위의 IP가 있는 작업자 노드에만 제한함으로써 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 프론트엔드 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
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 익스포트 정책)을 허용하는 익스포트 규칙과 함께 이전에 생성된 익스포트 정책이 있는지 확인해야 합니다. Astra Trident에 사용할 SVM을 지정하려면 항상 NetApp 권장 모범 사례를 따르십시오.

다음은 위의 예를 사용하여 이 기능이 작동하는 방식에 대한 설명입니다.

- 자동내보내기정책은 참으로 설정된다. 이는 Astra Trident가 'vm1' SVM에 대한 수출 정책을 만들고 'autoExportCIDR' 주소 블록을 사용하여 규칙 추가 및 삭제를 처리한다는 것을 의미합니다. 예를 들어 UUID 403b5326-8482-40db-96d0-d83fb3f4daec 및 "true"로 설정된 autoExportPolicy가 있는 백엔드는 SVM에 trident-403b5326-8482-40db-96d0-d83fb3f4daec라는 이름의 익스포트 정책을 생성합니다.
- autoExportCIDR에는 주소 블록 목록이 포함되어 있습니다. 이 필드는 선택 사항이며 기본적으로 ["0.0.0.0/0", ":/0"]입니다. 정의되지 않은 경우 Astra Trident는 작업자 노드에 있는 모든 전역 범위의 유니캐스트 주소를 추가합니다.

이 예에서는 192.168.0.0/24 주소 공간을 제공한다. 이 주소 범위에 속하는 Kubernetes 노드 IP가 Astra Trident가 생성하는 익스포트 정책에 추가됨을 나타냅니다. Astra Trident가 실행 중인 노드를 등록하면 해당 노드의 IP 주소를 조회하여 autoExportCIDR에서 제공하는 주소 블록과 대조합니다. IP를 필터링한 후 Astra Trident는 검색된

클라이언트 IP에 대한 익스포트 정책 규칙을 생성하며, 식별하는 각 노드에 대해 하나의 규칙을 사용합니다.

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.

Astra Trident가 백엔드를 생성하거나 업데이트한 후 'tridentctl' 또는 해당 'tridentbackend' CRD:

```
./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>
      fileType: ext4
```

노드가 Kubernetes 클러스터에 추가되고 Astra Trident 컨트롤러에 등록되면 기존 백엔드의 내보내기 정책이 업데이트됩니다(백엔드의 "autoExportCIDR"에 지정된 주소 범위에 속하는 경우).

노드가 제거되면 Astra Trident는 온라인 상태인 모든 백엔드를 검사하여 노드에 대한 액세스 규칙을 제거합니다. Astra Trident는 관리되는 백엔드의 내보내기 정책에서 이 노드 IP를 제거하여 불량 마운트를 방지합니다. 단, 클러스터의 새 노드에서 이 IP를 다시 사용하지 않는 한 마찬가지입니다.

기존 백엔드의 경우 백엔드를 로 업데이트합니다 `tridentctl update backend` Astra Trident가 익스포트 정책을 자동으로 관리하는지 확인합니다. 이렇게 하면 백엔드의 UUID를 기준으로 이름이 지정된 새 익스포트 정책이 생성되고 백엔드에 있는 볼륨은 다시 마운트될 때 새로 생성된 익스포트 정책을 사용합니다.



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

라이브 노드의 IP 주소가 업데이트되면 노드에서 Astra Trident POD를 다시 시작해야 합니다. 그런 다음 Astra Trident가 이 IP 변경 사항을 반영하도록 관리하는 백엔드에 대한 익스포트 정책을 업데이트합니다.

SMB 볼륨 프로비저닝을 위한 준비

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



SVM에서 NFS 및 SMB/CIFS 프로토콜을 모두 구성하여 을 생성해야 합니다 `ontap-nas-economy` ONTAP 사내를 위한 SMB 볼륨 이 두 프로토콜 중 하나를 구성하지 않으면 SMB 볼륨 생성에 실패합니다.

시작하기 전에

SMB 볼륨을 프로비저닝하려면 먼저 다음 항목이 있어야 합니다.

- Linux 컨트롤러 노드 및 Windows Server 2019를 실행하는 Windows 작업자 노드가 있는 Kubernetes 클러스터 Astra Trident는 Windows 노드에서 실행되는 Pod에만 마운트된 SMB 볼륨을 지원합니다.
- Active Directory 자격 증명에 포함된 Astra Trident 암호가 하나 이상 있어야 합니다. 암호를 생성합니다 `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 SMB 공유를 생성하거나 Astra Trident에서 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	다음 중 하나를 지정할 수 있습니다. Microsoft 관리 콘솔 또는 ONTAP CLI를 사용하여 생성된 SMB 공유의 이름, Astra Trident가 SMB 공유를 생성할 수 있도록 하는 이름 또는 볼륨에 대한 공통 공유 액세스를 방지하기 위해 매개 변수를 비워 둘 수 있습니다. 이 매개 변수는 사내 ONTAP의 경우 선택 사항입니다. 이 매개 변수는 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. *	""

ONTAP NAS 구성 옵션 및 예

Astra Trident 설치에 ONTAP NAS 드라이버를 생성하고 사용하는 방법을 알아보십시오. 이 섹션에서는 백엔드 구성 예제 및 Backend를 StorageClasses에 매핑하는 방법에 대한 세부 정보를 제공합니다.

백엔드 구성 옵션

백엔드 구성 옵션은 다음 표를 참조하십시오.

매개 변수	설명	기본값
'내전'		항상 1
'storageDriverName'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

