OpenShift Container Platform 4.10

백업 및 복원

OpenShift Container Platform 클러스터 백업 및 복원

Last Updated: 2024-01-10

OpenShift Container Platform 4.10 백업 및 복원

OpenShift Container Platform 클러스터 백업 및 복원

법적 공지

초록

이 문서는 클러스터 데이터를 백업하고 다양한 재해 시나리오에서 복구하는 방법에 대해 설명합니다.

차례

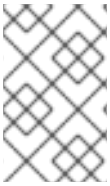
1장. 백업 및 복원

1.1. 컨트롤 플레인 백업 및 복원 작업

클러스터 관리자는 일정 기간 동안 OpenShift Container Platform 클러스터를 중지하고 나중에 다시 시작해야 할 수 있습니다. 클러스터를 다시 시작해야 하는 몇 가지 이유는 클러스터에서 유지 관리를 수행하거나 리소스 비용을 줄이기 때문입니다. OpenShift Container Platform에서는 나중에 클러스터를 쉽게 다시 시작할 수 있도록 클러스터를 정상적으로 종료 할 수 있습니다.

클러스터를 종료하기 전에 etcd 데이터를 백업 해야 합니다. etcd는 OpenShift Container Platform의 키-값 저장소이며 모든 리소스 오브젝트의 상태가 유지됩니다. etcd 백업은 재해 복구에서 중요한 역할을 합니다. OpenShift Container Platform에서 비정상적인 etcd 멤버를 교체할 수도 있습니다.

클러스터를 다시 실행하려면 클러스터를 정상적으로 다시 시작하십시오.



참고

클러스터의 인증서는 설치 날짜 이후 1년 후에 만료됩니다. 클러스터가 계속 유효한 상태에서 클러스터를 종료하고 정상적으로 다시 시작할 수 있습니다. 클러스터가 만료된 컨트롤 플레인 인증서를 자동으로 검색하지만 CSR(인증서 서명 요청)을 계속 승인해야 합니다.

OpenShift Container Platform이 예상대로 작동하지 않는 여러 상황에서 실행할 수 있습니다.

- 노드 장애 또는 네트워크 연결 문제와 같은 예기치 않은 상태로 인해 재시작 후 작동하지 않는 클러스터가 있습니다.
- 실수로 클러스터에서 중요한 작업을 삭제했습니다.
- 대부분의 컨트롤 플레인 호스트가 손실되어 etcd 쿼럼이 손실됩니다.

저장된 etcd 스냅샷을 사용하여 클러스터를 이전 상태로 복원 하면 재해 상황에서 항상 복구할 수 있습니다.

1.2. 애플리케이션 백업 및 복원 작업

클러스터 관리자는 OADP(OpenShift API for Data Protection)를 사용하여 OpenShift Container Platform에서 실행되는 애플리케이션을 백업하고 복원할 수 있습니다.

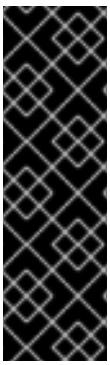
OADP는 Velero CLI 툴 다운로드 의 표에 따라 설치하는 OADP 버전에 적합한 Velero 버전을 사용하여 네임스페이스 단위로 Kubernetes 리소스 및 내부 이미지를 백업 및 복원합니다. OADP는 스냅샷 또는 Restic을 사용하여 PV(영구 볼륨)를 백업하고 복원합니다. 자세한 내용은 OADP 기능을 참조하십시오.

1.2.1. OADP 요구사항

OADP에는 다음과 같은 요구 사항이 있습니다.

- **cluster-admin** 역할의 사용자로 로그인해야 합니다.
- 다음 스토리지 유형 중 하나와 같이 백업을 저장하기 위한 오브젝트 스토리지가 있어야 합니다.
  - OpenShift Data Foundation
  - Amazon Web Services

- Microsoft Azure
- Google Cloud Platform
- S3 호환 오브젝트 스토리지



중요

S3 스토리지용 **CloudStorage** API는 기술 프리뷰 기능 전용입니다. 기술 프리뷰 기능은 Red Hat 프로덕션 서비스 수준 계약(SLA)에서 지원되지 않으며 기능적으로 완전하지 않을 수 있습니다. 따라서 프로덕션 환경에서 사용하는 것은 권장하지 않습니다. 이러한 기능을 사용하면 향후 제품 기능을 조기에 이용할 수 있어 개발 과정에서 고객이 기능을 테스트하고 피드백을 제공할 수 있습니다.

Red Hat 기술 프리뷰 기능의 지원 범위에 대한 자세한 내용은 기술 프리뷰 기능 지원 범위를 참조하십시오.

- 스냅샷으로 PV를 백업하려면 기본 스냅샷 API가 있거나 다음 공급자와 같은 CSI(Container Storage Interface) 스냅샷을 지원하는 클라우드 스토리지가 있어야 합니다.
  - Amazon Web Services
  - Microsoft Azure
  - Google Cloud Platform
  - CSI 스냅샷 지원 클라우드 스토리지(예: Ceph RBD 또는 Ceph FS)

참고

스냅샷을 사용하여 PV를 백업하지 않으려면 기본적으로 OADP Operator에 의해 설치된 Restic 를 사용할 수 있습니다.

1.2.2. 애플리케이션 백업 및 복원

Backup CR(사용자 정의 리소스)을 생성하여 애플리케이션을 백업합니다. 다음 백업 옵션을 구성할 수 있습니다.

- 백업 작업 전후에 명령을 실행하는 후크 백업
- 예약된 백업
- Restic 백업

Restore CR을 생성하여 애플리케이션을 복원합니다. 복원 작업 중에 init 컨테이너 또는 애플리케이션 컨테이너에서 명령을 실행하도록 복원 후크 를 구성할 수 있습니다.

2장. 클러스터를 안전하게 종료

이 문서에서는 클러스터를 안전하게 종료하는 프로세스를 설명합니다. 유지 관리를 위해 또는 리소스 비용을 절약하기 위해 일시적으로 클러스터를 종료해야 할 수 있습니다.

2.1. 전제 조건

- 클러스터를 종료하기 전에 etcd 백업을 수행하십시오.

2.2. 클러스터 종료

나중에 클러스터를 다시 시작하기 위해 안전한 방법으로 클러스터를 종료할 수 있습니다.

참고

설치 날짜부터 1년까지 클러스터를 종료하고 정상적으로 다시 시작할 수 있습니다. 설치 날짜로부터 1년 후에는 클러스터 인증서가 만료됩니다.

사전 요구 사항

- **cluster-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- etcd 백업이 수행되었습니다.

중요

클러스터를 다시 시작할 때 문제가 발생할 경우 클러스터를 복원 할 수 있도록 이 단계를 수행하기 전에 etcd 백업을 해 두는 것이 중요합니다.

예를 들어 다음 조건으로 인해 재시작된 클러스터가 손상될 수 있습니다.

- 종료 중 etcd 데이터 손상
- 하드웨어로 인한 노드 오류
- 네트워크 연결 문제

클러스터를 복구할 수 없는 경우 단계를 수행하여 이전 클러스터 상태로 복원합니다.

절차

1. 연장된 기간 동안 클러스터를 종료하는 경우 인증서가 만료되는 날짜를 확인합니다.

```
$ oc -n openshift-kube-apiserver-operator get secret kube-apiserver-to-kubelet-signer -o jsonpath='{.metadata.annotations.auth\.openshift\.io/certificate-not-after}'
```

출력 예

```
2022-08-05T14:37:50Zuser@user:~ $ 1
```

1 클러스터를 정상적으로 다시 시작할 수 있도록 지정된 날짜 또는 그 이전에 클러스터를 다시 시작하도록 계획합니다. 클러스터가 재시작되면 프로세스에서 kubelet 인증서를 복구하기

2. 클러스터의 모든 노드를 종료합니다. 클라우드 공급자의 웹 콘솔에서 이 작업을 수행하거나 다음 반복문을 실행할 수 있습니다.

```
$ for node in $(oc get nodes -o jsonpath='{.items[*].metadata.name}'); do oc debug
node/${node} -- chroot /host shutdown -h 1; done 1
```

1 **-h 1** 은 컨트롤 플레인 노드가 종료되기 전에 이 프로세스가 얼마나 오래 지속되는지를 나타냅니다. 노드가 10개 이상인 대규모 클러스터의 경우 모든 컴퓨팅 노드를 먼저 종료할 수 있도록 10분 이상으로 설정합니다.

출력 예

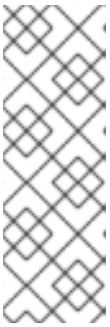
```
Starting pod/ip-10-0-130-169us-east-2computeinternal-debug ...
To use host binaries, run `chroot /host`
Shutdown scheduled for Mon 2021-09-13 09:36:17 UTC, use 'shutdown -c' to cancel.

Removing debug pod ...
Starting pod/ip-10-0-150-116us-east-2computeinternal-debug ...
To use host binaries, run `chroot /host`
Shutdown scheduled for Mon 2021-09-13 09:36:29 UTC, use 'shutdown -c' to cancel.
```

이러한 방법 중 하나를 사용하여 노드를 종료하면 pod가 정상적으로 종료되어 데이터 손상 가능성을 줄일 수 있습니다.

참고

대규모 클러스터에 대해 종료 시간을 더 길게 조정합니다.

```
$ for node in $(oc get nodes -o jsonpath='{.items[*].metadata.name}'); do oc
debug node/${node} -- chroot /host shutdown -h 10; done
```

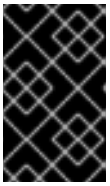


참고

종료하기 전에 OpenShift Container Platform과 함께 제공되는 표준 Pod의 컨트롤 플레인 노드를 드레인할 필요가 없습니다.

클러스터 관리자는 클러스터를 다시 시작한 후 워크로드를 완전히 다시 시작해야 합니다. 사용자 지정 워크로드로 인해 종료하기 전에 컨트롤 플레인 노드를 드레이한 경우 다시 시작한 후 클러스터가 다시 작동하기 전에 컨트롤 플레인 노드를 스케줄 대상으로 표시해야합니다.

3. 외부 스토리지 또는 LDAP 서버와 같이 더 이상 필요하지 않은 클러스터 종속성을 중지합니다. 이 작업을 수행하기 전에 공급 업체의 설명서를 확인하십시오.



중요

클라우드 공급자 플랫폼에 클러스터를 배포한 경우, 연결된 클라우드 리소스를 종료, 일시 중단 또는 삭제하지 마십시오. 중단된 가상 머신의 클라우드 리소스를 삭제하면 OpenShift Container Platform이 성공적으로 복원되지 않을 수 있습니다.

2.3. 추가 리소스

- 클러스터를 정상적으로 다시 시작
- 이전 클러스터 상태로 복원

3장. 클러스터를 정상적으로 다시 시작

이 문서에서는 정상 종료 후 클러스터를 다시 시작하는 프로세스에 대해 설명합니다.

다시 시작한 후 클러스터가 정상적으로 작동할 것으로 예상되지만 예상치 못한 상황으로 인해 클러스터가 복구되지 않을 수 있습니다. 예를 들면 다음과 같습니다.

- 종료 중 etcd 데이터 손상
- 하드웨어로 인한 노드 오류
- 네트워크 연결 문제

클러스터를 복구하지 못하면 단계에 따라 이전 클러스터 상태로 복원합니다.

3.1. 전제 조건

- 클러스터가 정상적으로 종료되었습니다.

3.2. 클러스터를 다시 시작

클러스터가 정상적으로 종료된 후 클러스터를 다시 시작할 수 있습니다.

전제 조건

- **cluster-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- 이 프로세스에서는 클러스터를 정상적으로 종료하고 있는 것을 전제로 하고 있습니다.

프로세스

1. 외부 스토리지 또는 LDAP 서버와 같은 클러스터의 종속 장치를 시작합니다.
2. 모든 클러스터 시스템을 시작합니다.
   클라우드 제공 업체의 웹 콘솔에서 시스템을 시작하는 것과 같이 클라우드 환경에 적합한 방법을 사용하여 시스템을 시작합니다.

   약 10 분 정도 기다린 후 컨트롤 플레인 노드의 상태를 확인합니다.
3. 모든 컨트롤 플레인 노드가 준비되었는지 확인합니다.

   ```
   $ oc get nodes -l node-role.kubernetes.io/master
   ```

   다음 출력에 표시된 대로 노드의 상태가 **Ready**인 경우 컨트롤 플레인 노드는 준비된 것입니다.

   ```
   NAME                          STATUS   ROLES    AGE   VERSION
   ip-10-0-168-251.ec2.internal   Ready    master   75m   v1.23.0
   ip-10-0-170-223.ec2.internal   Ready    master   75m   v1.23.0
   ip-10-0-211-16.ec2.internal    Ready    master   75m   v1.23.0
   ```
4. 컨트롤 플레인 노드가 준비 *되지 않은* 경우 승인해야하는 보류중인 인증서 서명 요청(CSR)이 있는지 확인합니다.

   a. 현재 CSR의 목록을 가져옵니다.

```
$ oc get csr
```

b. CSR의 세부 사항을 검토하여 CSR이 유효한지 확인합니다.

```
$ oc describe csr <csr_name> 1
```

1 **<csr_name>**은 현재 CSR 목록에 있는 CSR의 이름입니다.

c. 각각의 유효한 CSR을 승인합니다.

```
$ oc adm certificate approve <csr_name>
```

5. 컨트롤 플레인 노드가 준비되면 모든 작업자 노드가 준비되었는지 확인합니다.

```
$ oc get nodes -l node-role.kubernetes.io/worker
```

다음 출력에 표시된 대로 작업자 노드의 상태가 **Ready**인 경우 작업자 노드는 준비된 것입니다.

```
NAME                         STATUS   ROLES    AGE   VERSION
ip-10-0-179-95.ec2.internal  Ready    worker   64m   v1.23.0
ip-10-0-182-134.ec2.internal Ready    worker   64m   v1.23.0
ip-10-0-250-100.ec2.internal Ready    worker   64m   v1.23.0
```

6. 작업자 노드가 준비*되지 않은* 경우 승인해야하는 보류중인 인증서 서명 요청(CSR)이 있는지 확인합니다.

a. 현재 CSR의 목록을 가져옵니다.

```
$ oc get csr
```

b. CSR의 세부 사항을 검토하여 CSR이 유효한지 확인합니다.

```
$ oc describe csr <csr_name> 1
```

1 **<csr_name>**은 현재 CSR 목록에 있는 CSR의 이름입니다.

c. 각각의 유효한 CSR을 승인합니다.

```
$ oc adm certificate approve <csr_name>
```

7. 클러스터가 제대로 시작되었는지 확인합니다.

a. 성능이 저하된 클러스터 Operator가 없는지 확인합니다.

```
$ oc get clusteroperators
```

DEGRADED 조건이 **True**로 설정된 클러스터 Operator가 없는지 확인합니다.

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.10.0    True        False         False     59m
```

```
cloud-credential                           4.10.0    True        False         False      85m
cluster-autoscaler                         4.10.0    True        False         False      73m
config-operator                            4.10.0    True        False         False      73m
console                                    4.10.0    True        False         False      62m
csi-snapshot-controller                    4.10.0    True        False         False      66m
dns                                        4.10.0    True        False         False      76m
etcd                                       4.10.0    True        False         False      76m
...
```

b. 모든 노드가 **Ready** 상태에 있는지 확인합니다.

```
$ oc get nodes
```

모든 노드의 상태가 **Ready** 상태인지 확인합니다.

```
NAME                           STATUS   ROLES    AGE   VERSION
ip-10-0-168-251.ec2.internal   Ready    master   82m   v1.23.0
ip-10-0-170-223.ec2.internal   Ready    master   82m   v1.23.0
ip-10-0-179-95.ec2.internal    Ready    worker   70m   v1.23.0
ip-10-0-182-134.ec2.internal   Ready    worker   70m   v1.23.0
ip-10-0-211-16.ec2.internal    Ready    master   82m   v1.23.0
ip-10-0-250-100.ec2.internal   Ready    worker   69m   v1.23.0
```

클러스터가 제대로 시작되지 않은 경우 etcd 백업을 사용하여 클러스터를 복원해야 할 수 있습니다.

추가 리소스

- 클러스터를 다시 시작한 후 복구하지 못한 경우 etcd 백업을 사용하여 복원하는 방법은 이전 클러스터 상태로 복구할 수 있습니다.

4장. 애플리케이션 백업 및 복원

4.1. OADP 릴리스 정보

OADP(OpenShift API for Data Protection) 릴리스 노트에서는 새로운 기능 및 향상된 기능, 더 이상 사용되지 않는 기능, 제품 권장 사항, 알려진 문제 및 해결된 문제를 설명합니다.

4.1.1. OADP 1.1.2 릴리스 노트

OADP 1.1.2 릴리스 노트에는 제품 권장 사항, 수정된 버그 목록 및 알려진 문제에 대한 설명이 포함되어 있습니다.

4.1.1.1. 제품 권장 사항

VolSync

IKEvSync 0.5.1에서 dess **stable** 채널에서 사용 가능한 최신 버전으로 업그레이드를 준비하려면 다음 명령을 실행하여 **openshift-adp** 네임스페이스에 이 주석을 추가해야 합니다.

```
$ oc annotate --overwrite namespace/openshift-adp volsync.backube/privileged-movers='true'
```

Velero

이번 릴리스에서는 Velero가 버전 1.9.2에서 버전 1.9.5 로 업그레이드되었습니다.

Restic

이번 릴리스에서는 Restic이 버전 0.13.1에서 버전 0.14.0 으로 업그레이드되었습니다.

4.1.1.2. 수정된 버그

이번 릴리스에서는 다음 버그가 수정되었습니다.

- OADP-1150
- OADP-290
- OADP-1056

4.1.1.3. 확인된 문제

이 릴리스에는 다음과 같은 알려진 문제가 있습니다.

- OADP는 현재 Velero의 restic (**OADP-778**)을 사용하여 AWS EFS 볼륨의 백업 및 복원을 지원하지 않습니다.
- PVC당 **VolumeSnapshotContent** 스냅샷의 Ceph 제한으로 인해 CSI 백업이 실패할 수 있습니다.
  동일한 PVC(영구 볼륨 클레임)의 스냅샷을 여러 개 생성할 수 있지만 스냅샷의 주기적 생성은 예약할 수 없습니다.
  - CephFS의 경우 PVC당 최대 100개의 스냅샷을 생성할 수 있습니다. (**OADP-804**)
  - RADOS 블록 장치(RBD)의 경우 각 PVC에 대해 최대 512개의 스냅샷을 생성할 수 있습니다. (**OADP-975**)

자세한 내용은 볼륨 스냅샷 을 참조하십시오.

4.1.2. OADP 1.1.1 릴리스 노트

OADP 1.1.1 릴리스 노트에는 제품 권장 사항 및 알려진 문제에 대한 설명이 포함되어 있습니다.

4.1.2.1. 제품 권장 사항

OADP 1.1.1를 설치하기 전에ECDHESync 0.5.1을 설치하거나 업그레이드 하는 것이 좋습니다.

4.1.2.2. 확인된 문제

이 릴리스에는 다음과 같은 알려진 문제가 있습니다.

- OADP는 현재 Velero의 restic (OADP-778)을 사용하여 AWS EFS 볼륨의 백업 및 복원을 지원하지 않습니다.
- PVC당 **VolumeSnapshotContent** 스냅샷의 Ceph 제한으로 인해 CSI 백업이 실패할 수 있습니다.
  동일한 PVC(영구 볼륨 클레임)의 스냅샷을 여러 개 생성할 수 있지만 스냅샷의 주기적 생성은 예약할 수 없습니다.
  - CephFS의 경우 PVC당 최대 100개의 스냅샷을 생성할 수 있습니다.
  - RADOS 블록 장치(RBD)의 경우 각 PVC에 대해 최대 512개의 스냅샷을 생성할 수 있습니다. (OADP-804) 및 (OADP-975)
    자세한 내용은 볼륨 스냅샷 을 참조하십시오.

4.2. OADP 기능 및 플러그인

OADP(OpenShift API for Data Protection) 기능은 애플리케이션을 백업하고 복원하는 옵션을 제공합니다.

기본 플러그인을 사용하면 Velero가 특정 클라우드 공급자와 통합하고 OpenShift Container Platform 리소스를 백업 및 복원할 수 있습니다.

4.2.1. OADP 기능

OADP(OpenShift API for Data Protection)는 다음과 같은 기능을 지원합니다.

Backup

클러스터의 모든 리소스를 백업하거나 유형, 네임스페이스 또는 라벨에 따라 리소스를 필터링할 수 있습니다.
OADP는 Kubernetes 오브젝트 및 내부 이미지를 오브젝트 스토리지의 아카이브 파일로 저장하여 백업합니다. OADP는 기본 클라우드 스냅샷 API 또는 CSI(Container Storage Interface)로 스냅샷을 생성하여 PV(영구 볼륨)를 백업합니다. 스냅샷을 지원하지 않는 클라우드 공급자의 경우 OADP는 Restic을 사용하여 리소스 및 PV 데이터를 백업합니다.

Restore

백업에서 리소스와 PV를 복원할 수 있습니다. 백업의 모든 오브젝트를 복원하거나 네임스페이스, PV 또는 라벨에 따라 복원된 오브젝트를 필터링할 수 있습니다.

스케줄

지정된 간격으로 백업을 예약할 수 있습니다.

후크

후크를 사용하여 Pod의 컨테이너에서 명령을 실행할 수 있습니다(예: **fsfreeze**)를 사용하여 파일 시스템을 정지할 수 있습니다. 백업 또는 복원 전이나 후에 실행되도록 후크를 구성할 수 있습니다. 복원 후크는 init 컨테이너 또는 애플리케이션 컨테이너에서 실행할 수 있습니다.

4.2.2. OADP 플러그인

OADP(OpenShift API for Data Protection)는 백업 및 스냅샷 작업을 지원하기 위해 스토리지 공급자와 통합된 기본 Velero 플러그인을 제공합니다. Velero 플러그인을 기반으로 사용자 정의 플러그인을 생성할 수 있습니다.

OADP는 OpenShift Container Platform 리소스 백업, OpenShift Virtualization 리소스 백업 및 CSI(Container Storage Interface) 스냅샷에 대한 플러그인도 제공합니다.

표 4.1. OADP 플러그인

OADP 플러그인	함수	스토리지 위치
aws	Kubernetes 오브젝트를 백업하고 복원합니다.	AWS S3
	스냅샷을 사용하여 볼륨을 백업 및 복원합니다.	AWS EBS
azure	Kubernetes 오브젝트를 백업하고 복원합니다.	Microsoft Azure Blob 스토리지
	스냅샷을 사용하여 볼륨을 백업 및 복원합니다.	Microsoft Azure 관리형 디스크
gcp	Kubernetes 오브젝트를 백업하고 복원합니다.	Google Cloud Storage
	스냅샷을 사용하여 볼륨을 백업 및 복원합니다.	Google Compute Engine Disks
openshift	OpenShift Container Platform 리소스를 백업하고 복원합니다. [1]	오브젝트 저장소
kubevirt	OpenShift Virtualization 리소스를 백업하고 복원합니다. [2]	오브젝트 저장소
csi	CSI 스냅샷을 사용하여 볼륨을 백업 및 복원합니다. [3]	CSI 스냅샷을 지원하는 클라우드 스토리지

1. 필수 항목입니다.
2. 가상 머신 디스크는 CSI 스냅샷 또는 Restic을 사용하여 백업됩니다.
3. **csi** 플러그인은 Velero CSI 베타 스냅샷 API 를 사용합니다.

4.2.3. OADP Velero 플러그인 정보

Velero를 설치할 때 다음 두 가지 유형의 플러그인을 구성할 수 있습니다.

- 기본 클라우드 공급자 플러그인
- 사용자 정의 플러그인

두 가지 유형의 플러그인은 모두 선택 사항이지만 대부분의 사용자는 하나 이상의 클라우드 공급자 플러그인을 구성합니다.

4.2.3.1. 기본 Velero 클라우드 공급자 플러그인

배포 중에 **oadp_v1alpha1_dpa.yaml** 파일을 구성할 때 다음과 같은 기본 Velero 클라우드 공급자 플러그인을 설치할 수 있습니다.

- **AWS** (Amazon Web Services)
- **GCP** (Google Cloud Platform)
- **azure** (Microsoft Azure)
- **OpenShift** (OpenShift Velero 플러그인)
- **CSI** (컨테이너 스토리지 인터페이스)
- **Kube** Virt(KubeVirt)

배포 중에 **oadp_v1alpha1_dpa.yaml** 파일에 원하는 기본 플러그인을 지정합니다.

예제 파일

다음 **.yaml** 파일은 **openshift**,**aws**,**azure**, **gcp** 플러그인을 설치합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: dpa-sample
spec:
  configuration:
    velero:
      defaultPlugins:
      - openshift
      - aws
      - azure
      - gcp
```

4.2.3.2. 사용자 정의 Velero 플러그인

배포 중에 **oadp_v1alpha1_dpa.yaml** 파일을 구성할 때 플러그인 **이미지** 및 **이름을** 지정하여 사용자 정의 Velero 플러그인을 설치할 수 있습니다.

배포 중에 **oadp_v1alpha1_dpa.yaml** 파일에 원하는 사용자 지정 플러그인을 지정합니다.

