



모범 사례 및 권장사항

Trident

NetApp
February 02, 2026

목차

모범 사례 및 권장사항	1
구축	1
전용 네임스페이스에 구축	1
할당량 및 범위 제한을 사용하여 스토리지 사용량을 제어할 수 있습니다	1
스토리지 구성	1
플랫폼 개요	1
ONTAP 및 Cloud Volumes ONTAP 모범 사례	1
SolidFire 모범 사례	6
자세한 정보는 어디서 찾을 수 있습니까?	7
Trident 통합	8
운전자 선택 및 전개	8
스토리지 클래스 설계	10
가상 풀 설계	11
볼륨 작업입니다	12
메트릭 서비스	15
데이터 보호 및 재해 복구	16
Trident 복제 및 복구	16
SVM 복제 및 복구	17
볼륨 복제 및 복구	18
스냅샷 데이터 보호	18
Trident 사용하여 상태 저장 애플리케이션의 장애 조치 자동화	19
강제 분리에 대한 세부 정보	19
자동 장애 조치에 대한 세부 정보	19
보안	24
보안	24
Linux 통합 키 설정(LUKS)	25
Kerberos 전송 중 암호화	31

모범 사례 및 권장사항

구축

Trident를 구축할 때 여기에 나열된 권장 사항을 사용합니다.

전용 네임스페이스에 구축

"[네임스페이스](#)" 서로 다른 응용 프로그램 간의 관리 분리 기능을 제공하며 리소스 공유에 대한 장벽입니다. 예를 들어, 한 네임스페이스의 PVC는 다른 네임스페이스에서 사용할 수 없습니다. Trident는 Kubernetes 클러스터의 모든 네임스페이스에 PV 리소스를 제공하므로 Privileges가 상승한 서비스 계정을 활용합니다.

또한 Trident Pod에 액세스하면 사용자가 스토리지 시스템 자격 증명 및 기타 중요한 정보에 액세스할 수 있습니다. 애플리케이션 사용자 및 관리 애플리케이션에서 Trident 객체 정의 또는 POD 자체에 액세스할 수 없도록 하는 것이 중요합니다.

할당량 및 범위 제한을 사용하여 스토리지 사용량을 제어할 수 있습니다

Kubernetes에는 2가지 기능이 있으며, 이 기능을 조합하여 애플리케이션의 리소스 사용을 제한하는 강력한 메커니즘을 제공합니다. 를 클릭합니다. ["스토리지 할당량 메커니즘"](#) 관리자가 네임스페이스별로 글로벌 및 스토리지 클래스별, 용량 및 오브젝트 수 사용 제한을 구현할 수 있도록 지원 또한 를 사용합니다. ["범위 제한"](#) 요청이 프로비저닝 사용자에게 전달되기 전에 PVC 요청이 최소값 및 최대값 내에 있는지 확인합니다.

이러한 값은 네임스페이스 단위로 정의됩니다. 즉, 각 네임스페이스에는 리소스 요구 사항에 맞는 값이 정의되어 있어야 합니다. 에 대한 자세한 내용은 여기를 참조하십시오 ["할당량을 활용하는 방법"](#).

스토리지 구성

NetApp 포트폴리오의 각 스토리지 플랫폼은 컨테이너식으로 애플리케이션에 이점을 제공하는 고유한 기능을 제공합니다.

플랫폼 개요

Trident는 ONTAP 및 요소와 함께 작동합니다. 모든 애플리케이션과 시나리오에 적합한 플랫폼이 한 개 있는 것은 아니지만, 플랫폼을 선택할 때 애플리케이션과 장치를 관리하는 팀의 요구 사항을 고려해야 합니다.

활용 중인 프로토콜을 사용하여 호스트 운영 체제의 기준 모범 사례를 따라야 합니다. 필요에 따라 특정 애플리케이션에 맞게 스토리지를 최적화할 수 있도록 백엔드, 스토리지 클래스 및 PVC 설정과 함께 사용 가능한 경우 애플리케이션 Best Practice를 통합하는 것을 고려할 수 있습니다.

ONTAP 및 Cloud Volumes ONTAP 모범 사례

Trident를 위한 ONTAP 및 Cloud Volumes ONTAP를 구성하기 위한 모범 사례에 대해 알아보십시오.

다음 권장 사항은 Trident에서 동적으로 프로비저닝되는 볼륨을 사용하는 컨테이너식 워크로드에 대한 ONTAP 구성 지침입니다. 각 항목을 고려하여 작업 환경의 적절성을 판단해야 합니다.

Trident 전용 SVM을 사용하십시오

SVM(스토리지 가상 시스템)은 ONTAP 시스템의 테넌트 간에 격리하고 관리를 제공합니다. SVM을 애플리케이션 전용으로 사용하면 권한을 위임하고 리소스 사용을 제한하는 모범 사례를 적용할 수 있습니다.

SVM 관리를 위해 몇 가지 옵션을 사용할 수 있습니다.

- 백엔드 구성에서 클러스터 관리 인터페이스를 적절한 자격 증명과 함께 제공하고 SVM 이름을 지정합니다.
- ONTAP System Manager 또는 CLI를 사용하여 SVM을 위한 전용 관리 인터페이스를 생성합니다.
- 관리 역할을 NFS 데이터 인터페이스와 공유합니다.

각 경우 인터페이스가 DNS에 있어야 하며, Trident를 구성할 때 DNS 이름을 사용해야 합니다. 이렇게 하면 네트워크 ID 보존을 사용하지 않고 SVM-DR과 같은 일부 DR 시나리오를 간편하게 수행할 수 있습니다.

SVM에 전용 또는 공유 관리 LIF를 사용하는 것이 더 이상 선호되지 않지만, 선택한 접근 방식에 맞게 네트워크 보안 정책을 조정해야 합니다. 관리 LIF는 DNS를 통해 액세스할 수 있어야 하며, 유연성을 극대화해야 합니다. "SVM-DR" Trident와 함께 사용합니다.

최대 볼륨 수를 제한합니다

ONTAP 스토리지 시스템의 최대 볼륨 수는 소프트웨어 버전과 하드웨어 플랫폼에 따라 다릅니다. 을 참조하십시오 "NetApp Hardware Universe를 참조하십시오" 정확한 제한을 결정하는 특정 플랫폼 및 ONTAP 버전. 볼륨 수가 소진되면 프로비저닝 작업이 Trident뿐 아니라 모든 스토리지 요청에 대해 실패합니다.

Trident의 'ONTAP-NAS' 및 'ONTAP-SAN' 드라이버는 생성되는 각 Kubernetes 영구 볼륨(PV)에 대해 FlexVolume을 프로비저닝합니다. 'ONTAP-NAS-이코노미' 드라이버는 200개의 PVS에 대해 약 1개의 FlexVolume을 생성합니다(50개에서 300개로 구성 가능). 'ONTAP-SAN-이코노미' 드라이버는 PVS 100대당 약 1개의 FlexVolume을 생성합니다(50개에서 200개로 구성 가능). Trident가 스토리지 시스템에서 사용 가능한 모든 볼륨을 사용하지 않도록 하려면 SVM에 제한을 설정해야 합니다. 이 작업은 명령줄에서 수행할 수 있습니다.

```
vserver modify -vserver <svm_name> -max-volumes <num_of_volumes>
```

'최대 볼륨'의 값은 환경에 따라 몇 가지 기준에 따라 달라집니다.

- ONTAP 클러스터에 있는 기존 볼륨의 수입니다
- 다른 애플리케이션에 대해 Trident 외부에 프로비저닝할 것으로 예상되는 볼륨 수입니다
- Kubernetes 애플리케이션에서 사용할 것으로 예상되는 영구 볼륨의 수입니다

max-volumes 값은 개별 ONTAP 노드가 아닌 ONTAP 클러스터의 모든 노드에 프로비저닝된 총 볼륨입니다. 결과적으로, ONTAP 클러스터 노드에 다른 노드보다 훨씬 더 많은 Trident 프로비저닝 볼륨이 있을 수 있는 몇 가지 조건이 발생할 수 있습니다.

예를 들어, 2노드 ONTAP 클러스터에는 최대 2000개의 FlexVol 볼륨을 호스팅할 수 있습니다. 최대 볼륨 수를 1250으로 설정하면 매우 적절합니다. 그러나 한 노드에서만 SVM에 할당되거나 한 노드에서 할당된 애그리게이트를 용량 등)에 대해 프로비저닝할 수 없는 경우 "애그리게이트" 다른 노드는 모든 Trident 프로비저닝된 볼륨의 대상이 됩니다. 즉, 값에 도달하기 전에 해당 노드의 볼륨 제한에 도달하여 Trident 및 해당 노드를 사용하는 기타 볼륨 작업에 모두 영향을 줄 수 있습니다 max-volumes. * 클러스터의 각 노드에서 애그리게이트가 동일한 수의 Trident가 사용하는 SVM에 할당되도록 하면 이러한 상황을 방지할 수 있습니다. *

볼륨의 클론을 생성합니다

NetApp Trident 다음을 사용할 때 볼륨 복제를 지원합니다. `ontap-nas`, `ontap-san`, 그리고 `solidfire-san` 스토리지 드라이버. 사용 시 `ontap-nas-flexgroup` 또는 `ontap-nas-economy` 드라이버, 복제가 지원되지 않습니다. 기존 볼륨에서 새 볼륨을 생성하면 새로운 스냅샷이 생성됩니다.



다른 StorageClass와 연결된 PVC를 복제하지 마십시오. 호환성을 보장하고 예기치 않은 동작을 방지하려면 동일한 StorageClass 내에서 복제 작업을 수행하십시오.

Trident에서 생성한 볼륨의 최대 크기를 제한합니다

Trident에서 생성할 수 있는 볼륨의 최대 크기를 구성하려면 'backend.json' 정의에 있는 'limitVolumeSize' 매개 변수를 사용하십시오.

스토리지 어레이에서 볼륨 크기를 제어하는 것 외에도 Kubernetes 기능을 활용해야 합니다.

Trident에 의해 생성되는 FlexVols의 최대 크기를 제한합니다

ONTAP-SAN-Economy 및 ONTAP-NAS-Economy 드라이버의 풀로 사용되는 FlexVol의 최대 크기를 구성하려면 `limitVolumePoolSize` backend.json 정의에 매개 변수를 사용하십시오.

양방향 CHAP를 사용하도록 Trident를 구성합니다

백엔드 정의에 CHAP 이니시에이터와 타겟 사용자 이름 및 암호를 지정하고 SVM에서 Trident가 CHAP를 사용하도록 설정할 수 있습니다. 이를 사용합니다 `useCHAP` 매개 변수 백엔드 구성에서 Trident는 CHAP를 사용하여 ONTAP 백엔드에 대한 iSCSI 연결을 인증합니다.

SVM QoS 정책을 생성하고 사용합니다

SVM에 적용되는 ONTAP QoS 정책을 활용하여 Trident에서 프로비저닝된 볼륨에서 사용할 수 있는 IOPS 수를 제한합니다. 그러면 예 도움이 됩니다 "[괴롭힘을 방지합니다](#)" 또는 Trident SVM 외부의 워크로드에 영향을 주지 않는 제어 컨테이너

몇 가지 단계로 SVM에 대한 QoS 정책을 생성할 수 있습니다. 가장 정확한 정보는 사용 중인 ONTAP 버전 설명서를 참조하십시오. 아래 예는 SVM에 사용 가능한 총 IOPS를 5000으로 제한하는 QoS 정책을 생성합니다.

```
# create the policy group for the SVM
qos policy-group create -policy-group <policy_name> -vserver <svm_name>
-max-throughput 5000iops

# assign the policy group to the SVM, note this will not work
# if volumes or files in the SVM have existing QoS policies
vserver modify -vserver <svm_name> -qos-policy-group <policy_name>
```

또한 사용하는 ONTAP 버전에서 지원하는 경우에는 최소 QoS를 사용하여 컨테이너화된 워크로드에 대한 처리량을 보장하는 것을 고려할 수 있습니다. 적응형 QoS는 SVM 레벨 정책과 호환되지 않습니다.