매개 변수	설명	기본값
마나멘타LIF	<p>클러스터 또는 SVM 관리 LIF의 IP 주소입니다</p> <p>FQDN(정규화된 도메인 이름)을 지정할 수 있습니다.</p> <p>IPv6 플래그를 사용하여 Astra Trident를 설치한 경우 IPv6 주소를 사용하도록 설정할 수 있습니다. IPv6 주소는 과 같이 대괄호로 정의해야 합니다 [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].</p> <p>원활한 MetroCluster 전환은 를 참조하십시오 MetroCluster 예.</p>	"10.0.0.1", "[2001:1234:ABCD::fee]"
다타LIF	<p>프로토콜 LIF의 IP 주소입니다.</p> <p>지정할 것을 권장합니다 dataLIF. 제공되지 않는 경우 Astra Trident는 SVM에서 데이터 LIF를 가져옵니다. NFS 마운트 작업에 사용할 FQDN(정규화된 도메인 이름)을 지정하면 여러 데이터 LIF에서 로드 밸런싱을 위해 라운드 로빈 DNS를 생성할 수 있습니다.</p> <p>초기 설정 후에 변경할 수 있습니다. 을 참조하십시오 .</p> <p>IPv6 플래그를 사용하여 Astra Trident를 설치한 경우 IPv6 주소를 사용하도록 설정할 수 있습니다. IPv6 주소는 과 같이 대괄호로 정의해야 합니다 [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555].</p> <p>*MetroCluster의 경우 생략합니다. * 를 참조하십시오 MetroCluster 예.</p>	지정되지 않은 경우 SVM에서 지정 주소 또는 파생(권장하지 않음)
'VM'입니다	<p>사용할 스토리지 가상 머신입니다</p> <p>*MetroCluster의 경우 생략합니다. * 를 참조하십시오 MetroCluster 예.</p>	SVM 'managementLIF'가 지정된 경우에 파생됩니다
자동 내보내기 정책	<p>자동 익스포트 정책 생성 및 업데이트 [Boolean] 활성화 를 사용합니다 autoExportPolicy 및 autoExportCIDRs 옵션, Astra Trident는 익스포트 정책을 자동으로 관리할 수 있습니다.</p>	거짓
자동 내보내기	<p>언제 기준으로 Kubernetes 노드 IP를 필터링하는 CIDR 목록입니다 autoExportPolicy 가 활성화됩니다.</p> <p>를 사용합니다 autoExportPolicy 및 autoExportCIDRs 옵션, Astra Trident는 익스포트 정책을 자동으로 관리할 수 있습니다.</p>	["0.0.0.0/0",":/0"]
'라벨'	<p>볼륨에 적용할 임의의 JSON 형식 레이블 세트입니다</p>	""
'고객증명서'	<p>Base64 - 클라이언트 인증서의 인코딩된 값입니다. 인증서 기반 인증에 사용됩니다</p>	""

매개 변수	설명	기본값
'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	
"암호"	클러스터/SVM에 연결하는 암호 자격 증명 기반 인증에 사용됩니다	
'storagePrefix'	SVM에서 새 볼륨을 프로비저닝할 때 사용되는 접두사 설정한 후에는 업데이트할 수 없습니다	"트리덴트"
제한선택사용법	사용량이 이 비율을 초과하면 프로비저닝이 실패합니다. ONTAP * 용 아마존 FSx에는 * 가 적용되지 않습니다	""(기본적으로 적용되지 않음)
LimitVolumeSize	요청된 볼륨 크기가 이 값보다 큰 경우 용량 할당에 실패합니다. 또한 qtree 및 LUN에 대해 관리하는 볼륨의 최대 크기 및 을 제한합니다 qtreesPerFlexvol 옵션을 사용하면 FlexVol당 최대 qtree 수를 사용자 지정할 수 있습니다.	""(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} 사용하지 마십시오 debugTraceFlags 문제 해결 및 자세한 로그 덤프가 필요한 경우를 제외하고	null입니다
nasType	NFS 또는 SMB 볼륨 생성을 구성합니다. 옵션은 입니다 nfs, smb 또는 null입니다. Null로 설정하면 기본적으로 NFS 볼륨이 설정됩니다.	nfs
nfsMountOptions를 선택합니다	심표로 구분된 NFS 마운트 옵션 목록입니다. Kubernetes 영구 볼륨의 마운트 옵션은 일반적으로 스토리지 클래스에서 지정되지만, 스토리지 클래스에 마운트 옵션을 지정하지 않으면 Astra Trident가 스토리지 백엔드의 구성 파일에 지정된 마운트 옵션을 사용하여 로 돌아갑니다. 스토리지 클래스 또는 구성 파일에 마운트 옵션이 지정되지 않은 경우 Astra Trident는 연결된 영구 볼륨에 마운트 옵션을 설정하지 않습니다.	""
"케트리스퍼플렉스볼륨"	FlexVol당 최대 qtree, 범위 [50, 300]에 있어야 함	"200"

매개 변수	설명	기본값
smbShare	<p>다음 중 하나를 지정할 수 있습니다. Microsoft 관리 콘솔 또는 ONTAP CLI를 사용하여 생성된 SMB 공유의 이름, Astra Trident가 SMB 공유를 생성할 수 있도록 하는 이름 또는 볼륨에 대한 공통 공유 액세스를 방지하기 위해 매개 변수를 비워 둘 수 있습니다.</p> <p>이 매개 변수는 사내 ONTAP의 경우 선택 사항입니다.</p> <p>이 매개 변수는 ONTAP 백엔드에 대한 아마존 FSx에 필요하며 비워둘 수 없습니다.</p>	smb-share
'useREST'	<p>ONTAP REST API를 사용하는 부울 매개 변수입니다. * 기술 미리 보기 *</p> <p>useREST 프로덕션 작업 부하가 아닌 테스트 환경에 권장되는 기술 미리 보기로 제공됩니다. 를 로 설정한 경우 true, Astra Trident는 ONTAP REST API를 사용하여 백엔드와 통신합니다. 이 기능을 사용하려면 ONTAP 9.11.1 이상이 필요합니다. 또한 사용되는 ONTAP 로그인 역할에 에 대한 액세스 권한이 있어야 합니다 ontap 응용 프로그램. 이는 사전 정의된 에 의해 충족됩니다 vsadmin 및 cluster-admin 역할.</p> <p>useREST 는 MetroCluster에서 지원되지 않습니다.</p>	거짓

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"
'예비공간'	공간 예약 모드, "없음"(썸) 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	""
적응성 QosPolicy	생성된 볼륨에 할당할 적응형 QoS 정책 그룹입니다. 스토리지 풀/백엔드에서 qosPolicy 또는 adapativeQosPolicy 중 하나를 선택합니다. ONTAP에서 지원되지 않음 - NAS - 이코노미	""
안산예비역	스냅샷용으로 예약된 볼륨의 백분율입니다	"0"인 경우 snapshotPolicy "없음"이고, 그렇지 않으면""입니다.
'plitOnClone'을 선택합니다	생성 시 상위 클론에서 클론을 분할합니다	"거짓"

매개 변수	설명	기본값
암호화	새 볼륨에 NVE(NetApp Volume Encryption)를 사용하도록 설정하고 기본값은 'false'입니다. 이 옵션을 사용하려면 NVE 라이선스가 클러스터에서 활성화되어 있어야 합니다. 백엔드에서 NAE가 활성화된 경우 Astra Trident에 프로비저닝된 모든 볼륨은 NAE가 활성화됩니다. 자세한 내용은 다음을 참조하십시오. " Astra Trident가 NVE 및 NAE와 연동되는 방식 ".	"거짓"
'계층화 정책'	"없음"을 사용하는 계층화 정책	ONTAP 9.5 SVM-DR 이전 구성의 경우 "스냅샷 전용"
유니크권한	모드를 선택합니다	NFS 볼륨의 경우 "777", SMB 볼륨의 경우 비어 있음(해당 없음)
나프산디렉토리	에 액세스를 제어합니다 .snapshot 디렉토리	"거짓"
엑포트정책	사용할 엑포트 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.