예제 파일

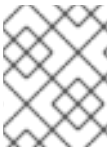
다음 **.yaml** 파일은 기본 **openshift**,**azure**, **gcp** 플러그인 및 이름이 **custom-plugin-example** 이고 **quay.io/example-repo/custom-velero-plugin** 이미지가 있는 사용자 정의 플러그인을 설치합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
 name: dpa-sample
spec:
 configuration:
   velero:
     defaultPlugins:
     - openshift
     - azure
     - gcp
     customPlugins:
     - name: custom-plugin-example
       image: quay.io/example-repo/custom-velero-plugin
```

4.3. OADP 설치 및 구성

4.3.1. OADP 설치 정보

클러스터 관리자는 OADP Operator를 설치하여 OADP(Data Protection)용 OpenShift API를 설치합니다. OADP Operator는 Velero 1.7 을 설치합니다.



참고

OADP 1.0.4부터 모든 OADP 1.0.*z* 버전은 MTC Operator의 종속성으로만 사용할 수 있으며 독립 실행형 Operator로 사용할 수 없습니다.

Kubernetes 리소스 및 내부 이미지를 백업하려면 다음 스토리지 유형 중 하나와 같은 오브젝트 스토리지를 백업 위치로 보유해야 합니다.

- Amazon Web Services
- Microsoft Azure
- Google Cloud Platform
- Multicloud Object Gateway
- S3 호환 오브젝트 스토리지(예: Noobaa 또는 Minio)

중요

개체 스토리지의 버킷 생성을 자동화하는 **CloudStorage** API는 기술 프리뷰 기능 전용입니다. 기술 프리뷰 기능은 Red Hat 프로덕션 서비스 수준 계약(SLA)에서 지원되지 않으며 기능적으로 완전하지 않을 수 있습니다. 따라서 프로덕션 환경에서 사용하는 것은 권장하지 않습니다. 이러한 기능을 사용하면 향후 제품 기능을 조기에 이용할 수 있어 개발 과정에서 고객이 기능을 테스트하고 피드백을 제공할 수 있습니다.

Red Hat 기술 프리뷰 기능의 지원 범위에 대한 자세한 내용은 기술 프리뷰 기능 지원 범위를 참조하십시오.

스냅샷 또는 Restic을 사용하여 PV(영구 볼륨)를 백업할 수 있습니다.

스냅샷을 사용하여 PV를 백업하려면 다음 클라우드 공급자 중 하나와 같은 기본 스냅샷 API 또는 CSI(Container Storage Interface) 스냅샷을 지원하는 클라우드 공급자가 있어야 합니다.

- Amazon Web Services
- Microsoft Azure
- Google Cloud Platform
- CSI 스냅샷 지원 클라우드 공급자(예: OpenShift Data Foundation)

클라우드 공급자가 스냅샷을 지원하지 않거나 스토리지가 NFS인 경우 개체 스토리지에서 Restic 백업을 사용하여 애플리케이션을 백업할 수 있습니다.

기본 보안을 생성한 다음 데이터 보호 애플리케이션을 설치합니다.

4.3.1.1. OpenShift Data Foundation에서 재해 복구를 위해 NooBaa 구성

OpenShift Data Foundation에서 NooBaa 버킷 backupStorageLocation 에 클러스터 스토리지를 사용하는 경우 NooBaa를 외부 오브젝트 저장소로 구성합니다.

주의

NooBaa를 외부 오브젝트 저장소로 구성하지 못하면 백업을 사용할 수 없을 수 있습니다.

절차

- **하이브리드 또는 Multicloud에 대한 스토리지 리소스 추가 에 설명된 대로 NooBaa를 외부 오브젝트 저장소로 구성합니다.**

추가 리소스

- **Velero 문서 의 백업 위치 및 스냅샷 위치 개요.**

/// 모듈은 다음 assembly에 포함되어 있습니다.

4.3.1.2. OADP 업데이트 채널 정보

OADP Operator를 설치할 때 *업데이트 채널*을 선택합니다. 이 채널은 **OADP Operator** 및 수신하는 **Velero**에 대한 업그레이드를 결정합니다. 언제든지 채널을 변경할 수 있습니다.

다음 세 가지 업데이트 채널이 있습니다.

- **stable** 채널에는 **OADP ClusterServiceVersion**의 최신 마이너 업데이트(**y-stream** 업데이트) 및 패치(**z-stream** 업데이트)가 포함되어 있습니다. 각 새 릴리스가 게시되면 **OADP Operator**의 사용 가능한 **ClusterServiceVersion** 에 사용 가능한 최신 마이너 패치가 추가됩니다.
- **stable-1.0** 채널에는 최신 **OADP 1.0 ClusterServiceVersion** 인 **oadp.v1.0.*z*** 가 포함되어 있습니다.
- **stable-1.1** 채널에는 최신 **OADP 1.1 ClusterServiceVersion** 인 **oadp.v1.1.*z*** 가 포함되어 있습니다.

어떤 업데이트 채널을 사용할 수 있습니까?

- 안정적인 최신 **OADP** 버전을 설치하고 마이너 업데이트와 패치를 모두 받으려면 **stable** 업데이트 채널을 선택합니다. 이 채널을 선택하면 ***x.y.z*** 의 모든 **y-**스트림 및 모든 **z-stream** 업데이트가 제공됩니다.
- **stable-1.*y*** 업데이트 채널을 선택하여 **OADP 1.*y*** 를 설치하고 계속 패치를 수신합니다. 이 채널을 선택하면 버전 **1**에 대한 모든 **z-stream** 패치가 제공됩니다.***y.z***.

언제 업데이트 채널을 전환해야합니까?

- **OADP 1.*y*** 가 설치되어 있고 해당 **y-stream** 전용 패치를 받으려면 **stable** 업데이트 채널에서 **stable-1.*y*** 업데이트 채널로 전환해야 합니다. 그런 다음 버전 **1.*y*. z**에 대한 모든***z* -stream** 패치가 제공됩니다.
- **OADP 1.0**이 설치되어 있고 **OADP 1.1**로 업그레이드하려는 경우 패치는 **stable-1.0** 업데이트 채널에서 **stable-1.1** 업데이트 채널로 전환해야 합니다. 그런 다음 버전 **1.1.*z*** 에 대한 모든 **z-stream** 패치가 제공됩니다.

- **OADP 1.*y*** 가 0보다 크게 설치되어 있고 **OADP 1.0**으로 전환하려는 경우 **OADP Operator**를 *제거한* 다음 **stable-1.0** 업데이트 채널을 사용하여 다시 설치해야 합니다. 그러면 버전 **1.0.*z***에 대한 모든 **z-stream** 패치가 제공됩니다.

참고

업데이트 채널을 전환하여 **OADP 1.*y*** 에서 **OADP 1.0**으로 전환할 수 없습니다. **Operator**를 제거한 다음 다시 설치해야 합니다.

추가 리소스

- 클러스터 서비스 버전

4.3.2. Amazon Web Services를 통한 데이터 보호를 위한 OpenShift API 설치 및 구성

OADP Operator를 설치하여 **AWS(Amazon Web Services)**에서 **OADP(Data Protection)**용 **OpenShift API**를 설치합니다. **Operator**는 **Velero 1.7** 을 설치합니다.

참고

OADP 1.0.4부터 모든 **OADP 1.0.*z*** 버전은 **MTC Operator**의 종속성으로만 사용할 수 있으며 독립 실행형 **Operator**로 사용할 수 없습니다.

Velero에 대해 **AWS**를 구성하고 기본 시크릿 을 생성한 다음 데이터 보호 애플리케이션을 설치합니다.

제한된 네트워크 환경에 **OADP Operator**를 설치하려면 먼저 기본 **OperatorHub** 소스를 비활성화하고 **Operator** 카탈로그를 미러링해야 합니다. 자세한 내용은 제한된 네트워크에서 **Operator Lifecycle Manager** 사용을 참조하십시오.

4.3.2.1. OADP Operator 설치

OLM(Operator Lifecycle Manager)을 사용하여 **OpenShift Container Platform 4.10**에 **OADP(OpenShift API for Data Protection) Operator**를 설치합니다.

OADP Operator는 **Velero 1.7** 을 설치합니다.

사전 요구 사항

- cluster-admin 권한이 있는 사용자로 로그인해야 합니다.

절차

1. OpenShift Container Platform 웹 콘솔에서 Operator → OperatorHub를 클릭합니다.
2. 키워드로 필터링 필드를 사용하여 OADP Operator 를 찾습니다.
3. OADP Operator 를 선택하고 설치를 클릭합니다.
4. openshift-adp 프로젝트에서 Operator를 설치하려면 설치를 클릭합니다.
5. Operators → 설치된 Operators 를 클릭하여 설치를 확인합니다.

4.3.2.2. Amazon Web Services 구성

OADP(OpenShift API for Data Protection)에 대해 AWS(Amazon Web Services)를 구성합니다.

사전 요구 사항

- AWS CLI 가 설치되어 있어야 합니다.

절차

1. BUCKET 변수를 설정합니다.

```
$ BUCKET=<your_bucket>
```

2. REGION 변수를 설정합니다.

```
$ REGION=<your_region>
```

3. AWS S3 버킷을 생성합니다.

```
$ aws s3api create-bucket \
  --bucket $BUCKET \
  --region $REGION \
  --create-bucket-configuration LocationConstraint=$REGION 1
```

1 us-east-1은 LocationConstraint를 지원하지 않습니다. 리전이 us-east-1인 경우 --create-bucket-configuration LocationConstraint=$REGION을 생략합니다.

4. IAM 사용자를 생성합니다.

```
$ aws iam create-user --user-name velero 1
```

1 Velero를 사용하여 여러 S3 버킷이 있는 여러 클러스터를 백업하려면 각 클러스터에 대해 고유한 사용자 이름을 생성합니다.

5. velero-policy.json 파일을 생성합니다.

```
$ cat > velero-policy.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeVolumes",
        "ec2:DescribeSnapshots",
        "ec2:CreateTags",
        "ec2:CreateVolume",
        "ec2:CreateSnapshot",
        "ec2:DeleteSnapshot"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:DeleteObject",
        "s3:PutObject",
        "s3:AbortMultipartUpload",
```

```
            "s3:ListMultipartUploadParts"
        ],
        "Resource": [
            "arn:aws:s3:::${BUCKET}/*"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "s3:ListBucket",
            "s3:GetBucketLocation",
            "s3:ListBucketMultipartUploads"
        ],
        "Resource": [
            "arn:aws:s3:::${BUCKET}"
        ]
    }
  ]
}
EOF
```

6. 정책을 연결하여 **velero** 사용자에게 필요한 최소 권한을 부여합니다.

```
$ aws iam put-user-policy \
  --user-name velero \
  --policy-name velero \
  --policy-document file://velero-policy.json
```

7. **velero** 사용자에 대한 액세스 키를 생성합니다.

```
$ aws iam create-access-key --user-name velero
```

출력 예

```
{
  "AccessKey": {
        "UserName": "velero",
        "Status": "Active",
        "CreateDate": "2017-07-31T22:24:41.576Z",
        "SecretAccessKey": <AWS_SECRET_ACCESS_KEY>,
        "AccessKeyId": <AWS_ACCESS_KEY_ID>
  }
}
```

8. **credentials-velero** 파일을 만듭니다.

```
$ cat << EOF > ./credentials-velero
[default]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
EOF
```

데이터 보호 애플리케이션을 설치하기 전에 **credentials-velero** 파일을 사용하여 **AWS**용 **Secret** 오브젝트를 생성합니다.

4.3.2.3. 백업 및 스냅샷 위치 및 보안 정보

DataProtectionApplication CR(사용자 정의 리소스)에서 백업 및 스냅샷 위치 및 해당 시크릿을 지정합니다.

백업 위치

Multicloud Object Gateway, Noobaa 또는 **Minio**와 같은 **S3** 호환 개체 스토리지를 백업 위치로 지정합니다.

Velero는 **OpenShift Container Platform** 리소스, **Kubernetes** 오브젝트 및 내부 이미지를 오브젝트 스토리지의 아카이브 파일로 백업합니다.

스냅샷 위치

영구 볼륨을 백업하기 위해 클라우드 공급자의 네이티브 스냅샷 **API**를 사용하는 경우 클라우드 공급자를 스냅샷 위치로 지정해야 합니다.

CSI(Container Storage Interface) 스냅샷을 사용하는 경우 **CSI** 드라이버를 등록하기 위해 **VolumeSnapshotClass CR**을 생성하기 때문에 스냅샷 위치를 지정할 필요가 없습니다.

Restic을 사용하는 경우 **Restic**이 개체 스토리지의 파일 시스템을 백업하므로 스냅샷 위치를 지정할 필요가 없습니다.

보안

백업 및 스냅샷 위치가 동일한 자격 증명을 사용하거나 스냅샷 위치가 필요하지 않은 경우 기본 시크릿 을 생성합니다.

백업 및 스냅샷 위치가 다른 인증 정보를 사용하는 경우 두 개의 시크릿 오브젝트를 생성합니다.

- **DataProtectionApplication CR**에서 지정하는 백업 위치에 대한 사용자 정의 시크릿 입니다.
- **DataProtectionApplication CR**에서 참조되지 않는 스냅샷 위치에 대한 기본 시크릿 입니다.

중요

데이터 보호 애플리케이션에는 기본 보안이 필요합니다. 그렇지 않으면 설치에 실패합니다.

설치 중에 백업 또는 스냅샷 위치를 지정하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 보안을 생성할 수 있습니다.

4.3.2.3.1. 기본 보안 생성

백업 및 스냅샷 위치가 동일한 자격 증명을 사용하거나 스냅샷 위치가 필요하지 않은 경우 기본 보안을 생성합니다.

Secret 의 기본 이름은 **cloud-credentials** 입니다.

참고

DataProtectionApplication CR(사용자 정의 리소스)에는 기본 시크릿 이 필요합니다. 그렇지 않으면 설치에 실패합니다. 백업 위치 **Secret** 의 이름이 지정되지 않은 경우 기본 이름이 사용됩니다.

설치 중에 백업 위치 자격 증명을 사용하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 이름으로 **Secret** 을 생성할 수 있습니다.

사전 요구 사항

- 오브젝트 스토리지 및 클라우드 스토리지는 동일한 인증 정보를 사용해야 합니다.

- Velero에 대한 오브젝트 스토리지를 구성해야 합니다.
- 오브젝트 스토리지의 credentials-velero 파일을 적절한 형식으로 만들어야 합니다.

절차

- 기본 이름으로 보안을 생성합니다.

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file
cloud=credentials-velero
```

Secret 은 데이터 보호 애플리케이션을 설치할 때 DataProtectionApplication CR의 spec.backupLocations.credential 블록에서 참조합니다.

4.3.2.3.2. 다양한 인증 정보에 대한 프로필 생성

백업 및 스냅샷 위치에 다른 인증 정보를 사용하는 경우 credentials-velero 파일에 별도의 프로필을 생성합니다.

그런 다음 Secret 오브젝트를 생성하고 DataProtectionApplication CR(사용자 정의 리소스)에서 프로필을 지정합니다.

절차

1. 다음 예제와 같이 백업 및 스냅샷 위치에 대한 별도의 프로필을 사용하여 credentials-velero 파일을 만듭니다.

```
[backupStorage]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>

[volumeSnapshot]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
```

2. credentials-velero 파일을 사용하여 Secret 오브젝트를 생성합니다.

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file
cloud=credentials-velero 1
```

3. 다음 예와 같이 **DataProtectionApplication CR**에 프로필을 추가합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket_name>
          prefix: <prefix>
        config:
          region: us-east-1
          profile: "backupStorage"
        credential:
          key: cloud
          name: cloud-credentials
  snapshotLocations:
    - name: default
      velero:
        provider: aws
        config:
          region: us-west-2
          profile: "volumeSnapshot"
```

4.3.2.4. 데이터 보호 애플리케이션 구성

Velero 리소스 할당을 설정하거나 자체 서명된 **CA** 인증서를 활성화하여 데이터 보호 애플리케이션을 구성할 수 있습니다.

4.3.2.4.1. Velero CPU 및 메모리 리소스 할당 설정

DataProtectionApplication CR(사용자 정의 리소스) 매니페스트를 편집하여 **Velero Pod**의 **CPU** 및 메모리 리소스 할당을 설정합니다.

사전 요구 사항

- **OADP(OpenShift API for Data Protection) Operator**가 설치되어 있어야 합니다.

절차

- 다음 예와 같이 **DataProtectionApplication CR** 매니페스트의 **spec.configuration.velero.podConfig.ResourceAllocations** 블록에서 값을 편집합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node selector> 1
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 512Mi
          requests:
            cpu: 500m
            memory: 256Mi
```

1 1 Velero podSpec에 제공할 노드 선택기 지정

4.3.2.4.2. 자체 서명 CA 인증서 활성화

알 수 없는 기관 오류로 서명된 인증서를 방지하려면 **DataProtectionApplication CR**(사용자 정의 리소스) 매니페스트를 편집하여 개체 스토리지에 대해 자체 서명된 **CA** 인증서를 활성화해야 합니다.

사전 요구 사항

- **OADP(OpenShift API for Data Protection) Operator**가 설치되어 있어야 합니다.

절차

- **DataProtectionApplication CR** 매니페스트의 **spec.backupLocations.velero.objectStorage.caCert** 매개변수 및 **spec.backupLocations.velero.config** 매개변수를 편집합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
```

```
  name: <dpa_sample>
spec:
...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket>
          prefix: <prefix>
          caCert: <base64_encoded_cert_string> 1
        config:
          insecureSkipTLSVerify: "false" 2
...
```

1

Base46 인코딩 CA 인증서 문자열을 지정합니다.

2

insecureSkipTLSVerify 구성은 "true" 또는 "false" 로 설정할 수 있습니다. "true" 로 설정하면 SSL/TLS 보안이 비활성화됩니다. "false" 로 설정하면 SSL/TLS 보안이 활성화됩니다.

4.3.2.5. 데이터 보호 애플리케이션 설치

Data ProtectionApplication API 인스턴스를 만들어 DPA(Data Protection Application)를 설치합니다.

사전 요구 사항

- OADP Operator를 설치해야 합니다.
- 오브젝트 스토리지를 백업 위치로 구성해야 합니다.
- 스냅샷을 사용하여 PV를 백업하는 경우 클라우드 공급자는 기본 스냅샷 API 또는 CSI(Container Storage Interface) 스냅샷을 지원해야 합니다.
- 백업 및 스냅샷 위치가 동일한 자격 증명을 사용하는 경우 기본 이름 cloud-credentials 를 사용하여 Secret 을 생성해야 합니다.

- 백업 및 스냅샷 위치에 다른 인증 정보를 사용하는 경우 백업 및 스냅샷 위치 자격 증명에 대한 별도의 프로필이 포함된 기본 이름 **cloud-credentials** 를 사용하여 **Secret** 을 생성해야 합니다.

참고

설치 중에 백업 또는 스냅샷 위치를 지정하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 보안을 생성할 수 있습니다. 기본 보안이 없으면 설치에 실패합니다.

절차

1. **Operators** → 설치된 **Operator** 를 클릭하고 **OADP Operator**를 선택합니다.

2. 제공된 **API** 아래의 **DataProtectionApplication** 상자에서 인스턴스 생성을 클릭합니다.

3. **YAML** 보기를 클릭하고 **DataProtectionApplication** 매니페스트의 매개변수를 업데이트합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - openshift (1)
        - aws
    restic:
      enable: true (2)
      podConfig:
        nodeSelector: <node selector> (3)
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket_name> (4)
          prefix: <prefix> (5)
        config:
          region: <region>
          profile: "default"
```

```
    credential:
      key: cloud
      name: cloud-credentials (6)
snapshotLocations: (7)
 - name: default
   velero:
     provider: aws
     config:
       region: <region> (8)
       profile: "default"
```

(1)

openshift 플러그인은 필수입니다.

(2)

Restic 설치를 비활성화하려면 false 로 설정합니다. Restic은 데몬 세트를 배포합니다. 즉, 각 작업자 노드에 Restic pod가 실행됩니다. Backup CR에 spec.defaultVolumesToRestic: true 를 추가하여 백업에 Restic을 구성합니다.

(3)

Restic podSpec에 제공할 노드 선택기를 지정합니다.

(4)

버킷을 백업 스토리지 위치로 지정합니다. 버킷이 Velero 백업용 전용 버킷이 아닌 경우 접두사를 지정해야 합니다.

(5)

버킷이 여러 용도로 사용되는 경우 Velero 백업의 접두사(예: velero)를 지정합니다.

(6)

생성한 Secret 오브젝트의 이름을 지정합니다. 이 값을 지정하지 않으면 기본 이름 cloud-credentials 가 사용됩니다. 사용자 지정 이름을 지정하면 백업 위치에 사용자 지정 이름이 사용됩니다.

(7)

PV를 백업하기 위해 CSI 스냅샷 또는 Restic을 사용하는 경우 스냅샷 위치를 지정할 필요가 없습니다.

(8)

스냅샷 위치는 PV와 동일한 리전에 있어야 합니다.

4. 생성을 클릭합니다.

5. OADP 리소스를 확인하여 설치를 확인합니다.

```
$ oc get all -n openshift-adp
```

출력 예

```
NAME                                                 READY   STATUS    RESTARTS   AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8   2/2     Running   0          2m8s
pod/restic-9cq4q                                     1/1     Running   0          94s
pod/restic-m4lts                                     1/1     Running   0          94s
pod/restic-pv4kr                                     1/1     Running   0          95s
pod/velero-588db7f655-n842v                          1/1     Running   0          95s

NAME                                                     TYPE        CLUSTER-IP       EXTERNAL-IP
PORT(S)    AGE
service/oadp-operator-controller-manager-metrics-service   ClusterIP   172.30.70.140
<none>        8443/TCP   2m8s

NAME                DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE
SELECTOR   AGE
daemonset.apps/restic   3         3         3       3            3           <none>          96s

NAME                                          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/oadp-operator-controller-manager   1/1     1            1           2m9s
deployment.apps/velero                        1/1     1            1           96s

NAME                                                   DESIRED   CURRENT   READY   AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47   1         1         1       2m9s
replicaset.apps/velero-588db7f655                      1         1         1       96s
```

4.3.2.5.1. DataProtectionApplication CR에서 CSI 활성화

CSI 스냅샷으로 영구 볼륨을 백업하기 위해 DataProtectionApplication CR(사용자 정의 리소스)에서 CSI(Container Storage Interface)를 활성화합니다.

사전 요구 사항

- 클라우드 공급자는 CSI 스냅샷을 지원해야 합니다.

절차

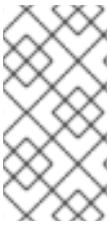
- 다음 예와 같이 **DataProtectionApplication CR**을 편집합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
      - openshift
      - csi 1
```

1 **csi** 기본 플러그인을 추가합니다.

4.3.3. Microsoft Azure를 사용하여 OpenShift API for Data Protection 설치 및 구성

OADP Operator를 설치하여 **Microsoft Azure**와 함께 **OADP(Data Protection)**용 **OpenShift API**를 설치합니다. **Operator**는 **Velero 1.7** 을 설치합니다.



참고

OADP 1.0.4부터 모든 **OADP 1.0.*z*** 버전은 **MTC Operator**의 종속성으로만 사용할 수 있으며 독립 실행형 **Operator**로 사용할 수 없습니다.

Velero에 대해 **Azure**를 구성하고 기본 시크릿 을 생성한 다음 데이터 보호 애플리케이션을 설치합니다.

제한된 네트워크 환경에 **OADP Operator**를 설치하려면 먼저 기본 **OperatorHub** 소스를 비활성화하고 **Operator** 카탈로그를 미러링해야 합니다. 자세한 내용은 제한된 네트워크에서 **Operator Lifecycle Manager** 사용을 참조하십시오.

4.3.3.1. OADP Operator 설치

OLM(Operator Lifecycle Manager)을 사용하여 **OpenShift Container Platform 4.10**에 **OADP(OpenShift API for Data Protection) Operator**를 설치합니다.

OADP Operator는 Velero 1.7 을 설치합니다.

사전 요구 사항

- cluster-admin 권한이 있는 사용자로 로그인해야 합니다.

절차

1. OpenShift Container Platform 웹 콘솔에서 Operator → OperatorHub를 클릭합니다.
2. 키워드로 필터링 필드를 사용하여 OADP Operator 를 찾습니다.
3. OADP Operator 를 선택하고 설치를 클릭합니다.
4. openshift-adp 프로젝트에서 Operator를 설치하려면 설치를 클릭합니다.
5. Operators → 설치된 Operators 를 클릭하여 설치를 확인합니다.

4.3.3.2. Microsoft Azure 구성

OADP(OpenShift API for Data Protection)를 위해 Microsoft Azure를 구성합니다.

사전 요구 사항

- Azure CLI 가 설치되어 있어야 합니다.

절차

1. Azure에 로그인합니다.

```
$ az login
```

2. AZURE_RESOURCE_GROUP 변수를 설정합니다.

```
$ AZURE_RESOURCE_GROUP=Velero_Backups
```

3. Azure 리소스 그룹을 생성합니다.

```
$ az group create -n $AZURE_RESOURCE_GROUP --location CentralUS 1
```

1 위치를 지정합니다.