컨테이너화된 워크로드 전용 IOPS 수는 다양한 측면에 따라 다릅니다. 그 밖의 다른 사항으로는 다음과 같은 것들이 있습니다.

- 기타 워크로드는 스토리지 어레이를 사용합니다. 스토리지 리소스를 활용하여 Kubernetes 구축과 관련되지 않은 다른 워크로드가 있는 경우, 해당 워크로드가 실수로 영향을 받지 않도록 주의해야 합니다.
- 컨테이너에서 실행 중인 예상 워크로드 IOPS 요구사항이 높은 워크로드를 컨테이너에서 실행할 경우 QoS 정책이 낮으면 잘못된 경험이 될 수 있습니다.

SVM 레벨에서 할당된 QoS 정책을 사용하면 동일한 IOPS 풀을 공유하는 SVM에 프로비저닝된 모든 볼륨이 생성된다는 점을 기억해야 합니다. 컨테이너화된 애플리케이션 중 하나 또는 그 수가 적은 경우 높은 IOPS 요구사항이 있으면 다른 컨테이너화된 워크로드에 문제가 될 수 있습니다. 이 경우 외부 자동화를 사용하여 볼륨당 QoS 정책을 할당하는 것을 고려할 수 있습니다.



ONTAP 버전이 9.8 이전인 경우 SVM * 에만 QoS 정책 그룹을 할당해야 합니다.

Trident에 대한 QoS 정책 그룹을 생성합니다

QoS(서비스 품질)는 경쟁 워크로드로부터 주요 워크로드의 성능이 저하되지 않도록 보장합니다. ONTAP QoS 정책 그룹은 볼륨에 대한 QoS 옵션을 제공하고 사용자가 하나 이상의 워크로드에 대한 처리량 한도를 정의할 수 있도록 지원합니다. QoS에 대한 자세한 내용은 ["QoS를 통해 처리량 보장"](#). 백엔드에서 또는 스토리지 풀에 QoS 정책 그룹을 지정할 수 있으며, 이러한 그룹은 해당 풀 또는 백엔드에서 생성된 각 볼륨에 적용됩니다.

ONTAP에는 기존 QoS 정책과 적응형 서비스 두 가지 QoS 정책 그룹이 있습니다. 기존 정책 그룹은 IOPS 단위로 최대 또는 최소 단위의 고정 처리량을 제공합니다. 적응형 QoS는 워크로드 크기에 따라 처리량을 자동으로 확장하므로 워크로드 크기에 따라 IOPS와 TB|GB의 비율을 유지합니다. 따라서 대규모 구축 환경에서 수백 또는 수천 개의 워크로드를 관리할 경우 상당한 이점이 있습니다.

QoS 정책 그룹을 생성할 때는 다음 사항을 고려하십시오.

- 백엔드 구성의 "defaults" 블록에 qosPolicy 키를 설정해야 합니다. 다음 백엔드 구성 예를 참조하십시오.

```

---
version: 1
storageDriverName: ontap-nas
managementLIF: 0.0.0.0
dataLIF: 0.0.0.0
svm: svm0
username: user
password: pass
defaults:
  qosPolicy: standard-pg
storage:
  - labels:
      performance: extreme
    defaults:
      adaptiveQosPolicy: extremely-adaptive-pg
  - labels:
      performance: premium
    defaults:
      qosPolicy: premium-pg

```

- 각 볼륨이 정책 그룹에서 지정한 전체 처리량을 얻을 수 있도록 볼륨별로 정책 그룹을 적용해야 합니다. 공유 정책 그룹은 지원되지 않습니다.

QoS 정책 그룹에 대한 자세한 내용은 ["ONTAP 명령 참조입니다"](#) 참조하십시오.

스토리지 리소스에 대한 액세스 권한을 **Kubernetes** 클러스터 구성원으로 제한합니다

Trident에서 생성한 NFS 볼륨, iSCSI LUN 및 FC LUN에 대한 액세스를 제한하는 것은 Kubernetes 배포를 위한 보안 상태의 중요 구성 요소입니다. 이렇게 하면 Kubernetes 클러스터의 일부가 아닌 호스트가 볼륨에 액세스하고 예기치 않게 데이터를 수정하는 것을 방지할 수 있습니다.

네임스페이스가 Kubernetes의 리소스에 대한 논리적 경계라는 것을 이해하는 것이 중요합니다. 동일한 네임스페이스의 리소스를 공유할 수 있다고 가정하지만, 특히 상호 네임스페이스 기능이 없다는 것이 중요합니다. 즉, PVS는 글로벌 객체이지만 PVC에 바인딩되면 동일한 네임스페이스에 있는 Pod에서만 액세스할 수 있습니다. * 적절한 경우 네임스페이스를 사용하여 구분을 제공하는 것이 중요합니다. *

Kubernetes 컨테이너에서 데이터 보안과 관련하여 대부분의 조직은 컨테이너 내의 프로세스가 호스트에 마운트된 스토리지에 액세스할 수 있지만 컨테이너용 프로세스는 아닙니다. ["네임스페이스"](#) 이러한 유형의 손상을 방지하도록 설계되었습니다. 그러나 권한 있는 컨테이너에는 한 가지 예외가 있습니다.

권한 있는 컨테이너는 일반적인 것보다 훨씬 더 많은 호스트 수준 권한으로 실행되는 컨테이너입니다. 이러한 기능은 기본적으로 거부되지 않으므로 이를 사용하여 기능을 사용하지 않도록 설정해야 합니다. ["POD 보안 정책"](#).

Kubernetes 및 외부 호스트 모두에서 액세스가 필요한 볼륨의 경우, Trident에서 관리하지 않고 관리자가 PV를 도입한 상태로 스토리지를 기존 방식으로 관리해야 합니다. 이렇게 하면 Kubernetes 및 외부 호스트의 연결이 모두 끊기고 볼륨을 더 이상 사용하지 않는 경우에만 스토리지 볼륨이 폐기됩니다. 또한, 맞춤형 엑스포트 정책을 적용하여 Kubernetes 클러스터 노드 및 Kubernetes 클러스터 외부의 타겟 서버에서 액세스할 수 있습니다.

전용 인프라 노드(예: OpenShift) 또는 사용자 애플리케이션을 예약할 수 없는 다른 노드를 구축하는 경우, 별도의

엑스포트 정책을 사용하여 스토리지 리소스에 대한 액세스를 더욱 제한해야 합니다. 여기에는 해당 인프라 노드에 배포된 서비스(예: OpenShift Metrics 및 Logging 서비스)에 대한 엑스포트 정책과 비인프라 노드에 배포되는 표준 애플리케이션이 포함됩니다.

전용 엑스포트 정책을 사용하십시오

Kubernetes 클러스터에 있는 노드에만 액세스할 수 있도록 각 백엔드에 대한 엑스포트 정책이 있어야 합니다. Trident는 엑스포트 정책을 자동으로 생성하고 관리할 수 있습니다. 이러한 방법으로 Trident는 Kubernetes 클러스터의 노드에 프로비저닝되는 볼륨에 대한 액세스를 제한하고 노드 추가/삭제를 단순화합니다.

또는 수동으로 엑스포트 정책을 생성하여 각 노드 액세스 요청을 처리하는 하나 이상의 엑스포트 규칙으로 채울 수도 있습니다.

- vserver export-policy create ONTAP CLI 명령을 사용하여 엑스포트 정책을 생성합니다.
- vserver export-policy rule create ONTAP CLI 명령을 사용하여 엑스포트 정책에 규칙을 추가합니다.

이러한 명령을 실행하면 데이터에 액세스할 수 있는 Kubernetes 노드를 제한할 수 있습니다.

사용 안 함 showmount 애플리케이션 SVM을 위해

이 showmount 기능을 사용하면 NFS 클라이언트가 SVM에서 사용 가능한 NFS 엑스포트 목록을 쿼리할 수 있습니다. Kubernetes 클러스터에 배포된 Pod는 에 대해 명령을 실행하고 액세스 권한이 없는 마운트를 포함하여 사용 가능한 마운트 목록을 받을 수 있습니다 showmount -e. 이는 그 자체로 보안 문제가 아니라, 권한이 없는 사용자가 NFS 내보내기에 연결하는 데 도움이 될 수 있는 불필요한 정보를 제공합니다.

SVM 레벨의 ONTAP CLI 명령을 사용하여 'howmount'를 비활성화해야 합니다.

```
vserver nfs modify -vserver <svm_name> -showmount disabled
```

SolidFire 모범 사례

Trident를 위한 SolidFire 스토리지를 구성하기 위한 모범 사례에 대해 알아보십시오.

SolidFire 계정을 만듭니다

각 SolidFire 계정은 고유한 볼륨 소유자를 나타내며 자체 CHAP(Challenge-Handshake 인증 프로토콜) 자격 증명을 받습니다. 계정 이름 및 상대 CHAP 자격 증명을 사용하거나 볼륨 액세스 그룹을 통해 계정에 할당된 볼륨에 액세스할 수 있습니다. 계정에는 최대 2천 개의 볼륨이 할당될 수 있지만 볼륨은 하나의 계정에만 속할 수 있습니다.

QoS 정책을 생성합니다

여러 볼륨에 적용할 수 있는 표준화된 서비스 품질 설정을 만들어 저장하려면 SolidFire 서비스 품질(QoS) 정책을 사용하십시오.

볼륨별로 QoS 매개 변수를 설정할 수 있습니다. QoS를 정의하는 세 가지 구성 가능한 매개 변수, 즉 Min IOPS, Max IOPS, Burst IOPS를 설정하여 각 볼륨의 성능을 보장할 수 있습니다.

4KB 블록 크기에 대해 가능한 최소, 최대 및 버스트 IOPS 값입니다.

IOPS 매개 변수입니다	정의	최소 값	기본값	최대 가치(4KB)
최소 IOPS	볼륨에 대한 보장된 성능 수준.	50	50	15000
최대 IOPS	성능은 이 제한을 초과하지 않습니다.	50	15000	200,000
버스트 IOPS	짧은 버스트 시나리오에서 허용되는 최대 IOPS입니다.	50	15000	200,000



최대 IOPS와 버스트 IOPS는 최대 200,000으로 설정할 수 있지만, 실제 볼륨의 최대 성능은 클러스터 사용량 및 노드당 성능에 의해 제한됩니다.

블록 크기와 대역폭은 IOPS 수에 직접적인 영향을 미칩니다. 블록 크기가 증가함에 따라 시스템에서 더 큰 블록 크기를 처리하는 데 필요한 수준까지 대역폭을 높일 수 있습니다. 대역폭이 증가할수록 시스템에서 달성할 수 있는 IOPS의 수가 감소합니다. 을 참조하십시오 ["SolidFire 서비스 품질"](#) QoS 및 성능에 대한 자세한 내용은 를 참조하십시오.

SolidFire 인증

요소는 CHAP 및 vag(볼륨 액세스 그룹)의 두 가지 인증 방법을 지원합니다. CHAP는 CHAP 프로토콜을 사용하여 호스트를 백엔드에 인증합니다. 볼륨 액세스 그룹은 프로비전되는 볼륨에 대한 액세스를 제어합니다. NetApp은 CHAP를 사용하여 인증을 수행하는 것이 더 간단하고 확장 제한이 없기 때문에 CHAP를 사용하는 것이 좋습니다.



CSI 프로비저닝이 강화된 Trident는 CHAP 인증 사용을 지원합니다. VAG는 일반적인 비 CSI 작동 모드에서만 사용해야 합니다.