Astra Trident와 함께 QoS 정책 그룹을 사용하려면 ONTAP 9.8 이상이 필요합니다. 비공유 QoS 정책 그룹을 사용하고 정책 그룹이 각 구성요소별로 적용되었는지 확인하는 것이 좋습니다. 공유 QoS 정책 그룹은 모든 워크로드의 총 처리량에 대해 상한을 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'

```

ONTAP-NAS와 ONTAP-NAS-Flexgroups의 경우, Astra Trident는 이제 새로운 계산을 통해 스냅샷 예비 공간 비율 및 PVC로 FlexVol의 크기를 올바르게 조정했습니다. 사용자가 PVC를 요청하면 Astra Trident는 새 계산을 사용하여 더 많은 공간을 가진 원본 FlexVol을 생성합니다. 이 계산을 통해 사용자는 PVC에서 요청한 쓰기 가능 공간을 확보할 수 있으며 요청된 공간보다 적은 공간을 확보할 수 있습니다. v21.07 이전에는 사용자가 스냅샷 보존 공간을 50%로 하여 PVC(예: 5GiB)를 요청할 때 쓰기 가능한 공간은 2.5GiB에 불과합니다. 이는 사용자가 요청한 전체 볼륨과 스냅샷 보존 비율이 다르기 때문입니다. Trident 21.07을 사용하면 쓰기 가능한 공간이 요청되고 Astra Trident는 '스냅샷 보존' 번호를 전체 볼륨의 백분율로 정의합니다. 이는 ONTAP-NAS-이코노미(ONTAP-NAS-이코노미)에는 적용되지 않습니다. 이 작동 방식을 보려면 다음 예를 참조하십시오.

계산은 다음과 같습니다.

```

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

```

snapshotReserve = 50%, PVC request = 5GiB의 경우, 총 볼륨 크기는 $2/5 = 10\text{GiB}$ 이고 사용 가능한 크기는 5GiB입니다. 이는 사용자가 PVC 요청에서 요청한 것입니다. 'volume show' 명령은 다음 예와 유사한 결과를 표시합니다.

Vserver	Volume	Aggregate	State	Type	Size	Available	Used%
		_pvc_89f1c156_3801_4de4_9f9d_034d54c395f4	online	RW	10GB	5.00GB	0%
		_pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba	online	RW	1GB	511.8MB	0%

2 entries were displayed.

이전 설치에서 기존 백엔드는 Astra Trident를 업그레이드할 때 위에서 설명한 대로 볼륨을 프로비저닝합니다. 업그레이드하기 전에 생성한 볼륨의 경우 변경 사항을 관찰하기 위해 볼륨의 크기를 조정해야 합니다. 예를 들어, '스냅샷 보존 = 50'의 2GiB PVC는 쓰기 가능한 공간 1GiB를 제공하는 볼륨을 낳았습니다. 예를 들어, 볼륨을 3GiB로 조정하면 애플리케이션에 6GiB 볼륨의 쓰기 가능 공간이 3GiB로 표시됩니다.

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 예 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 익스포트 정책의 예

이 예에서는 Astra Trident가 동적 익스포트 정책을 사용하여 익스포트 정책을 자동으로 생성하고 관리하도록 지시하는 방법을 보여 줍니다. 이 기능은 예 대해서도 동일하게 작동합니다 `ontap-nas-economy` 및 `ontap-nas-flexgroup` 드라이버.

```
---
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_
```

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.

Astra Trident가 "Comments" 필드에 프로비저닝 레이블을 설정합니다. 설명은 FlexVol for에서 설정됩니다 ontap-nas 또는 FlexGroup for ontap-nas-flexgroup. Astra Trident는 프로비저닝할 때 가상 풀에 있는 모든 레이블을 스토리지 볼륨에 복사합니다. 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.

```

---
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: mysqlldb
    cost: '25'
    zone: us_east_1d
    defaults:
      spaceReserve: volume
      encryption: 'false'
      unixPermissions: '0775'
```

```

---
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'
```

```

---
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"

```

Astra Trident가 선택한 가상 풀을 결정하고 스토리지 요구 사항을 충족시킵니다.

업데이트 dataLIF 초기 구성 후

다음 명령을 실행하여 초기 구성 후에 데이터 LIF를 변경할 수 있으며, 업데이트된 데이터 LIF가 포함된 새 백엔드 JSON 파일을 제공할 수 있습니다.