4. AZURE_STORAGE_ACCOUNT_ID 변수를 설정합니다.

```
$ AZURE_STORAGE_ACCOUNT_ID="velero$(uuidgen | cut -d '-' -f5 | tr '[A-Z]' '[a-z]')"
```

5. Azure 스토리지 계정을 생성합니다.

```
$ az storage account create \
  --name $AZURE_STORAGE_ACCOUNT_ID \
  --resource-group $AZURE_RESOURCE_GROUP \
  --sku Standard_GRS \
  --encryption-services blob \
  --https-only true \
  --kind BlobStorage \
  --access-tier Hot
```

6. BLOB_CONTAINER 변수를 설정합니다.

```
$ BLOB_CONTAINER=velero
```

7. Azure Blob 스토리지 컨테이너를 생성합니다.

```
$ az storage container create \
  -n $BLOB_CONTAINER \
  --public-access off \
  --account-name $AZURE_STORAGE_ACCOUNT_ID
```

8. 스토리지 계정 액세스 키를 가져옵니다.

```
$ AZURE_STORAGE_ACCOUNT_ACCESS_KEY=`az storage account keys list \
  --account-name $AZURE_STORAGE_ACCOUNT_ID \
  --query "[?keyName == 'key1'].value" -o tsv`
```

9. 필요한 최소 권한이 있는 사용자 지정 역할을 생성합니다.

```
AZURE_ROLE=Velero
az role definition create --role-definition '{
   "Name": "'$AZURE_ROLE'",
   "Description": "Velero related permissions to perform backups, restores and
deletions",
   "Actions": [
      "Microsoft.Compute/disks/read",
      "Microsoft.Compute/disks/write",
      "Microsoft.Compute/disks/endGetAccess/action",
      "Microsoft.Compute/disks/beginGetAccess/action",
      "Microsoft.Compute/snapshots/read",
      "Microsoft.Compute/snapshots/write",
      "Microsoft.Compute/snapshots/delete",
      "Microsoft.Storage/storageAccounts/listkeys/action",
      "Microsoft.Storage/storageAccounts/regeneratekey/action"
   ],
   "AssignableScopes": ["/subscriptions/'$AZURE_SUBSCRIPTION_ID'"]
   }'
```

10. credentials-velero 파일을 만듭니다.

```
$ cat << EOF > ./credentials-velero
AZURE_SUBSCRIPTION_ID=${AZURE_SUBSCRIPTION_ID}
AZURE_TENANT_ID=${AZURE_TENANT_ID}
AZURE_CLIENT_ID=${AZURE_CLIENT_ID}
AZURE_CLIENT_SECRET=${AZURE_CLIENT_SECRET}
AZURE_RESOURCE_GROUP=${AZURE_RESOURCE_GROUP}
AZURE_STORAGE_ACCOUNT_ACCESS_KEY=${AZURE_STORAGE_ACCOUNT_ACCE
SS_KEY} (1)
AZURE_CLOUD_NAME=AzurePublicCloud
EOF
```

(1) 필수 항목입니다. credentials-velero 파일에 서비스 주체 자격 증명만 포함된 경우 내부 이미지를 백업할 수 없습니다.

credentials-velero 파일을 사용하여 데이터 보호 애플리케이션을 설치하기 전에 Azure용 Secret 오브젝트를 생성합니다.

4.3.3.3. 백업 및 스냅샷 위치 및 보안 정보

DataProtectionApplication CR(사용자 정의 리소스)에서 백업 및 스냅샷 위치 및 해당 시크릿을 지정합니다.

백업 위치

Multicloud Object Gateway, Noobaa 또는 **Minio**와 같은 **S3** 호환 개체 스토리지를 백업 위치로 지정합니다.

Velero는 **OpenShift Container Platform** 리소스, **Kubernetes** 오브젝트 및 내부 이미지를 오브젝트 스토리지의 아카이브 파일로 백업합니다.

스냅샷 위치

영구 볼륨을 백업하기 위해 클라우드 공급자의 네이티브 스냅샷 **API**를 사용하는 경우 클라우드 공급자를 스냅샷 위치로 지정해야 합니다.

CSI(Container Storage Interface) 스냅샷을 사용하는 경우 **CSI** 드라이버를 등록하기 위해 **VolumeSnapshotClass CR**을 생성하기 때문에 스냅샷 위치를 지정할 필요가 없습니다.

Restic을 사용하는 경우 **Restic**이 개체 스토리지의 파일 시스템을 백업하므로 스냅샷 위치를 지정할 필요가 없습니다.

보안

백업 및 스냅샷 위치가 동일한 자격 증명을 사용하거나 스냅샷 위치가 필요하지 않은 경우 기본 시크릿 을 생성합니다.

백업 및 스냅샷 위치가 다른 인증 정보를 사용하는 경우 두 개의 시크릿 오브젝트를 생성합니다.

- **DataProtectionApplication CR**에서 지정하는 백업 위치에 대한 사용자 정의 시크릿 입니다.
- **DataProtectionApplication CR**에서 참조되지 않는 스냅샷 위치에 대한 기본 시크릿 입니다.

중요

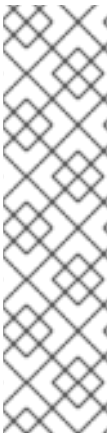
데이터 보호 애플리케이션에는 기본 보안이 필요합니다. 그렇지 않으면 설치에 실패합니다.

설치 중에 백업 또는 스냅샷 위치를 지정하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 보안을 생성할 수 있습니다.

4.3.3.3.1. 기본 보안 생성

백업 및 스냅샷 위치가 동일한 자격 증명을 사용하거나 스냅샷 위치가 필요하지 않은 경우 기본 보안을 생성합니다.

Secret 의 기본 이름은 **cloud-credentials-azure** 입니다.



참고

DataProtectionApplication CR(사용자 정의 리소스)에는 기본 시크릿 이 필요합니다. 그렇지 않으면 설치에 실패합니다. 백업 위치 **Secret** 의 이름이 지정되지 않은 경우 기본 이름이 사용됩니다.

설치 중에 백업 위치 자격 증명을 사용하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 이름으로 **Secret** 을 생성할 수 있습니다.

사전 요구 사항

- 오브젝트 스토리지 및 클라우드 스토리지는 동일한 인증 정보를 사용해야 합니다.
- **Velero**에 대한 오브젝트 스토리지를 구성해야 합니다.
- 오브젝트 스토리지의 **credentials-velero** 파일을 적절한 형식으로 만들어야 합니다.

절차

- 기본 이름으로 보안을 생성합니다.

```
$ oc create secret generic cloud-credentials-azure -n openshift-adp --from-file
cloud=credentials-velero
```

Secret 은 데이터 보호 애플리케이션을 설치할 때 DataProtectionApplication CR의 spec.backupLocations.credential 블록에서 참조합니다.

4.3.3.3.2. 다른 인증 정보의 보안 생성

백업 및 스냅샷 위치에 다른 인증 정보를 사용하는 경우 두 개의 Secret 오브젝트를 생성해야 합니다.

- 사용자 지정 이름을 사용한 백업 위치 시크릿. 사용자 정의 이름은 DataProtectionApplication CR(사용자 정의 리소스)의 spec.backupLocations 블록에 지정됩니다.
- 기본 이름 cloud-credentials-azure 인 스냅샷 위치 시크릿. 이 Secret 은 DataProtectionApplication CR에 지정되지 않습니다.

절차

1. 클라우드 공급자에 적합한 형식으로 스냅샷 위치에 대한 자격 증명-velero 파일을 만듭니다.
2. 기본 이름으로 스냅샷 위치에 대한 Secret 을 생성합니다.

```
$ oc create secret generic cloud-credentials-azure -n openshift-adp --from-file
cloud=credentials-velero
```

3. 오브젝트 스토리지의 적절한 형식으로 백업 위치에 대한 credentials-velero 파일을 만듭니다.
4. 사용자 정의 이름으로 백업 위치에 대한 보안을 생성합니다.

```
$ oc create secret generic <custom_secret> -n openshift-adp --from-file
cloud=credentials-velero
```

5. 다음 예와 같이 사용자 지정 이름이 있는 Secret 을 DataProtectionApplication CR에 추가합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
...
  backupLocations:
    - velero:
        config:
          resourceGroup: <azure_resource_group>
          storageAccount: <azure_storage_account_id>
          subscriptionId: <azure_subscription_id>
          storageAccountKeyEnvVar: AZURE_STORAGE_ACCOUNT_ACCESS_KEY
        credential:
          key: cloud
          name: <custom_secret> (1)
        provider: azure
        default: true
        objectStorage:
          bucket: <bucket_name>
          prefix: <prefix>
  snapshotLocations:
    - velero:
        config:
          resourceGroup: <azure_resource_group>
          subscriptionId: <azure_subscription_id>
          incremental: "true"
        name: default
        provider: azure
```

(1) 사용자 정의 이름으로 백업 위치 시크릿.

4.3.3.4. 데이터 보호 애플리케이션 구성

Velero 리소스 할당을 설정하거나 자체 서명된 **CA** 인증서를 활성화하여 데이터 보호 애플리케이션을 구성할 수 있습니다.

4.3.3.4.1. Velero CPU 및 메모리 리소스 할당 설정

DataProtectionApplication CR(사용자 정의 리소스) 매니페스트를 편집하여 **Velero Pod**의 **CPU** 및 메모리 리소스 할당을 설정합니다.

사전 요구 사항

- OADP(OpenShift API for Data Protection) Operator가 설치되어 있어야 합니다.

절차

- 다음 예와 같이 DataProtectionApplication CR 매니페스트의 spec.configuration.velero.podConfig.ResourceAllocations 블록에서 값을 편집합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node selector> 1
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 512Mi
          requests:
            cpu: 500m
            memory: 256Mi
```

1 Velero podSpec에 제공할 노드 선택기 지정

4.3.3.4.2. 자체 서명 CA 인증서 활성화

알 수 없는 기관 오류로 서명된 인증서를 방지하려면 DataProtectionApplication CR(사용자 정의 리소스) 매니페스트를 편집하여 개체 스토리지에 대해 자체 서명된 CA 인증서를 활성화해야 합니다.

사전 요구 사항

- OADP(OpenShift API for Data Protection) Operator가 설치되어 있어야 합니다.

절차

- DataProtectionApplication CR 매니페스트의 spec.backupLocations.velero.objectStorage.caCert 매개변수 및 spec.backupLocations.velero.config 매개변수를 편집합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket>
          prefix: <prefix>
          caCert: <base64_encoded_cert_string> 1
        config:
          insecureSkipTLSVerify: "false" 2
...
```

1 Base46 인코딩 CA 인증서 문자열을 지정합니다.

2 insecureSkipTLSVerify 구성은 "true" 또는 "false" 로 설정할 수 있습니다. "true" 로 설정하면 SSL/TLS 보안이 비활성화됩니다. "false" 로 설정하면 SSL/TLS 보안이 활성화됩니다.

4.3.3.5. 데이터 보호 애플리케이션 설치

Data ProtectionApplication API 인스턴스를 만들어 DPA(Data Protection Application)를 설치합니다.

사전 요구 사항

- OADP Operator를 설치해야 합니다.
- 오브젝트 스토리지를 백업 위치로 구성해야 합니다.
- 스냅샷을 사용하여 PV를 백업하는 경우 클라우드 공급자는 기본 스냅샷 API 또는 CSI(Container Storage Interface) 스냅샷을 지원해야 합니다.
-

백업 및 스냅샷 위치가 동일한 자격 증명을 사용하는 경우 기본 이름 **cloud-credentials-azure** 를 사용하여 **Secret** 을 생성해야 합니다.

- 백업 및 스냅샷 위치가 다른 인증 정보를 사용하는 경우 두 개의 시크릿 을 생성해야 합니다.
  - 백업 위치에 대한 사용자 지정 이름이 있는 시크릿 입니다. 이 보안을 **DataProtectionApplication CR**에 추가합니다.
  - 스냅샷 위치에 대한 기본 이름 **cloud-credentials-azure** 가 있는 시크릿 입니다. 이 보안 은 **DataProtectionApplication CR**에서 참조되지 않습니다.

참고

설치 중에 백업 또는 스냅샷 위치를 지정하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 보안을 생성할 수 있습니다. 기본 보안이 없으면 설치에 실패합니다.

절차

1. **Operators** → 설치된 **Operator** 를 클릭하고 **OADP Operator**를 선택합니다.
2. 제공된 **API** 아래의 **DataProtectionApplication** 상자에서 인스턴스 생성을 클릭합니다.
3. **YAML** 보기를 클릭하고 **DataProtectionApplication** 매니페스트의 매개변수를 업데이트합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - azure
        - openshift 1
    restic:
      enable: true 2
      podConfig:
```

```
    nodeSelector: <node selector> 3
backupLocations:
 - velero:
    config:
     resourceGroup: <azure_resource_group> 4
     storageAccount: <azure_storage_account_id> 5
     subscriptionId: <azure_subscription_id> 6
     storageAccountKeyEnvVar: AZURE_STORAGE_ACCOUNT_ACCESS_KEY
    credential:
     key: cloud
     name: cloud-credentials-azure 7
    provider: azure
    default: true
    objectStorage:
     bucket: <bucket_name> 8
     prefix: <prefix> 9
snapshotLocations: 10
 - velero:
    config:
     resourceGroup: <azure_resource_group>
     subscriptionId: <azure_subscription_id>
     incremental: "true"
    name: default
    provider: azure
```

1

openshift 플러그인은 필수입니다.

2

Restic 설치를 비활성화하려면 false 로 설정합니다. Restic은 데몬 세트를 배포합니다. 즉, 각 작업자 노드에 Restic pod가 실행됩니다. Backup CR에 spec.defaultVolumesToRestic: true 를 추가하여 백업에 Restic을 구성합니다.

3

Restic podSpec에 제공할 노드 선택기를 지정합니다.

4

Azure 리소스 그룹을 지정합니다.

5

Azure 스토리지 계정 ID를 지정합니다.

6

Azure 서브스크립션 **ID**를 지정합니다.

7

이 값을 지정하지 않으면 기본 이름 **cloud-credentials-azure** 가 사용됩니다. 사용자 지정 이름을 지정하면 백업 위치에 사용자 지정 이름이 사용됩니다.

8

버킷을 백업 스토리지 위치로 지정합니다. 버킷이 **Velero** 백업용 전용 버킷이 아닌 경우 접두사를 지정해야 합니다.

9

버킷이 여러 용도로 사용되는 경우 **Velero** 백업의 접두사(예: **velero**)를 지정합니다.

10

PV를 백업하기 위해 **CSI** 스냅샷 또는 **Restic**을 사용하는 경우 스냅샷 위치를 지정할 필요가 없습니다.

4. 생성을 클릭합니다.

5. **OADP** 리소스를 확인하여 설치를 확인합니다.

```
$ oc get all -n openshift-adp
```

출력 예

```
NAME                                                 READY   STATUS    RESTARTS   AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8   2/2     Running   0          2m8s
pod/restic-9cq4q                                     1/1     Running   0          94s
pod/restic-m4lts                                     1/1     Running   0          94s
pod/restic-pv4kr                                     1/1     Running   0          95s
pod/velero-588db7f655-n842v                          1/1     Running   0          95s

NAME                                                 TYPE        CLUSTER-IP      EXTERNAL-IP
PORT(S)    AGE
service/oadp-operator-controller-manager-metrics-service   ClusterIP   172.30.70.140
<none>        8443/TCP   2m8s

NAME                  DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE
```

```
SELECTOR   AGE
daemonset.apps/restic   3        3        3      3          3         <none>        96s

NAME                                    READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/oadp-operator-controller-manager    1/1     1          1         2m9s
deployment.apps/velero                              1/1     1          1         96s

NAME                                               DESIRED   CURRENT   READY   AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47    1         1         1       2m9s
replicaset.apps/velero-588db7f655                              1         1         1       96s
```

4.3.3.5.1. DataProtectionApplication CR에서 CSI 활성화

CSI 스냅샷으로 영구 볼륨을 백업하기 위해 DataProtectionApplication CR(사용자 정의 리소스)에서 CSI(Container Storage Interface)를 활성화합니다.

사전 요구 사항

- 클라우드 공급자는 **CSI** 스냅샷을 지원해야 합니다.

절차

- 다음 예와 같이 **DataProtectionApplication CR**을 편집합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
      - openshift
      - csi (1)
```

(1) csi 기본 플러그인을 추가합니다.

4.3.4. Google Cloud Platform으로 데이터 보호를 위한 OpenShift API 설치 및 구성

OADP Operator를 설치하여 GCP(Google Cloud Platform)에서 OADP(Data Protection)용 OpenShift API를 설치합니다. Operator는 Velero 1.7 을 설치합니다.

참고

OADP 1.0.4부터 모든 OADP 1.0.*z* 버전은 MTC Operator의 종속성으로만 사용할 수 있으며 독립 실행형 Operator로 사용할 수 없습니다.

Velero에 대한 GCP를 구성하고 기본 시크릿 을 생성한 다음 데이터 보호 애플리케이션을 설치합니다.

제한된 네트워크 환경에 OADP Operator를 설치하려면 먼저 기본 OperatorHub 소스를 비활성화하고 Operator 카탈로그를 미러링해야 합니다. 자세한 내용은 제한된 네트워크에서 Operator Lifecycle Manager 사용을 참조하십시오.

4.3.4.1. OADP Operator 설치

OLM(Operator Lifecycle Manager)을 사용하여 OpenShift Container Platform 4.10에 OADP(OpenShift API for Data Protection) Operator를 설치합니다.

OADP Operator는 Velero 1.7 을 설치합니다.

사전 요구 사항

- cluster-admin 권한이 있는 사용자로 로그인해야 합니다.

절차

1. OpenShift Container Platform 웹 콘솔에서 Operator → OperatorHub를 클릭합니다.
2. 키워드로 필터링 필드를 사용하여 OADP Operator 를 찾습니다.
3. OADP Operator 를 선택하고 설치를 클릭합니다.
4. openshift-adp 프로젝트에서 Operator를 설치하려면 설치를 클릭합니다.

5. **Operators** → 설치된 **Operators** 를 클릭하여 설치를 확인합니다.

4.3.4.2. GCP(Google Cloud Platform) 구성

OADP(OpenShift API for Data Protection)에 대해 GCP(Google Cloud Platform)를 구성합니다.

사전 요구 사항

- **gcloud** 및 **gsutil CLI** 툴이 설치되어 있어야 합니다. 자세한 내용은 **Google** 클라우드 설명서를 참조하십시오.

절차

1. **GCP**에 로그인합니다.

```
$ gcloud auth login
```

2. **BUCKET** 변수를 설정합니다.

```
$ BUCKET=<bucket> 1
```

1 버킷 이름을 지정합니다.

3. 스토리지 버킷을 생성합니다.

```
$ gsutil mb gs://$BUCKET/
```

4. **PROJECT_ID** 변수를 활성 프로젝트로 설정합니다.

```
$ PROJECT_ID=$(gcloud config get-value project)
```

5. 서비스 계정을 생성합니다.

```
$ gcloud iam service-accounts create velero \
    --display-name "Velero service account"
```

6. 서비스 계정을 나열합니다.

```
$ gcloud iam service-accounts list
```

7. email 값과 일치하도록 SERVICE_ACCOUNT_EMAIL 변수를 설정합니다.

```
$ SERVICE_ACCOUNT_EMAIL=$(gcloud iam service-accounts list \
    --filter="displayName:Velero service account" \
    --format 'value(email)')
```

8. 정책을 연결하여 velero 사용자에게 필요한 최소 권한을 부여합니다.

```
$ ROLE_PERMISSIONS=(
    compute.disks.get
    compute.disks.create
    compute.disks.createSnapshot
    compute.snapshots.get
    compute.snapshots.create
    compute.snapshots.useReadOnly
    compute.snapshots.delete
    compute.zones.get
)
```

9. velero.server 사용자 정의 역할을 생성합니다.

```
$ gcloud iam roles create velero.server \
    --project $PROJECT_ID \
    --title "Velero Server" \
    --permissions "$(IFS=","; echo "${ROLE_PERMISSIONS[*]}")"
```

10. 프로젝트에 IAM 정책 바인딩을 추가합니다.

```
$ gcloud projects add-iam-policy-binding $PROJECT_ID \
    --member serviceAccount:$SERVICE_ACCOUNT_EMAIL \
    --role projects/$PROJECT_ID/roles/velero.server
```

11. IAM 서비스 계정을 업데이트합니다.

```
$ gsutil iam ch serviceAccount:$SERVICE_ACCOUNT_EMAIL:objectAdmin
gs://${BUCKET}
```

12. IAM 서비스 계정 키를 현재 디렉터리의 credentials-velero 파일에 저장합니다.

```
$ gcloud iam service-accounts keys create credentials-velero \
  --iam-account $SERVICE_ACCOUNT_EMAIL
```

credentials-velero 파일을 사용하여 데이터 보호 애플리케이션을 설치하기 전에 GCP용 Secret 오브젝트를 생성합니다.

4.3.4.3. 백업 및 스냅샷 위치 및 보안 정보

DataProtectionApplication CR(사용자 정의 리소스)에서 백업 및 스냅샷 위치 및 해당 시크릿을 지정합니다.

백업 위치

Multicloud Object Gateway, Noobaa 또는 Minio와 같은 S3 호환 개체 스토리지를 백업 위치로 지정합니다.

Velero는 OpenShift Container Platform 리소스, Kubernetes 오브젝트 및 내부 이미지를 오브젝트 스토리지의 아카이브 파일로 백업합니다.

스냅샷 위치

영구 볼륨을 백업하기 위해 클라우드 공급자의 네이티브 스냅샷 API를 사용하는 경우 클라우드 공급자를 스냅샷 위치로 지정해야 합니다.

CSI(Container Storage Interface) 스냅샷을 사용하는 경우 CSI 드라이버를 등록하기 위해 VolumeSnapshotClass CR을 생성하기 때문에 스냅샷 위치를 지정할 필요가 없습니다.

Restic을 사용하는 경우 Restic이 개체 스토리지의 파일 시스템을 백업하므로 스냅샷 위치를 지정할 필요가 없습니다.

보안

백업 및 스냅샷 위치가 동일한 자격 증명을 사용하거나 스냅샷 위치가 필요하지 않은 경우 기본 시크릿 을 생성합니다.

백업 및 스냅샷 위치가 다른 인증 정보를 사용하는 경우 두 개의 시크릿 오브젝트를 생성합니다.

- **DataProtectionApplication CR**에서 지정하는 백업 위치에 대한 사용자 정의 시크릿 입니다.
- **DataProtectionApplication CR**에서 참조되지 않는 스냅샷 위치에 대한 기본 시크릿 입니다.

중요

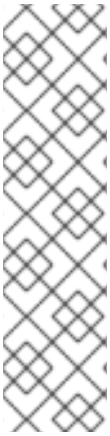
데이터 보호 애플리케이션에는 기본 보안이 필요합니다. 그렇지 않으면 설치에 실패합니다.

설치 중에 백업 또는 스냅샷 위치를 지정하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 보안을 생성할 수 있습니다.

4.3.4.3.1. 기본 보안 생성

백업 및 스냅샷 위치가 동일한 자격 증명을 사용하거나 스냅샷 위치가 필요하지 않은 경우 기본 보안을 생성합니다.

Secret 의 기본 이름은 **cloud-credentials-gcp** 입니다.



참고

DataProtectionApplication CR(사용자 정의 리소스)에는 기본 시크릿 이 필요합니다. 그렇지 않으면 설치에 실패합니다. 백업 위치 **Secret** 의 이름이 지정되지 않은 경우 기본 이름이 사용됩니다.

설치 중에 백업 위치 자격 증명을 사용하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 이름으로 **Secret** 을 생성할 수 있습니다.

사전 요구 사항

- 오브젝트 스토리지 및 클라우드 스토리지는 동일한 인증 정보를 사용해야 합니다.

- Velero에 대한 오브젝트 스토리지를 구성해야 합니다.
- 오브젝트 스토리지의 credentials-velero 파일을 적절한 형식으로 만들어야 합니다.

절차

- 기본 이름으로 보안을 생성합니다.

```
$ oc create secret generic cloud-credentials-gcp -n openshift-adp --from-file
cloud=credentials-velero
```

Secret 은 데이터 보호 애플리케이션을 설치할 때 DataProtectionApplication CR의 spec.backupLocations.credential 블록에서 참조합니다.

4.3.4.3.2. 다른 인증 정보의 보안 생성

백업 및 스냅샷 위치에 다른 인증 정보를 사용하는 경우 두 개의 Secret 오브젝트를 생성해야 합니다.

- 사용자 지정 이름을 사용한 백업 위치 시크릿. 사용자 정의 이름은 DataProtectionApplication CR(사용자 정의 리소스)의 spec.backupLocations 블록에 지정됩니다.
- 기본 이름 cloud-credentials-gcp 를 사용한 스냅샷 위치 시크릿. 이 Secret 은 DataProtectionApplication CR에 지정되지 않습니다.

절차

1. 클라우드 공급자에 적합한 형식으로 스냅샷 위치에 대한 자격 증명-velero 파일을 만듭니다.
2. 기본 이름으로 스냅샷 위치에 대한 Secret 을 생성합니다.

```
$ oc create secret generic cloud-credentials-gcp -n openshift-adp --from-file
cloud=credentials-velero
```

3. 오브젝트 스토리지의 적절한 형식으로 백업 위치에 대한 credentials-velero 파일을 만듭니

다.

4. 사용자 정의 이름으로 백업 위치에 대한 보안을 생성합니다.

```
$ oc create secret generic <custom_secret> -n openshift-adp --from-file
cloud=credentials-velero
```

5. 다음 예와 같이 사용자 지정 이름이 있는 **Secret** 을 **DataProtectionApplication CR**에 추가합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
...
  backupLocations:
    - velero:
        provider: gcp
        default: true
        credential:
          key: cloud
          name: <custom_secret> 1
        objectStorage:
          bucket: <bucket_name>
          prefix: <prefix>
  snapshotLocations:
    - velero:
        provider: gcp
        default: true
        config:
          project: <project>
          snapshotLocation: us-west1
```

1 사용자 정의 이름으로 백업 위치 시크릿.

4.3.4.4. 데이터 보호 애플리케이션 구성

Velero 리소스 할당을 설정하거나 자체 서명된 **CA** 인증서를 활성화하여 데이터 보호 애플리케이션을 구성할 수 있습니다.

4.3.4.4.1. Velero CPU 및 메모리 리소스 할당 설정

DataProtectionApplication CR(사용자 정의 리소스) 매니페스트를 편집하여 Velero Pod의 CPU 및 메모리 리소스 할당을 설정합니다.

사전 요구 사항

- OADP(OpenShift API for Data Protection) Operator가 설치되어 있어야 합니다.

절차

- 다음 예와 같이 DataProtectionApplication CR 매니페스트의 spec.configuration.velero.podConfig.ResourceAllocations 블록에서 값을 편집합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node selector> 1
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 512Mi
          requests:
            cpu: 500m
            memory: 256Mi
```

1 Velero podSpec에 제공할 노드 선택기 지정

4.3.4.4.2. 자체 서명 CA 인증서 활성화

알 수 없는 기관 오류로 서명된 인증서를 방지하려면 DataProtectionApplication CR(사용자 정의 리소스) 매니페스트를 편집하여 개체 스토리지에 대해 자체 서명된 CA 인증서를 활성화해야 합니다.

사전 요구 사항

- OADP(OpenShift API for Data Protection) Operator가 설치되어 있어야 합니다.

절차

- DataProtectionApplication CR 매니페스트의 spec.backupLocations.velero.objectStorage.caCert 매개변수 및 spec.backupLocations.velero.config 매개변수를 편집합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket>
          prefix: <prefix>
          caCert: <base64_encoded_cert_string> 1
        config:
          insecureSkipTLSVerify: "false" 2
...
```

1. Base46 인코딩 CA 인증서 문자열을 지정합니다.

2. insecureSkipTLSVerify 구성은 "true" 또는 "false" 로 설정할 수 있습니다. "true" 로 설정하면 SSL/TLS 보안이 비활성화됩니다. "false" 로 설정하면 SSL/TLS 보안이 활성화됩니다.