CHAP 인증(이니시에이터가 대상 볼륨 사용자인지 확인)은 계정 기반 액세스 제어에서만 지원됩니다. CHAP를 인증에 사용하는 경우 단방향 CHAP 및 양방향 CHAP의 두 가지 옵션을 사용할 수 있습니다. 단방향 CHAP는 SolidFire 계정 이름 및 이니시에이터 암호를 사용하여 볼륨 액세스를 인증합니다. 양방향 CHAP 옵션은 볼륨이 계정 이름과 이니시에이터 암호를 통해 호스트를 인증한 다음 호스트가 계정 이름과 타겟 암호를 통해 볼륨을 인증하기 때문에 볼륨을 인증하는 가장 안전한 방법을 제공합니다.

그러나 CHAP를 설정할 수 없고 VAG가 필요한 경우 액세스 그룹을 생성하고 호스트 이니시에이터 및 볼륨을 액세스 그룹에 추가합니다. 액세스 그룹에 추가하는 각 IQN은 CHAP 인증을 사용하거나 사용하지 않고 그룹의 각 볼륨에 액세스할 수 있습니다. iSCSI 이니시에이터가 CHAP 인증을 사용하도록 구성된 경우 계정 기반 액세스 제어가 사용됩니다. iSCSI 초기자가 CHAP 인증을 사용하도록 구성되지 않은 경우 볼륨 액세스 그룹 액세스 제어가 사용됩니다.

자세한 정보는 어디서 찾을 수 있습니까?

다음은 몇 가지 모범 사례 문서입니다. 를 검색합니다 ["NetApp 라이브러리"](#) 최신 버전의 경우.

- ONTAP *
- ["NFS Best Practice and Implementation Guide를 참조하십시오"](#)
- ["SAN 관리" \(iSCSI용\)](#)

- "[RHEL용 iSCSI Express 구성](#)"

Element 소프트웨어 *

- "[Linux용 SolidFire 구성](#)"
- NetApp HCI *
- "[NetApp HCI 구축 사전 요구 사항](#)"
- "[NetApp 배포 엔진에 액세스합니다](#)"
- 응용 프로그램 모범 사례 정보 *
- "[ONTAP 기반 MySQL의 모범 사례](#)"
- "[SolidFire 기반 MySQL의 모범 사례](#)"
- "[NetApp SolidFire 및 Cassandra](#)"
- "[SolidFire에 대한 Oracle 모범 사례](#)"
- "[SolidFire에 대한 PostgreSQL Best Practice](#)"

모든 애플리케이션에 구체적인 지침이 있는 것은 아니며 NetApp 팀과 함께 을 사용하는 것이 중요합니다 "[NetApp 라이브러리](#)" 최신 설명서를 참조하십시오.

Trident 통합

Trident를 통합하려면 드라이버 선택 및 배포, 스토리지 클래스 설계, 가상 풀 설계, PVC(영구적 볼륨 클레임)가 스토리지 프로비저닝, 볼륨 운영, Trident를 사용하는 OpenShift 서비스 배포에 미치는 영향 등 통합이 필요합니다.

운전자 선택 및 전개

스토리지 시스템에 사용할 백엔드 드라이버를 선택하고 구축합니다.

ONTAP 백엔드 드라이버

ONTAP 백엔드 드라이버는 사용된 프로토콜과 스토리지 시스템에서 볼륨을 프로비저닝하는 방법에 따라 다릅니다. 따라서 배포할 드라이버를 결정할 때는 신중하게 고려해야 합니다.

상위 레벨에서는 애플리케이션에 공유 스토리지가 필요한 구성 요소(동일한 PVC에 액세스하는 여러 Pod)가 있는 경우 NAS 기반 드라이버가 기본 선택이고 블록 기반 iSCSI 드라이버는 비공유 스토리지의 요구를 충족합니다. 애플리케이션의 요구사항 및 스토리지 및 인프라 팀의 편안함 수준을 기준으로 프로토콜을 선택합니다. 일반적으로 대부분의 애플리케이션에서 두 서버 간에 차이가 거의 없기 때문에 공유 스토리지(둘 이상의 POD에 동시 액세스가 필요한 경우)가 필요한지 여부에 따라 결정하는 경우가 많습니다.

사용 가능한 ONTAP 백엔드 드라이버는 다음과 같습니다.

- ONTAP-NAS: 프로비저닝되는 각 PV는 ONTAP FlexVolume입니다.
- ONTAP-NAS-이코노미: 각 PV 프로비저닝은 qtree이며 FlexVolume당 qtree(기본값 200)를 구성할 수 있습니다.
- ONTAP-NAS-flexgroup: 각 PV는 전체 ONTAP FlexGroup으로 프로비저닝되고 SVM에 할당된 모든 애그리게이트가 사용됩니다.

- ONTAP-SAN: 각 PV는 자체 FlexVolume 내의 LUN입니다.
- ONTAP-SAN-이코노미: 각 PV는 LUN으로 프로비저닝되며 FlexVolume당 구성 가능한 LUN 수(기본값 100)가 있습니다.

세 개의 NAS 드라이버 중 하나를 선택할 경우 해당 기능에 약간의 영향을 줍니다. 이 기능은 응용 프로그램에서 사용할 수 있습니다.

아래 표에서 모든 기능이 Trident를 통해 표시되는 것은 아닙니다. 용량 할당 후 기능을 적용하려면 스토리지 관리자가 일부 기능을 적용해야 합니다. 위 첨자 각주는 기능 및 드라이버별 기능을 구별합니다.

ONTAP NAS 드라이버	스냅샷 수	복제	동적 엑스포트 정책	다중 연결	QoS를 참조하십시오	크기 조정	복제
'ONTAP-NAS'	예	예	Yesfootnote: 5[]	예	Yesfootnote: 1[]	예	Yesfootnote: 1[]
ONTAP-NAS-이코노미	메모:3[]	메모:3[]	Yesfootnote: 5[]	예	메모:3[]	예	메모:3[]
'ONTAP-NAS-Flexgroup'	Yesfootnote: 1[]	아니요	Yesfootnote: 5[]	예	Yesfootnote: 1[]	예	Yesfootnote: 1[]

Trident는 ONTAP용 SAN 드라이버 2개를 제공합니다. 이러한 드라이버는 다음과 같습니다.

ONTAP SAN 드라이버	스냅샷 수	복제	다중 연결	양방향 CHAP	QoS를 참조하십시오	크기 조정	복제
'ONTAP-SAN'	예	예	Yesfootnote: 4[]	예	Yesfootnote: 1[]	예	Yesfootnote: 1[]
ONTAP-SAN-이코노미	예	예	Yesfootnote: 4[]	예	메모:3[]	예	메모:3[]

위 표의 각주: Yes [1]: Trident에서 관리되지 않음 Yes [2]: Trident에서 관리하지만 PV 세밀한 NO [3]: Trident에서 관리하지 않으며 PV 세밀한 예각주:4[]: 원시 블록 볼륨에 대해 지원됨 예: 5[]: Trident에서 지원합니다

PV 세분화되지 않은 기능은 전체 FlexVolume에 적용되고 모든 PVS(즉, 공유 FlexVol의 qtree 또는 LUN)는 공통 스케줄을 공유합니다.

위 표에서 볼 수 있듯이, ONTAP-NAS와 ONTAP-NAS-이코노미 간의 기능 대부분은 동일합니다. 그러나 ONTAP-NAS-이코노미 드라이버는 PV 단위의 일정 제어 기능을 제한하므로 특히 재해 복구 및 백업 계획에 영향을 줄 수 있습니다. ONTAP 스토리지에서 PVC 클론 기능을 활용하고자 하는 개발팀은 ONTAP-NAS, ONTAP-SAN 또는 ONTAP-SAN 절약 드라이버를 사용할 때만 가능합니다.



줄idfire-san의 드라이버도 PVC를 클로닝할 수 있습니다.

Cloud Volumes ONTAP 백엔드 드라이버

Cloud Volumes ONTAP은 NAS 및 SAN 프로토콜(NFS, SMB/CIFS 및 iSCSI)을 지원하는 파일 공유 및 블록 레벨 스토리지 등 다양한 활용 사례에 맞게 엔터프라이즈급 스토리지 기능과 함께 데이터 제어를 제공합니다. Cloud Volume

ONTAP와 호환되는 드라이버는 ONTAP-NAS, ONTAP-NAS-이코노미, ONTAP-SAN, ONTAP-SAN 경제입니다. Cloud Volume ONTAP for Azure, Cloud Volume ONTAP for GCP에 적용할 수 있습니다.

ONTAP 백엔드 드라이버용 Amazon FSx

Amazon FSx for NetApp ONTAP를 사용하면 익숙한 NetApp 기능, 성능 및 관리 기능을 활용하면서 AWS에 데이터를 저장할 때의 단순성, 민첩성, 보안 및 확장성을 활용할 수 있습니다. FSx for ONTAP은 많은 ONTAP 파일 시스템 기능과 관리 API를 지원합니다. Cloud Volume ONTAP와 호환되는 드라이버는 입니다 ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san 및 ontap-san-economy.

NetApp HCI/SolidFire 백엔드 드라이버

NetApp HCI/SolidFire 플랫폼과 함께 사용되는 'olidfire-SAN' 드라이버는 관리자가 QoS 제한을 기반으로 Trident에 대한 Element 백엔드를 구성하는 데 도움이 됩니다. Trident에서 프로비저닝한 볼륨에 대한 특정 QoS 제한을 설정하기 위해 백엔드를 설계하려면 백엔드 파일에 'type' 매개 변수를 사용하십시오. 또한 관리자는 'limitVolumeSize' 매개 변수를 사용하여 스토리지에 생성할 수 있는 볼륨 크기를 제한할 수 있습니다. 현재 볼륨 크기 조정 및 볼륨 복제와 같은 Element 스토리지 기능은 'olidfire-SAN' 드라이버를 통해 지원되지 않습니다. 이러한 작업은 Element 소프트웨어 웹 UI를 통해 수동으로 수행해야 합니다.

SolidFire 드라이버	스냅샷 수	복제	다중 연결	CHAP	QoS를 참조하십시오	크기 조정	복제
'솔더불-산'	예	예	Yes [2]	예	예	예	Yesfootnote: 1[]

각주: Yes [1]: Trident에서 관리하지 않습니다. 예각주:2[]: 원시 블록 볼륨에 대해 지원됩니다

Azure NetApp Files 백엔드 드라이버

Trident는 드라이버를 "Azure NetApp Files" 사용하여 azure-netapp-files 서비스를 관리합니다.

이 드라이버에 대한 자세한 내용 및 구성 방법은 ["Azure NetApp Files에 대한 Trident 백엔드 구성"](#) 참조하십시오.

Azure NetApp Files 드라이버	스냅샷 수	복제	다중 연결	QoS를 참조하십시오	비즈니스	복제
'Azure-NetApp-파일'	예	예	예	예	예	Yesfootnote: 1[]

각주: Yes [1]: Trident에서 관리하지 않습니다

스토리지 클래스 설계

Kubernetes Storage Class 객체를 생성하려면 개별 스토리지 클래스를 구성 및 적용해야 합니다. 이 섹션에서는 애플리케이션에 대한 스토리지 클래스를 설계하는 방법에 대해 설명합니다.

특정 백엔드 활용도

특정 스토리지 클래스 객체 내에서 필터링을 사용하여 해당 스토리지 클래스에 사용할 스토리지 폴 또는 폴 세트를 결정할 수 있습니다. Storage Class에서 'storagePools', 'additionalStoragePools', 'excludeStoragePools' 등의 세

가지 필터를 설정할 수 있습니다.

`storagePools` 매개 변수를 사용하면 지정된 속성과 일치하는 풀 세트로 스토리지를 제한할 수 있습니다. `additionalStoragePools` 매개 변수는 Trident가 프로비저닝에 사용하는 풀 세트를 속성 및 매개 변수로 선택한 풀 세트와 함께 확장하는 데 `storagePools` 사용됩니다. 매개 변수만 사용하거나 둘 모두를 함께 사용하여 적절한 스토리지 풀 세트가 선택되었는지 확인할 수 있습니다.

excludeStoragePools 매개 변수는 속성과 일치하는 나열된 풀 세트를 명시적으로 제외하는 데 사용됩니다.