```

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

```



PVC가 하나 이상의 포드에 연결된 경우 해당 포드를 모두 내린 다음 다시 불러와서 새 데이터 LIF가 적용되도록 해야 합니다.

NetApp ONTAP용 Amazon FSx

NetApp ONTAP용 Amazon FSx와 함께 Astra Trident를 사용하십시오

"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.

개요

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. NetApp ONTAP용 Amazon FSx를 사용하면 클라우드에서 Data ONTAP가 관리형 파일 시스템으로 제공됩니다. 새로운 파일 시스템 유형을 * NetApp ONTAP * 라고 합니다.

NetApp ONTAP용 Amazon FSx와 Astra Trident를 사용하면 Amazon EKS(Elastic Kubernetes Service)에서 실행되는 Kubernetes 클러스터가 ONTAP에서 지원하는 블록 및 파일 영구 볼륨을 프로비저닝할 수 있습니다.

고려 사항

- SMB 볼륨:
 - SMB 볼륨은 를 사용하여 지원됩니다 `ontap-nas` 드라이버만 해당.
 - Astra Trident EKS 애드온에서는 SMB 볼륨이 지원되지 않습니다.
 - Astra Trident는 Windows 노드에서 실행되는 Pod에만 마운트된 SMB 볼륨을 지원합니다.
- Astra Trident 24.02 이전에는 자동 백업이 활성화된 Amazon FSx 파일 시스템에서 생성된 볼륨을 Trident에서 삭제할 수 없었습니다. Astra Trident 24.02 이상에서 이 문제를 방지하려면 를 지정합니다 `fsxFilesystemID`` 있습니다 ``apiRegion`` 있습니다 ``apiKey`` 있습니다 ``secretKey`` AWS FSx for ONTAP의 백엔드 구성 파일에 있습니다.



Astra Trident에 IAM 역할을 지정하는 경우 를 지정하는 것을 생략할 수 있습니다 `apiRegion`, `apiKey`, 및 `secretKey` 필드를 Astra Trident로 명시적으로 전환합니다. 자세한 내용은 을 참조하십시오 ["ONTAP 구성 옵션 및 예제용 FSX"](#).

FSx for ONTAP 드라이버 세부 정보

다음 드라이버를 사용하여 Astra Trident를 NetApp ONTAP용 Amazon FSx와 통합할 수 있습니다.

- 'ONTAP-SAN': 프로비저닝되는 각 PV는 NetApp ONTAP 볼륨용 자체 Amazon FSx 내에 있는 LUN입니다.
- 'ONTAP-SAN-이코노미': 프로비저닝되는 각 PV는 NetApp ONTAP 볼륨에 대해 Amazon FSx당 구성 가능한 LUN 수를 가진 LUN입니다.

- ONTAP-NAS: 프로비저닝되는 각 PV는 NetApp ONTAP 볼륨용 전체 Amazon FSx입니다.
- '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"](#).

인증

Astra Trident는 두 가지 인증 모드를 제공합니다.

- 인증서 기반: Astra Trident는 SVM에 설치된 인증서를 사용하여 FSx 파일 시스템의 SVM과 통신합니다.
- 자격 증명 기반: 파일 시스템에 대해 'fsxadmin' 사용자를 사용하거나 SVM용으로 구성된 'vsadmin' 사용자를 사용할 수 있습니다.



Astra Trident는 한 대로 실행될 것으로 예상합니다 vsadmin SVM 사용자 또는 동일한 역할을 가진 다른 이름의 사용자 NetApp ONTAP용 Amazon FSx에는 가 있습니다 fsxadmin ONTAP의 제한된 교체인 사용자입니다 admin 클러스터 사용자. 을 사용하는 것이 좋습니다 vsadmin Astra Trident와 함께.

자격 증명 기반 방법과 인증서 기반 방법 간에 이동하도록 백엔드를 업데이트할 수 있습니다. 그러나 * 자격 증명 및 인증서 * 를 제공하려고 하면 백엔드 생성이 실패합니다. 다른 인증 방법으로 전환하려면 백엔드 구성에서 기존 방법을 제거해야 합니다.

인증 활성화에 대한 자세한 내용은 드라이버 유형에 대한 인증을 참조하십시오.

- ["ONTAP NAS 인증"](#)
- ["ONTAP SAN 인증"](#)

EKS용 클라우드 ID

클라우드 ID를 사용하면 Kubernetes Pod에서 명시적 AWS 자격 증명을 제공하지 않고 AWS IAM 역할로 인증하여 AWS 리소스에 액세스할 수 있습니다.

AWS에서 클라우드 ID를 활용하려면 다음이 필요합니다.

- EKS를 사용하여 구축된 Kubernetes 클러스터
- 가 포함된 Astra Trident가 설치되었습니다 cloudProvider 지정 "AWS" 및 cloudIdentity AWS IAM 역할 지정

Trident 운영자

Trident 연산자를 사용하여 Astra Trident를 설치하려면 편집을 진행합니다

tridentorchestrator_cr.yaml 를 눌러 설정합니다 cloudProvider 를 선택합니다 "AWS" 그리고 설정합니다 cloudIdentity AWS IAM 역할로 이동합니다.

예를 들면 다음과 같습니다.

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "AWS"
  cloudIdentity: "'eks.amazonaws.com/role-arn:
arn:aws:iam::123456:role/astratrident-role'"
```

헬름

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

```
export CP="AWS"
export CI="'eks.amazonaws.com/role-arn:
arn:aws:iam::123456:role/astratrident-role'"
```

다음 예에서는 Astra Trident 및 Sets를 설치합니다 cloudProvider 를 선택합니다 AWS 환경 변수를 사용합니다 \$CP 및 환경 변수를 사용하여 'cloudIdentity'를 설정합니다 \$CI:

```
helm install trident trident-operator-100.2402.0.tgz --set
cloudProvider=$CP --set cloudIdentity=$CI
```

<code>tridentctl</code>

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

```
export CP="AWS"
export CI="'eks.amazonaws.com/role-arn:
arn:aws:iam::123456:role/astratrident-role'"
```

다음 예에서는 Astra Trident를 설치하고 를 설정합니다 cloud-provider 에 플래그 지정 \$CP, 및 cloud-identity 를 선택합니다 \$CI:

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

자세한 내용을 확인하십시오

- ["NetApp ONTAP용 Amazon FSx 문서"](#)
- ["NetApp ONTAP용 Amazon FSx 블로그 게시물"](#)

NetApp ONTAP용 Amazon FSx를 통합합니다

Amazon Elastic Kubernetes Service(EKS)에서 실행되는 Kubernetes 클러스터가 ONTAP에서 지원하는 블록 및 파일 영구 볼륨을 프로비저닝할 수 있도록 NetApp ONTAP 파일 시스템용 Amazon FSx를 Astra Trident와 통합할 수 있습니다.

요구 사항

또한 ["Astra Trident 요구사항"](#), ONTAP용 FSx와 Astra Trident를 통합하려면 다음이 필요합니다.

- kubectl이 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.

- Astra Trident는 Windows 노드에서 실행되는 Pod에만 마운트된 SMB 볼륨을 지원합니다. 을 참조하십시오 [SMB 볼륨 프로비저닝을 위한 준비](#) 를 참조하십시오.

ONTAP SAN 및 NAS 드라이버 통합



SMB 볼륨에 대해 를 구성하는 경우 을 읽어야 합니다 [SMB 볼륨 프로비저닝을 위한 준비](#) 백엔드를 생성하기 전에

단계

1. 중 하나를 사용하여 Astra Trident를 배포합니다 ["배포 방법"](#).
2. SVM 관리 LIF DNS 이름을 수집합니다. 예를 들어, AWS CLI를 사용하여 를 찾습니다 `DNSName` 에 입력 `Endpoints → Management` 다음 명령을 실행한 후:

```
aws fsx describe-storage-virtual-machines --region <file system region>
```

3. 에 대한 인증서를 만들고 설치합니다 ["NAS 백엔드 인증"](#) 또는 ["SAN 백엔드 인증"](#).



파일 시스템에 연결할 수 있는 모든 위치에서 SSH를 사용하여 파일 시스템(예: 인증서 설치)에 로그인할 수 있습니다. 파일 시스템 생성 시 구성한 fsxadmin 사용자, AWS FSX 기술파일 시스템 "의 관리 DNS 이름을 사용합니다.

4. 아래 예에 표시된 대로 인증서와 관리 LIF의 DNS 이름을 사용하여 백엔드 파일을 생성합니다.