4.3.4.5. 데이터 보호 애플리케이션 설치

Data ProtectionApplication API 인스턴스를 만들어 DPA(Data Protection Application)를 설치합니다.

사전 요구 사항

- OADP Operator를 설치해야 합니다.

- 오브젝트 스토리지를 백업 위치로 구성해야 합니다.

- 스냅샷을 사용하여 **PV**를 백업하는 경우 클라우드 공급자는 기본 스냅샷 **API** 또는 **CSI(Container Storage Interface)** 스냅샷을 지원해야 합니다.

- 백업 및 스냅샷 위치가 동일한 자격 증명을 사용하는 경우 기본 이름 **cloud-credentials-gcp**를 사용하여 **Secret** 을 생성해야 합니다.

- 백업 및 스냅샷 위치가 다른 인증 정보를 사용하는 경우 두 개의 시크릿 을 생성해야 합니다.

  - 백업 위치에 대한 사용자 지정 이름이 있는 시크릿 입니다. 이 보안을 **DataProtectionApplication CR**에 추가합니다.

  - 스냅샷 위치에 대한 기본 이름 **cloud-credentials-gcp** 가 있는 시크릿 입니다. 이 보안은 **DataProtectionApplication CR**에서 참조되지 않습니다.

참고

설치 중에 백업 또는 스냅샷 위치를 지정하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 보안을 생성할 수 있습니다. 기본 보안이 없으면 설치에 실패합니다.

절차

1. **Operators → 설치된 Operator** 를 클릭하고 **OADP Operator**를 선택합니다.

2. 제공된 **API** 아래의 **DataProtectionApplication** 상자에서 인스턴스 생성을 클릭합니다.

3. **YAML** 보기를 클릭하고 **DataProtectionApplication** 매니페스트의 매개변수를 업데이트합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
```

```
spec:
 configuration:
  velero:
   defaultPlugins:
    - gcp
    - openshift (1)
  restic:
   enable: true (2)
   podConfig:
    nodeSelector: <node selector> (3)
 backupLocations:
  - velero:
     provider: gcp
     default: true
     credential:
      key: cloud
      name: cloud-credentials-gcp (4)
     objectStorage:
      bucket: <bucket_name> (5)
      prefix: <prefix> (6)
 snapshotLocations: (7)
  - velero:
     provider: gcp
     default: true
     config:
      project: <project>
      snapshotLocation: us-west1 (8)
```

(1) **openshift** 플러그인은 필수입니다.

(2) **Restic** 설치를 비활성화하려면 **false** 로 설정합니다. **Restic**은 데몬 세트를 배포합니다. 즉, 각 작업자 노드에 **Restic pod**가 실행됩니다. **Backup CR**에 **spec.defaultVolumesToRestic: true** 를 추가하여 백업에 **Restic**을 구성합니다.

(3) **Restic podSpec**에 제공할 노드 선택기를 지정합니다.

(4) 이 값을 지정하지 않으면 기본 이름 **cloud-credentials-gcp** 가 사용됩니다. 사용자 지정 이름을 지정하면 백업 위치에 사용자 지정 이름이 사용됩니다.

(5) 버킷을 백업 스토리지 위치로 지정합니다. 버킷이 **Velero** 백업용 전용 버킷이 아닌 경우 접두사를 지정해야 합니다.

6

버킷이 여러 용도로 사용되는 경우 **Velero** 백업의 접두사(예: **velero**)를 지정합니다.

7

PV를 백업하기 위해 **CSI** 스냅샷 또는 **Restic**을 사용하는 경우 스냅샷 위치를 지정할 필요가 없습니다.

8

스냅샷 위치는 **PV**와 동일한 리전에 있어야 합니다.

4. 생성을 클릭합니다.

5. **OADP** 리소스를 확인하여 설치를 확인합니다.

```
$ oc get all -n openshift-adp
```

출력 예

```
NAME                                                READY   STATUS    RESTARTS   AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8   2/2     Running   0          2m8s
pod/restic-9cq4q                                    1/1     Running   0          94s
pod/restic-m4lts                                    1/1     Running   0          94s
pod/restic-pv4kr                                    1/1     Running   0          95s
pod/velero-588db7f655-n842v                         1/1     Running   0          95s

NAME                                                     TYPE        CLUSTER-IP       EXTERNAL-IP
PORT(S)    AGE
service/oadp-operator-controller-manager-metrics-service   ClusterIP   172.30.70.140
<none>        8443/TCP   2m8s

NAME                    DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE
SELECTOR   AGE
daemonset.apps/restic   3         3         3       3            3           <none>          96s

NAME                                          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/oadp-operator-controller-manager   1/1     1            1           2m9s
deployment.apps/velero                        1/1     1            1           96s
```

```
NAME                                                 DESIRED   CURRENT   READY   AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47   1         1         1       2m9s
replicaset.apps/velero-588db7f655                    1         1         1       96s
```

4.3.4.5.1. DataProtectionApplication CR에서 CSI 활성화

CSI 스냅샷으로 영구 볼륨을 백업하기 위해 DataProtectionApplication CR(사용자 정의 리소스)에서 CSI(Container Storage Interface)를 활성화합니다.

사전 요구 사항

- 클라우드 공급자는 **CSI** 스냅샷을 지원해야 합니다.

절차

- 다음 예와 같이 **DataProtectionApplication CR**을 편집합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
      - openshift
      - csi (1)
```

(1) csi 기본 플러그인을 추가합니다.

4.3.5. Multicloud Object Gateway를 사용하여 데이터 보호를 위한 OpenShift API 설치 및 구성

OADP Operator를 설치하여 MCG(Multicloud Object Gateway)를 사용하여 OADP(Data Protection)용 OpenShift API를 설치합니다. Operator는 Velero 1.7 을 설치합니다.

참고

OADP 1.0.4부터 모든 **OADP 1.0.*z*** 버전은 **MTC Operator**의 종속성으로만 사용할 수 있으며 독립 실행형 **Operator**로 사용할 수 없습니다.

Multicloud Object Gateway 를 백업 위치로 구성합니다. **MCG**는 **OpenShift Data Foundation**의 구성 요소입니다. **DataProtectionApplication CR**(사용자 정의 리소스)에서 **MCG**를 백업 위치로 구성합니다.

중요

개체 스토리지의 버킷 생성을 자동화하는 **CloudStorage API**는 기술 프리뷰 기능 전용입니다. 기술 프리뷰 기능은 **Red Hat** 프로덕션 서비스 수준 계약(**SLA**)에서 지원되지 않으며 기능적으로 완전하지 않을 수 있습니다. 따라서 프로덕션 환경에서 사용하는 것은 권장하지 않습니다. 이러한 기능을 사용하면 향후 제품 기능을 조기에 이용할 수 있어 개발 과정에서 고객이 기능을 테스트하고 피드백을 제공할 수 있습니다.

Red Hat 기술 프리뷰 기능의 지원 범위에 대한 자세한 내용은 기술 프리뷰 기능 지원 범위를 참조하십시오.

백업 위치에 대한 시크릿 을 생성한 다음 데이터 보호 애플리케이션을 설치합니다.

제한된 네트워크 환경에 **OADP Operator**를 설치하려면 먼저 기본 **OperatorHub** 소스를 비활성화하고 **Operator** 카탈로그를 미러링해야 합니다. 자세한 내용은 제한된 네트워크에서 **Operator Lifecycle Manager** 사용을 참조하십시오.

4.3.5.1. OADP Operator 설치

OLM(Operator Lifecycle Manager)을 사용하여 **OpenShift Container Platform 4.10**에 **OADP(OpenShift API for Data Protection) Operator**를 설치합니다.

OADP Operator는 Velero 1.7 을 설치합니다.

사전 요구 사항

- **cluster-admin** 권한이 있는 사용자로 로그인해야 합니다.

절차

1. OpenShift Container Platform 웹 콘솔에서 Operator → OperatorHub를 클릭합니다.
2. 키워드로 필터링 필드를 사용하여 OADP Operator 를 찾습니다.
3. OADP Operator 를 선택하고 설치를 클릭합니다.
4. openshift-adp 프로젝트에서 Operator를 설치하려면 설치를 클릭합니다.
5. Operators → 설치된 Operators 를 클릭하여 설치를 확인합니다.

4.3.5.2. 멀티 클라우드 오브젝트 게이트웨이 인증 정보 검색

OADP(데이터 보호)용 OpenShift API에 대한 Secret CR(사용자 정의 리소스)을 생성하려면 MCG(Multicloud Object Gateway) 인증 정보를 검색해야 합니다.

MCG는 OpenShift Data Foundation의 구성 요소입니다.

사전 요구 사항

- 적절한 OpenShift Data Foundation 배포 가이드를 사용하여 OpenShift Data Foundation 을 배포 해야 합니다.

절차

1. NooBaa 사용자 정의 리소스에서 describe 명령을 실행하여 S3 끝점, AWS_ACCESS_ID 및 AWS_SECRET_ACCESS_KEY 를 가져옵니다.
2. credentials-velero 파일을 만듭니다.

```
$ cat << EOF > ./credentials-velero
[default]
aws_access_key_id=<AWS_ACCESS_KEY_ID>
aws_secret_access_key=<AWS_SECRET_ACCESS_KEY>
EOF
```

credentials-velero 파일을 사용하여 데이터 보호 애플리케이션을 설치할 때 Secret 오브젝트를 생성합니다.

4.3.5.3. 백업 및 스냅샷 위치 및 보안 정보

DataProtectionApplication CR(사용자 정의 리소스)에서 백업 및 스냅샷 위치 및 해당 시크릿을 지정합니다.

백업 위치

Multicloud Object Gateway, Noobaa 또는 Minio와 같은 S3 호환 개체 스토리지를 백업 위치로 지정합니다.

Velero는 OpenShift Container Platform 리소스, Kubernetes 오브젝트 및 내부 이미지를 오브젝트 스토리지의 아카이브 파일로 백업합니다.

스냅샷 위치

영구 볼륨을 백업하기 위해 클라우드 공급자의 네이티브 스냅샷 API를 사용하는 경우 클라우드 공급자를 스냅샷 위치로 지정해야 합니다.

CSI(Container Storage Interface) 스냅샷을 사용하는 경우 CSI 드라이버를 등록하기 위해 VolumeSnapshotClass CR을 생성하기 때문에 스냅샷 위치를 지정할 필요가 없습니다.

Restic을 사용하는 경우 Restic이 개체 스토리지의 파일 시스템을 백업하므로 스냅샷 위치를 지정할 필요가 없습니다.

보안

백업 및 스냅샷 위치가 동일한 자격 증명을 사용하거나 스냅샷 위치가 필요하지 않은 경우 기본 시크릿 을 생성합니다.

백업 및 스냅샷 위치가 다른 인증 정보를 사용하는 경우 두 개의 시크릿 오브젝트를 생성합니다.

- DataProtectionApplication CR에서 지정하는 백업 위치에 대한 사용자 정의 시크릿 입니다.
- DataProtectionApplication CR에서 참조되지 않는 스냅샷 위치에 대한 기본 시크릿 입니

다.

중요

데이터 보호 애플리케이션에는 기본 보안이 필요합니다. 그렇지 않으면 설치에 실패합니다.

설치 중에 백업 또는 스냅샷 위치를 지정하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 보안을 생성할 수 있습니다.

4.3.5.3.1. 기본 보안 생성

백업 및 스냅샷 위치가 동일한 자격 증명을 사용하거나 스냅샷 위치가 필요하지 않은 경우 기본 보안을 생성합니다.

Secret 의 기본 이름은 **cloud-credentials** 입니다.

참고

DataProtectionApplication CR(사용자 정의 리소스)에는 기본 시크릿 이 필요합니다. 그렇지 않으면 설치에 실패합니다. 백업 위치 **Secret** 의 이름이 지정되지 않은 경우 기본 이름이 사용됩니다.

설치 중에 백업 위치 자격 증명을 사용하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 이름으로 **Secret** 을 생성할 수 있습니다.

사전 요구 사항

- 오브젝트 스토리지 및 클라우드 스토리지는 동일한 인증 정보를 사용해야 합니다.
- **Velero**에 대한 오브젝트 스토리지를 구성해야 합니다.
- 오브젝트 스토리지의 **credentials-velero** 파일을 적절한 형식으로 만들어야 합니다.

절차

- 기본 이름으로 보안을 생성합니다.

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file
cloud=credentials-velero
```

Secret 은 데이터 보호 애플리케이션을 설치할 때 DataProtectionApplication CR의 spec.backupLocations.credential 블록에서 참조합니다.

4.3.5.3.2. 다른 인증 정보의 보안 생성

백업 및 스냅샷 위치에 다른 인증 정보를 사용하는 경우 두 개의 Secret 오브젝트를 생성해야 합니다.

- 사용자 지정 이름을 사용한 백업 위치 시크릿. 사용자 정의 이름은 DataProtectionApplication CR(사용자 정의 리소스)의 spec.backupLocations 블록에 지정됩니다.
- 기본 이름 cloud-credentials 가 있는 스냅샷 위치 시크릿. 이 Secret 은 DataProtectionApplication CR에 지정되지 않습니다.

절차

1. 클라우드 공급자에 적합한 형식으로 스냅샷 위치에 대한 자격 증명-velero 파일을 만듭니다.
2. 기본 이름으로 스냅샷 위치에 대한 Secret 을 생성합니다.

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file
cloud=credentials-velero
```

3. 오브젝트 스토리지의 적절한 형식으로 백업 위치에 대한 credentials-velero 파일을 만듭니다.
4. 사용자 정의 이름으로 백업 위치에 대한 보안을 생성합니다.

```
$ oc create secret generic <custom_secret> -n openshift-adp --from-file
cloud=credentials-velero
```

5. 다음 예와 같이 사용자 지정 이름이 있는 **Secret** 을 **DataProtectionApplication CR**에 추가합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
...
  backupLocations:
    - velero:
        config:
          profile: "default"
          region: minio
          s3Url: <url>
          insecureSkipTLSVerify: "true"
          s3ForcePathStyle: "true"
        provider: aws
        default: true
        credential:
          key: cloud
          name:  <custom_secret> (1)
        objectStorage:
          bucket: <bucket_name>
          prefix: <prefix>
```

(1) 사용자 정의 이름으로 백업 위치 시크릿.

4.3.5.4. 데이터 보호 애플리케이션 구성

Velero 리소스 할당을 설정하거나 자체 서명된 **CA** 인증서를 활성화하여 데이터 보호 애플리케이션을 구성할 수 있습니다.

4.3.5.4.1. Velero CPU 및 메모리 리소스 할당 설정

DataProtectionApplication CR(사용자 정의 리소스) 매니페스트를 편집하여 **Velero Pod**의 **CPU** 및 메모리 리소스 할당을 설정합니다.

사전 요구 사항

- **OADP(OpenShift API for Data Protection) Operator**가 설치되어 있어야 합니다.

절차

- 다음 예와 같이 **DataProtectionApplication CR** 매니페스트의 **spec.configuration.velero.podConfig.ResourceAllocations** 블록에서 값을 편집합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node selector> 1
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 512Mi
          requests:
            cpu: 500m
            memory: 256Mi
```

1 Velero podSpec에 제공할 노드 선택기 지정

4.3.5.4.2. 자체 서명 CA 인증서 활성화

알 수 없는 기관 오류로 서명된 인증서를 방지하려면 **DataProtectionApplication CR**(사용자 정의 리소스) 매니페스트를 편집하여 개체 스토리지에 대해 자체 서명된 **CA** 인증서를 활성화해야 합니다.

사전 요구 사항

- **OADP(OpenShift API for Data Protection) Operator**가 설치되어 있어야 합니다.

절차

- **DataProtectionApplication CR** 매니페스트의 **spec.backupLocations.velero.objectStorage.caCert** 매개변수 및 **spec.backupLocations.velero.config** 매개변수를 편집합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
```

```
  name: <dpa_sample>
spec:
...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket>
          prefix: <prefix>
          caCert: <base64_encoded_cert_string> 1
        config:
          insecureSkipTLSVerify: "false" 2
...
```

1

Base46 인코딩 CA 인증서 문자열을 지정합니다.

2

insecureSkipTLSVerify 구성은 "true" 또는 "false" 로 설정할 수 있습니다. "true" 로 설정하면 SSL/TLS 보안이 비활성화됩니다. "false" 로 설정하면 SSL/TLS 보안이 활성화됩니다.

4.3.5.5. 데이터 보호 애플리케이션 설치

Data ProtectionApplication API 인스턴스를 만들어 DPA(Data Protection Application)를 설치합니다.

사전 요구 사항

- OADP Operator를 설치해야 합니다.
- 오브젝트 스토리지를 백업 위치로 구성해야 합니다.
- 스냅샷을 사용하여 PV를 백업하는 경우 클라우드 공급자는 기본 스냅샷 API 또는 CSI(Container Storage Interface) 스냅샷을 지원해야 합니다.
- 백업 및 스냅샷 위치가 동일한 자격 증명을 사용하는 경우 기본 이름 cloud-credentials 를 사용하여 Secret 을 생성해야 합니다.

- 백업 및 스냅샷 위치가 다른 인증 정보를 사용하는 경우 두 개의 시크릿 을 생성해야 합니다.
  - 백업 위치에 대한 사용자 지정 이름이 있는 시크릿 입니다. 이 보안을 **DataProtectionApplication CR**에 추가합니다.
  - 스냅샷 위치에 대한 기본 이름 **cloud-credentials** 가 있는 시크릿 입니다. 이 보안 은 **DataProtectionApplication CR**에서 참조되지 않습니다.

참고

설치 중에 백업 또는 스냅샷 위치를 지정하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 보안을 생성할 수 있습니다. 기본 보안이 없으면 설치에 실패합니다.

절차

1. **Operators** → 설치된 **Operator** 를 클릭하고 **OADP Operator**를 선택합니다.
2. 제공된 **API** 아래의 **DataProtectionApplication** 상자에서 인스턴스 생성을 클릭합니다.
3. **YAML** 보기를 클릭하고 **DataProtectionApplication** 매니페스트의 매개변수를 업데이트합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - aws
        - openshift (1)
    restic:
      enable: true (2)
      podConfig:
        nodeSelector: <node selector> (3)
  backupLocations:
    - velero:
        config:
```

```
    profile: "default"
    region: minio
    s3Url: <url> 4
    insecureSkipTLSVerify: "true"
    s3ForcePathStyle: "true"
  provider: aws
  default: true
  credential:
    key: cloud
    name: cloud-credentials 5
  objectStorage:
    bucket: <bucket_name> 6
    prefix: <prefix> 7
```

1

openshift 플러그인은 필수입니다.

2

Restic 설치를 비활성화하려면 false 로 설정합니다. Restic은 데몬 세트를 배포합니다. 즉, 각 작업자 노드에 Restic pod가 실행됩니다. Backup CR에 spec.defaultVolumesToRestic: true 를 추가하여 백업에 Restic을 구성합니다.

3

Restic podSpec에 제공할 노드 선택기를 지정합니다.

4

S3 끝점의 URL을 지정합니다.

5

이 값을 지정하지 않으면 기본 이름 cloud-credentials 가 사용됩니다. 사용자 지정 이름을 지정하면 백업 위치에 사용자 지정 이름이 사용됩니다.

6

버킷을 백업 스토리지 위치로 지정합니다. 버킷이 Velero 백업용 전용 버킷이 아닌 경우 접두사를 지정해야 합니다.

7

버킷이 여러 용도로 사용되는 경우 Velero 백업의 접두사(예: velero)를 지정합니다.

4. 생성을 클릭합니다.

5. OADP 리소스를 확인하여 설치를 확인합니다.

```
$ oc get all -n openshift-adp
```

출력 예

```
NAME                                                 READY   STATUS    RESTARTS   AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8   2/2     Running   0          2m8s
pod/restic-9cq4q                                     1/1     Running   0          94s
pod/restic-m4lts                                     1/1     Running   0          94s
pod/restic-pv4kr                                     1/1     Running   0          95s
pod/velero-588db7f655-n842v                          1/1     Running   0          95s

NAME                                                     TYPE        CLUSTER-IP       EXTERNAL-IP
PORT(S)    AGE
service/oadp-operator-controller-manager-metrics-service   ClusterIP   172.30.70.140
<none>        8443/TCP   2m8s

NAME                DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE
SELECTOR   AGE
daemonset.apps/restic   3         3         3       3            3           <none>          96s

NAME                                          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/oadp-operator-controller-manager   1/1     1            1           2m9s
deployment.apps/velero                        1/1     1            1           96s

NAME                                                   DESIRED   CURRENT   READY   AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47   1         1         1       2m9s
replicaset.apps/velero-588db7f655                      1         1         1       96s
```

4.3.5.5.1. DataProtectionApplication CR에서 CSI 활성화

CSI 스냅샷으로 영구 볼륨을 백업하기 위해 DataProtectionApplication CR(사용자 정의 리소스)에서 CSI(Container Storage Interface)를 활성화합니다.

사전 요구 사항

- 클라우드 공급자는 CSI 스냅샷을 지원해야 합니다.

절차

- 다음 예와 같이 **DataProtectionApplication CR**을 편집합니다.

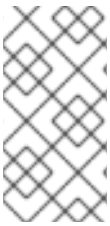
```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
      - openshift
      - csi 1
```

1

csi 기본 플러그인을 추가합니다.

4.3.6. OpenShift Data Foundation으로 OpenShift Data Protection을 위한 OpenShift API 설치 및 구성

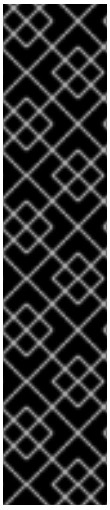
OADP Operator를 설치하고 백업 위치와 스냅샷 위치를 구성하여 **OpenShift Data Foundation**과 함께 **OADP(Data Protection)**용 **OpenShift API**를 설치합니다. 그런 다음 데이터 보호 애플리케이션을 설치합니다.



참고

OADP 1.0.4부터 모든 **OADP 1.0.*z*** 버전은 **MTC Operator**의 종속성으로만 사용할 수 있으며 독립 실행형 **Operator**로 사용할 수 없습니다.

Multicloud Object Gateway 또는 **S3** 호환 개체 스토리지를 백업 위치로 구성할 수 있습니다.



중요

개체 스토리지의 버킷 생성을 자동화하는 **CloudStorage API**는 기술 프리뷰 기능 전용입니다. 기술 프리뷰 기능은 **Red Hat** 프로덕션 서비스 수준 계약(SLA)에서 지원되지 않으며 기능적으로 완전하지 않을 수 있습니다. 따라서 프로덕션 환경에서 사용하는 것은 권장하지 않습니다. 이러한 기능을 사용하면 향후 제품 기능을 조기에 이용할 수 있어 개발 과정에서 고객이 기능을 테스트하고 피드백을 제공할 수 있습니다.

Red Hat 기술 프리뷰 기능의 지원 범위에 대한 자세한 내용은 기술 프리뷰 기능 지원 범위를 참조하십시오.

백업 위치에 대한 시크릿 을 생성한 다음 데이터 보호 애플리케이션을 설치합니다.

제한된 네트워크 환경에 **OADP Operator**를 설치하려면 먼저 기본 **OperatorHub** 소스를 비활성화하고 **Operator** 카탈로그를 미러링해야 합니다. 자세한 내용은 제한된 네트워크에서 **Operator Lifecycle Manager** 사용을 참조하십시오.

4.3.6.1. OADP Operator 설치

OLM(Operator Lifecycle Manager)을 사용하여 **OpenShift Container Platform 4.10**에 **OADP(OpenShift API for Data Protection) Operator**를 설치합니다.

OADP Operator는 **Velero 1.7** 을 설치합니다.

사전 요구 사항

- **cluster-admin** 권한이 있는 사용자로 로그인해야 합니다.

절차

1. **OpenShift Container Platform** 웹 콘솔에서 **Operator → OperatorHub**를 클릭합니다.
2. 키워드로 필터링 필드를 사용하여 **OADP Operator** 를 찾습니다.
3. **OADP Operator** 를 선택하고 설치를 클릭합니다.

4. **openshift-adp** 프로젝트에서 **Operator**를 설치하려면 설치를 클릭합니다.

5. **Operators** → 설치된 **Operators** 를 클릭하여 설치를 확인합니다.

4.3.6.2. 백업 및 스냅샷 위치 및 보안 정보

DataProtectionApplication CR(사용자 정의 리소스)에서 백업 및 스냅샷 위치 및 해당 시크릿을 지정합니다.

백업 위치

Multicloud Object Gateway, Noobaa 또는 **Minio**와 같은 **S3** 호환 개체 스토리지를 백업 위치로 지정합니다.

Velero는 **OpenShift Container Platform** 리소스, **Kubernetes** 오브젝트 및 내부 이미지를 오브젝트 스토리지의 아카이브 파일로 백업합니다.

스냅샷 위치

영구 볼륨을 백업하기 위해 클라우드 공급자의 네이티브 스냅샷 **API**를 사용하는 경우 클라우드 공급자를 스냅샷 위치로 지정해야 합니다.

CSI(Container Storage Interface) 스냅샷을 사용하는 경우 **CSI** 드라이버를 등록하기 위해 **VolumeSnapshotClass CR**을 생성하기 때문에 스냅샷 위치를 지정할 필요가 없습니다.

Restic을 사용하는 경우 **Restic**이 개체 스토리지의 파일 시스템을 백업하므로 스냅샷 위치를 지정할 필요가 없습니다.