QoS 정책을 에뮬레이트합니다

서비스 품질 정책을 에뮬레이트하기 위해 스토리지 클래스를 설계하려면 '미디어' 속성을 HDD 또는 'SSD'로 사용하여 스토리지 클래스를 생성합니다. 스토리지 클래스에 언급된 미디어 특성에 따라, Trident는 미디어 속성과 일치하도록 HDD 또는 SSD 애티리게이트를 제공하는 적절한 백엔드를 선택한 다음 볼륨 프로비저닝을 특정 애티리게이트로 전달합니다. 따라서 프리미엄 QoS 정책으로 분류될 수 있는 '미디어' 속성이 'SD'로 설정된 스토리지 클래스 Premium을 생성할 수 있습니다. 표준 QoS 정책으로 분류될 수 있는 미디어 속성을 'HDD'로 설정하는 또 다른 스토리지 클래스 표준을 생성할 수 있습니다. 또한 스토리지 클래스에서 ""IOPS"" 속성을 사용하여 QoS 정책으로 정의할 수 있는 Element 어플라이언스로 프로비저닝을 리디렉션할 수도 있습니다.

특정 기능을 기반으로 백엔드를 활용합니다

스토리지 클래스는 씬 및 일반 프로비저닝, 스냅샷, 클론 및 암호화와 같은 기능이 설정된 특정 백엔드에서 볼륨 프로비저닝을 수행하도록 설계되었습니다. 사용할 스토리지를 지정하려면 필요한 기능이 설정된 적절한 백엔드를 지정하는 스토리지 클래스를 생성합니다.

가상 풀

모든 Trident 백엔드에서 가상 풀을 사용할 수 있습니다. Trident에서 제공하는 모든 드라이버를 사용하여 백엔드에 대한 가상 풀을 정의할 수 있습니다.

가상 풀을 사용하면 관리자가 저장소 클래스를 통해 참조할 수 있는 백엔드에 대한 추상화 수준을 생성하여 백엔드에 볼륨을 보다 유연하고 효율적으로 배치할 수 있습니다. 동일한 서비스 클래스로 다른 백엔드를 정의할 수 있습니다. 또한 동일한 백엔드에서 여러 스토리지 풀을 생성할 수 있지만 특성이 다릅니다. 특정 레이블이 있는 선택기를 사용하여 스토리지 클래스를 구성하면 Trident에서 볼륨을 배치할 모든 선택기 레이블과 일치하는 백엔드를 선택합니다. 스토리지 클래스 선택기 레이블이 여러 스토리지 풀과 일치하는 경우 Trident는 볼륨 용량 할당을 위해 둘 중 하나를 선택합니다.

가상 풀 설계

백엔드를 생성할 때 일반적으로 매개변수 집합을 지정할 수 있습니다. 이전에는 관리자가 동일한 스토리지 자격 증명과 다른 매개변수 집합을 사용하여 다른 백엔드를 생성하는 것이 불가능했습니다. 가상 풀이 도입되면서 이 문제가 해결되었습니다. 가상 풀은 백엔드와 Kubernetes 스토리지 클래스 사이에 도입된 레벨 추상화로, 관리자가 백엔드에 관계없이 Kubernetes 스토리지 클래스를 통해 선택기로 참조할 수 있는 레이블과 함께 매개변수를 정의할 수 있도록 합니다. 가상 풀은 Trident를 통해 지원되는 모든 NetApp 백엔드에 대해 정의할 수 있습니다. 해당 목록에는 SolidFire/NetApp HCI, ONTAP, Azure NetApp Files 포함됩니다.



가상 풀을 정의할 때는 백엔드 정의에서 기존 가상 풀의 순서를 재정렬하지 않는 것이 좋습니다. 또한 기존 가상 풀의 속성을 편집/수정하고 대신 새 가상 풀을 정의하는 것이 좋습니다.

다양한 서비스 수준/QoS 에뮬레이션

서비스 클래스를 에뮬레이트하기 위한 가상 풀을 설계할 수 있습니다. Azure NetApp Files용 Cloud Volume Service에 대한 가상 풀 구현을 사용하여 다양한 서비스 클래스를 설정하는 방법을 살펴보겠습니다. 다양한 성능 수준을 나타내는 여러 레이블을 사용하여 Azure NetApp Files 백엔드를 구성합니다. 설정 `servicelevel` 적절한 성과 수준에 맞게 종횡비를 지정하고 각 레이블 아래에 다른 필요한 요소를 추가합니다. 이제 다른 가상 풀에 매핑할 다른 Kubernetes 스토리지 클래스를 생성합니다. 를 사용합니다 `parameters.selector` 필드에서 각 StorageClass는 볼륨을 호스팅하는 데 사용할 수 있는 가상 풀을 호출합니다.

특정 측면 지정

특정 측면의 여러 가상 풀을 단일 스토리지 백엔드에서 설계할 수 있습니다. 이를 위해 백엔드에 여러 레이블을 구성하고 각 레이블 아래에 필요한 측면을 설정합니다. 이제 를 사용하여 다양한 Kubernetes Storage 클래스를 생성할 수 있습니다 `parameters.selector` 다른 가상 풀에 매핑될 필드입니다. 백엔드에서 프로비저닝되는 볼륨에는 선택한 가상 풀에 정의된 측면이 있습니다.

스토리지 프로비저닝에 영향을 미치는 PVC 특성

요청된 스토리지 클래스를 벗어난 일부 매개변수는 PVC 생성 시 Trident 프로비저닝 결정 프로세스에 영향을 미칠 수 있습니다.

액세스 모드

PVC를 통한 저장 요청 시 필수 필드 중 하나가 액세스 모드입니다. 원하는 모드는 스토리지 요청을 호스팅하기 위해 선택한 백엔드에 영향을 줄 수 있습니다.

Trident는 다음 매트릭스에 따라 지정된 액세스 방법과 함께 사용되는 스토리지 프로토콜을 일치시키려고 시도합니다. 이는 기본 스토리지 플랫폼과 무관합니다.

	ReadWriteOnce 를 참조하십시오	ReadOnlyMany 를 참조하십시오	ReadWriteMany 를 참조하십시오
iSCSI	예	예	예(원시 블록)
NFS 를 참조하십시오	예	예	예

NFS 백엔드가 구성되지 않은 상태로 Trident 배포에 제출된 `ReadWriteMany` PVC에 대한 요청은 볼륨이 프로비저닝되지 않습니다. 이러한 이유로 요청자는 자신의 응용 프로그램에 적합한 액세스 모드를 사용해야 합니다.

볼륨 작업입니다

영구 볼륨 수정

영구 볼륨은 Kubernetes에서 두 가지 예외, 영구적 객체입니다. 생성된 후에는 부가세 반환 청구액 정책 및 크기를 수정할 수 있습니다. 그러나 이렇게 해도 볼륨의 일부 측면이 Kubernetes 외부에서 수정되는 것을 방지할 수 없습니다. 특정 애플리케이션에 맞게 볼륨을 사용자 지정하거나, 실수로 용량이 소비되지 않도록 하거나, 어떠한 이유로든 볼륨을 다른 스토리지 컨트롤러로 이동하는 것이 좋을 수 있습니다.



Kubernetes 트리 내 프로비저닝은 현재 NFS, iSCSI 또는 FC PVS에 대한 볼륨 크기 조정 작업을 지원하지 않습니다. Trident은 NFS, iSCSI 및 FC 볼륨의 확장을 지원합니다.

PV의 접속 세부 정보는 생성 후 수정할 수 없습니다.

주문형 볼륨 스냅샷을 생성합니다

Trident는 CSI 프레임워크를 사용하여 주문형 볼륨 스냅샷 생성 및 스냅샷으로부터 PVC 생성을 지원합니다. 스냅샷은 편리한 데이터 시점 복사본을 유지 관리하는 방법을 제공하며 Kubernetes의 소스 PV와 독립적인 라이프사이클을 갖고 있습니다. 이러한 스냅샷을 사용하여 PVC를 복제할 수 있습니다.

스냅샷으로부터 볼륨을 생성합니다

Trident는 또한 볼륨 스냅샷에서 PersistentVolumes 생성을 지원합니다. 이를 위해 PersistentVolumeClaim을 생성하고 볼륨을 생성해야 하는 필수 스냅샷으로 을 언급하면 datasource 됩니다. Trident는 스냅샷에 데이터가 있는 볼륨을 생성하여 이 PVC를 처리합니다. 이 기능을 사용하면 지역 간에 데이터를 복제하거나 테스트 환경을 생성하거나 손상되거나 손상된 운영 볼륨을 전체적으로 교체하거나 특정 파일 및 디렉토리를 검색하여 연결된 다른 볼륨으로 전송할 수 있습니다.

클러스터에서 볼륨 이동

스토리지 관리자는 ONTAP 클러스터의 Aggregate와 컨트롤러 간에 볼륨을 스토리지 소비자로 중단 없이 이동할 수 있습니다. 대상 애그리게이트가 Trident가 사용 중인 SVM에 액세스할 수 있는 경우에만 이 작업이 Trident 또는 Kubernetes 클러스터에는 영향을 미치지 않습니다. 중요한 것은, aggregate가 SVM에 새로 추가된 경우 이를 Trident에 다시 추가하여 백엔드를 새로 고쳐야 한다는 것입니다. 이 경우 Trident가 SVM의 인벤토리를 다시 작성하도록 트리거하여 새 애그리게이트가 인식됩니다.

하지만 Trident에서는 백엔드 볼륨 이동이 자동으로 지원되지 않습니다. 여기에는 동일한 클러스터 내, 클러스터 간 또는 다른 스토리지 플랫폼(스토리지 시스템이 Trident에 연결된 SVM인 경우에도)에 있는 SVM이 포함됩니다.

볼륨이 다른 위치에 복사되면 볼륨 가져오기 기능을 사용하여 현재 볼륨을 Trident로 가져올 수 있습니다.

볼륨 확장

Trident NFS, iSCSI, FC PV의 크기 조정을 지원합니다. 이를 통해 사용자는 Kubernetes 계층을 통해 볼륨 크기를 직접 조정할 수 있습니다. ONTAP, SolidFire/ NetApp HCI 백엔드를 포함한 모든 주요 NetApp 스토리지 플랫폼에서 볼륨 확장이 가능합니다. 나중에 확장이 가능하도록 설정하세요. allowVolumeExpansion 에게 true 볼륨과 연결된 StorageClass에서. 영구 볼륨의 크기를 조정해야 할 때마다 다음을 편집합니다.
spec.resources.requests.storage 영구 볼륨 클레임에 필요한 볼륨 크기에 대한 주석을 추가합니다. Trident 스토리지 클러스터의 볼륨 크기를 자동으로 조정합니다.

기존 볼륨을 **Kubernetes**로 임포트

볼륨 가져오는 기존 스토리지 볼륨을 Kubernetes 환경으로 가져오는 기능을 제공합니다. 이는 현재 다음에서 지원됩니다. `ontap-nas`, `ontap-nas-flexgroup`, `solidfire-san`, 그리고 `azure-netapp-files` 운전자. 이 기능은 기존 애플리케이션을 Kubernetes로 이식할 때나 재해 복구 시나리오에서 유용합니다.

ONTAP 및 드라이버를 사용할 `solidfire-san` 경우 명령을 `tridentctl import volume <backend-name> <volume-name> -f /path/pvc.yaml` 사용하여 기존 볼륨을 Kubernetes로 임포트하여 Trident에서 관리할 수 있습니다. 가져오기 볼륨 명령에 사용된 PVC YAML 또는 JSON 파일은 Trident를 프로비저닝자로 식별하는 저장소 클래스를 가리킵니다. NetApp HCI/SolidFire 백엔드를 사용할 경우 볼륨 이름이 고유한지 확인합니다. 볼륨 이름이 중복되면 볼륨을 고유한 이름으로 복제하여 볼륨 가져오기 기능에서 볼륨 이름을 구분할 수 있도록 합니다.