YAML

```
version: 1
storageDriverName: ontap-san
backendName: customBackendName
managementLIF: svm-XXXXXXXXXXXXXXXXXXXX.fs-XXXXXXXXXXXXXXXXXXXX.fsx.us-
east-2.aws.internal
svm: svm01
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
```

JSON을 참조하십시오

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "customBackendName",
  "managementLIF": "svm-XXXXXXXXXXXXXXXXXXXX.fs-
XXXXXXXXXXXXXXXXXXXX.fsx.us-east-2.aws.internal",
  "svm": "svm01",
  "clientCertificate": "ZXR0ZXJwYXB...ICMgJ3BhcGVyc2",
  "clientPrivateKey": "vciwKIyAgZG...0cnksIGRlc2NyaX",
  "trustedCACertificate": "zcyBbaG...b3Igb3duIGNsYXNz"
}
```

또는 다음 예와 같이 AWS Secret Manager에 저장된 SVM 자격 증명(사용자 이름 및 암호)을 사용하여 백엔드 파일을 생성할 수 있습니다.

YAML

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

JSON을 참조하십시오

```
{
  "apiVersion": "trident.netapp.io/v1",
  "kind": "TridentBackendConfig",
  "metadata": {
    "name": "backend-tbc-ontap-nas"
  },
  "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:xxxxxxx:secret:secret-name",
      "type": "awsarn"
    }
  }
}
```


백엔드 만들기에 대한 자세한 내용은 다음 링크를 참조하십시오.

- ["ONTAP NAS 드라이버를 사용하여 백엔드를 구성합니다"](#)
- ["ONTAP SAN 드라이버를 사용하여 백엔드를 구성합니다"](#)

SMB 볼륨 프로비저닝을 위한 준비

를 사용하여 SMB 볼륨을 프로비저닝할 수 있습니다 `ontap-nas` 드라이버. 를 누릅니다 [ONTAP SAN 및 NAS 드라이버 통합](#) 다음 단계를 완료합니다.

시작하기 전에

를 사용하여 SMB 볼륨을 프로비저닝하기 전에 `ontap-nas` 드라이버, 다음이 있어야 합니다.

- Linux 컨트롤러 노드 및 Windows Server 2019를 실행하는 Windows 작업자 노드가 있는 Kubernetes 클러스터 Astra Trident는 Windows 노드에서 실행되는 Pod에만 마운트된 SMB 볼륨을 지원합니다.
- Active Directory 자격 증명이 포함된 Astra Trident 암호가 하나 이상 있어야 합니다. 암호를 생성합니다 `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. SMB 공유를 생성합니다. 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 또는 Astra Trident가 SMB 공유를 생성할 수 있도록 이름을 지정할 수 있습니다. 이 매개변수는 ONTAP 백엔드에 대한 Amazon FSx에 필요합니다.	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. *	""

ONTAP 구성 옵션 및 예제용 FSX

Amazon FSx for ONTAP의 백엔드 구성 옵션에 대해 알아보십시오. 이 섹션에서는 백엔드 구성 예를 제공합니다.

백엔드 구성 옵션

백엔드 구성 옵션은 다음 표를 참조하십시오.

매개 변수	설명	예
'내전'		항상 1
'storageDriverName'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

매개 변수	설명	예
마나멘타LIF	<p>클러스터 또는 SVM 관리 LIF의 IP 주소입니다</p> <p>FQDN(정규화된 도메인 이름)을 지정할 수 있습니다.</p> <p>IPv6 플래그를 사용하여 Astra Trident를 설치한 경우 IPv6 주소를 사용하도록 설정할 수 있습니다. IPv6 주소는 대괄호로 묶어야 합니다(예: [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]).</p>	"10.0.0.1", "[2001:1234:ABCD::fee]"
다타LIF	<p>프로토콜 LIF의 IP 주소입니다.</p> <p>* ONTAP NAS 드라이버 *: 데이터 LIF를 지정하는 것이 좋습니다. 제공되지 않는 경우 Astra Trident는 SVM에서 데이터 LIF를 가져옵니다. NFS 마운트 작업에 사용할 FQDN(정규화된 도메인 이름)을 지정하면 여러 데이터 LIF에서 로드 밸런싱을 위해 라운드 로빈 DNS를 생성할 수 있습니다. 초기 설정 후에 변경할 수 있습니다. 을 참조하십시오 .</p> <p>* ONTAP SAN 드라이버 *: iSCSI에 대해 지정하지 마십시오. Astra Trident는 ONTAP 선택적 LUN 맵을 사용하여 다중 경로 세션을 설정하는데 필요한 iSCSI LIF를 검색합니다. 데이터 LIF가 명시적으로 정의되어 있으면 경고가 생성됩니다.</p> <p>IPv6 플래그를 사용하여 Astra Trident를 설치한 경우 IPv6 주소를 사용하도록 설정할 수 있습니다. IPv6 주소는 대괄호로 묶어야 합니다(예: [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]).</p>	
자동 내보내기 정책	<p>자동 익스포트 정책 생성 및 업데이트 [Boolean] 활성화 를 사용합니다 autoExportPolicy 및 autoExportCIDRs 옵션, Astra Trident는 익스포트 정책을 자동으로 관리할 수 있습니다.</p>	거짓입니다

매개 변수	설명	예
자동 내보내기	언제 기준으로 Kubernetes 노드 IP를 필터링하는 CIDR 목록입니다 autoExportPolicy 가 활성화됩니다. 를 사용합니다 autoExportPolicy 및 autoExportCIDRs 옵션, Astra Trident는 익스포트 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	""
신택인증서다	Base64 - 신뢰할 수 있는 CA 인증서의 인코딩된 값입니다. 선택 사항. 인증서 기반 인증에 사용됩니다.	""
'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.
'트러agePrefix'	SVM에서 새 볼륨을 프로비저닝할 때 사용되는 접두사 생성 후에는 수정할 수 없습니다. 이 매개 변수를 업데이트하려면 새 백엔드를 생성해야 합니다.	trident
제한선택사용법	* NetApp ONTAP * 용 Amazon FSx는 지정하지 마십시오 fsxadmin 및 vsadmin 애그리게이트 사용을 검색하고 Astra 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 LUN에 대해 관리하는 볼륨의 최대 크기 및 을 제한합니다 qtreesPerFlexvol 옵션을 사용하면 FlexVol당 최대 qtree 수를 사용자 지정할 수 있습니다.	""(기본적으로 적용되지 않음)

매개 변수	설명	예
'오만유연한'	FlexVol당 최대 LUN 수는 [50, 200] 범위 내에 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
nfsMountOptions를 선택합니다	심표로 구분된 NFS 마운트 옵션 목록입니다. Kubernetes 영구 볼륨의 마운트 옵션은 일반적으로 스토리지 클래스에서 지정되지만, 스토리지 클래스에 마운트 옵션을 지정하지 않으면 Astra Trident가 스토리지 백엔드의 구성 파일에 지정된 마운트 옵션을 사용하여 로 돌아갑니다. 스토리지 클래스 또는 구성 파일에 마운트 옵션이 지정되지 않은 경우 Astra Trident는 연결된 영구 볼륨에 마운트 옵션을 설정하지 않습니다.	""
nasType	NFS 또는 SMB 볼륨 생성을 구성합니다. 옵션은 입니다 nfs, smb 또는 null입니다. * 를 로 설정해야 합니다 `smb` SMB 볼륨의 경우. * null로 설정하면 기본적으로 NFS 볼륨이 설정됩니다.	nfs