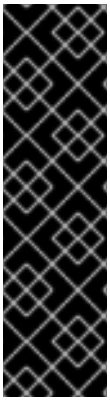
보안

백업 및 스냅샷 위치가 동일한 자격 증명을 사용하거나 스냅샷 위치가 필요하지 않은 경우 기본 시크릿 을 생성합니다.

백업 및 스냅샷 위치가 다른 인증 정보를 사용하는 경우 두 개의 시크릿 오브젝트를 생성합니다.

- **DataProtectionApplication CR**에서 지정하는 백업 위치에 대한 사용자 정의 시크릿 입니다.

- **DataProtectionApplication CR**에서 참조되지 않는 스냅샷 위치에 대한 기본 시크릿 입니다.



중요

데이터 보호 애플리케이션에는 기본 보안이 필요합니다. 그렇지 않으면 설치에 실패합니다.

설치 중에 백업 또는 스냅샷 위치를 지정하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 보안을 생성할 수 있습니다.

4.3.6.2.1. 기본 보안 생성

백업 및 스냅샷 위치가 동일한 자격 증명을 사용하거나 스냅샷 위치가 필요하지 않은 경우 기본 보안을 생성합니다.

백업 스토리지 공급자에 **aws,azure, gcp** 와 같은 기본 플러그인이 없으면 **Secret** 의 기본 이름은 **cloud-credentials** 입니다. 이 경우 기본 이름은 공급자별 **OADP** 설치 절차에 지정됩니다.

참고

DataProtectionApplication CR(사용자 정의 리소스)에는 기본 시크릿 이 필요합니다. 그렇지 않으면 설치에 실패합니다. 백업 위치 **Secret** 의 이름이 지정되지 않은 경우 기본 이름이 사용됩니다.

설치 중에 백업 위치 자격 증명을 사용하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 이름으로 **Secret** 을 생성할 수 있습니다.

사전 요구 사항

- 오브젝트 스토리지 및 클라우드 스토리지는 동일한 인증 정보를 사용해야 합니다.
- **Velero**에 대한 오브젝트 스토리지를 구성해야 합니다.
- 오브젝트 스토리지의 **credentials-velero** 파일을 적절한 형식으로 만들어야 합니다.

절차

- 기본 이름으로 보안을 생성합니다.

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file
cloud=credentials-velero
```

Secret 은 데이터 보호 애플리케이션을 설치할 때 DataProtectionApplication CR의 spec.backupLocations.credential 블록에서 참조합니다.

4.3.6.2.2. 다른 인증 정보의 보안 생성

백업 및 스냅샷 위치에 다른 인증 정보를 사용하는 경우 두 개의 Secret 오브젝트를 생성해야 합니다.

- 사용자 지정 이름을 사용한 백업 위치 시크릿. 사용자 정의 이름은 DataProtectionApplication CR(사용자 정의 리소스)의 spec.backupLocations 블록에 지정됩니다.
- 기본 이름 cloud-credentials 가 있는 스냅샷 위치 시크릿. 이 Secret 은 DataProtectionApplication CR에 지정되지 않습니다.

절차

1. 클라우드 공급자에 적합한 형식으로 스냅샷 위치에 대한 자격 증명-velero 파일을 만듭니다.
2. 기본 이름으로 스냅샷 위치에 대한 Secret 을 생성합니다.

```
$ oc create secret generic cloud-credentials -n openshift-adp --from-file
cloud=credentials-velero
```

3. 오브젝트 스토리지의 적절한 형식으로 백업 위치에 대한 credentials-velero 파일을 만듭니다.
4. 사용자 정의 이름으로 백업 위치에 대한 보안을 생성합니다.

```
$ oc create secret generic <custom_secret> -n openshift-adp --from-file
cloud=credentials-velero
```

5. 다음 예와 같이 사용자 지정 이름이 있는 **Secret** 을 **DataProtectionApplication CR**에 추가합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
...
  backupLocations:
    - velero:
        provider: <provider>
        default: true
        credential:
          key: cloud
          name: <custom_secret> (1)
        objectStorage:
          bucket: <bucket_name>
          prefix: <prefix>
```

(1) 사용자 정의 이름으로 백업 위치 시크릿.

4.3.6.3. 데이터 보호 애플리케이션 구성

Velero 리소스 할당을 설정하거나 자체 서명된 **CA** 인증서를 활성화하여 데이터 보호 애플리케이션을 구성할 수 있습니다.

4.3.6.3.1. Velero CPU 및 메모리 리소스 할당 설정

DataProtectionApplication CR(사용자 정의 리소스) 매니페스트를 편집하여 **Velero Pod**의 **CPU** 및 메모리 리소스 할당을 설정합니다.

사전 요구 사항

- **OADP(OpenShift API for Data Protection) Operator**가 설치되어 있어야 합니다.

절차

- 다음 예와 같이 **DataProtectionApplication CR** 매니페스트의 **spec.configuration.velero.podConfig.ResourceAllocations** 블록에서 값을 편집합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
...
  configuration:
    velero:
      podConfig:
        nodeSelector: <node selector> 1
        resourceAllocations:
          limits:
            cpu: "1"
            memory: 512Mi
          requests:
            cpu: 500m
            memory: 256Mi
```

1 **Velero podSpec**에 제공할 노드 선택기 지정

4.3.6.3.2. 자체 서명 CA 인증서 활성화

알 수 없는 기관 오류로 서명된 인증서를 방지하려면 **DataProtectionApplication CR**(사용자 정의 리소스) 매니페스트를 편집하여 개체 스토리지에 대해 자체 서명된 **CA** 인증서를 활성화해야 합니다.

사전 요구 사항

- **OADP(OpenShift API for Data Protection) Operator**가 설치되어 있어야 합니다.

절차

- **DataProtectionApplication CR** 매니페스트의 **spec.backupLocations.velero.objectStorage.caCert** 매개변수 및 **spec.backupLocations.velero.config** 매개변수를 편집합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
spec:
```

```
...
  backupLocations:
    - name: default
      velero:
        provider: aws
        default: true
        objectStorage:
          bucket: <bucket>
          prefix: <prefix>
          caCert: <base64_encoded_cert_string> 1
        config:
          insecureSkipTLSVerify: "false" 2
...
```

1

Base46 인코딩 CA 인증서 문자열을 지정합니다.

2

insecureSkipTLSVerify 구성은 "true" 또는 "false" 로 설정할 수 있습니다. "true" 로 설정하면 SSL/TLS 보안이 비활성화됩니다. "false" 로 설정하면 SSL/TLS 보안이 활성화됩니다.

4.3.6.4. 데이터 보호 애플리케이션 설치

Data ProtectionApplication API 인스턴스를 만들어 DPA(Data Protection Application)를 설치합니다.

사전 요구 사항

- OADP Operator를 설치해야 합니다.
- 오브젝트 스토리지를 백업 위치로 구성해야 합니다.
- 스냅샷을 사용하여 PV를 백업하는 경우 클라우드 공급자는 기본 스냅샷 API 또는 CSI(Container Storage Interface) 스냅샷을 지원해야 합니다.
- 백업 및 스냅샷 위치가 동일한 자격 증명을 사용하는 경우 기본 이름 cloud-credentials 를 사용하여 Secret 을 생성해야 합니다.

- 백업 및 스냅샷 위치가 다른 인증 정보를 사용하는 경우 두 개의 시크릿 을 생성해야 합니다.
  - 백업 위치에 대한 사용자 지정 이름이 있는 시크릿 입니다. 이 보안을 **DataProtectionApplication CR**에 추가합니다.
  - 스냅샷 위치에 대한 기본 이름 **cloud-credentials** 가 있는 시크릿 입니다. 이 보안 은 **DataProtectionApplication CR**에서 참조되지 않습니다.

참고

설치 중에 백업 또는 스냅샷 위치를 지정하지 않으려면 빈 **credentials-velero** 파일을 사용하여 기본 보안을 생성할 수 있습니다. 기본 보안이 없으면 설치에 실패합니다.

절차

1. **Operators** → 설치된 **Operator** 를 클릭하고 **OADP Operator**를 선택합니다.
2. 제공된 **API** 아래의 **DataProtectionApplication** 상자에서 인스턴스 생성을 클릭합니다.
3. **YAML** 보기를 클릭하고 **DataProtectionApplication** 매니페스트의 매개변수를 업데이트합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: <dpa_sample>
  namespace: openshift-adp
spec:
  configuration:
    velero:
      defaultPlugins:
        - kubevirt (1)
        - gcp (2)
        - csi (3)
        - openshift (4)
    restic:
      enable: true (5)
      podConfig:
        nodeSelector: <node selector> (6)
  backupLocations:
```

```
- velero:
    provider: gcp 7
    default: true
    credential:
      key: cloud
      name: <default_secret> 8
    objectStorage:
      bucket: <bucket_name> 9
      prefix: <prefix> 10
```

1

선택 사항: **kubevirt** 플러그인이 **OpenShift Virtualization**과 함께 사용됩니다.

2

필요한 경우 백업 공급자의 기본 플러그인을 지정합니다(예: **gcp**).

3

CSI 스냅샷을 사용하여 **PV**를 백업하는 경우 **csi** 기본 플러그인을 지정합니다. **csi** 플러그인은 **Velero CSI** 베타 스냅샷 **API** 를 사용합니다. 스냅샷 위치를 구성할 필요가 없습니다.

4

openshift 플러그인은 필수입니다.

5

Restic 설치를 비활성화하려면 **false** 로 설정합니다. **Restic**은 데몬 세트를 배포합니다. 즉, 각 작업자 노드에 **Restic pod**가 실행됩니다. **Backup CR**에 **spec.defaultVolumesToRestic: true** 를 추가하여 백업에 **Restic**을 구성합니다.

6

Restic podSpec에 제공할 노드 선택기 지정

7

백업 공급자를 지정합니다.

8

백업 공급자에 기본 플러그인을 사용하는 경우 **Secret** 에 대해 올바른 기본 이름을 지정해야 합니다(예: **cloud-credentials-gcp**). 사용자 지정 이름을 지정하면 백업 위치에 사용자 지정 이름이 사용됩니다. **Secret** 이름을 지정하지 않으면 기본 이름이 사용됩니다.

9 버킷을 백업 스토리지 위치로 지정합니다. 버킷이 **Velero** 백업용 전용 버킷이 아닌 경우 접두사를 지정해야 합니다.

10 버킷이 여러 용도로 사용되는 경우 **Velero** 백업의 접두사(예: **velero**)를 지정합니다.

4. **생성**을 클릭합니다.

5. **OADP** 리소스를 확인하여 설치를 확인합니다.

```
$ oc get all -n openshift-adp
```

출력 예

```
NAME                                                 READY   STATUS    RESTARTS   AGE
pod/oadp-operator-controller-manager-67d9494d47-6l8z8   2/2     Running   0          2m8s
pod/restic-9cq4q                                     1/1     Running   0          94s
pod/restic-m4lts                                     1/1     Running   0          94s
pod/restic-pv4kr                                     1/1     Running   0          95s
pod/velero-588db7f655-n842v                          1/1     Running   0          95s

NAME                                                     TYPE        CLUSTER-IP       EXTERNAL-IP
PORT(S)    AGE
service/oadp-operator-controller-manager-metrics-service   ClusterIP   172.30.70.140
<none>        8443/TCP   2m8s

NAME                  DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE
SELECTOR   AGE
daemonset.apps/restic   3         3         3       3            3           <none>          96s

NAME                                            READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/oadp-operator-controller-manager   1/1     1            1           2m9s
deployment.apps/velero                             1/1     1            1           96s

NAME                                                       DESIRED   CURRENT   READY   AGE
replicaset.apps/oadp-operator-controller-manager-67d9494d47   1         1         1       2m9s
replicaset.apps/velero-588db7f655                          1         1         1       96s
```

4.3.6.4.1. OpenShift Data Foundation에서 재해 복구를 위해 NooBaa 구성

OpenShift Data Foundation에서 NooBaa 버킷 backupStorageLocation 에 클러스터 스토리지를 사용하는 경우 NooBaa를 외부 오브젝트 저장소로 구성합니다.

주의

NooBaa를 외부 오브젝트 저장소로 구성하지 못하면 백업을 사용할 수 없을 수 있습니다.

절차

- 하이브리드 또는 Multicloud에 대한 스토리지 리소스 추가 에 설명된 대로 NooBaa를 외부 오브젝트 저장소로 구성합니다.

4.3.6.4.2. DataProtectionApplication CR에서 CSI 활성화

CSI 스냅샷으로 영구 볼륨을 백업하기 위해 DataProtectionApplication CR(사용자 정의 리소스)에서 CSI(Container Storage Interface)를 활성화합니다.

사전 요구 사항

- 클라우드 공급자는 CSI 스냅샷을 지원해야 합니다.

절차

- 다음 예와 같이 DataProtectionApplication CR을 편집합니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      defaultPlugins:
      - openshift
      - csi (1)
```

1 csi 기본 플러그인을 추가합니다.

4.3.7. 데이터 보호를 위한 OpenShift API 설치 제거

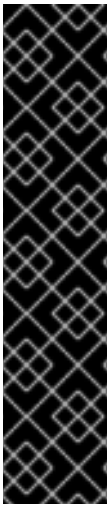
OADP Operator를 삭제하여 OADP(Data Protection)용 OpenShift API를 설치 제거합니다. 자세한 내용은 클러스터에서 Operator 삭제를 참조하십시오.

4.4. 백업 및 복원

4.4.1. 애플리케이션 백업

Backup CR(사용자 정의 리소스)을 생성하여 애플리케이션을 백업합니다.

Backup CR은 Kubernetes 리소스 및 내부 이미지, S3 개체 스토리지에서 내부 이미지, 영구 볼륨(PV)에 대한 백업 파일을 생성합니다. 클라우드 공급자가 네이티브 스냅샷 API 또는 CSI(Container Storage Interface) 를 사용하여 OpenShift Data Foundation 4와 같은 스냅샷을 생성합니다. 자세한 내용은 CSI 볼륨 스냅샷을 참조하십시오.



중요

S3 스토리지용 CloudStorage API는 기술 프리뷰 기능 전용입니다. 기술 프리뷰 기능은 Red Hat 프로덕션 서비스 수준 계약(SLA)에서 지원되지 않으며 기능적으로 완전하지 않을 수 있습니다. 따라서 프로덕션 환경에서 사용하는 것은 권장하지 않습니다. 이러한 기능을 사용하면 향후 제품 기능을 조기에 이용할 수 있어 개발 과정에서 고객이 기능을 테스트하고 피드백을 제공할 수 있습니다.

Red Hat 기술 프리뷰 기능의 지원 범위에 대한 자세한 내용은 기술 프리뷰 기능 지원 범위를 참조하십시오.

클라우드 공급자에 기본 스냅샷 API가 있거나 CSI(Container Storage Interface) 스냅샷을 지원하는 경우 Backup CR은 스냅샷을 생성하여 영구 볼륨을 백업합니다. 자세한 내용은 OpenShift Container Platform 문서 의 CSI 볼륨 스냅샷 개요 를 참조하십시오.

클라우드 공급자가 스냅샷을 지원하지 않거나 애플리케이션이 NFS 데이터 볼륨에 있는 경우 Restic을 사용하여 백업을 생성할 수 있습니다.

백업 작업 전후에 명령을 실행하는 백업 후크 를 생성할 수 있습니다.

Backup CR 대신 Schedule CR 을 생성하여 백업을 예약할 수 있습니다.

4.4.1.1. Backup CR 생성

Backup CR(사용자 정의 리소스)을 생성하여 Kubernetes 이미지, 내부 이미지 및 PV(영구 볼륨)를 백업 합니다.

사전 요구 사항

- OADP(OpenShift API for Data Protection) Operator를 설치해야 합니다.
- DataProtectionApplication CR은 Ready 상태여야 합니다.
- 백업 위치 사전 요구 사항:
  - Velero용으로 S3 오브젝트 스토리지가 구성되어 있어야 합니다.
  - DataProtectionApplication CR에 백업 위치가 구성되어 있어야 합니다.
- 스냅샷 위치 사전 요구 사항:
  - 클라우드 공급자에는 기본 스냅샷 API가 있거나 CSI(Container Storage Interface) 스냅샷을 지원해야 합니다.
  - CSI 스냅샷의 경우 CSI 드라이버를 등록하려면 VolumeSnapshotClass CR을 생성해야 합니다.
  - DataProtectionApplication CR에 구성된 볼륨 위치가 있어야 합니다.

절차

1. 다음 명령을 입력하여 **backupStorageLocations CR**을 검색합니다.

```
$ oc get backupStorageLocations
```

출력 예

```
NAME           PHASE       LAST VALIDATED   AGE   DEFAULT
velero-sample-1   Available   11s              31m
```

2. 다음 예와 같이 **Backup CR**을 생성합니다.

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: <backup>
  labels:
    velero.io/storage-location: default
  namespace: openshift-adp
spec:
  hooks: {}
  includedNamespaces:
  - <namespace> 1
  includedResources: [] 2
  excludedResources: [] 3
  storageLocation: <velero-sample-1> 4
  ttl: 720h0m0s
  labelSelector: 5
  - matchLabels:
      app=<label_1>
  - matchLabels:
      app=<label_2>
  - matchLabels:
      app=<label_3>
  orlabelSelectors: 6
  - matchLabels:
      app=<label_1>
  - matchLabels:
      app=<label_2>
  - matchLabels:
      app=<label_3>
```

1

2

선택 사항: 백업에 포함할 리소스 배열을 지정합니다. 리소스는 바로 가기(예: 'pods'의 경우 'po')이거나 정규화될 수 있습니다. 지정하지 않으면 모든 리소스가 포함됩니다.

3

선택 사항: 백업에서 제외할 리소스 배열을 지정합니다. 리소스는 바로 가기(예: 'pods'의 경우 'po')이거나 정규화될 수 있습니다.

4

backupStorageLocations CR의 이름을 지정합니다.

5

지정된 레이블이 모두 있는 백업 리소스입니다.

6

지정된 레이블이 하나 이상 있는 백업 리소스입니다.

3. Backup CR의 상태가 Completed 인지 확인합니다.

```
$ oc get backup -n openshift-adp <backup> -o jsonpath='{.status.phase}'
```

4.4.1.2. CSI 스냅샷을 사용하여 영구 볼륨 백업

Backup CR을 생성하기 전에 클라우드 스토리지의 VolumeSnapshotClass CR(사용자 정의 리소스)을 편집하여 CSI(Container Storage Interface) 스냅샷을 사용하여 영구 볼륨을 백업 합니다.

사전 요구 사항

- 클라우드 공급자는 CSI 스냅샷을 지원해야 합니다.
- DataProtectionApplication CR에서 CSI를 활성화해야 합니다.

절차

- metadata.labels.velero.io/csi-volumesnapshot-class: "true" 키-값 쌍을 VolumeSnapshotClass CR에 추가합니다.

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: <volume_snapshot_class_name>
  labels:
    velero.io/csi-volumesnapshot-class: "true"
driver: <csi_driver>
deletionPolicy: Retain
```

이제 Backup CR을 생성할 수 있습니다.

4.4.1.3. Restic을 사용하여 애플리케이션 백업

Restic을 사용하여 Kubernetes 리소스, 내부 이미지 및 영구 볼륨을 백업하여 Backup CR(사용자 정의 리소스)을 편집합니다.

DataProtectionApplication CR에서 스냅샷 위치를 지정할 필요가 없습니다.

중요

Restic은 hostPath 볼륨 백업을 지원하지 않습니다. 자세한 내용은 추가 Rustic 제한을 참조하십시오.

사전 요구 사항

- OADP(OpenShift API for Data Protection) Operator를 설치해야 합니다.
- DataProtectionApplication CR에서 spec.configuration.restic.enable 을 false 로 설정하여 기본 Restic 설치를 비활성화할 수 없습니다.
- DataProtectionApplication CR은 Ready 상태여야 합니다.

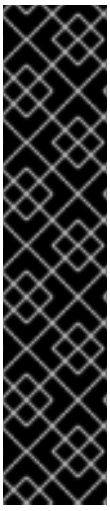
절차

- 다음 예와 같이 **Backup CR**을 편집합니다.

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: <backup>
  labels:
    velero.io/storage-location: default
  namespace: openshift-adp
spec:
  defaultVolumesToRestic: true (1)
...
```

(1) **spec** 블록에 **defaultVolumesToRestic: true** 를 추가합니다.

4.4.1.4. CSI 스냅샷에 Data Mover 사용



중요

CSI 스냅샷의 **Data Mover**는 기술 프리뷰 기능 전용입니다. 기술 프리뷰 기능은 **Red Hat** 프로덕션 서비스 수준 계약(**SLA**)에서 지원되지 않으며 기능적으로 완전하지 않을 수 있습니다. 따라서 프로덕션 환경에서 사용하는 것은 권장하지 않습니다. 이러한 기능을 사용하면 향후 제품 기능을 조기에 이용할 수 있어 개발 과정에서 고객이 기능을 테스트하고 피드백을 제공할 수 있습니다.

Red Hat 기술 프리뷰 기능의 지원 범위에 대한 자세한 내용은 기술 프리뷰 기능 지원 범위를 참조하십시오.

OADP 1.1.0 Data Mover를 사용하면 고객이 **CSI(Container Storage Interface)** 볼륨 스냅샷을 원격 오브젝트 저장소에 백업할 수 있습니다. **Data Mover**가 활성화되면 클러스터 장애, 실수로 삭제 또는 클러스터 손상이 발생하면 저장소에서 상태 저장 애플리케이션을 복원할 수 있습니다. **OADP 1.1.0 Data Mover** 솔루션은 **VolSync**의 **Restic** 옵션을 사용합니다.

참고

Data Mover는 **CSI** 볼륨 스냅샷의 백업 및 복원만 지원합니다.

현재 **Data Mover**는 **Google Cloud Storage(GCS)** 버킷을 지원하지 않습니다.

사전 요구 사항

- StorageClass 및 VolumeSnapshotClass CR(사용자 정의 리소스)에서 CSI를 지원하는지 확인했습니다.
- 하나의 volumeSnapshotClass CR만 snapshot.storage.kubernetes.io/is-default-class: true 가 있음을 확인했습니다.
- storageClass CR이 하나만 storageclass.kubernetes.io/is-default-class: true 가 있음을 확인했습니다.
- VolumeSnapshotClass CR에 velero.io/csi-volumesnapshot-class: 'true' 레이블이 포함되어 있습니다.
- OLM(Operator Lifecycle Manager)을 사용하여 VolSync Operator를 설치했습니다.

참고

VolSync Operator는 기술 프리뷰 Data Mover에서만 사용해야 합니다. OADP 프로덕션 기능을 사용하는 데 Operator가 필요하지 않습니다.

- OLM을 사용하여 OADP Operator를 설치했습니다.

절차

1. 다음과 같이 .yaml 파일을 생성하여 Restic 시크릿을 구성합니다.

```
apiVersion: v1
kind: Secret
metadata:
  name: <secret_name>
  namespace: openshift-adp
type: Opaque
stringData:
  RESTIC_PASSWORD: <secure_restic_password>
```

참고

기본적으로 **Operator**는 **dm-credential** 이라는 시크릿을 찾습니다. 다른 이름을 사용하는 경우 **dpa.spec.features.dataMover.credentialName** 을 사용하여 **DPA(Data Protection Application) CR**을 통해 이름을 지정해야 합니다.

2. 다음 예와 유사한 **DPA CR**을 생성합니다. 기본 플러그인에는 **CSI**가 포함됩니다.

DPA(Data Protection Application) CR 예

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: velero-sample
  namespace: openshift-adp
spec:
  features:
    dataMover:
      enable: true
      credentialName: <secret_name> 1
  backupLocations:
    - velero:
        config:
          profile: default
          region: us-east-1
        credential:
          key: cloud
          name: cloud-credentials
        default: true
        objectStorage:
          bucket: <bucket_name>
          prefix: <bucket_prefix>
        provider: aws
  configuration:
    restic:
      enable: <true_or_false>
    velero:
      defaultPlugins:
        - openshift
        - aws
        - csi
```

1

OADP Operator는 두 개의 CRD(사용자 정의 리소스 정의), VolumeSnapshotBackup 및 VolumeSnapshotRestore 를 설치합니다.

VolumeSnapshotBackup CRD의 예

```
apiVersion: datamover.oadp.openshift.io/v1alpha1
kind: VolumeSnapshotBackup
metadata:
  name: <vsb_name>
  namespace: <namespace_name> 1
spec:
  volumeSnapshotContent:
    name: <snapcontent_name>
  protectedNamespace: <adp_namespace>
  resticSecretRef:
    name: <restic_secret_name>
```

1

볼륨 스냅샷이 존재하는 네임스페이스를 지정합니다.

VolumeSnapshotRestore CRD의 예

```
apiVersion: datamover.oadp.openshift.io/v1alpha1
kind: VolumeSnapshotRestore
metadata:
  name: <vsr_name>
  namespace: <namespace_name> 1
spec:
  protectedNamespace: <protected_ns> 2
  resticSecretRef:
    name: <restic_secret_name>
  volumeSnapshotMoverBackupRef:
    sourcePVCData:
      name: <source_pvc_name>
      size: <source_pvc_size>
    resticrepository: <your_restic_repo>
    volumeSnapshotClassName: <vsclass_name>
```

1

볼륨 스냅샷이 존재하는 네임스페이스를 지정합니다.

2

Operator가 설치된 네임스페이스를 지정합니다. 기본값은 openshift-adp 입니다.

3. 다음 단계를 수행하여 볼륨 스냅샷을 백업할 수 있습니다.

a. 백업 CR을 생성합니다.

```
apiVersion: velero.io/v1
kind: Backup
metadata:
  name: <backup_name>
  namespace: <protected_ns> 1
spec:
  includedNamespaces:
  - <app_ns>
  storageLocation: velero-sample-1
```

1

Operator가 설치된 네임스페이스를 지정합니다. 기본 네임스페이스는 openshift-adp 입니다.

b. 최대 10분 정도 기다린 후 다음 명령을 입력하여 VolumeSnapshotBackup CR 상태가 완료되었는지 확인합니다.

```
$ oc get vsb -n <app_ns>
```

```
$ oc get vsb <vsb_name> -n <app_ns> -o jsonpath="{.status.phase}"
```

오브젝트는 DPA에서 구성된 오브젝트 저장소에서 생성됩니다.