만약 `azure-netapp-files` 드라이버가 사용되면 명령을 사용하세요 `tridentctl import volume <backend-name> <volume path> -f /path/pvc.yaml` 볼륨을 Kubernetes로 가져와서 Trident에서 관리하도록 합니다. 이를 통해 고유한 볼륨 참조가 보장됩니다.

위의 명령을 실행하면 Trident는 백엔드에서 볼륨을 찾고 크기를 읽습니다. 구성된 PVC 볼륨 크기를 자동으로 추가

(필요한 경우 덮어쓰기)합니다. 그런 다음 Trident는 새 PV를 만들고 Kubernetes는 PV에 PVC를 바인딩합니다.

특정 가져온 PVC가 필요한 컨테이너를 배포한 경우 PVC/PV 쌍이 볼륨 가져오기 프로세스를 통해 바인딩될 때까지 보류 상태로 유지됩니다. PVC/PV 쌍이 바인딩되면 다른 문제가 없는 한 컨테이너가 나타나야 합니다.

레지스트리 서비스

레지스트리의 스토리지 배포 및 관리는 예 설명되어 있습니다 "[NetApp.IO를 참조하십시오](#)" 에 있습니다 "[블로그](#)".

로깅 서비스

다른 OpenShift 서비스와 마찬가지로 로깅 서비스는 Ansible을 사용하여 인벤토리 파일에서 제공하는 구성 매개 변수로 배포됩니다 호스트가 플레이북에 제공됩니다. OpenShift를 설치한 후 초기 OpenShift 설치 중에 로깅을 배포하고 로깅을 배포하는 두 가지 설치 방법이 제공됩니다.



Red Hat OpenShift 버전 3.9를 기준으로 공식 문서는 데이터 손상 관련 우려 때문에 로깅 서비스에 NFS를 사용할 것을 권장합니다. 이는 제품에 대한 Red Hat 테스트를 기반으로 합니다. ONTAP NFS 서버에는 이러한 문제가 없으며 로깅 구축을 쉽게 되돌릴 수 있습니다. 궁극적으로, 로깅 서비스를 위한 프로토콜을 선택할 수 있습니다. 두 가지 모두 NetApp 플랫폼을 사용할 때 효과가 있으며 원활 경우 NFS를 피할 이유가 없습니다.

로깅 서비스에서 NFS를 사용하도록 선택한 경우 설치 관리자의 실패를 방지하려면 Ansible 변수 "openshift_enable_unsupported_configurations"를 "true"로 설정해야 합니다.

시작하십시오

로깅 서비스는 필요에 따라 두 애플리케이션 및 OpenShift 클러스터 자체의 핵심 운영에 구축할 수 있습니다. 작업 로깅을 배포하려는 경우 변수 "openshift_logging_use_ops"를 "true"로 지정하면 서비스의 인스턴스 두 개가 만들어집니다. 작업에 대한 로깅 인스턴스를 제어하는 변수에는 "ops"가 포함되어 있지만 응용 프로그램의 인스턴스는 그렇지 않습니다.

기본 서비스에서 올바른 스토리지를 활용하기 위해서는 배포 방법에 따라 Ansible 변수를 구성하는 것이 중요합니다. 각 배포 방법에 대한 옵션을 살펴보겠습니다.



아래 표에는 로깅 서비스와 관련된 스토리지 구성과 관련된 변수만 나와 있습니다. 배포에 따라 검토, 구성 및 사용해야 하는 다른 옵션을 찾을 수 "[Red Hat OpenShift 로깅 문서](#)"입니다.

아래 표의 변수는 제공된 세부 정보를 사용하여 로깅 서비스에 대한 PV 및 PVC를 생성하는 Ansible 플레이북을 만듭니다. 이 방법은 OpenShift 설치 후 구성 요소 설치 플레이북을 사용하는 것보다 훨씬 덜 유연하지만, 기존 볼륨을 사용할 수 있는 경우 옵션으로 제공됩니다.

변수	세부 정보
"openshift_logging_storage_kind"	설치 프로그램이 로깅 서비스에 대한 NFS PV를 생성하도록 'NFS'로 설정합니다.
"openshift_logging_storage_host"를 선택합니다	NFS 호스트의 호스트 이름 또는 IP 주소입니다. 가상 머신의 데이터 LIF로 설정해야 합니다.
'openshift_logging_storage_nfs_directory'	NFS 내보내기의 마운트 경로입니다. 예를 들어 볼륨이 '/openshift_logging'으로 가정되는 경우 이 변수에 해당 경로를 사용합니다.

변수	세부 정보
'openshift_logging_storage_volume_name'	생성할 PV의 이름(예: PV_ose_logs)입니다.
"openshift_logging_storage_volume_size"	NFS 내보내기의 크기(예: 100Gi)입니다.

OpenShift 클러스터가 이미 실행 중이고 Trident가 배포 및 구성된 경우 설치 관리자는 동적 프로비저닝을 사용하여 볼륨을 생성할 수 있습니다. 다음 변수를 구성해야 합니다.

변수	세부 정보
"openshift_logging_es_pvc_dynamic"	동적으로 프로비저닝된 볼륨을 사용하려면 true로 설정합니다.
'openshift_logging_es_pvc_storage_class_name'	PVC에 사용될 스토리지 클래스의 이름입니다.
"openshift_logging_es_pvc_size"를 선택합니다	PVC에서 요청된 체적의 크기입니다.
"openshift_logging_es_pvc_prefix"	로깅 서비스에서 사용하는 PVC의 접두사입니다.
"openshift_logging_es_ops_pvc_dynamic"	작업 로깅 인스턴스에 동적으로 프로비저닝된 볼륨을 사용하려면 "true"로 설정합니다.
'openshift_logging_es_ops_pvc_storage_class_name'	작업 로깅 인스턴스에 대한 스토리지 클래스의 이름입니다.
"openshift_logging_es_ops_pvc_size"를 선택합니다	작업 인스턴스에 대한 볼륨 요청의 크기입니다.
"openshift_logging_es_ops_pvc_prefix"	ops instance PVCs(ops 인스턴스 PVC)의 접두사입니다.

로깅 스택을 배포합니다

초기 OpenShift 설치 프로세스의 일부로 로깅을 배포하는 경우 표준 배포 프로세스만 따르면 됩니다. Ansible이 완료되는 즉시 서비스를 이용할 수 있도록 필요한 서비스와 OpenShift 개체를 구성 및 배포합니다.

하지만 초기 설치 후에 구축할 경우 구성 요소 플레이북을 Ansible에서 사용해야 합니다. 이 프로세스는 OpenShift의 다른 버전에 따라 약간 변경될 수 있으므로 해당 버전에 대해 읽고 ["Red Hat OpenShift Container Platform 3.11 설명서"](#)따르십시오.

메트릭 서비스

메트릭 서비스는 관리자에게 OpenShift 클러스터의 상태, 리소스 활용도 및 가용성에 대한 중요한 정보를 제공합니다. 또한 POD 자동 확장 기능도 필요하며, 많은 조직에서 비용 청구 및/또는 애플리케이션 표시를 위해 메트릭 서비스의 데이터를 사용합니다.

로깅 서비스 및 OpenShift와 마찬가지로 Ansible을 사용하여 메트릭 서비스를 배포합니다. 또한 로깅 서비스와 마찬가지로 클러스터 초기 설정 중에 또는 구성 요소 설치 방법을 사용하여 작동 후에 메트릭 서비스를 구축할 수 있습니다. 다음 표에는 메트릭 서비스에 대한 영구 스토리지를 구성할 때 중요한 변수가 나와 있습니다.



아래 표에는 메트릭 서비스와 관련된 스토리지 구성과 관련된 변수만 포함되어 있습니다. 문서에 나와 있는 다른 많은 옵션은 배포 내용에 따라 검토, 구성 및 사용해야 합니다.

변수	세부 정보
"openshift_metrics_storage_kind"	설치 프로그램이 로깅 서비스에 대한 NFS PV를 생성하도록 'NFS'로 설정합니다.

변수	세부 정보
'openshift_metrics_storage_host'	NFS 호스트의 호스트 이름 또는 IP 주소입니다. SVM의 데이터 LIF로 설정해야 합니다.
'openshift_metrics_storage_nfs_directory'	NFS 내보내기의 마운트 경로입니다. 예를 들어, 볼륨이 '/openshift_metrics'로 가정되는 경우 이 변수에 해당 경로를 사용합니다.
'openshift_metrics_storage_volume_name'	생성할 PV의 이름(예: PV_ose_metrics).
'openshift_metrics_storage_volume_size'	NFS 내보내기의 크기(예: 100Gi)입니다.

OpenShift 클러스터가 이미 실행 중이고 Trident가 배포 및 구성된 경우 설치 관리자는 동적 프로비저닝을 사용하여 볼륨을 생성할 수 있습니다. 다음 변수를 구성해야 합니다.

변수	세부 정보
'openshift_metrics_cassandra_pvc_prefix'	지표 PVC에 사용할 접두사입니다.
'openshift_metrics_cassandra_pvc_size'	요청할 볼륨의 크기입니다.
'openshift_metrics_cassandra_storage_type'	메트릭에 사용할 스토리지 유형으로, 적절한 스토리지 클래스로 PVC를 생성하려면 Ansible에서 이를 동적으로 설정해야 합니다.
'openshift_metrics_cassandra_pvc_storage_class_name'	사용할 스토리지 클래스의 이름입니다.

메트릭 서비스를 구축합니다

호스트/인벤토리 파일에 정의된 적절한 Ansible 변수를 사용하여 서비스를 구축하십시오. OpenShift 설치 시 배포하는 경우 PV가 자동으로 생성되고 사용됩니다. 구성 요소 플레이북을 사용하여 배포하려는 경우 OpenShift 설치 후 Ansible은 필요한 PVC를 생성하고 Trident에서 이를 위해 스토리지를 프로비저닝한 후 서비스를 배포합니다.

위의 변수와 배포 프로세스는 각 OpenShift 버전에 따라 변경될 수 있습니다. 사용자 환경에 맞게 구성할 수 있도록 버전을 검토하고 그에 따라 ["Red Hat의 OpenShift 배포 가이드"](#)수행하십시오.

데이터 보호 및 재해 복구

Trident 및 Trident를 사용하여 생성된 볼륨의 보호 및 복구 옵션에 대해 알아보십시오. 지속성 요구사항이 있는 각 애플리케이션에 대한 데이터 보호 및 복구 전략이 있어야 합니다.

Trident 복제 및 복구

재해 발생 시 Trident 복원을 위한 백업을 생성할 수 있습니다.

Trident 복제

Trident는 Kubernetes CRD를 사용하여 자체 상태를 저장 및 관리하고 Kubernetes 클러스터 etcd를 사용하여 메타데이터를 저장합니다.

단계

1. 를 사용하여 Kubernetes 클러스터를 백업합니다 "[Kubernetes: etcd 클러스터 백업](#)".

2. FlexVol volume에 백업 아티팩트를 배치합니다



NetApp은 FlexVol이 다른 SVM과 SnapMirror 관계를 맺고 있는 SVM을 보호하는 것이 좋습니다.

Trident 복구

Kubernetes CRD 및 Kubernetes 클러스터 etcd 스냅샷을 사용하여 Trident를 복구할 수 있습니다.

단계

- 대상 SVM에서 Kubernetes etcd 데이터 파일 및 인증서가 포함된 볼륨을 마스터 노드로 설정할 호스트에 마운트합니다.
- 에서 Kubernetes 클러스터와 관련된 모든 필수 인증서를 복사합니다 /etc/kubernetes/pki 및 에 있는 etcd 멤버 파일 /var/lib/etcd.
- 를 사용하여 etcd 백업에서 Kubernetes 클러스터를 복원합니다 "[Kubernetes: etcd 클러스터 복구](#)".
- 실행 kubectl get crd 모든 Trident 사용자 지정 리소스가 준비되었는지 확인하고 Trident 객체를 검색하여 모든 데이터를 사용할 수 있는지 확인합니다.