"케트리스퍼플렉스볼륨"	FlexVol당 최대 qtree, 범위 [50, 300]에 있어야 함	200
smbShare	Microsoft 관리 콘솔 또는 ONTAP CLI를 사용하여 생성된 SMB 공유의 이름 또는 Astra Trident가 SMB 공유를 생성할 수 있도록 이름을 지정할 수 있습니다. 이 매개변수는 ONTAP 백엔드에 대한 Amazon FSx에 필요합니다.	smb-share

매개 변수	설명	예
'useREST'	ONTAP REST API를 사용하는 부울 매개 변수입니다. * 기술 미리 보기 * useREST 프로덕션 작업 부하가 아닌 테스트 환경에 권장되는 기술 미리 보기로 제공됩니다. 를 로 설정한 경우 true, Astra Trident는 ONTAP REST API를 사용하여 백엔드와 통신합니다. 이 기능을 사용하려면 ONTAP 9.11.1 이상이 필요합니다. 또한 사용되는 ONTAP 로그인 역할에 대한 액세스 권한이 있어야 합니다 ontap 응용 프로그램. 이는 사전 정의된 에 의해 충족됩니다 vsadmin 및 cluster-admin 역할.	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.	"" "" ""
credentials	AWS Secret Manager에 저장할 FSx SVM 자격 증명을 지정합니다. - name: SVM의 자격 증명에 포함된 비밀의 ARN(Amazon Resource Name). - type: 로 설정합니다 awsarn. 을 참조하십시오 "AWS Secrets Manager 암호를 생성합니다" 를 참조하십시오.	

업데이트 dataLIF 초기 구성 후

다음 명령을 실행하여 초기 구성 후에 데이터 LIF를 변경할 수 있으며, 업데이트된 데이터 LIF가 포함된 새 백엔드 JSON 파일을 제공할 수 있습니다.

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



PVC가 하나 이상의 포드에 연결된 경우 해당 포드를 모두 내린 다음 다시 불러와서 새 데이터 LIF가 적용되도록 해야 합니다.

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"(싌) 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. Astra Trident와 함께 QoS 정책 그룹을 사용하려면 ONTAP 9.8 이상이 필요합니다. 비공유 QoS 정책 그룹을 사용하고 정책 그룹이 각 구성요소별로 적용되도록 하는 것이 좋습니다. 공유 QoS 정책 그룹은 모든 워크로드의 총 처리량에 대해 상한을 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 예약된 볼륨의 백분율	If(경우 snapshotPolicy 있습니다 none, else ""
'plitOnClone'을 선택합니다	생성 시 상위 클론에서 클론을 분할합니다	거짓입니다
암호화	새 볼륨에 NVE(NetApp Volume Encryption)를 사용하도록 설정하고 기본값은 'false'입니다. 이 옵션을 사용하려면 NVE 라이선스가 클러스터에서 활성화되어 있어야 합니다. 백엔드에서 NAE가 활성화된 경우 Astra Trident에 프로비저닝된 모든 볼륨은 NAE가 활성화됩니다. 자세한 내용은 다음을 참조하십시오. "Astra Trident가 NVE 및 NAE와 연동되는 방식" .	거짓입니다
luksEncryption	LUKS 암호화를 사용합니다. 을 참조하십시오 "LUKS(Linux Unified Key Setup) 사용" . SAN만 해당.	""
'계층화 정책'	사용할 계층화 정책 none	snapshot-only ONTAP 9.5 이전 SVM-DR 구성용
유니크권한	모드를 선택합니다. * SMB 볼륨의 경우 비워 둡니다. *	""

매개 변수	설명	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.

예제 설정

SMB 볼륨에 대한 스토리지 클래스를 구성합니다

사용 nasType, node-stage-secret-name, 및 node-stage-secret-namespace, SMB 볼륨을 지정하고 필요한 Active Directory 자격 증명을 제공할 수 있습니다. SMB 볼륨은 를 사용하여 지원됩니다 ontap-nas 드라이버만 해당.

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



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

EKS 클러스터에서 **Astra Trident EKS** 애드온 버전 **23.10**을 구성합니다

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

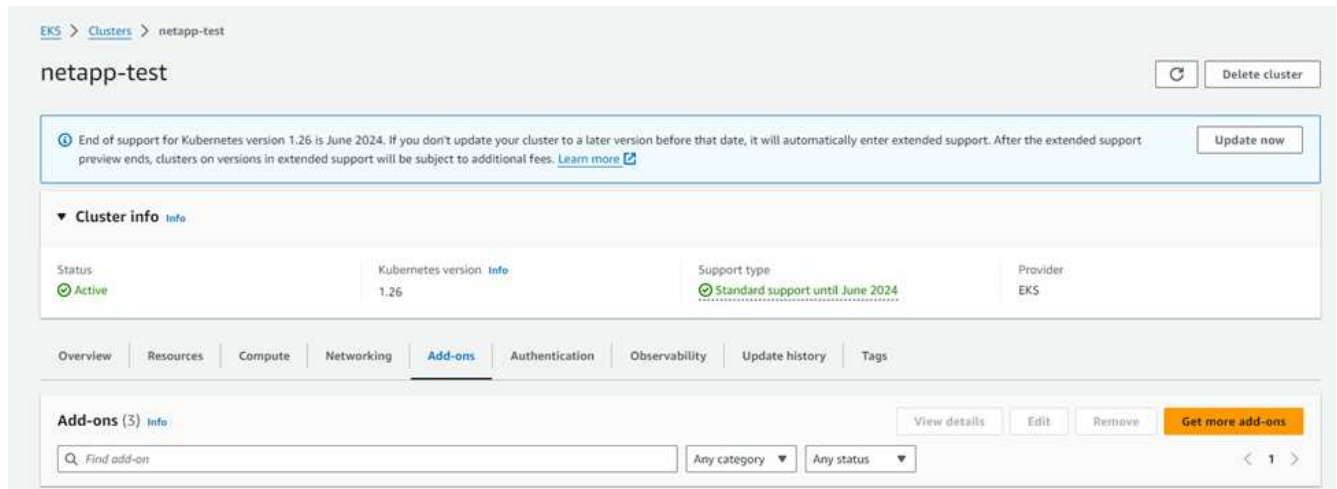
필수 구성 요소

AWS EKS용 Astra Trident 애드온을 구성하기 전에 다음 사항을 확인하십시오.

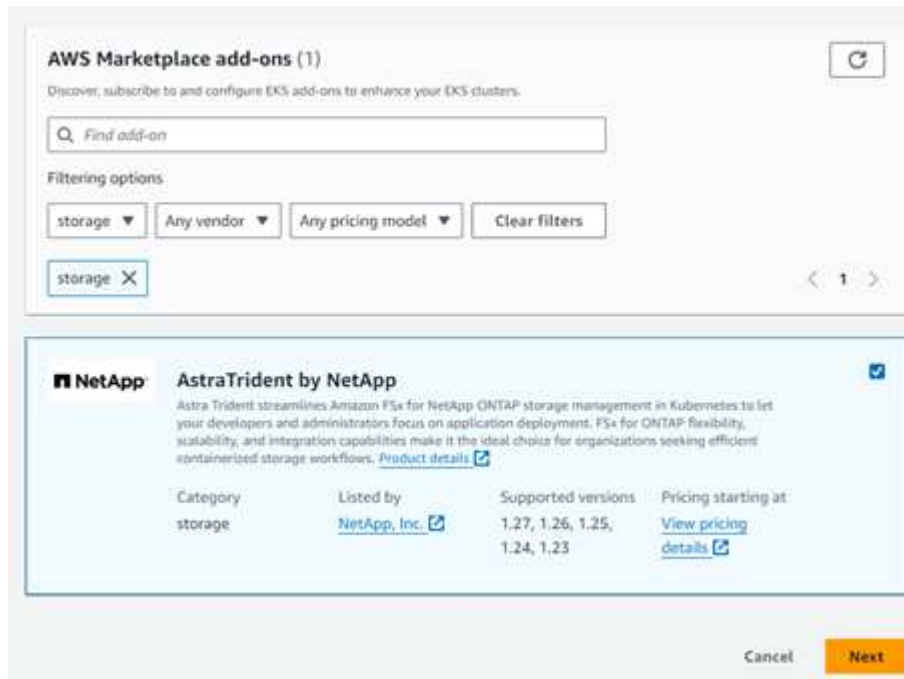
- 애드온 가입이 있는 Amazon EKS 클러스터 계정입니다
- AWS 마켓플레이스에 대한 AWS 권한:
"aws-marketplace:ViewSubscriptions",
"aws-marketplace:Subscribe",
"aws-marketplace:Unsubscribe"
- AMI 유형: Amazon Linux 2 (AL2_x86_64) 또는 Amazon Linux 2 Arm (AL2_ARM_64)
- 노드 유형: AMD 또는 ARM
- 기존 Amazon FSx for NetApp ONTAP 파일 시스템

단계

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



2. AWS Marketplace 애드온 * 으로 이동하여 _STORAGE_CATEGORY를 선택합니다.



3. AstraTrident by NetApp * 를 찾고 Astra Trident 애드온 확인란을 선택합니다.
4. 원하는 추가 기능 버전을 선택합니다.

Astra Trident by NetApp Remove add-on

Listed by **NetApp** Category storage Status Ready to install

You're subscribed to this software View subscription ×
 You can view the terms and pricing details for this product or choose another offer if one is available.

Version
 Select the version for this add-on.
 v23.10.0-eksbuild.1

Select IAM role
 Select an IAM role to use with this add-on. To create a new role, follow the instructions in the [Amazon EKS User Guide](#).
 Not set ↕ ↻
 Filter roles
 Not set ✓
 This add-on will use the IAM role of the node where it runs.

Cancel Previous Next

5. 노드에서 상속할 IAM 역할 옵션을 선택합니다.
6. 필요에 따라 옵션 설정을 구성하고 * 다음 * 을 선택합니다.

Review and add

Step 1: Select add-ons Edit

Selected add-ons

< 1 >

Add-on name	Type	Status
netapp_trident-operator	storage	Ready to install

Step 2: Configure selected add-ons settings Edit

Selected add-ons version

Add-on name	Version	IAM role
netapp_trident-operator	v23.10.0-eksbuild.1	Inherit from node

Cancel Previous Create

7. Create * 를 선택합니다.
8. 애드온의 상태가 _Active_인지 확인합니다.



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

CLI를 사용하여 **Astra Trident EKS** 애드온을 설치합니다.

다음 명령 예에서는 Astra Trident EKS 애드온을 설치합니다.