참고

VolumeSnapshotBackup CR의 상태가 실패 가 되는 경우 문제를 해결하려면 Velero 로그를 참조하십시오.

4. 다음 단계를 수행하여 볼륨 스냅샷을 복원할 수 있습니다.

a. Velero CSI 플러그인에서 생성한 애플리케이션 네임스페이스 및 volumeSnapshotContent 를 삭제합니다.

b. Restore CR을 생성하고 restorePV 를 true 로 설정합니다.

Restore CR의 예

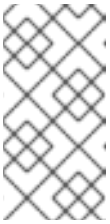
```
apiVersion: velero.io/v1
kind: Restore
metadata:
  name: <restore_name>
  namespace: <protected_ns>
spec:
  backupName: <previous_backup_name>
  restorePVs: true
```

c. 최대 10분 정도 기다린 후 다음 명령을 입력하여 VolumeSnapshotRestore CR 상태가 완료되었는지 확인합니다.

```
$ oc get vsr -n <app_ns>
```

```
$ oc get vsr <vsr_name> -n <app_ns> -o jsonpath="{.status.phase}"
```

d. 애플리케이션 데이터 및 리소스가 복원되었는지 확인합니다.



참고

VolumeSnapshotRestore CR의 상태가 **'Failed'**가 되면 문제 해결을 위해 **Velero** 로그를 참조하십시오.

추가 리소스

- 관리자를 위해 클러스터에 Operator 설치
- 관리자가 아닌 경우 네임스페이스에 Operator 설치

4.4.1.5. 백업 후크 생성

Backup CR(사용자 정의 리소스)을 편집하여 **Pod**의 컨테이너에서 명령을 실행하는 백업 후크를 생성합니다.

포드를 백업하기 전에 *사전* 후크가 실행됩니다. 백업 후 후크가 실행됩니다.

절차

- *다음 예와 같이 **Backup CR**의 **spec.hooks** 블록에 후크를 추가합니다.*

```
apiVersion: velero.io/v1
kind: Backup
metadata:
 name: <backup>
 namespace: openshift-adp
spec:
 hooks:
  resources:
   - name: <hook_name>
     includedNamespaces:
     - <namespace> 1
     excludedNamespaces: 2
     - <namespace>
     includedResources: []
     - pods 3
     excludedResources: [] 4
     labelSelector: 5
       matchLabels:
         app: velero
         component: server
     pre: 6
```

```
      - exec:
          container: <container> 7
          command:
          - /bin/uname 8
          - -a
          onError: Fail 9
          timeout: 30s 10
        post: 11
...
```

1

선택 사항: 후크가 적용되는 네임스페이스를 지정할 수 있습니다. 이 값을 지정하지 않으면 후크가 모든 네임스페이스에 적용됩니다.

2

선택 사항: 후크가 적용되지 않는 네임스페이스를 지정할 수 있습니다.

3

4

선택 사항: 후크가 적용되지 않는 리소스를 지정할 수 있습니다.

5

선택 사항: 이 후크는 레이블과 일치하는 오브젝트에만 적용됩니다. 이 값을 지정하지 않으면 후크가 모든 네임스페이스에 적용됩니다.

6

백업 전에 실행할 후크 배열입니다.

7

선택 사항: 컨테이너를 지정하지 않으면 **Pod**의 첫 번째 컨테이너에서 명령이 실행됩니다.

8

이는 추가되는 **init** 컨테이너의 진입점입니다.

9

오류 처리에 허용되는 값은 **Fail** 및 **Continue** 입니다. 기본값은 **Fail** 입니다.

10

선택 사항: 명령이 실행될 때까지 대기하는 시간입니다. 기본값은 **30s** 입니다.

11

이 블록은 백업 후 실행할 후크 배열과 **pre-backup** 후크와 동일한 매개변수를 정의합니다.

4.4.1.6. 백업 예약

Backup CR 대신 **Schedule CR**(사용자 정의 리소스)을 생성하여 백업을 예약합니다.

주의

다른 백업이 생성되기 전에 백업이 완료될 수 있도록 백업 일정에 충분한 시간을 남겨 둡니다.

예를 들어 네임스페이스의 백업이 일반적으로 **10**분 정도 걸리는 경우 **15**분마다 백업을 더 자주 예약하지 마십시오.

사전 요구 사항

- **OADP(OpenShift API for Data Protection) Operator**를 설치해야 합니다.
- **DataProtectionApplication CR**은 **Ready** 상태여야 합니다.

절차

1. **backupStorageLocations CR**을 검색합니다.

```
$ oc get backupStorageLocations
```

출력 예

```
NAME              PHASE       LAST VALIDATED   AGE   DEFAULT
velero-sample-1   Available   11s              31m
```

2. 다음 예와 같이 **Schedule CR**을 생성합니다.

```
$ cat << EOF | oc apply -f -
apiVersion: velero.io/v1
kind: Schedule
metadata:
  name: <schedule>
  namespace: openshift-adp
spec:
  schedule: 0 7 * * * 1
  template:
    hooks: {}
    includedNamespaces:
    - <namespace> 2
    storageLocation: <velero-sample-1> 3
    defaultVolumesToRestic: true 4
    ttl: 720h0m0s
EOF
```

1. 백업을 예약하는 Cron 표현식(예: **0 7 * * ***)은 **7:00**에 매일 백업을 수행합니다.

2. 백업할 네임스페이스의 배열입니다.

3. **backupStorageLocations CR**의 이름입니다.

4. 선택 사항: **Restic**을 사용하여 볼륨을 백업하는 경우 **defaultVolumesToRestic: true** 키-값 쌍을 추가합니다.

3. 예약된 백업이 실행된 후 **Schedule CR**의 상태가 **Completed** 인지 확인합니다.

```
$ oc get schedule -n openshift-adp <schedule> -o jsonpath='{.status.phase}'
```

4.4.1.7. 백업 삭제

Backup CR(사용자 정의 리소스)을 삭제하여 백업 파일을 제거할 수 있습니다.

주의

Backup CR 및 관련 개체 스토리지 데이터를 삭제한 후에는 삭제된 데이터를 복구할 수 없습니다.

사전 요구 사항

- **Backup CR**을 생성하셨습니다.
- **Backup CR**의 이름과 이를 포함하는 네임스페이스를 알고 있습니다.
- **Velero CLI** 툴을 다운로드했습니다.
- 클러스터에서 **Velero** 바이너리에 액세스할 수 있습니다.

절차

- **Backup CR**을 삭제하려면 다음 작업 중 하나를 선택합니다.
  - **Backup CR**을 삭제하고 관련 오브젝트 스토리지 데이터를 유지하려면 다음 명령을 실행합니다.

```
$ oc delete backup <backup_CR_name> -n <velero_namespace>
```

- Backup CR을 삭제하고 관련 오브젝트 스토리지 데이터를 삭제하려면 다음 명령을 실행합니다.

```
$ velero backup delete <backup_CR_name> -n <velero_namespace>
```

다음과 같습니다.

<backup_CR_name>

Backup 사용자 정의 리소스의 이름을 지정합니다.

<velero_namespace>

Backup 사용자 정의 리소스가 포함된 네임스페이스를 지정합니다.

추가 리소스

- Velero CLI 툴 다운로드

4.4.2. 애플리케이션 복원

Restore CR(사용자 정의 리소스) 을 생성하여 애플리케이션 백업을 복원합니다.

복원 후크 를 생성하여 init 컨테이너에서 명령을 실행하거나 애플리케이션 컨테이너를 시작하기 전에 또는 애플리케이션 컨테이너 자체에서 명령을 실행할 수 있습니다.

4.4.2.1. Restore CR 생성

Restore CR을 생성하여 Backup CR(사용자 정의 리소스)을 복원합니다.

사전 요구 사항

- OADP(OpenShift API for Data Protection) Operator를 설치해야 합니다.
- DataProtectionApplication CR은 Ready 상태여야 합니다.

- *Velero Backup CR이 있어야 합니다.*
- *PV(영구 볼륨) 용량이 백업 시 요청된 크기와 일치하도록 요청된 크기를 조정합니다.*

절차

1. *다음 예와 같이 Restore CR을 생성합니다.*

```
apiVersion: velero.io/v1
kind: Restore
metadata:
 name: <restore>
 namespace: openshift-adp
spec:
 backupName: <backup> 1
 includedResources: [] 2
 excludedResources:
 - nodes
 - events
 - events.events.k8s.io
 - backups.velero.io
 - restores.velero.io
 - resticrepositories.velero.io
 restorePVs: true
```

1. *Backup CR의 이름입니다.*
2. *선택 사항: 복원 프로세스에 포함할 리소스 배열을 지정합니다. 리소스는 바로 가기(예: 'pods'의 경우 'po')이거나 정규화될 수 있습니다. 지정하지 않으면 모든 리소스가 포함됩니다.*

2. *다음 명령을 입력하여 Restore CR의 상태가 Completed 인지 확인합니다.*

```
$ oc get restore -n openshift-adp <restore> -o jsonpath='{.status.phase}'
```

3. *다음 명령을 입력하여 백업 리소스가 복원되었는지 확인합니다.*

```
$ oc get all -n <namespace> 1
```

1

백업한 네임스페이스입니다.

4. Restic을 사용하여 DeploymentConfig 오브젝트를 복원하거나 post-restore 후크를 사용하는 경우 다음 명령을 입력하여 dc-restic-post-restore.sh cleanup 스크립트를 실행합니다.

```
$ bash dc-restic-post-restore.sh <restore-name>
```

참고

복원 프로세스 과정에서 OADP Velero 플러그인은 DeploymentConfig 오브젝트를 축소하고 Pod를 독립 실행형 Pod로 복원하여 클러스터가 복원 시 복원된 DeploymentConfig Pod를 즉시 삭제하지 못하도록 하고 Restic 및 post-restore 후크를 허용하여 복원된 Pod에서 작업을 완료할 수 있습니다. cleanup 스크립트는 이러한 연결이 끊긴 Pod를 제거하고 DeploymentConfig 오브젝트를 적절한 복제본 수로 백업합니다.

예 4.1. dc-restic-post-restore.sh cleanup script

```
#!/bin/bash
set -e

# if sha256sum exists, use it to check the integrity of the file
if command -v sha256sum >/dev/null 2>&1; then
  CHECKSUM_CMD="sha256sum"
else
  CHECKSUM_CMD="shasum -a 256"
fi

label_name () {
    if [ "${#1}" -le "63" ]; then
 echo $1
 return
    fi
    sha=$(echo -n $1|$CHECKSUM_CMD)
    echo "${1:0:57}${sha:0:6}"
}

OADP_NAMESPACE=${OADP_NAMESPACE:=openshift-adp}

if [[ $# -ne 1 ]]; then
    echo "usage: ${BASH_SOURCE} restore-name"
    exit 1
fi

echo using OADP Namespace $OADP_NAMESPACE
echo restore: $1
```

```
label=$(label_name $1)
echo label: $label

echo Deleting disconnected restore pods
oc delete pods -l oadp.openshift.io/disconnected-from-dc=$label

for dc in $(oc get dc --all-namespaces -l oadp.openshift.io/replicas-modified=$label
-o jsonpath='{range .items[*]}{.metadata.namespace}{","}{.metadata.name}{","}
{.metadata.annotations.oadp\.openshift\.io/original-replicas}{","}
{.metadata.annotations.oadp\.openshift\.io/original-paused}{"\n"}')
do
  IFS=',' read -ra dc_arr <<< "$dc"
  if [ ${#dc_arr[0]} -gt 0 ]; then
 echo Found deployment ${dc_arr[0]}/${dc_arr[1]}, setting replicas: ${dc_arr[2]},
paused: ${dc_arr[3]}
 cat <<EOF | oc patch dc  -n ${dc_arr[0]} ${dc_arr[1]} --patch-file /dev/stdin
spec:
  replicas: ${dc_arr[2]}
  paused: ${dc_arr[3]}
EOF
  fi
done
```

4.4.2.2. 복원 후크 생성

Restore CR(사용자 정의 리소스)을 편집하여 애플리케이션을 복원하는 동안 **Pod**의 컨테이너에서 명령을 실행하는 복원 후크를 생성합니다.

두 가지 유형의 복원 후크를 생성할 수 있습니다.

- **init** 후크는 애플리케이션 컨테이너가 시작되기 전에 설정 작업을 수행하기 위해 **Pod**에 **init** 컨테이너를 추가합니다.

  Restic 백업을 복원하는 경우 복원 후크 **init** 컨테이너 앞에 **restic-wait init** 컨테이너가 추가됩니다.

- **exec** 후크는 복원된 **Pod**의 컨테이너에서 명령 또는 스크립트를 실행합니다.

절차

- 다음 예와 같이 **Restore CR**의 **spec.hooks** 블록에 후크를 추가합니다.

```
apiVersion: velero.io/v1
kind: Restore
metadata:
  name: <restore>
  namespace: openshift-adp
spec:
  hooks:
    resources:
      - name: <hook_name>
        includedNamespaces:
        - <namespace> 1
        excludedNamespaces:
        - <namespace>
        includedResources:
        - pods 2
        excludedResources: []
        labelSelector: 3
          matchLabels:
            app: velero
            component: server
        postHooks:
        - init:
            initContainers:
            - name: restore-hook-init
              image: alpine:latest
              volumeMounts:
              - mountPath: /restores/pvc1-vm
                name: pvc1-vm
              command:
              - /bin/ash
              - -c
            timeout: 4
        - exec:
            container: <container> 5
            command:
            - /bin/bash 6
            - -c
            - "psql < /backup/backup.sql"
            waitTimeout: 5m 7
            execTimeout: 1m 8
            onError: Continue 9
```

1

선택 사항: 후크가 적용되는 네임스페이스 배열입니다. 이 값을 지정하지 않으면 후크가 모든 네임스페이스에 적용됩니다.

2

현재 Pod는 후크를 적용할 수 있는 유일한 지원 리소스입니다.

3

4

선택 사항: **Timeout**은 **Velero**가 **initContainers** 가 완료될 때까지의 최대 시간을 지정합니다.

5

선택 사항: 컨테이너를 지정하지 않으면 **Pod**의 첫 번째 컨테이너에서 명령이 실행됩니다.

6

이는 추가되는 **init** 컨테이너의 진입점입니다.

7

선택 사항: 컨테이너가 준비될 때까지 대기하는 시간입니다. 컨테이너가 시작되고 동일한 컨테이너의 이전 후크가 완료될 때까지 충분히 길어야 합니다. 설정되지 않은 경우 복원 프로세스는 무기한 대기합니다.

8

선택 사항: 명령이 실행될 때까지 대기하는 시간입니다. 기본값은 **30s** 입니다.

9

오류 처리에 허용되는 값은 **Fail and Continue** 입니다.

- **continue**: 명령 실패만 기록됩니다.
- **fail**: **Pod**의 컨테이너에서 더 이상 복원 후크가 실행되지 않습니다. **Restore CR**의 상태는 **PartiallyFailed** 가 됩니다.

4.5. 문제 해결

OpenShift CLI 툴 또는 **Velero CLI** 툴 을 사용하여 **Velero** 사용자 정의 리소스(**CR**)를 디버깅할 수 있습니다. https://access.redhat.com/documentation/en-us/openshift_container_platform/4.10/html-single/backup_and_restore/#migration-debugging-velero-resources_oadp-troubleshooting **Velero CLI** 툴에서는 더 자세한 로그 및 정보를 제공합니다.

설치 문제,백업 및 복원 CR 문제 , Restic 문제를 확인할 수 있습니다.

must-gather 툴을 사용하여 로그, CR 정보 및 Prometheus 지표 데이터를 수집할 수 있습니다.

다음을 통해 Velero CLI 툴을 가져올 수 있습니다.

- Velero CLI 툴 다운로드
- 클러스터의 Velero 배포에서 Velero 바이너리에 액세스

4.5.1. Velero CLI 툴 다운로드

Velero 설명서 페이지의 지침에 따라 Velero CLI 툴을 다운로드하여 설치할 수 있습니다.

페이지에는 다음에 대한 지침이 포함되어 있습니다.

- Homebrew를 사용하여 macOS
- GitHub
- Chocolatey를 사용하여 Windows

사전 요구 사항

- DNS 및 컨테이너 네트워킹이 활성화된 Kubernetes 클러스터 v1.16 이상에 액세스할 수 있습니다.
- kubectl 을 로컬로 설치했습니다.

절차

1. 브라우저를 열고 Verleo 웹 사이트에서 "Install the CLI" 로 이동합니다.

2. macOS, GitHub 또는 Windows에 대한 적절한 절차를 따르십시오.

3. 다음 표에 따라 OADP 및 OpenShift Container Platform 버전에 적합한 Velero 버전을 다운로드합니다.

표 4.2. OADP-Velero-OpenShift Container Platform 버전 관계

OADP 버전	Velero 버전	OpenShift Container Platform 버전
1.0.0	1.7	4.6 이상
1.0.1	1.7	4.6 이상
1.0.2	1.7	4.6 이상
1.0.3	1.7	4.6 이상
1.1.0	{velero-version}	4.9 이상
1.1.1	{velero-version}	4.9 이상
1.1.2	{velero-version}	4.9 이상

4.5.2. 클러스터의 Velero 배포에서 Velero 바이너리에 액세스

shell 명령을 사용하여 클러스터의 Velero 배포에서 Velero 바이너리에 액세스할 수 있습니다.

사전 요구 사항

- DataProtectionApplication 사용자 정의 리소스의 상태는 Reconcile complete.

절차

- 다음 명령을 입력하여 필요한 별칭을 설정합니다.

```
$ alias velero='oc -n openshift-adp exec deployment/velero -c velero -it -- ./velero'
```

4.5.3. OpenShift CLI 툴을 사용하여 Velero 리소스 디버깅

OpenShift CLI 툴을 사용하여 Velero 사용자 정의 리소스(CR) 및 Velero Pod 로그를 확인하여 실패한 백업 또는 복원을 디버깅할 수 있습니다.

Velero CR

oc describe 명령을 사용하여 Backup 또는 Restore CR과 관련된 경고 및 오류 요약을 검색합니다.

```
$ oc describe <velero_cr> <cr_name>
```

Velero 포드 로그

oc logs 명령을 사용하여 Velero 포드 로그를 검색합니다.

```
$ oc logs pod/<velero>
```

Velero pod 디버그 로그

다음 예와 같이 DataProtectionApplication 리소스에서 Velero 로그 수준을 지정할 수 있습니다.

참고

이 옵션은 OADP 1.0.3부터 사용할 수 있습니다.

```
apiVersion: oadp.openshift.io/v1alpha1
kind: DataProtectionApplication
metadata:
  name: velero-sample
spec:
  configuration:
    velero:
      logLevel: warning
```

다음 logLevel 값을 사용할 수 있습니다.

- trace
- debug

- *info*
- *경고*
- *error*
- *fatal*
- *panic*

*대부분의 로그에 **debug** 를 사용하는 것이 좋습니다.*

4.5.4. Velero CLI 툴을 사용하여 Velero 리소스 디버깅

Backup 및 Restore CR(사용자 정의 리소스)을 디버그하고 Velero CLI 툴을 사용하여 로그를 검색할 수 있습니다.

Velero CLI 툴은 OpenShift CLI 툴보다 자세한 정보를 제공합니다.

구문

oc exec 명령을 사용하여 Velero CLI 명령을 실행합니다.

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  <backup_restore_cr> <command> <cr_name>
```

예제

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  backup describe 0e44ae00-5dc3-11eb-9ca8-df7e5254778b-2d8ql
```

도움말 옵션

velero --help 옵션을 사용하여 모든 Velero CLI 명령을 나열합니다.

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  --help
```

Describe 명령

velero describe 명령을 사용하여 Backup 또는 Restore CR과 관련된 경고 및 오류 요약을 검색합니다.

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  <backup_restore_cr> describe <cr_name>
```

예제

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  backup describe 0e44ae00-5dc3-11eb-9ca8-df7e5254778b-2d8ql
```

Logs 명령

velero logs 명령을 사용하여 Backup 또는 Restore CR의 로그를 검색합니다.

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  <backup_restore_cr> logs <cr_name>
```

예제

```
$ oc -n openshift-adp exec deployment/velero -c velero -- ./velero \
  restore logs ccc7c2d0-6017-11eb-afab-85d0007f5a19-x4lbf
```

4.5.5. Velero 및 승인 Webhook 관련 문제

Velero는 복원 중 승인 Webhook 문제를 해결할 수 있는 기능이 제한되어 있습니다. 승인 Webhook가 있는 워크로드가 있는 경우 추가 Velero 플러그인을 사용하거나 워크로드를 복원하는 방법을 변경해

야 할 수 있습니다.

일반적으로 승인 Webhook가 있는 워크로드에서는 먼저 특정 종류의 리소스를 생성해야 합니다. 승인 Webhook가 일반적으로 하위 리소스를 차단하기 때문에 워크로드에 하위 리소스가 있는 경우 특히 중요합니다.

예를 들어 service.serving.knative.dev 와 같은 최상위 오브젝트를 생성하거나 복원하면 일반적으로 하위 리소스가 자동으로 생성됩니다. 먼저 이 작업을 수행하는 경우 Velero를 사용하여 이러한 리소스를 생성하고 복원할 필요가 없습니다. 그러면 Velero가 사용할 수 있는 승인 Webhook에 의해 차단되는 하위 리소스의 문제를 방지할 수 있습니다.

4.5.5.1. 승인 Webhook를 사용하는 Velero 백업의 해결방법 복원

이 섹션에서는 승인 Webhook를 사용하는 여러 유형의 Velero 백업에 대한 리소스를 복원하는 데 필요한 추가 단계를 설명합니다.

4.5.5.1.1. Knative 리소스 복원

승인 Webhook를 사용하는 Knative 리소스를 백업하려면 Velero를 사용하여 문제가 발생할 수 있습니다.

승인 Webhook를 사용하는 Knative 리소스를 백업 및 복원할 때마다 최상위 서비스 리소스를 복원하여 이러한 문제를 방지할 수 있습니다.

절차

- 최상위 서비스.serving.knavtive.dev Service 리소스를 복원합니다.

```
$ velero restore <restore_name> \
  --from-backup=<backup_name> --include-resources \
  service.serving.knavtive.dev
```

4.5.5.1.2. IBM AppConnect 리소스 복원

승인 Webhook가 있는 IBM AppConnect 리소스를 복원하기 위해 Velero를 사용할 때 문제가 발생하는 경우 이 프로세스에서 검사를 실행할 수 있습니다.

절차

1. 클러스터에 변경 승인 플러그인이 있는지 확인합니다 : **MutatingWebhookConfiguration.**

```
$ oc get mutatingwebhookconfigurations
```

2. 각 **kind: MutatingWebhookConfiguration** 의 **YAML** 파일을 검사하여 문제가 발생한 오브젝트의 규칙 블록 생성이 없는지 확인합니다. 자세한 내용은 공식 **Kuberbetes** 설명서를 참조하십시오.

3. 유형에 있는 **spec.version: Configuration.appconnect.ibm.com/v1beta1** 이 설치된 **Operator**에서 지원되는지 확인합니다.

추가 리소스

- 허용 플러그인
- **Webhook** 승인 플러그인
- 웹 후크 승인 플러그인의 유형

4.5.6. 설치 문제

데이터 보호 애플리케이션을 설치할 때 유효하지 않은 디렉토리 또는 잘못된 인증 정보를 사용하여 발생한 문제가 발생할 수 있습니다.

4.5.6.1. 백업 스토리지에는 잘못된 디렉터리가 포함되어 있습니다.

Velero 포드 로그에 오류 메시지가 표시되고 **Backup** 스토리지에 잘못된 최상위 디렉터리가 포함되어 있습니다.

원인

오브젝트 스토리지에는 **Velero** 디렉터리가 아닌 최상위 디렉터리가 포함되어 있습니다.

해결책

오브젝트 스토리지가 **Velero** 전용이 아닌 경우 **DataProtectionApplication** 매니페스트에서 **spec.backupLocations.velero.objectStorage.prefix** 매개변수를 설정하여 버킷에 대한 접두사를 지정

해야 합니다.

4.5.6.2. 잘못된 AWS 인증 정보

oadp-aws-registry Pod 로그에는 오류 메시지, InvalidAccessKeyId: AWS Access Key Id가 저희 레코드에 존재하지 않습니다.

Velero Pod 로그에 오류 메시지 NoCredentialProviders: no valid providers in chain.

원인

Secret 오브젝트를 생성하는 데 사용되는 credentials-velero 파일이 잘못 포맷됩니다.

해결책

다음 예와 같이 credentials-velero 파일이 올바르게 포맷되어 있는지 확인합니다.

credentials-velero 파일 예

```
[default] 1
aws_access_key_id=AKIAIOSFODNN7EXAMPLE 2
aws_secret_access_key=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

1 *AWS 기본 프로필.*

2 *따옴표로 값을 묶지 마십시오(", '.).*

4.5.7. CR 백업 및 복원 문제

백업 및 CR(사용자 정의 리소스)과 관련된 일반적인 문제가 발생할 수 있습니다.

4.5.7.1. 백업 CR은 볼륨을 검색할 수 없음

Backup CR에 오류 메시지 InvalidVolume.NotFound: 볼륨 'vol-xxxx'가 존재하지 않습니다.

원인

PV(영구 볼륨) 및 스냅샷 위치는 다른 지역에 있습니다.

해결책

1. DataProtectionApplication 매니페스트에서 spec.snapshotLocations.velero.config.region 키 값을 편집하여 스냅샷 위치가 PV와 동일한 리전에 있도록 합니다.

2. 새 Backup CR을 생성합니다.

4.5.7.2. 백업 CR 상태가 진행 중임

Backup CR의 상태는 InProgress 단계에 남아 있으며 완료되지 않습니다.

원인

백업이 중단되면 다시 시작할 수 없습니다.

해결책

1. Backup CR의 세부 정보를 검색합니다.