SVM 복제 및 복구

Trident은 복제 관계를 구성할 수 없지만 스토리지 관리자는 를 사용하여 SVM을 복제할 수 있습니다 "[ONTAP SnapMirror를 참조하십시오](#)".

재해가 발생할 경우 SnapMirror 대상 SVM을 활성화하여 데이터 제공을 시작할 수 있습니다. 시스템이 복원되면 운영 시스템으로 다시 전환할 수 있습니다.

이 작업에 대해

SnapMirror SVM 복제 기능을 사용할 때는 다음 사항을 고려하십시오.

- SVM-DR이 활성화된 각 SVM에 대해 별개의 백엔드를 생성해야 합니다.
- SVM-DR을 지원하는 백엔드에 복제를 프로비저닝하지 않아도 되는 볼륨을 가질 필요가 없도록 필요한 경우에만 복제된 백엔드를 선택하도록 스토리지 클래스를 구성합니다.
- 애플리케이션 관리자는 복제와 관련된 추가 비용 및 복잡성을 이해하고 이 프로세스를 시작하기 전에 복구 계획을 신중하게 고려해야 합니다.

SVM 복제

을 사용할 수 있습니다 "[ONTAP: SnapMirror SVM 복제](#)" SVM 복제 관계를 생성합니다.

SnapMirror를 사용하여 복제할 항목을 제어하는 옵션을 설정할 수 있습니다. 미리 만들 때 선택한 옵션을 알아야 [Trident를 사용한 SVM 복구](#)합니다.

- "[-identity -true 를 유지합니다](#)" 전체 SVM 구성을 복제합니다.
- "[-discard-configs 네트워크](#)" LIF 및 관련 네트워크 설정은 제외됩니다.
- "[-identity -false 를 유지합니다](#)" 볼륨 및 보안 구성만 복제합니다.

Trident를 사용한 SVM 복구

Trident는 SVM 장애를 자동으로 감지하지 못합니다. 재해가 발생할 경우 관리자는 새로운 SVM으로 Trident 페일오버를 수동으로 시작할 수 있습니다.

단계

1. SnapMirror의 예약된 전송 및 진행 중인 전송을 취소하고 복제 관계를 중지한 다음, 소스 SVM을 중지하고 SnapMirror 대상 SVM을 활성화합니다.
2. 를 지정한 경우 `-identity-preserve false` 또는 `-discard-config network` SVM 복제를 구성할 때 를 업데이트합니다 `managementLIF` 및 `dataLIF` Trident 백엔드 정의 파일
3. 확인합니다 `storagePrefix` Trident 백엔드 정의 파일에 있습니다. 이 매개 변수는 변경할 수 없습니다. 생략합니다 `storagePrefix` 백엔드 업데이트가 실패합니다.
4. 다음을 사용하여 새로운 대상 SVM 이름을 반영하도록 필수 백엔드를 업데이트합니다.

```
./tridentctl update backend <backend-name> -f <backend-json-file> -n  
<namespace>
```

5. 를 지정한 경우 `-identity-preserve false` 또는 `'discard-config network'` 모든 애플리케이션 포드를 바운스해야 합니다.



지정한 경우 `-identity-preserve true`, 대상 SVM이 활성화될 때 Trident에서 프로비저닝한 모든 볼륨이 데이터 제공을 시작합니다.

볼륨 복제 및 복구

Trident는 SnapMirror 복제 관계를 구성할 수 없지만, 스토리지 관리자는 를 사용하여 Trident에서 생성된 볼륨을 복제할 수 ["ONTAP SnapMirror 복제 및 복구"](#)입니다.

그런 다음 를 사용하여 복구된 볼륨을 Trident로 가져올 수 ["Tridentctl 볼륨 가져오기"](#)입니다.



에서 가져오기가 지원되지 않습니다 `ontap-nas-economy`, `ontap-san-economy`, 또는 `ontap-flexgroup-economy` 드라이버.

스냅샷 데이터 보호

다음을 사용하여 데이터를 보호하고 복원할 수 있습니다.

- PVS(영구 볼륨)의 Kubernetes 볼륨 스냅샷을 생성하는 외부 스냅샷 컨트롤러 및 CRD

["볼륨 스냅샷"](#)

- ONTAP 스냅샷 - 볼륨의 전체 내용을 복원하거나 개별 파일 또는 LUN을 복구합니다.

["ONTAP 스냅샷"](#)

Trident 사용하여 상태 저장 애플리케이션의 장애 조치 자동화

Trident의 강제 분리 기능을 사용하면 Kubernetes 클러스터에서 비정상적인 노드의 볼륨을 자동으로 분리하여 데이터 손상을 방지하고 애플리케이션 가용성을 보장할 수 있습니다. 이 기능은 노드가 응답하지 않거나 유지 관리를 위해 오프라인으로 전환되는 시나리오에서 특히 유용합니다.

강제 분리에 대한 세부 정보

강제 분리가 가능합니다. `ontap-san`, `ontap-san-economy`, `ontap-nas`, 그리고 `ontap-nas-economy` 오직 강제 분리를 활성화하기 전에 Kubernetes 클러스터에서 비정상적 노드 종료(NGNS)를 활성화해야 합니다. NGNS는 Kubernetes 1.28 이상에서 기본적으로 활성화되어 있습니다. 자세한 내용은 다음을 참조하세요. "[Kubernetes: 노드 정상 종료 아님](#)".



또는 `ontap-nas-economy` 드라이버를 사용할 경우 `ontap-nas`, Trident이 관리형 엑스포트 정책을 사용하여 적용된 태그로 인해 Kubernetes 노드에서 액세스를 제한할 수 있도록 백엔드 구성에서 매개 변수를 `true` 설정해야 `autoExportPolicy` 합니다.



Trident는 Kubernetes `nngs`를 사용하기 때문에 허용할 수 없는 모든 워크로드의 일정이 재조정될 때까지 비정상 노드에서 테인트를 제거하지 마십시오 `out-of-service`. 무모하게 타트를 적용하거나 제거하면 백엔드 데이터 보호가 위태롭게 될 수 있습니다.

Kubernetes 클러스터 관리자가 노드에 태그를 `enableForceDetach` 적용하고 `node.kubernetes.io/out-of-service=nodeshutdown:NoExecute`로 설정하면 `true` Trident이 노드 상태와 다음을 확인합니다.

1. 해당 노드에 마운트된 볼륨에 대한 백엔드 I/O 액세스를 중지합니다.
2. Trident 노드 개체를 `dirty` 표시합니다(새 발행물에 안전하지 않음).



Trident 컨트롤러는 Trident 노드 포드에 의해 노드가 다시 검증될 때까지(로 표시된 후) 새로운 게시 볼륨 요청을 거부합니다 `dirty`. Trident가 노드를 확인할 수 있을 때까지(새 발행물에 안전함) 마운트된 PVC로 예약된 모든 작업 부하(클러스터 노드가 정상 및 준비 상태임 이후에도)는 수락되지 `clean` 않습니다.

노드 상태가 복원되고 정점이 제거되면 Trident는 다음을 수행합니다.

1. 노드에서 오래된 게시된 경로를 식별하고 제거합니다.
2. 노드가 상태(서비스 중단 시간이 제거되고 노드가 상태)이고 모든 오래되고 `Ready` 게시된 경로가 정리된 경우 `cleanable`, Trident는 노드를 `reclaim`하고 게시된 새로운 볼륨을 노드에 허용합니다. `clean`

자동 장애 조치에 대한 세부 정보

통합을 통해 강제 분리 프로세스를 자동화할 수 있습니다. "[노드 상태 점검\(NHC\) 운영자](#)". 노드 장애가 발생하면 NHC는 Trident 노드 복구(TNR)를 트리거하고 Trident의 네임스페이스에 장애가 발생한 노드를 정의하는 `TridentNodeRemediation` CR을 생성하여 자동으로 강제 분리합니다. TNR은 노드 장애가 발생한 경우에만 생성되며, 노드가 다시 온라인 상태가 되거나 노드가 삭제되면 NHC에서 제거됩니다.

실패한 노드 포드 제거 프로세스

자동 장애 조치는 실패한 노드에서 제거할 작업 부하를 선택합니다. TNR이 생성되면 TNR 컨트롤러는 노드를 더티로 표시하여 새로운 볼륨 게시를 방지하고 강제 분리가 지원되는 포드와 해당 볼륨 첨부 파일을 제거하기 시작합니다.

강제 분리가 지원하는 모든 볼륨/PVC는 자동 장애 조치가 지원됩니다.

- NAS 및 자동 내보내기 정책을 사용하는 NAS 경제 볼륨(SMB는 아직 지원되지 않음).
- SAN 및 SAN 경제 볼륨.

[참조하다](#)[강제 분리에 대한 세부 정보](#).

기본 동작:

- 강제 분리가 지원하는 볼륨을 사용하는 Pod는 실패한 노드에서 제거됩니다. 쿠버네티스는 이를 정상적인 노드로 다시 예약합니다.
- Trident 볼륨이 아닌 볼륨을 포함하여 강제 분리가 지원되지 않는 볼륨을 사용하는 Pod는 실패한 노드에서 제거되지 않습니다.
- 상태 비저장 포드(PVC 아님)는 포드 주석이 없는 한 실패한 노드에서 제거되지 않습니다.
`trident.netapp.io/podRemediationPolicy: delete` 설정되었습니다.

포드 제거 동작 재정의:

Pod 제거 동작은 Pod 주석을 사용하여 사용자 정의할 수 있습니다.

`trident.netapp.io/podRemediationPolicy[retain, delete]`. 이러한 주석은 장애 조치가 발생할 때 검사되어 사용됩니다. 페일오버 후 주석이 사라지는 것을 방지하려면 Kubernetes 배포/복제 세트 포드 사양에 주석을 적용합니다.

- `retain`- 자동 장애 조치 중에는 실패한 노드에서 Pod가 제거되지 않습니다.
- `delete`- 자동 장애 조치 중에 실패한 노드에서 Pod가 제거됩니다.

이러한 주석은 모든 포드에 적용될 수 있습니다.

-  • 강제 분리를 지원하는 볼륨의 경우 실패한 노드에서만 I/O 작업이 차단됩니다.
- 강제 분리를 지원하지 않는 볼륨의 경우 데이터 손상 및 다중 연결 문제가 발생할 위험이 있습니다.

트라이던트노드복구 CR

TridentNodeRemediation(TNR) CR은 실패한 노드를 정의합니다. TNR의 이름은 실패한 노드의 이름입니다.

TNR 예시:

```
apiVersion: trident.netapp.io/v1
kind: TridentNodeRemediation
metadata:
  name: <K8s-node-name>
spec: {}
```

TNR 상태: 다음 명령을 사용하여 TNR 상태를 확인하세요.

```
kubectl get tnr <name> -n <trident-namespace>
```

TNR은 다음 상태 중 하나에 속할 수 있습니다.

- 수정:

- 해당 노드에 마운트된 강제 분리가 지원하는 볼륨에 대한 백엔드 I/O 액세스를 중단합니다.
- Trident 노드 객체가 더티(새로운 게시물에 안전하지 않음)로 표시되었습니다.
- 노드에서 포드 및 볼륨 첨부 파일 제거

- 노드 복구 보류:

- 컨트롤러는 노드가 다시 온라인 상태가 될 때까지 기다리고 있습니다.
- 노드가 온라인 상태가 되면 publish-enforcement를 통해 노드가 정리되고 새로운 볼륨 출판에 적합한 상태가 됩니다.

- 노드가 K8s에서 삭제되면 TNR 컨트롤러는 TNR을 제거하고 조정을 중단합니다.