```
eksctl create addon --cluster K8s-arm --name netapp_trident-operator --version v23.10.0-eksbuild.1
eksctl create addon --cluster K8s-arm --name netapp_trident-operator --version v23.10.0-eksbuild.1 (전용 버전 포함)
```

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

다음 명령을 실행하면 Astra Trident EKS 애드온이 제거됩니다.

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

kubectl로 백엔드를 만듭니다

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

'트리펜엔드구성' 사용자 정의 리소스 정의(CRD)를 사용하면 Kubernetes 인터페이스를 통해 Trident 백엔드를 직접 생성하고 관리할 수 있습니다. 쿠버네티스나 이와 동등한 CLI 툴을 사용하여 Kubernetes를 배포할 수 있습니다.

TridentBackendConfig

트리젠벤디Config('tbc', 'tbconfig', 'tbackendconfig')는 쿠버네티스로 아스트라 트리덴트 백엔드를 관리할 수 있는 프론트이자 이름있는 CRD입니다. 이제 Kubernetes 및 스토리지 관리자는 전용 명령줄 유틸리티('tridentctl')를 사용하지 않고 Kubernetes CLI를 통해 직접 백엔드를 생성 및 관리할 수 있습니다.

'트리펜엔드구성' 객체가 생성되면 다음과 같은 현상이 발생합니다.

- 백엔드는 사용자가 제공하는 구성에 따라 Astra Trident에서 자동으로 생성합니다. 이는 내부적으로 트리덴트백엔드(트리덴트백엔드) CR로 표현된다.
- 트리젠트백엔드구성은 아스트라 트리덴트(Astra Trident)가 만든 트리젠백엔드(트리젠백엔드)에 고유하게 바인딩됩니다.

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.



트리젠백엔드 CRS는 아스트라 트리덴트(Astra Trident)에 의해 자동으로 생성됩니다. 수정할 수 없습니다. 백엔드에 대한 업데이트를 하려면 '트리엔백구성' 개체를 수정하여 이 작업을 수행하십시오.

'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 도구"](#) 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 트리엔백엔드Config CR에 새로 도입된 자격 증명과 deletionPolicy 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 트리엔BackendConfig가 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]: 트리엔BackendConfig 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.



tridentctl로 만든 백엔드에는 관련된 트리엔BackendConfig 객체가 없습니다. 트리엔백엔드Config CR을 만들어 kubctl로 백엔드를 관리할 수 있습니다. 동일한 구성 매개 변수(예: 'pec.backendName', 'pec.storagePrefix', 'pec.storageDriverName' 등)를 지정할 때는 주의해야 합니다. Astra Trident는 새로 생성된 '트리엔백엔드구성'을 기존 백엔드와 자동으로 바인딩합니다.

단계 개요

kubbeck을 사용하여 새 백엔드를 생성하려면 다음을 수행해야 합니다.

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

백엔드를 생성한 후 "kubectl 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.