```
$ oc -n {namespace} exec deployment/velero -c velero -- ./velero \
  backup describe <backup>
```

2. Backup CR을 삭제합니다.

```
$ oc delete backup <backup> -n openshift-adp
```

진행중인 Backup CR이 오브젝트 스토리지에 파일을 업로드하지 않았기 때문에 백업 위치를 정리할 필요가 없습니다.

3. *새 Backup CR을 생성합니다.*

4.5.7.3. Backup CR 상태는 PartiallyFailed에 남아 있습니다.

Restic이 사용되지 않은 Backup CR의 상태는 PartiallyFailed 단계에 남아 있으며 완료되지 않습니다. 관련 PVC의 스냅샷이 생성되지 않습니다.

원인

CSI 스냅샷 클래스를 기반으로 백업을 생성하지만 라벨이 누락된 경우 CSI 스냅샷 플러그인이 스냅샷을 생성하지 못합니다. 결과적으로 Velero pod는 다음과 유사한 오류를 기록합니다.

+

```
time="2023-02-17T16:33:13Z" level=error msg="Error backing up item" backup=openshift-adp/user1-backup-check5 error="error executing custom action (groupResource=persistentvolumeclaims, namespace=busy1, name=pvc1-user1): rpc error: code = Unknown desc = failed to get volumesnapshotclass for storageclass ocs-storagecluster-ceph-rbd: failed to get volumesnapshotclass for provisioner openshift-storage.rbd.csi.ceph.com, ensure that the desired volumesnapshot class has the velero.io/csi-volumesnapshot-class label" logSource="/remote-source/velero/app/pkg/backup/backup.go:417" name=busybox-79799557b5-vprq
```

해결책

1. *Backup CR을 삭제합니다.*

```
$ oc delete backup <backup> -n openshift-adp
```

2. *필요하면 BackupStorageLocation 에서 저장된 데이터를 정리하여 공간을 확보합니다.*

3. *VolumeSnapshotClass 오브젝트에 velero.io/csi-volumesnapshot-class=true 라벨을 적용합니다.*

```
$ oc label volumesnapshotclass/<snapclass_name> velero.io/csi-volumesnapshot-class=true
```

4. *새 Backup CR을 생성합니다.*

4.5.8. Restic 문제

Restic을 사용하여 애플리케이션을 백업할 때 이러한 문제가 발생할 수 있습니다.

4.5.8.1. root_squash가 활성화된 NFS 데이터 볼륨에 대한 Restic 권한 오류

Restic pod 로그에 오류 메시지 controller=pod-volume-backup error="fork/exec/usr/bin/restic: permission denied" 가 표시됩니다.

원인

NFS 데이터 볼륨에 root_squash 가 활성화된 경우 Restic 은 nfsnobody 에 매핑되고 백업을 생성할 수 있는 권한이 없습니다.

해결책

Restic 에 대한 추가 그룹을 생성하고 DataProtectionApplication 매니페스트에 그룹 ID를 추가하여 이 문제를 해결할 수 있습니다.

1. NFS 데이터 볼륨에서 Restic 에 대한 추가 그룹을 생성합니다.
2. 그룹 소유권이 상속되도록 NFS 디렉터리에 setgid 비트를 설정합니다.
3. 다음 예와 같이 spec.configuration.restic.supplementalGroups 매개변수와 그룹 ID를 DataProtectionApplication 매니페스트에 추가합니다.

```
spec:
  configuration:
    restic:
      enable: true
      supplementalGroups:
      - <group_id> 1
```

1. 보조 그룹 ID를 지정합니다.

4. 변경 사항을 적용할 수 있도록 Restic Pod가 다시 시작될 때까지 기다립니다.

4.5.8.2. Restic Backup CR은 버킷을 꺼진 후 다시 생성할 수 없습니다.

네임스페이스에 대한 Restic Backup CR을 생성하고 오브젝트 스토리지 버킷을 비우고 동일한 네임스페이스에 Backup CR을 다시 생성하면 다시 만든 Backup CR이 실패합니다.

velero Pod 로그에는 다음과 같은 오류 메시지가 표시됩니다. stderr=Fatal: unable to open config file: criteria: specified key does not exist.\nIs there a repository at the following location?.

원인

Velero는 Restic 디렉터리가 오브젝트 스토리지에서 삭제되면 ResticRepository 매니페스트에서 Restic 리포지토리를 다시 생성하거나 업데이트하지 않습니다. 자세한 내용은 Velero issue 4421 을 참조하십시오.

해결책

- 다음 명령을 실행하여 네임스페이스에서 관련 Restic 리포지토리를 제거합니다.

```
$ oc delete resticrepository openshift-adp <name_of_the_restic_repository>
```

다음 오류 로그에서 mysql-persistent 는 문제가 있는 Restic 리포지토리입니다. 저장소 이름은 명확성을 위해 이주에 표시됩니다.

```
time="2021-12-29T18:29:14Z" level=info msg="1 errors
encountered backup up item" backup=velero/backup65
logSource="pkg/backup/backup.go:431" name=mysql-7d99fc949-qbkds
time="2021-12-29T18:29:14Z" level=error msg="Error backing up item"
backup=velero/backup65 error="pod volume backup failed: error running
restic backup, stderr=Fatal: unable to open config file: Stat: The
specified key does not exist.\nIs there a repository at the following
location?\ns3:http://minio-minio.apps.mayap-oadp-
veleo-1234.qe.devcluster.openshift.com/mayapvelerooadp2/velero1/
restic/mysql-persistent\n: exit status 1" error.file="/remote-source/
src/github.com/vmware-tanzu/velero/pkg/restic/backupper.go:184"
error.function="github.com/vmware-tanzu/velero/
pkg/restic.(*backupper).BackupPodVolumes"
logSource="pkg/backup/backup.go:435" name=mysql-7d99fc949-qbkds
```

4.5.9. must-gather 툴 사용

must-gather 툴을 사용하여 OADP 사용자 정의 리소스에 대한 로그, 메트릭 및 정보를 수집할 수 있습니다.

must-gather 데이터는 모든 고객 사례에 첨부되어야 합니다.

사전 요구 사항

- cluster-admin 역할의 사용자로 OpenShift Container Platform 클러스터에 로그인해야 합니다.
- OpenShift CLI(oc)가 설치되어 있어야 합니다.

절차

1. must-gather 데이터를 저장하려는 디렉터리로 이동합니다.
2. 다음 데이터 수집 옵션 중 하나에 대해 oc adm must-gather 명령을 실행합니다.

```
$ oc adm must-gather --image=registry.redhat.io/oadp/oadp-mustgather-rhel8:v1.1
```

데이터는 must-gather/must-gather.tar.gz 로 저장됩니다. Red Hat 고객 포털에서 해당 지원 사례에 이 파일을 업로드할 수 있습니다.

```
$ oc adm must-gather --image=registry.redhat.io/oadp/oadp-mustgather-rhel8:v1.1 \
  -- /usr/bin/gather_metrics_dump
```

이 작업에는 오랜 시간이 걸릴 수 있습니다. 데이터는 must-gather/metrics/prom_data.tar.gz 로 저장됩니다.

Prometheus 콘솔을 사용하여 메트릭 데이터 보기

Prometheus 콘솔을 사용하여 메트릭 데이터를 볼 수 있습니다.

절차

1. prom_data.tar.gz 파일의 압축을 풉니다.

```
$ tar -xvzf must-gather/metrics/prom_data.tar.gz
```

2. 로컬 Prometheus 인스턴스를 생성합니다.

```
$ make prometheus-run
```

이 명령은 Prometheus URL을 출력합니다.

출력 결과

```
Started Prometheus on http://localhost:9090
```

3. 웹 브라우저를 시작하고 URL로 이동하여 Prometheus 웹 콘솔을 사용하여 데이터를 확인합니다.

4. 데이터를 보고 나면 Prometheus 인스턴스 및 데이터를 삭제합니다.

```
$ make prometheus-cleanup
```

4.6. OADP와 함께 사용되는 API

이 문서에서는 OADP와 함께 사용할 수 있는 다음 API에 대한 정보를 제공합니다.

- Velero API
- OADP API

4.6.1. Velero API

Velero API 문서는 Red Hat이 아닌 Velero API에서 유지 관리합니다. Velero API 유형에서 찾을 수 있습니다.

4.6.2. OADP API

*다음 표에서는 **OADP API**의 구조를 제공합니다.*

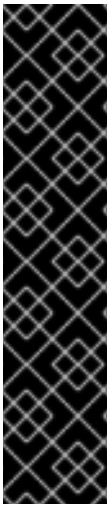
표 4.3. DataProtectionApplicationSpec

속성	유형	설명
backupLocations	[] **BackupLocation**	**BackupStorageLocations** 에 사용할 구성 목록을 정의합니다.
snapshotLocations	[] **SnapshotLocation**	**VolumeSnapshotLocation에 사용할 구성 목록을 정의합니다**.
unsupportedOverrides	map [UnsupportedImageKey] string	개발을 위해 배포된 종속 이미지를 재정의하는 데 사용할 수 있습니다. 옵션은 **veleroImageFqin**,**awsPluginImageFqin**,**openshiftPluginImageFqin**,**azurePluginImageFqin**,**gcpPluginImageImageFqin** , dataMover**ImageFqin**,**dataMoverImageFqin**,**resticRestoreImageFqin**, VirticRestore**ImageFqin** , **awsPlugin** ImageFqin 입니다.
podAnnotations	map [string] string	Operator에서 배포한 Pod에 주석을 추가하는 데 사용됩니다.
podDnsPolicy	**DNSPolicy**	Pod의 DNS 구성을 정의합니다.
podDnsConfig	**PodDNSConfig**	**DNSPolicy** 에서 생성된 포드 외에도 Pod의 DNS 매개변수를 정의합니다.
backupImages	*bool	이미지 백업 및 복원을 위해 레지스트리를 배포할지 여부를 지정하는 데 사용됩니다.
설정	***ApplicationConfig**	데이터 보호 애플리케이션의 서버 구성을 정의하는 데 사용됩니다.
기능	*기능	기술 프리뷰 기능을 활성화하기 위한 DPA 구성을 정의합니다.

OADP API에 대한 전체 스키마 정의

표 4.4. BackupLocation

속성	유형	설명
velero	*velero.BackupStorageLocationSpec	Backup Storage Location에 설명된 대로 볼륨 스냅샷을 저장할 위치입니다.
bucket	*CloudStorageLocation	[기술 프리뷰] 일부 클라우드 스토리지 공급자에서 백업 스토리지 위치로 사용하기 위해 버킷 생성을 자동화합니다.



중요

***bucket** 매개변수는 기술 프리뷰 기능 전용입니다. 기술 프리뷰 기능은 **Red Hat** 프로덕션 서비스 수준 계약(**SLA**)에서 지원되지 않으며 기능적으로 완전하지 않을 수 있습니다. 따라서 프로덕션 환경에서 사용하는 것은 권장하지 않습니다. 이러한 기능을 사용하면 향후 제품 기능을 조기에 이용할 수 있어 개발 과정에서 고객이 기능을 테스트하고 피드백을 제공할 수 있습니다.*

***Red Hat** 기술 프리뷰 기능의 지원 범위에 대한 자세한 내용은 기술 프리뷰 기능 지원 범위를 참조하십시오.*

BackupLocation유형의 전체 스키마 정의 입니다.

표 4.5. SnapshotLocation

속성	유형	설명
velero	*VolumeSnapshotLocationSpec	볼륨 스냅샷 위치에 설명된 대로 볼륨 스냅샷을 저장할 위치입니다.

SnapshotLocation유형의 전체 스키마 정의 입니다.

표 4.6. ApplicationConfig

속성	유형	설명
velero	*VeleroConfig	Velero 서버의 구성을 정의합니다.
restic	*ResticConfig	Restic 서버의 구성을 정의합니다.

ApplicationConfig유형의 전체 스키마 정의.

표 4.7. VeleroConfig

속성	유형	설명
featureFlags	[] string	Velero 인스턴스에 사용할 기능 목록을 정의합니다.
defaultPlugins	[] string	다음과 같은 기본 Velero 플러그인을 설치할 수 있습니다. **aws**,**azure**,**csi**,**gcp**,**kubevirt**, **openshift**.
customPlugins	[]CustomPlugin	사용자 지정 Velero 플러그인 설치에 사용됩니다. 기본 플러그인 및 사용자 정의 플러그인은 OADP 플러그인에설명되어 있습니다.
restoreResourcesVersionPriority	string	EnableAPIGroupVersions 기능 플래그와 함께 사용할 수 있도록 정의된 경우 생성되는 구성 맵을 나타냅니다.Represents a config map that is created if defined for use in conjunction with the **EnableAPIGroupVersions** feature flag. 이 필드를 정의하면 Velero 서버 기능 플래그에 **EnableAPIGroupVersions** 가 자동으로 추가됩니다.
noDefaultBackupLocation	bool	기본 백업 스토리지 위치 없이 Velero를 설치하려면 설치를 확인하려면 **noDefaultBackupLocation** 플래그를 설정해야 합니다.
podConfig	***PodConfig**	**Velero** 포드의 구성을 정의합니다.

속성	유형	설명
logLevel	string	Velero 서버의 로그 수준(가장 세분화된 로깅에 **debug** 사용)은 Velero 기본값으로 설정되지 않은 상태로 둡니다. 유효한 옵션은 **trace**,**debug**,**info**,**warning**,**error**,**fatal**, **panic** 입니다.

VeleroConfig유형의 전체 스키마 정의.

표 4.8. CustomPlugin

속성	유형	설명
name	string	사용자 정의 플러그인의 이름입니다.
image	string	사용자 지정 플러그인의 이미지입니다.

CustomPlugin유형에 대한 전체 스키마 정의 입니다.

표 4.9. ResticConfig

속성	유형	설명
enable	*bool	**true** 로 설정하면 Restic을 사용하여 백업 및 복원을 활성화합니다. **false** 로 설정하면 스냅샷이 필요합니다.
supplementalGroups	[]int64	**Restic** pod에 적용할 Linux 그룹을 정의합니다.
timeout	string	Restic 타임아웃을 정의하는 사용자가 제공하는 기간 문자열입니다. 기본값은 **1hr** (1시간)입니다. duration 문자열은 10진수 숫자의 부호 있는 시퀀스이며, 각각 선택적인 fraction 및 단위 접미사(예: **300ms**, -1.5h' 또는 **2h45m**)입니다. 유효한 시간 단위는 **ns**,**us** (또는 tekton **s**), **ms**,**s**,**m** 및 **h** 입니다.
podConfig	*PodConfig	**Restic** pod의 구성을 정의합니다.

ResticConfig유형의 전체 스키마 정의.

표 4.10. PodConfig

속성	유형	설명
nodeSelector	map [string] string	**Velero podSpec** 또는 **Restic podSpec** 에 제공할 **nodeSelector** 를 정의합니다.
허용 오차	[] 허용 오차	Velero 배포 또는 Restic **데몬** 세트에 적용할 허용 오차 목록을 정의합니다.
resourceAllocations	ResourceRequirements	Velero CPU 및 메모리 리소스 할당 설정에 설명된 대로 **Velero** Pod 또는 **Restic** 포드에 대한 특정 리소스 **제한** 및 **요청을** 설정합니다.
labels	map [string] string	Pod에 추가할 라벨입니다.

PodConfig유형의 전체 스키마 정의.

표 4.11. 기능

속성	유형	설명
dataMover	***DataMover**	데이터 장애의 구성을 정의합니다.

유형 기능에대한 전체 스키마 정의.

표 4.12. DataMover

속성	유형	설명
enable	bool	**true** 로 설정하면 볼륨 스냅샷 이동기 컨트롤러와 수정된 CSI Data Mover 플러그인을 배포합니다. **false** 로 설정하면 배포되지 않습니다.
credentialName	string	Data Mover에 대한 사용자 제공 Restic **시크릿** 이름입니다.

속성	유형	설명
timeout	string	**VolumeSnapshotBackup** 및 **VolumeSnapshotRestore** 가 완료되는 사용자 제공 기간 문자열입니다. 기본값은 **10m** (10분)입니다. duration 문자열은 10진수 숫자의 부호 있는 시퀀스이며, 각각 선택적인 fraction 및 단위 접미사(예: **300ms**, -1.5h' 또는 **2h45m**)입니다. 유효한 시간 단위는 **ns**,**us** (또는 tekton **s**), **ms**,**s**,**m** 및 **h** 입니다.

OADP API는 OADP Operator 에 더 자세히 설명되어 있습니다.

4.7. 고급 OADP 기능 및 기능

이 문서에서는 OADP(OpenShift API for Data Protection)의 고급 기능 및 기능에 대한 정보를 제공합니다.

4.7.1. 동일한 클러스터에서 다양한 Kubernetes API 버전 작업

4.7.1.1. 클러스터의 Kubernetes API 그룹 버전 나열

소스 클러스터는 이러한 버전 중 하나가 기본 API 버전인 여러 버전의 API를 제공할 수 있습니다. 예를 들어 Example 이라는 API가 있는 소스 클러스터는 example.com/v1 및 example.com/v1beta2 API 그룹에서 사용할 수 있습니다.

Velero를 사용하여 이러한 소스 클러스터를 백업하고 복원하는 경우 Velero는 Kubernetes API의 기본 버전을 사용하는 해당 리소스의 버전만 백업합니다.

위 예제로 돌아가려면 example.com/v1 이 기본 API인 경우 Velero는 example.com/v1 을 사용하는 리소스 버전만 백업합니다. 또한 Velero가 대상 클러스터에서 리소스를 복원하기 위해서는 대상 클러스터에 example.com/v1 이 사용 가능한 API 리소스 세트에 등록되어 있어야 합니다.

따라서 선호하는 API 버전이 사용 가능한 API 리소스 세트에 등록되도록 대상 클러스터에서 Kubernetes API 그룹 버전 목록을 생성해야 합니다.

프로세스

- 다음 명령을 실행합니다.

```
$ oc api-resources
```

4.7.1.2. API 그룹 버전 사용 정보

기본적으로 Velero는 Kubernetes API의 기본 버전을 사용하는 리소스만 백업합니다. 그러나 Velero에는 이러한 제한을 해결하는 기능인 Enable API Group Versions 도 포함되어 있습니다. 소스 클러스터에서 이 기능을 활성화하면 Velero가 기본 기능뿐만 아니라 클러스터에서 지원되는 모든 Kubernetes API 그룹 버전을 백업합니다. 백업 .tar 파일에 버전이 저장된 후 대상 클러스터에서 복원할 수 있습니다.

예를 들어 Example 이라는 API가 있는 소스 클러스터는 example.com/v1 및 example.com/v1beta2 API 그룹에서 사용할 수 있으며 example.com/v1 은 기본 API입니다.

Enable API Group Versions 기능을 활성화하지 않으면 Velero는 예의 기본 API 그룹 버전(예: example.com/v 1)만 백업합니다. 기능이 활성화된 경우 Velero도 example.com/v1beta2 를 백업합니다.

대상 클러스터에서 Enable API Group Versions 기능이 활성화되면 Velero는 API 그룹 버전의 우선 순위 순서에 따라 복원할 버전을 선택합니다.

참고

API 그룹 버전 활성화는 아직 베타 버전입니다.

Velero는 다음 알고리즘을 사용하여 API 버전에 우선 순위를 지정하고 1 를 최상위 우선 순위로 할당합니다.

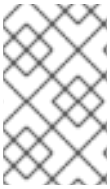
1. 대상 클러스터의 기본 버전
2. source_ 클러스터의 기본 버전
3. Kubernetes 버전 우선 순위가 가장 높은 일반적인 지원되지 않는 버전

추가 리소스

- API 그룹 버전 기능 사용

4.7.1.3. API 그룹 버전 사용

Velero의 Enable API Group Versions 기능을 사용하여 기본 설정뿐만 아니라 클러스터에서 지원되는 모든 Kubernetes API 그룹 버전을 백업할 수 있습니다.



참고

API 그룹 버전 활성화는 아직 베타 버전입니다.

프로세스

- EnableAPIGroupVersions 기능을 구성합니다.

```
apiVersion: oadp.openshift.io/vialpha1
kind: DataProtectionApplication
...
spec:
  configuration:
    velero:
      featureFlags:
      - EnableAPIGroupVersions
```

추가 리소스

- API 그룹 버전 기능 사용

5장. 컨트롤 플레인 백업 및 복원

5.1. ETCD 백업

etcd는 모든 리소스 개체의 상태를 저장하는 OpenShift Container Platform의 키-값 형식의 저장소입니다.

클러스터의 etcd 데이터를 정기적으로 백업하고 OpenShift Container Platform 환경 외부의 안전한 위치에 백업 데이터를 저장하십시오. 설치 후 24 시간 내에 발생하는 첫 번째 인증서 교체가 완료되기 전까지 etcd 백업을 수행하지 마십시오. 인증서 교체가 완료되기 전에 실행하면 백업에 만료된 인증서가 포함됩니다. etcd 스냅샷에 I/O 비용이 높기 때문에 사용량이 많지 않은 동안 etcd 백업을 수행하는 것이 좋습니다.

클러스터를 업그레이드한 후 etcd 백업을 수행해야 합니다. 이는 클러스터를 복원할 때 동일한 z-stream 릴리스에서 가져온 etcd 백업을 사용해야 하므로 중요합니다. 예를 들어 OpenShift Container Platform 4.y.z 클러스터는 4.y.z에서 가져온 etcd 백업을 사용해야 합니다.

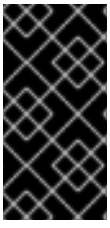
중요

컨트롤 플레인 호스트에서 백업 스크립트를 실행하여 클러스터의 etcd 데이터를 백업합니다. 클러스터의 각 컨트롤 플레인 호스트마다 백업을 수행하지 마십시오.

etcd 백업 후 이전 클러스터 상태로 복원할 수 있습니다.

5.1.1. etcd 데이터 백업

다음 단계에 따라 etcd 스냅샷을 작성하고 정적 pod의 리소스를 백업하여 etcd 데이터를 백업합니다. 이 백업을 저장하여 etcd를 복원해야하는 경우 나중에 사용할 수 있습니다.



중요

단일 컨트롤 플레인 호스트의 백업만 저장합니다. 클러스터의 각 컨트롤 플레인 호스트에서 백업을 수행하지 마십시오.

사전 요구 사항

- cluster-admin 역할의 사용자로 클러스터에 액세스할 수 있어야 합니다.

- 클러스터 전체의 프록시가 활성화되어 있는지 확인해야 합니다.

작은 정보

oc get proxy cluster -o yaml의 출력을 확인하여 프록시가 사용 가능한지 여부를 확인할 수 있습니다. **httpProxy, httpsProxy** 및 **noProxy** 필드에 값이 설정되어 있으면 프록시가 사용됩니다.

프로세스

1. 컨트롤 플레인 노드의 디버그 세션을 시작합니다.

```
$ oc debug node/<node_name>
```

2. 루트 디렉토리를 **/host** 로 변경합니다.

```
sh-4.2# chroot /host
```

3. 클러스터 전체의 프록시가 활성화되어 있는 경우 **NO_PROXY, HTTP_PROXY** 및 **https_proxy** 환경 변수를 내보내고 있는지 확인합니다.

4. **cluster-backup.sh** 스크립트를 실행하고 백업을 저장할 위치를 입력합니다.

작은 정보

cluster-backup.sh 스크립트는 **etcd Cluster Operator**의 구성 요소로 유지 관리되며 **etcdctl snapshot save** 명령 관련 래퍼입니다.