- 성공:

- 모든 수정 및 노드 복구 단계가 성공적으로 완료되었습니다. 노드가 정리되어 새로운 권의 출판에 적합합니다.

- 실패한:

- 복구할 수 없는 오류입니다. 오류 이유는 CR의 status.message 필드에 설정됩니다.

자동 장애 조치 활성화

필수 조건:

- 자동 장애 조치를 활성화하기 전에 강제 분리가 활성화되어 있는지 확인하세요. 자세한 내용은 다음을 참조하세요.
[\[강제 분리에 대한 세부 정보\]](#).
- Kubernetes 클러스터에 노드 상태 점검(NHC)을 설치합니다.
 - ["operator-sdk 설치"](#).
 - 클러스터에 Operator Lifecycle Manager(OLM)가 아직 설치되지 않았다면 설치하세요. `operator-sdk olm install`.
 - 노드 상태 점검 운영자 설치: `kubectl create -f https://operatorhub.io/install/node-healthcheck-operator.yaml`.



또한 지정된 대로 노드 실패를 감지하기 위해 대체 방법을 사용할 수도 있습니다. [\[Integrating Custom Node Health Check Solutions\]](#) 아래 섹션을 참조하세요.

보다 "노드 상태 점검 운영자" 자세한 내용은.

단계

1. 클러스터의 워커 노드를 모니터링하기 위해 Trident 네임스페이스에 NodeHealthCheck(NHC) CR을 만듭니다. 예:

```

apiVersion: remediation.medik8s.io/v1alpha1
kind: NodeHealthCheck
metadata:
  name: <CR name>
spec:
  selector:
    matchExpressions:
      - key: node-role.kubernetes.io/control-plane
        operator: DoesNotExist
      - key: node-role.kubernetes.io/master
        operator: DoesNotExist
  remediationTemplate:
    apiVersion: trident.netapp.io/v1
    kind: TridentNodeRemediationTemplate
    namespace: <Trident installation namespace>
    name: trident-node-remediation-template
  minHealthy: 0 # Trigger force-detach upon one or more node failures
  unhealthyConditions:
    - type: Ready
      status: "False"
      duration: 0s
    - type: Ready
      status: Unknown
      duration: 0s

```

2. 노드 상태 점검 CR을 적용합니다. trident 네임스페이스.

```
kubectl apply -f <nhc-cr-file>.yaml -n <trident-namespace>
```

위 CR은 노드 조건 '준비: 거짓' 및 '알 수 없음'에 대해 K8s 워커 노드를 감시하도록 구성되어 있습니다. 자동 장애 조치는 노드가 준비: 거짓 또는 준비: 알 수 없음 상태로 전환되면 실행됩니다.

그만큼 unhealthyConditions CR에서는 0초의 유예 기간을 사용합니다. 이렇게 하면 K8s가 노드 조건 Ready: false를 설정하는 즉시 자동 장애 조치가 트리거됩니다. 이 조건은 K8s가 노드에서 하트비트를 잃은 후에 설정됩니다. K8s는 마지막 하트비트 이후 Ready: false로 설정되기 전에 기본적으로 40초간 기다립니다. 이 유예 기간은 K8s 배포 옵션에서 사용자 정의할 수 있습니다.

추가 구성 옵션은 다음을 참조하세요. ["Node-Healthcheck-Operator 문서"](#).

추가 설정 정보

Trident 강제 분리가 활성화된 상태로 설치하면 NHC와의 통합을 용이하게 하기 위해 Trident 네임스페이스에 두 개의 추가 리소스인 TridentNodeRemediationTemplate(TNRT)과 ClusterRole이 자동으로 생성됩니다.

TridentNodeRemediationTemplate(TNRT):

TNRT는 NHC 컨트롤러를 위한 템플릿 역할을 하며, NHC 컨트롤러는 필요에 따라 TNRT를 사용하여 TNR 리소스를

생성합니다.

```
apiVersion: trident.netapp.io/v1
kind: TridentNodeRemediationTemplate
metadata:
  name: trident-node-remediation-template
  namespace: trident
spec:
  template:
    spec: {}
```

클러스터 역할:

강제 분리가 활성화된 경우 설치 중에 클러스터 역할도 추가됩니다. 이를 통해 NHC는 Trident 네임스페이스의 TNR에 대한 권한을 얻습니다.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  labels:
    rbac.ext-remediation/aggregate-to-ext-remediation: "true"
  name: tridentnoderemediation-access
rules:
- apiGroups:
  - trident.netapp.io
  resources:
  - tridentnoderemediationtemplates
  - tridentnoderemediations
  verbs:
  - get
  - list
  - watch
  - create
  - update
  - patch
  - delete
```

K8s 클러스터 업그레이드 및 유지 관리

장애 조치를 방지하려면 노드가 다운되거나 재부팅될 것으로 예상되는 K8s 유지 관리 또는 업그레이드 중에 자동 장애 조치를 일시 중지합니다. 위에 설명된 NHC CR을 일시 중지하려면 CR을 패치하면 됩니다.

```
kubectl patch NodeHealthCheck <cr-name> --patch
'{"spec": {"pauseRequests": ["<description-for-reason-of-pause>"]}}' --type=merge
```

이렇게 하면 자동 장애 조치가 일시 중지됩니다. 자동 장애 조치를 다시 활성화하려면 유지 관리가 완료된 후 사양에서

`pauseRequests`를 제거하세요.

제한 사항

- 강제 분리가 지원되는 볼륨의 경우 실패한 노드에서만 I/O 작업이 방지됩니다. 강제 분리를 지원하는 볼륨/PVC를 사용하는 포드만 자동으로 제거됩니다.
- 자동 장애 조치 및 강제 분리 기능은 트라이던트 컨트롤러 포드 내부에서 실행됩니다. 트라이던트 컨트롤러를 호스팅하는 노드에 장애가 발생하면 K8s가 포드를 정상적인 노드로 옮길 때까지 자동 장애 조치가 지연됩니다.

사용자 정의 노드 상태 점검 솔루션 통합

Node Healthcheck Operator를 대체 노드 장애 감지 도구로 교체하여 자동 장애 조치를 트리거할 수 있습니다. 자동 장애 조치 메커니즘과의 호환성을 보장하려면 사용자 지정 솔루션이 다음을 수행해야 합니다.

- 노드 장애가 감지되면 장애가 발생한 노드의 이름을 TNR CR 이름으로 사용하여 TNR을 생성합니다.
- 노드가 복구되고 TNR이 성공 상태가 되면 TNR을 삭제합니다.

보안

보안

여기에서 나열된 권장 사항을 사용하여 Trident가 안전하게 설치되었는지 확인하십시오.

자체 네임스페이스에서 **Trident** 실행

안정적인 스토리지를 확보하고 잠재적 악의적인 활동을 차단하려면 애플리케이션, 애플리케이션 관리자, 사용자 및 관리 애플리케이션이 Trident 오브젝트 정의 또는 포드에 액세스하지 못하도록 방지하는 것이 중요합니다.

Trident에서 다른 애플리케이션과 사용자를 분리하려면 항상 자체 Kubernetes 네임스페이스에 Trident를 (`trident` 설치해야 합니다.) Trident를 자체 네임스페이스에 배치하면 Kubernetes 관리 담당자만 Trident Pod와 이름 속도의 CRD 개체에 저장된 아티팩트(예: 백엔드 및 CHAP 암호(해당하는 경우)에 액세스할 수 있습니다. 관리자만 Trident 네임스페이스에 액세스하여 응용 프로그램에 액세스할 수 있도록 해야 `tridentctl 합니다.`)

ONTAP SAN 백엔드에 **CHAP** 인증을 사용합니다

Trident는 및 `ontap-san-economy` 드라이버를 사용하여 ONTAP SAN 워크로드에 대한 CHAP 기반 인증을 `ontap-san` 지원합니다. NetApp에서는 Trident 호스트와 스토리지 백엔드 간의 인증에 양방향 CHAP를 사용할 것을 권장합니다.

SAN 스토리지 드라이버를 사용하는 ONTAP 백엔드의 경우 Trident에서 양방향 CHAP를 설정하고 이를 통해 CHAP 사용자 이름 및 암호를 관리할 수 `tridentctl` 있습니다. Trident가 ONTAP 백엔드에서 CHAP를 구성하는 방법에 대한 자세한 내용은 ["ONTAP SAN 드라이버를 사용하여 백엔드를 구성할 준비를 합니다"](#) 참조하십시오.

NetApp HCI 및 SolidFire 백엔드에서 **CHAP** 인증을 사용합니다

양방향 CHAP를 구축하여 호스트와 NetApp HCI 및 SolidFire 백엔드 간의 인증을 보장하는 것이 좋습니다. Trident는 테넌트당 두 개의 CHAP 암호를 포함하는 비밀 개체를 사용합니다. Trident가 설치되면 CHAP 암호를 관리하고 `tridentvolume` 해당 PV에 대한 CR 개체에 저장합니다. PV를 생성할 때 Trident는 CHAP 암호를 사용하여 iSCSI 세션을 시작하고 CHAP를 통해 NetApp HCI 및 SolidFire 시스템과 통신합니다.



Trident에 의해 생성된 볼륨은 볼륨 액세스 그룹과 연결되지 않습니다.

NVE 및 NAE에 Trident를 사용하십시오

NetApp ONTAP는 유휴 데이터 암호화를 제공하여 디스크를 도난, 반환 또는 용도 변경할 때 중요한 데이터를 보호합니다. 자세한 내용은 ["NetApp 볼륨 암호화 구성 개요"](#).

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

NAE 지원 백엔드의 Trident에서 생성된 볼륨은 NVE 또는 NAE로 암호화되어야 합니다.



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

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

백엔드 구성 옵션에 대한 자세한 내용은 다음을 참조하십시오.

- "[ONTAP SAN 구성 옵션](#)"
- "[ONTAP NAS 구성 옵션](#)"

Linux 통합 키 설정(LUKS)

Linux LUKS(통합 키 설정)를 활성화하여 Trident에서 ONTAP SAN 및 ONTAP SAN 경제 볼륨을 암호화할 수 있습니다. Trident는 LUKS로 암호화된 볼륨에 대해 암호 순환 및 볼륨 확장을 지원합니다.

Trident에서 LUKS로 암호화된 볼륨은 에서 권장하는 AES-XTS-plain64 사이퍼 및 모드를 "[NIST](#)" 사용합니다.



ASA r2 시스템에서는 LUKS 암호화가 지원되지 않습니다. ASA r2 시스템에 대한 자세한 내용은 다음을 참조하세요. "[ASA R2 스토리지 시스템에 대해 알아보십시오](#)".

시작하기 전에

- 작업자 노드에는 Cryptsetup 2.1 이상(3.0 이하)이 설치되어 있어야 합니다. 자세한 내용은 [를 참조하십시오](#) "[Gitlab: cryptsetup](#)".
- 성능상의 이유로 NetApp에서는 작업자 노드가 AES-NI(Advanced Encryption Standard New Instructions)를 지원하는 것이 좋습니다. AES-NI 지원을 확인하려면 다음 명령을 실행합니다.

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

아무 것도 반환되지 않으면 프로세서는 AES-NI를 지원하지 않습니다. AES-NI에 대한 자세한 내용은 다음 웹 사이트를 참조하십시오. "인텔: AES-NI(Advanced Encryption Standard Instructions)".

LUKS 암호화를 사용합니다

ONTAP SAN 및 ONTAP SAN 이코노미 볼륨에 대해 Linux 통합 키 설정(LUKS)을 사용하여 볼륨별 호스트 측 암호화를 활성화할 수 있습니다.

단계

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

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

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

```

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

```

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

```

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

```

제한 사항

LUKS - 암호화된 볼륨은 ONTAP 중복 제거 및 압축을 활용할 수 없습니다.

LUKS 볼륨을 가져오기 위한 백엔드 구성입니다

LUKS 볼륨을 가져오려면 백엔드에서 로(true 설정해야 luksEncryption 합니다. 0| luksEncryption 옵션은 볼륨이 LUKS 호환(true) 또는 LUKS 규격이 아닌 경우 Trident에 지시합니다.(false