```
kubectll -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: t@Ax@7q(>
```

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
GCP용 Cloud Volumes Service	private_key_id	개인 키의 ID입니다. CVS 관리자 역할을 가진 GCP 서비스 계정에 대한 API 키의 일부
GCP용 Cloud Volumes Service	개인 키	개인 키. CVS 관리자 역할을 가진 GCP 서비스 계정에 대한 API 키의 일부
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

스토리지 플랫폼 암호 필드 설명입니다	비밀	필드 설명입니다
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	채터 사용자 이름	인바운드 사용자 이름입니다. useCHAP = TRUE인 경우 필수입니다. ONTAP-SAN과 ONTAP-SAN 경제입니다
ONTAP	채터시토키트	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
ONTAP	채터타겟이니터시키키트	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
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(PVCs)이 없는 경우, 트리엔백엔드구성을 삭제하면 Astra Trident가 백엔드를 삭제하고 '트리엔백구성' 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 '트리엔백엔드구성' CR은 삭제할 수 있습니다.
- **Unknown:** Astra Trident가 와 연결된 백엔드의 상태 또는 존재를 확인할 수 없습니다 TridentBackendConfig 있습니다. 예를 들어, API 서버가 응답하지 않거나 가 응답하지 않는 경우

tridentbackends.trident.netapp.io CRD가 누락되었습니다. 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'에는 '트리젠BackendConfig' CR에 대응하여 만든 백엔드의 'backendName'과 'backendUUID'가 포함되어 있습니다. 'lastOperationStatus' 필드는 사용자 트리거(예: 사용자가 'spec'에서 무언가를 변경한 경우) 또는 Astra Trident(예: Astra Trident 재시작 시)에 의해 트리거될 수 있는 '트리젠백엔드 Config' CR의 마지막 작업 상태를 나타냅니다. 성공 또는 실패일 수 있습니다. 단계는 트리젠백엔드Config CR과 백엔드 간의 관계를 나타냅니다. 위의 예에서 'phase'는 값이 바인딩되어 있어 '트리젠백엔드구성' CR이 백엔드와 연결되어 있음을 의미합니다.

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



tridentctl을 사용하여 연결된 'TridentBackendConfig' 객체가 포함된 백엔드는 업데이트하거나 삭제할 수 없습니다. tridentctl과 트리젠BackendConfig의 전환 단계를 이해하려면 [여기](#)를 참조하십시오.

백엔드 관리

kubeck을 사용하여 백엔드 관리 수행

kubctl을 사용하여 백엔드 관리 작업을 수행하는 방법에 대해 알아보십시오.

백엔드를 삭제합니다

'트리엔BackendConfig'를 삭제하면 Astra Trident가 백엔드 삭제/보존('정책' 기준)을 수행하도록 지시합니다. 백엔드를 삭제하려면 deletionPolicy가 delete로 설정되어 있는지 확인합니다. 트리엔백엔드구성만 삭제하려면, '정책'이 '유지'로 설정되어 있는지 확인하십시오. 이렇게 하면 백엔드가 계속 존재하고 'tridentctl'을 사용하여 관리할 수 있습니다.

다음 명령을 실행합니다.

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

Astra Trident는 트리엔엔드구성에서 사용 중인 Kubernetes 비밀을 삭제하지 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 '트리엔엔드구성' 객체에 사용되는 Kubernetes Secret를 업데이트해야 합니다. Astra Trident가 자동으로 백엔드를 제공된 최신 자격 증명으로 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. 'kubctl get tbc<tbname>-o YAML-n trident' 또는 'kubctl t설명해 tbc<tbname>-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

Astra Trident에서 백엔드를 삭제하려면 다음을 수행합니다.

1. 백엔드 이름 검색:

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

2. 백엔드를 삭제합니다.

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



Astra Trident가 백엔드에서 여전히 존재하는 볼륨 및 스냅샷을 프로비저닝한 경우 백엔드를 삭제하면 새 볼륨이 백엔드에서 프로비저닝되지 않습니다. 백엔드는 계속해서 "삭제" 상태에 있으며, Trident는 삭제될 때까지 해당 볼륨 및 스냅샷을 계속 관리합니다.

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

Astra 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`.
- `tridentctl`로 만든 새 백엔드와 다른 트리젠BackendConfig 개체가 있습니다.

두 시나리오 모두 Astra Trident가 볼륨을 예약하고 운영하면서 백엔드가 계속 존재할 것입니다. 관리자는 다음 두 가지 옵션 중 하나를 선택할 수 있습니다.

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

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

1. Kubernetes 암호를 생성하십시오. 비밀에는 Astra Trident가 스토리지 클러스터/서비스와 통신하는 데 필요한 자격 증명에 포함되어 있습니다.
2. '트리젠백엔드구성' 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: A를 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
tbc-ontap-nas-backend	ontap-nas-backend	52f2eb10-e4c6-4160-99fc-96b3be5ab5d7

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

NAME	STORAGE DRIVER	UUID
ontap-nas-backend	ontap-nas	52f2eb10-e4c6-4160-99fc-96b3be5ab5d7

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

관리 TridentBackendConfig 을 사용하여 백엔드를 만듭니다 tridentctl

트리덴ctl은 트리엔백구성(TridentBackendConfig)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
backend-tbc-ontap-san	ontap-san-backend	81abcb27-ea63-49bb-b606-0a5315ac5f82

```
tridentctl get backend ontap-san-backend -n trident
```

NAME	STORAGE DRIVER	UUID
ontap-san-backend	ontap-san	81abcb27-ea63-49bb-b606-0a5315ac5f82

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

1단계: 확인 deletionPolicy 가 로 설정되어 있습니다 retain

정책이라는 가치를 한번 살펴보자. 이를 보철로 설정해야 합니다. 이렇게 하면 'TridentBackendConfig' CR을 삭제할 때 백엔드 정의가 계속 존재하고 'tridentctl'을 사용하여 관리할 수 있습니다.

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

NAME	BACKEND NAME	BACKEND UUID
backend-tbc-ontap-san	ontap-san-backend	81abcb27-ea63-49bb-b606-0a5315ac5f82

```
# 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
backend-tbc-ontap-san	ontap-san-backend	81abcb27-ea63-49bb-b606-0a5315ac5f82



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

트리젠벤트Config 객체를 삭제하면 Astra Trident는 백엔드 자체를 삭제하지 않고 간단히 해당 객체를 제거합니다.

저작권 정보

Copyright © 2024 NetApp, Inc. All Rights Reserved. 미국에서 인쇄된 본 문서의 어떠한 부분도 저작권 소유자의 사전 서면 승인 없이는 어떠한 형식이나 수단(복사, 녹음, 녹화 또는 전자 검색 시스템에 저장하는 것을 비롯한 그래픽, 전자적 또는 기계적 방법)으로도 복제될 수 없습니다.

NetApp이 저작권을 가진 자료에 있는 소프트웨어에는 아래의 라이선스와 고지사항이 적용됩니다.

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

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

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

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

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

상표 정보

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