```

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

```

LUKS 볼륨을 가져오기 위한 PVC 구성입니다

LUKS 볼륨을 동적으로 가져오려면 `trident.netapp.io/luksEncryption true` 이 예제에 표시된 대로 주석을
로 설정하고 PVC에 LUKS 사용 스토리지 클래스를 포함합니다.

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

LUKS 암호를 회전합니다

LUKS 암호를 회전하고 회전을 확인할 수 있습니다.

 볼륨, 스냅샷 또는 비밀이 더 이상 참조하지 않음을 확인할 때까지 암호문을 잊지 마십시오. 참조된
암호가 손실된 경우 볼륨을 마운트할 수 없으며 데이터가 암호화된 상태로 유지되고 액세스할 수 없게
됩니다.

이 작업에 대해

LUKS 암호 회전은 새 LUKS 암호를 지정한 후 볼륨을 마운트하는 POD가 생성될 때 발생합니다. 새 Pod가 생성되면 Trident는 볼륨의 LUKS 암호를 활성 암호와 비교합니다.

- 볼륨의 암호가 비밀의 활성 암호와 일치하지 않으면 회전이 발생합니다.
- 볼륨의 암호가 비밀의 활성 암호와 일치하면 가 됩니다 `previous-luks-passphrase` 매개 변수는 무시됩니다.

단계

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

```

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

```

2. 볼륨 또는 스냅숏에서 기존 암호를 식별합니다.

볼륨

```

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

...luksPassphraseNames: ["A"]

```

스냅샷

```

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

...luksPassphraseNames: ["A"]

```

3. 볼륨에 대한 LUKS 암호를 업데이트하여 새 암호 및 이전 암호 문자를 지정합니다. 확인합니다 previous-luke-passphrase-name 및 previous-luks-passphrase 이전 패스프레이즈를 일치시킵니다.

```

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

```

4. 볼륨을 마운트하는 새 포드를 생성합니다. 이 작업은 회전을 시작하는 데 필요합니다.

5. 패스프레이즈가 회전되었는지 확인합니다.

볼륨

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

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

스냅샷

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

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

결과

볼륨과 스냅샷에 새 암호문만 반환되면 암호가 회전되었습니다.



예를 들어, 두 개의 암호 구문이 반환되는 경우 luksPassphraseNames: ["B", "A"], 회전이 완료되지 않았습니다. 새 포드를 트리거하여 회전을 완료할 수 있습니다.

볼륨 확장을 설정합니다

LUKS 암호화 볼륨에서 볼륨 확장을 활성화할 수 있습니다.

단계

1. 를 활성화합니다 `CSINodeExpandSecret` 기능 게이트(베타 1.25+). 을 참조하십시오 "[Kubernetes 1.25: CSI 볼륨의 노드 기반 확장에 비밀을 사용합니다](#)" 를 참조하십시오.
2. 를 추가합니다 `node-expand-secret-name` 및 `node-expand-secret-namespace` `StorageClass` 매개 변수입니다. 예를 들면 다음과 같습니다.

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

결과

온라인 저장소 확장을 시작할 때 kublet은 적절한 자격 증명을 드라이버에 전달합니다.

Kerberos 전송 중 암호화

Kerberos 전송 중 암호화를 사용하면 관리 클러스터와 스토리지 백엔드 사이의 트래픽에 암호화를 활성화하여 데이터 액세스 보안을 강화할 수 있습니다.

Trident는 ONTAP를 스토리지 백엔드로 위해 Kerberos 암호화를 지원합니다.

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

따라서 스냅샷을 생성하고, 삭제하고, 크기 조정하고, 스냅샷, 클론 읽기 전용 클론 생성 및 NFS 암호화를 사용하는 볼륨 가져오기

사내 ONTAP 볼륨을 사용하여 전송 중인 **Kerberos** 암호화를 구성합니다

관리 클러스터와 온프레미스 ONTAP 스토리지 백엔드 사이의 스토리지 트래픽에 Kerberos 암호화를 활성화할 수 있습니다.



온-프레미스 ONTAP 스토리지 백엔드를 사용하는 NFS 트래픽에 대한 Kerberos 암호화는 스토리지 드라이버를 사용하는 경우에만 ontap-nas 지원됩니다.

시작하기 전에

- 유ти리티에 액세스할 수 있는지 tridentctl 확인합니다.
- ONTAP 스토리지 백엔드에 대한 관리자 액세스 권한이 있어야 합니다.
- ONTAP 스토리지 백엔드에서 공유할 볼륨의 이름을 알고 있어야 합니다.
- NFS 볼륨에 대한 Kerberos 암호화를 지원하는 ONTAP 스토리지 VM을 준비했는지 확인합니다. 자세한 내용은 ["데이터 LIF에서 Kerberos 활성화"](#) 참조하십시오.
- Kerberos 암호화로 사용하는 NFSv4 볼륨이 올바르게 구성되어 있는지 확인합니다. 의 NetApp NFSv4 도메인 구성 섹션(13페이지)을 ["NetApp NFSv4의 향상된 기능 및 모범 사례 가이드"](#) 참조하십시오.

ONTAP 엑스포트 정책을 추가 또는 수정합니다

기존 ONTAP 엑스포트 정책에 규칙을 추가하거나, ONTAP 스토리지 VM 루트 볼륨뿐 아니라 업스트림 Kubernetes 클러스터와 공유된 ONTAP 볼륨에 대해 Kerberos 암호화를 지원하는 새 엑스포트 정책을 생성해야 합니다. 추가한 내보내기 정책 규칙 또는 새로 만든 내보내기 정책은 다음 액세스 프로토콜 및 액세스 권한을 지원해야 합니다.

액세스 프로토콜

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

액세스 세부 정보

볼륨에 대한 필요에 따라 세 가지 Kerberos 암호화 버전 중 하나를 구성할 수 있습니다.

- * Kerberos 5 * - (인증 및 암호화)
- * Kerberos 5i * - (인증 및 ID 보호 암호화)
- * Kerberos 5p * - (인증 및 암호화, ID 및 개인 정보 보호)

적절한 액세스 권한을 사용하여 ONTAP 엑스포트 정책 규칙을 구성합니다. 예를 들어, 클러스터가 Kerberos 5i 및 Kerberos 5p 암호화가 혼합된 NFS 볼륨을 마운트하는 경우 다음 액세스 설정을 사용합니다.

유형	읽기 전용 액세스	읽기/쓰기 권한	고급 사용자 액세스
Unix	활성화됨	활성화됨	활성화됨
Kerberos 5i	활성화됨	활성화됨	활성화됨
Kerberos 5p	활성화됨	활성화됨	활성화됨

ONTAP 엑스포트 정책과 엑스포트 정책 규칙을 생성하는 방법은 다음 문서를 참조하십시오.

- ["엑스포트 정책을 생성합니다"](#)
- ["엑스포트 정책에 규칙 추가"](#)

스토리지 백엔드를 생성합니다

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

이 작업에 대해

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

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

Kerberos 수준을 하나만 지정하십시오. 매개 변수 목록에서 Kerberos 암호화 수준을 두 개 이상 지정하면 첫 번째 옵션만 사용됩니다.

단계

1. 관리되는 클러스터에서 다음 예제를 사용하여 스토리지 백엔드 구성 파일을 생성합니다. 괄호 안의 값을 사용자 환경의 정보로 대체:

```

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

```

2. 이전 단계에서 생성한 구성 파일을 사용하여 백엔드를 생성합니다.

```
tridentctl create backend -f <backend-configuration-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.

스토리지 클래스를 생성합니다

스토리지 클래스를 만들어 Kerberos 암호화를 사용하여 볼륨을 프로비저닝할 수 있습니다.

이 작업에 대해

저장소 클래스 객체를 만들 때 매개 변수를 사용하여 Kerberos 암호화의 세 가지 버전 중 하나를 지정할 수 있는 `mountOptions` 있습니다.

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

Kerberos 수준을 하나만 지정하십시오. 매개 변수 목록에서 Kerberos 암호화 수준을 두 개 이상 지정하면 첫 번째 옵션만 사용됩니다. 스토리지 백엔드 구성에서 지정한 암호화 수준이 스토리지 클래스 객체에 지정한 레벨과 다른 경우 스토리지 클래스 객체가 우선합니다.

단계

1. 다음 예제를 사용하여 StorageClass Kubernetes 객체를 생성합니다.

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

2. 스토리지 클래스를 생성합니다.

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

3. 스토리지 클래스가 생성되었는지 확인합니다.

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

다음과 유사한 출력이 표시됩니다.

NAME	PROVISIONER	AGE
ontap-nas-sc	csi.trident.netapp.io	15h

볼륨 프로비저닝

스토리지 백엔드와 스토리지 클래스를 생성한 후 이제 볼륨을 프로비저닝할 수 있습니다. 자세한 내용은 ["볼륨을 프로비저닝합니다" 참조하십시오.](#)

Azure NetApp Files 볼륨과 함께 전송 중인 Kerberos 암호화를 구성합니다

관리 클러스터와 단일 Azure NetApp Files 스토리지 백엔드 또는 Azure NetApp Files 스토리지 백엔드의 가상 풀 사이의 스토리지 트래픽에 Kerberos 암호화를 활성화할 수 있습니다.

시작하기 전에

- 관리되는 Red Hat OpenShift 클러스터에서 Trident를 활성화했는지 확인합니다.
- 유틸리티에 액세스할 수 있는지 `tridentctl` 확인합니다.
- 요구 사항을 확인하고 의 지침에 따라 Kerberos 암호화용 Azure NetApp Files 스토리지 백엔드를 준비했는지 ["Azure NetApp Files 설명서"](#) 확인합니다.
- Kerberos 암호화로 사용하는 NFSv4 볼륨이 올바르게 구성되어 있는지 확인합니다. 의 NetApp NFSv4 도메인 구성 섹션(13페이지)을 ["NetApp NFSv4의 향상된 기능 및 모범 사례 가이드" 참조하십시오.](#)

스토리지 백엔드를 생성합니다

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

이 작업에 대해

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

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

가상 풀 레벨에서 구성을 정의하면 스토리지 클래스의 레이블을 사용하여 풀이 선택됩니다.

두 레벨에서 Kerberos 암호화의 세 가지 버전 중 하나를 지정할 수 있습니다.

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

단계

1. 관리되는 클러스터에서 스토리지 백엔드(스토리지 백엔드 레벨 또는 가상 풀 레벨)를 정의해야 하는 위치에 따라 다음 예제 중 하나를 사용하여 스토리지 백엔드 구성 파일을 생성합니다. 괄호 안의 값을 사용자 환경의 정보로 대체:

스토리지 백엔드 레벨의 예

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

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

가상 풀 레벨 예

```

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

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

```

2. 이전 단계에서 생성한 구성 파일을 사용하여 백엔드를 생성합니다.

```
tridentctl create backend -f <backend-configuration-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.

스토리지 클래스를 생성합니다

스토리지 클래스를 만들어 Kerberos 암호화를 사용하여 볼륨을 프로비저닝할 수 있습니다.

단계

1. 다음 예제를 사용하여 StorageClass Kubernetes 개체를 생성합니다.

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

2. 스토리지 클래스를 생성합니다.

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

3. 스토리지 클래스가 생성되었는지 확인합니다.

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

다음과 유사한 출력이 표시됩니다.

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

볼륨 프로비저닝

스토리지 백엔드와 스토리지 클래스를 생성한 후 이제 볼륨을 프로비저닝할 수 있습니다. 자세한 내용은 ["볼륨을 프로비저닝합니다"](#) 참조하십시오.

저작권 정보

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

NetApp이 저작권을 가진 자료에 있는 소프트웨어에는 아래의 라이센스와 고지사항이 적용됩니다.

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

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

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

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

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

상표 정보

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