```
sh-4.4# /usr/local/bin/cluster-backup.sh /home/core/assets/backup
```

스크립트 출력 예

```
found latest kube-apiserver: /etc/kubernetes/static-pod-resources/kube-apiserver-pod-
6
found latest kube-controller-manager: /etc/kubernetes/static-pod-resources/kube-
controller-manager-pod-7
```

```
found latest kube-scheduler: /etc/kubernetes/static-pod-resources/kube-scheduler-pod-6
found latest etcd: /etc/kubernetes/static-pod-resources/etcd-pod-3
ede95fe6b88b87ba86a03c15e669fb4aa5bf0991c180d3c6895ce72eaade54a1
etcdctl version: 3.4.14
API version: 3.4
{"level":"info","ts":1624647639.0188997,"caller":"snapshot/v3_snapshot.go:119","msg":"created temporary db file","path":"/home/core/assets/backup/snapshot_2021-06-25_190035.db.part"}
{"level":"info","ts":"2021-06-25T19:00:39.030Z","caller":"clientv3/maintenance.go:200","msg":"opened snapshot stream; downloading"}
{"level":"info","ts":1624647639.0301006,"caller":"snapshot/v3_snapshot.go:127","msg":"fetching snapshot","endpoint":"https://10.0.0.5:2379"}
{"level":"info","ts":"2021-06-25T19:00:40.215Z","caller":"clientv3/maintenance.go:208","msg":"completed snapshot read; closing"}
{"level":"info","ts":1624647640.6032252,"caller":"snapshot/v3_snapshot.go:142","msg":"fetched snapshot","endpoint":"https://10.0.0.5:2379","size":"114 MB","took":1.584090459}
{"level":"info","ts":1624647640.6047094,"caller":"snapshot/v3_snapshot.go:152","msg":"saved","path":"/home/core/assets/backup/snapshot_2021-06-25_190035.db"}
Snapshot saved at /home/core/assets/backup/snapshot_2021-06-25_190035.db
{"hash":3866667823,"revision":31407,"totalKey":12828,"totalSize":114446336}
snapshot db and kube resources are successfully saved to /home/core/assets/backup
```

이 예제에서는 컨트롤 플레인 호스트의 **/home/core/assets/backup/** 디렉토리에 두 개의 파일이 생성됩니다.

- **snapshot_<datetimestamp>.db**:이 파일은 **etcd** 스냅샷입니다. **cluster-backup.sh** 스크립트는 유효성을 확인합니다.

- **static_kuberesources_<datetimestamp>.tar.gz**: 이 파일에는 정적 **pod** 리소스가 포함되어 있습니다. **etcd** 암호화가 활성화되어 있는 경우 **etcd** 스냅 샷의 암호화 키도 포함됩니다.

참고

etcd 암호화가 활성화되어 있는 경우 보안상의 이유로 이 두 번째 파일을 etcd 스냅 샷과 별도로 저장하는 것이 좋습니다. 그러나 이 파일은 etcd 스냅 샷에서 복원하는데 필요합니다.

etcd 암호화는 키가 아닌 값만 암호화합니다. 이는 리소스 유형, 네임 스페이스 및 개체 이름은 암호화되지 않습니다.

5.2. 비정상적인 ETCD 멤버 교체

이 문서는 비정상적인 단일 etcd 멤버를 교체하는 프로세스를 설명합니다.

이 프로세스는 시스템이 실행 중이 아니거나 노드가 준비되지 않았거나 etcd pod가 크래시 루프 상태에있는 등 etcd 멤버의 비정상적인 상태에 따라 다릅니다.

참고

대부분의 컨트롤 플레인 호스트가 손실된 경우 재해 복구 절차에 따라 이 프로세스 대신 이전 클러스터 상태로 복원 합니다.

교체되는 멤버 에서 컨트롤 플레인 인증서가 유효하지 않은 경우 이 절차 대신 만료된 컨트롤 플레인 인증서 복구 절차를 따라야 합니다.

컨트롤 플레인 노드가 유실되고 새 노드가 생성되는 경우 etcd 클러스터 Operator는 새 TLS 인증서 생성 및 노드를 etcd 멤버로 추가하는 프로세스를 진행합니다.

5.2.1. 사전 요구 사항

- 비정상적인 etcd 멤버를 교체하기 전에 etcd 백업을 수행하십시오.

5.2.2. 비정상 etcd 멤버 식별

클러스터에 비정상적인 etcd 멤버가 있는지 여부를 확인할 수 있습니다.

사전 요구 사항

- *cluster-admin 역할의 사용자로 클러스터에 액세스할 수 있어야 합니다.*

프로세스

1. *다음 명령을 사용하여 EtcdMembersAvailable 상태의 상태 조건을 확인하십시오.*

```
$ oc get etcd -o=jsonpath='{range .items[0].status.conditions[?
(@.type=="EtcdMembersAvailable")]}{.message}{"\n"}'
```

2. *출력을 확인합니다.*

```
2 of 3 members are available, ip-10-0-131-183.ec2.internal is unhealthy
```

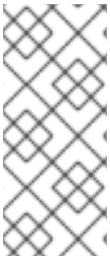
이 출력 예는 ip-10-0-131-183.ec2.internal etcd 멤버가 비정상임을 보여줍니다.

5.2.3. 비정상적인 etcd 멤버의 상태 확인

비정상적인 etcd 멤버를 교체하는 프로세스는 etcd가 다음의 어떤 상태에 있는지에 따라 달라집니다.

- *컴퓨터가 실행 중이 아니거나 노드가 준비되지 않았습니다.*
- *etcd pod가 크래시 루프 상태에 있습니다.*

다음 프로세스에서는 etcd 멤버가 어떤 상태에 있는지를 확인합니다. 이를 통해 비정상 etcd 멤버를 대체하기 위해 수행해야하는 단계를 확인할 수 있습니다.



참고

시스템이 실행되고 있지 않거나 노드가 준비되지 않았지만 곧 정상 상태로 돌아올 것으로 예상되는 경우 etcd 멤버를 교체하기 위한 절차를 수행할 필요가 없습니다. etcd 클러스터 Operator는 머신 또는 노드가 정상 상태로 돌아 오면 자동으로 동기화됩니다.

사전 요구 사항

- *cluster-admin 역할의 사용자로 클러스터에 액세스할 수 있어야 합니다.*
- *비정상적인 etcd 멤버를 식별하고 있습니다.*

프로세스

1. *시스템이 실행되고 있지 않은지를 확인합니다.*

```
$ oc get machines -A -ojsonpath='{range .items[*]}{@.status.nodeRef.name}{"\t"}
{@.status.providerStatus.instanceState}{"\n"}' | grep -v running
```

출력 예

```
ip-10-0-131-183.ec2.internal  stopped (1)
```

(1) *이 출력은 노드와 노드 시스템의 상태를 나열합니다. 상태가 running이 아닌 경우 시스템은 실행되지 않습니다.*

시스템이 실행되고 있지 않은 경우, 시스템이 실행되고 있지 않거나 노드가 준비되지 않은 비정상적인 etcd 멤버 교체 프로세스를 수행하십시오.

2. *노드가 준비되지 않았는지 확인합니다.*

다음 조건 중 하나에 해당하면 노드가 준비되지 않은 것입니다.

- *시스템이 실행중인 경우 노드에 액세스할 수 있는지 확인하십시오.*

```
$ oc get nodes -o jsonpath='{range .items[*]}{"\n"}{.metadata.name}{"\t"}{range
.spec.taints[*]}{.key}{" "}' | grep unreachable
```

출력 예

```
ip-10-0-131-183.ec2.internal node-role.kubernetes.io/master
node.kubernetes.io/unreachable node.kubernetes.io/unreachable 1
```

1 unreachable 상태의 노드가 나열되면 노드가 준비되지 않은 것 입니다.

- 노드에 여전히 액세스할 수 있는 경우 노드가 NotReady로 나열되어 있는지 확인하십시오.

```
$ oc get nodes -l node-role.kubernetes.io/master | grep "NotReady"
```

출력 예

```
ip-10-0-131-183.ec2.internal   NotReady   master   122m   v1.23.0 1
```

1 노드가 NotReady로 표시되면 노드가 준비되지 않은 것입니다.

노드가 준비되지 않은 경우 시스템이 실행되고 있지 않거나 노드가 준비되지 않은 비정상적인 etcd 멤버 교체 프로세스를 수행하십시오.

3. etcd pod가 크래시 루프 상태인지 확인합니다.

시스템이 실행되고 있고 노드가 준비된 경우 etcd pod가 크래시 루프 상태인지 확인하십시오.

a.

모든 컨트롤 플레인 노드가 **Ready** 로 나열되어 있는지 확인합니다.

```
$ oc get nodes -l node-role.kubernetes.io/master
```

출력 예

```
NAME                           STATUS   ROLES    AGE     VERSION
ip-10-0-131-183.ec2.internal   Ready    master   6h13m   v1.23.0
ip-10-0-164-97.ec2.internal    Ready    master   6h13m   v1.23.0
ip-10-0-154-204.ec2.internal   Ready    master   6h13m   v1.23.0
```

b.

etcd pod의 상태가 **Error** 또는 **CrashloopBackoff**인지 확인하십시오.

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

출력 예

```
etcd-ip-10-0-131-183.ec2.internal          2/3     Error     7          6h9m   1
etcd-ip-10-0-164-97.ec2.internal           3/3     Running   0          6h6m
etcd-ip-10-0-154-204.ec2.internal          3/3     Running   0          6h6m
```

1

이 **pod**의 상태는 **Error**이므로 **etcd pod**는 크래시 루프 상태입니다.

etcd pod가 크래시 루프 상태인 경우**etcd pod**가 크래시 루프 상태인 비정상적인 **etcd** 멤버 교체 프로세스를 수행하십시오.

5.2.4. 비정상적인 etcd 멤버 교체

비정상적인 **etcd** 멤버의 상태에 따라 다음 절차 중 하나를 사용합니다.

- 시스템이 실행되고 있지 않거나 노드가 준비되지 않은 비정상적인 **etcd** 멤버 교체
- **etcd pod**가 크래시 루프 상태인 비정상적인 **etcd** 멤버 교체
- 비정상 중지된 **baremetal etcd** 멤버 교체

5.2.4.1. 시스템이 실행되고 있지 않거나 노드가 준비되지 않은 비정상적인 etcd 멤버 교체

다음에서는 시스템이 실행되고 있지 않거나 노드가 준비되지 않은 경우의 비정상적인 **etcd** 멤버를 교체하는 프로세스에 대해 자세히 설명합니다.

전제 조건

- 비정상적인 **etcd** 멤버를 식별했습니다.
- 시스템이 실행되고 있지 않거나 노드가 준비되지 않았음을 확인했습니다.
- **cluster-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.
- **etcd** 백업이 수행되었습니다.

중요

문제가 발생할 경우 클러스터를 복원할 수 있도록 이 프로세스를 수행하기 전에 **etcd** 백업을 수행해야합니다.

프로세스

1. 비정상적인 멤버를 제거합니다.

   a. 영향을 받는 노드에 없는 **pod**를 선택합니다.

      클러스터에 액세스할 수 있는 터미널에서 **cluster-admin** 사용자로 다음 명령을 실행합

니다.

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

출력 예

```
etcd-ip-10-0-131-183.ec2.internal             3/3     Running     0          123m
etcd-ip-10-0-164-97.ec2.internal              3/3     Running     0          123m
etcd-ip-10-0-154-204.ec2.internal             3/3     Running     0          124m
```

b. *실행중인 **etcd** 컨테이너에 연결하고 영향을 받는 노드에 없는 **pod** 이름을 전달합니다.*

*클러스터에 액세스할 수 있는 터미널에서 **cluster-admin** 사용자로 다음 명령을 실행합니다.*

```
$ oc rsh -n openshift-etcd etcd-ip-10-0-154-204.ec2.internal
```

c. *멤버 목록을 확인합니다.*

```
sh-4.2# etcdctl member list -w table
```

출력 예

```
+------------------+---------+--------------------------+--------------------------+--------------------------+
| ID               | STATUS  |          NAME            |       PEER ADDRS         |       CLIENT ADDRS       |
+------------------+---------+--------------------------+--------------------------+--------------------------+
| 6fc1e7c9db35841d | started | ip-10-0-131-183.ec2.internal | https://10.0.131.183:2380 | https://10.0.131.183:2379 |
| 757b6793e2408b6c | started |  ip-10-0-164-97.ec2.internal | https://10.0.164.97:2380 |  https://10.0.164.97:2379 |
| ca8c2990a0aa29d1 | started | ip-10-0-154-204.ec2.internal | https://10.0.154.204:2380 | https://10.0.154.204:2379 |
+------------------+---------+--------------------------+--------------------------+--------------------------+
```

*이러한 값은 프로세스의 뒷부분에서 필요하므로 비정상 **etcd** 멤버의 **ID**와 이름을 기록해 두십시오. **$ etcdctl endpoint health** 명령은 교체 절차가 완료되고 새 멤버가 추가될 때까지 제거된 멤버를 나열합니다.*

d.

*etcdctl member remove 명령에 **ID**를 지정하여 비정상적인 **etcd** 멤버를 제거합니다.*

```
sh-4.2# etcdctl member remove 6fc1e7c9db35841d
```

출력 예

```
Member 6fc1e7c9db35841d removed from cluster ead669ce1fbfb346
```

e.

멤버 목록을 다시 표시하고 멤버가 제거되었는지 확인합니다.

```
sh-4.2# etcdctl member list -w table
```

출력 예

```
+-----------------+---------+-------------------------+-------------------------+--------------
----------+
|       ID        | STATUS  |          NAME           |       PEER ADDRS        |      CLIENT
ADDRS       |
+-----------------+---------+-------------------------+-------------------------+--------------
----------+
| 757b6793e2408b6c | started |  ip-10-0-164-97.ec2.internal |
https://10.0.164.97:2380 |  https://10.0.164.97:2379 |
| ca8c2990a0aa29d1 | started | ip-10-0-154-204.ec2.internal |
https://10.0.154.204:2380 | https://10.0.154.204:2379 |
+-----------------+---------+-------------------------+-------------------------+--------------
----------+
```

이제 노드 쉘을 종료할 수 있습니다.

중요

멤버를 제거한 후 나머지 **etcd** 인스턴스가 재부팅되는 동안 잠시 동안 클러스터에 연결할 수 없습니다.

2. 다음 명령을 입력하여 쿼럼 보호기를 끄십시오.

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides":
{"useUnsupportedUnsafeNonHANonProductionUnstableEtcd": true}}}'
```

이 명령을 사용하면 보안을 다시 생성하고 정적 **Pod**를 롤아웃할 수 있습니다.

3. 삭제된 비정상 **etcd** 멤버의 이전 암호를 제거합니다.

a. 삭제된 비정상 **etcd** 멤버의 시크릿(**secrets**)을 나열합니다.

```
$ oc get secrets -n openshift-etcd | grep ip-10-0-131-183.ec2.internal 1
```

1 이 프로세스의 앞부분에서 기록한 비정상 **etcd** 멤버의 이름을 전달합니다.

다음 출력에 표시된대로 피어, 서빙 및 메트릭 시크릿이 있습니다.

출력 예

```
etcd-peer-ip-10-0-131-183.ec2.internal             kubernetes.io/tls                     2
47m
etcd-serving-ip-10-0-131-183.ec2.internal          kubernetes.io/tls                     2
47m
etcd-serving-metrics-ip-10-0-131-183.ec2.internal  kubernetes.io/tls                     2
47m
```

b.

*제거된 비정상 **etcd** 멤버의 시크릿을 삭제합니다.*

i.

피어 시크릿을 삭제합니다.

```
$ oc delete secret -n openshift-etcd etcd-peer-ip-10-0-131-183.ec2.internal
```

ii.

서빙 시크릿을 삭제합니다.

```
$ oc delete secret -n openshift-etcd etcd-serving-ip-10-0-131-183.ec2.internal
```

iii.

메트릭 시크릿을 삭제합니다.

```
$ oc delete secret -n openshift-etcd etcd-serving-metrics-ip-10-0-131-183.ec2.internal
```

4.

*컨트롤 플레인 시스템을 삭제하고 다시 생성합니다. 이 시스템을 다시 만든 후에는 새 버전이 강제 실행되고 **etcd**는 자동으로 확장됩니다.*

*설치 프로그램에서 제공한 인프라를 실행 중이거나 **Machine API**를 사용하여 컴퓨터를 만든 경우 다음 단계를 수행합니다. 그렇지 않으면 원래 마스터를 만들 때 사용한 방법과 동일한 방법을 사용하여 새 마스터를 작성해야합니다.*

a.

비정상 멤버의 컴퓨터를 가져옵니다.

*클러스터에 액세스할 수 있는 터미널에서 **cluster-admin** 사용자로 다음 명령을 실행합니다.*

```
$ oc get machines -n openshift-machine-api -o wide
```

출력 예

```
NAME                              PHASE     TYPE        REGION      ZONE         AGE
NODE                          PROVIDERID                          STATE
clustername-8qw5l-master-0        Running   m4.xlarge   us-east-1   us-east-1a
```

```
3h37m  ip-10-0-131-183.ec2.internal  aws:///us-east-1a/i-0ec2782f8287dfb7e
stopped (1)
clustername-8qw5l-master-1                 Running  m4.xlarge  us-east-1  us-east-1b
3h37m  ip-10-0-154-204.ec2.internal  aws:///us-east-1b/i-096c349b700a19631
running
clustername-8qw5l-master-2                 Running  m4.xlarge  us-east-1  us-east-1c
3h37m  ip-10-0-164-97.ec2.internal  aws:///us-east-1c/i-02626f1dba9ed5bba
running
clustername-8qw5l-worker-us-east-1a-wbtgd  Running  m4.large  us-east-1  us-
east-1a  3h28m  ip-10-0-129-226.ec2.internal  aws:///us-east-1a/i-
010ef6279b4662ced  running
clustername-8qw5l-worker-us-east-1b-lrdxb  Running  m4.large  us-east-1  us-
east-1b  3h28m  ip-10-0-144-248.ec2.internal  aws:///us-east-1b/i-
0cb45ac45a166173b  running
clustername-8qw5l-worker-us-east-1c-pkg26  Running  m4.large  us-east-1  us-
east-1c  3h28m  ip-10-0-170-181.ec2.internal  aws:///us-east-1c/i-
06861c00007751b0a  running
```

(1)

*이는 비정상 노드의 컨트롤 플레인 시스템 **ip-10-0-131-183.ec2.internal**입니다.*

b.

시스템 설정을 파일 시스템의 파일에 저장합니다.

```
$ oc get machine clustername-8qw5l-master-0 \ (1)
  -n openshift-machine-api \
  -o yaml \
  > new-master-machine.yaml
```

(1)

비정상 노드의 컨트롤 플레인 시스템의 이름을 지정합니다.

c.

*이전 단계에서 만든 **new-master-machine.yaml** 파일을 편집하여 새 이름을 할당하고 불필요한 필드를 제거합니다.*

i.

*전체 **status** 섹션을 삭제합니다.*

```
status:
  addresses:
  - address: 10.0.131.183
    type: InternalIP
```

```
  - address: ip-10-0-131-183.ec2.internal
    type: InternalDNS
  - address: ip-10-0-131-183.ec2.internal
    type: Hostname
  lastUpdated: "2020-04-20T17:44:29Z"
  nodeRef:
    kind: Node
    name: ip-10-0-131-183.ec2.internal
    uid: acca4411-af0d-4387-b73e-52b2484295ad
  phase: Running
  providerStatus:
    apiVersion: awsproviderconfig.openshift.io/v1beta1
    conditions:
    - lastProbeTime: "2020-04-20T16:53:50Z"
      lastTransitionTime: "2020-04-20T16:53:50Z"
      message: machine successfully created
      reason: MachineCreationSucceeded
      status: "True"
      type: MachineCreation
    instanceId: i-0fdb85790d76d0c3f
    instanceState: stopped
    kind: AWSMachineProviderStatus
```

ii. **metadata.name** 필드를 새 이름으로 변경합니다.

이전 시스템과 동일한 기본 이름을 유지하고 마지막 번호를 사용 가능한 다음 번호로 변경하는 것이 좋습니다. 이 예에서 **clustername-8qw5l-master-0**은 **clustername-8qw5l-master-3**으로 변경되어 있습니다.

예를 들어 다음과 같습니다.

```
apiVersion: machine.openshift.io/v1beta1
kind: Machine
metadata:
  ...
  name: clustername-8qw5l-master-3
  ...
```

iii. **spec.providerID** 필드를 삭제합니다.

```
  providerID: aws:///us-east-1a/i-0fdb85790d76d0c3f
```

d. 다음 명령을 실행하여 **BareMetalHost** 오브젝트를 삭제하고 **< host_name >**을 비정상 노드의 베어 메탈 호스트 이름으로 교체합니다.

```
$ oc delete bmh -n openshift-machine-api <host_name>
```

e. *다음 명령을 실행하여 비정상 멤버의 머신을 삭제하고 **< machine_name >**을 비정상 노드의 컨트롤 플레인 시스템의 이름으로 교체합니다(예: **clustername-8qw5l-master -0**).*

```
$ oc delete machine -n openshift-machine-api <machine_name>
```

f. *시스템이 삭제되었는지 확인합니다.*

```
$ oc get machines -n openshift-machine-api -o wide
```

출력 예

```
NAME                                 PHASE     TYPE        REGION      ZONE         AGE
NODE                           PROVIDERID                          STATE
clustername-8qw5l-master-1           Running   m4.xlarge   us-east-1   us-east-1b   3h37m   ip-10-0-154-204.ec2.internal   aws:///us-east-1b/i-096c349b700a19631   running
clustername-8qw5l-master-2           Running   m4.xlarge   us-east-1   us-east-1c   3h37m   ip-10-0-164-97.ec2.internal    aws:///us-east-1c/i-02626f1dba9ed5bba   running
clustername-8qw5l-worker-us-east-1a-wbtgd   Running   m4.large   us-east-1   us-east-1a   3h28m   ip-10-0-129-226.ec2.internal   aws:///us-east-1a/i-010ef6279b4662ced   running
clustername-8qw5l-worker-us-east-1b-lrdxb   Running   m4.large   us-east-1   us-east-1b   3h28m   ip-10-0-144-248.ec2.internal   aws:///us-east-1b/i-0cb45ac45a166173b   running
clustername-8qw5l-worker-us-east-1c-pkg26   Running   m4.large   us-east-1   us-east-1c   3h28m   ip-10-0-170-181.ec2.internal   aws:///us-east-1c/i-06861c00007751b0a   running
```

g. ***new-master-machine.yaml** 파일을 사용하여 새 시스템을 만듭니다.*

```
$ oc apply -f new-master-machine.yaml
```

h. *새 시스템이 생성되었는지 확인합니다.*

```
$ oc get machines -n openshift-machine-api -o wide
```

출력 예

```
NAME                                 PHASE          TYPE        REGION      ZONE         AGE
NODE                           PROVIDERID                          STATE
clustername-8qw5l-master-1           Running        m4.xlarge   us-east-1   us-east-
1b   3h37m   ip-10-0-154-204.ec2.internal   aws:///us-east-1b/i-096c349b700a19631
running
clustername-8qw5l-master-2           Running        m4.xlarge   us-east-1   us-east-
1c   3h37m   ip-10-0-164-97.ec2.internal    aws:///us-east-1c/i-02626f1dba9ed5bba
running
clustername-8qw5l-master-3           Provisioning   m4.xlarge   us-east-1   us-
east-1a  85s     ip-10-0-133-53.ec2.internal    aws:///us-east-1a/i-015b0888fe17bc2c8
running (1)
clustername-8qw5l-worker-us-east-1a-wbtgd  Running      m4.large    us-east-1
us-east-1a  3h28m   ip-10-0-129-226.ec2.internal   aws:///us-east-1a/i-
010ef6279b4662ced   running
clustername-8qw5l-worker-us-east-1b-lrdxb   Running      m4.large    us-east-1
us-east-1b  3h28m   ip-10-0-144-248.ec2.internal   aws:///us-east-1b/i-
0cb45ac45a166173b   running
clustername-8qw5l-worker-us-east-1c-pkg26   Running      m4.large    us-east-1
us-east-1c  3h28m   ip-10-0-170-181.ec2.internal   aws:///us-east-1c/i-
06861c00007751b0a   running
```

(1)

새 시스템 **clustername-8qw5l-master-3**이 생성되고 단계가 **Provisioning**에서 **Running**으로 변경되면 시스템이 준비 상태가 됩니다.

새 시스템을 만드는 데 몇 분이 소요될 수 있습니다. **etcd** 클러스터 **Operator**는 머신 또는 노드가 정상 상태로 돌아 오면 자동으로 동기화됩니다.

5. 다음 명령을 입력하여 쿼럼 보호기를 다시 켭니다.

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides":
null}}'
```

6. 다음 명령을 입력하여 **unsupportedConfigOverrides** 섹션이 오브젝트에서 제거되었는지 확인할 수 있습니다.

```
$ oc get etcd/cluster -oyaml
```

7.

단일 노드 OpenShift를 사용하는 경우 노드를 다시 시작합니다. 그렇지 않으면 etcd 클러스터 Operator에서 다음 오류가 발생할 수 있습니다.

출력 예

```
EtcdCertSignerControllerDegraded: [Operation cannot be fulfilled on secrets "etcd-peer-sno-0": the object has been modified; please apply your changes to the latest version and try again, Operation cannot be fulfilled on secrets "etcd-serving-sno-0": the object has been modified; please apply your changes to the latest version and try again, Operation cannot be fulfilled on secrets "etcd-serving-metrics-sno-0": the object has been modified; please apply your changes to the latest version and try again]
```

검증

1. 모든 etcd pod가 올바르게 실행되고 있는지 확인합니다.

   클러스터에 액세스할 수 있는 터미널에서 cluster-admin 사용자로 다음 명령을 실행합니다.

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

출력 예

```
etcd-ip-10-0-133-53.ec2.internal          3/3     Running     0          7m49s
etcd-ip-10-0-164-97.ec2.internal          3/3     Running     0          123m
etcd-ip-10-0-154-204.ec2.internal         3/3     Running     0          124m
```

이전 명령의 출력에 두 개의 pod만 나열되는 경우 수동으로 etcd 재배포를 강제 수행할 수 있습니다. 클러스터에 액세스할 수 있는 터미널에서 cluster-admin 사용자로 다음 명령을 실행합니다.

```
$ oc patch etcd cluster -p='{"spec": {"forceRedeploymentReason": "recovery-'"$( date --rfc-3339=ns )"'"}}' --type=merge (1)
```

1

forceRedeploymentReason 값은 고유해야하므로 타임 스탬프가 추가됩니다.

2. *정확히 세 개의 etcd 멤버가 있는지 확인합니다.*

a. *실행중인 etcd 컨테이너에 연결하고 영향을 받는 노드에 없는 pod 이름을 전달합니다.*

클러스터에 액세스할 수 있는 터미널에서 cluster-admin 사용자로 다음 명령을 실행합니다.

```
$ oc rsh -n openshift-etcd etcd-ip-10-0-154-204.ec2.internal
```

b. *멤버 목록을 확인합니다.*

```
sh-4.2# etcdctl member list -w table
```

출력 예

```
+-----------------+---------+--------------------------+--------------------------+--------------
----------+
|        ID       | STATUS  |          NAME            |       PEER ADDRS         |      CLIENT
ADDRS       |
+-----------------+---------+--------------------------+--------------------------+--------------
----------+
| 5eb0d6b8ca24730c | started |  ip-10-0-133-53.ec2.internal |
https://10.0.133.53:2380 |  https://10.0.133.53:2379 |
| 757b6793e2408b6c | started |  ip-10-0-164-97.ec2.internal |
https://10.0.164.97:2380 |  https://10.0.164.97:2379 |
| ca8c2990a0aa29d1 | started | ip-10-0-154-204.ec2.internal |
https://10.0.154.204:2380 | https://10.0.154.204:2379 |
+-----------------+---------+--------------------------+--------------------------+--------------
----------+
```

이전 명령의 출력에 세 개 이상의 etcd 멤버가 나열된 경우 원하지 않는 멤버를 신중하게 제거해야 합니다.

주의

*올바른 **etcd** 멤버를 제거하십시오. **etcd** 멤버를 제거하면 쿼럼이 손실될 수 있습니다.*

5.2.4.2. etcd pod가 크래시 루프 상태인 비정상적인 etcd 멤버 교체

*이 단계에서는 **etcd pod**가 크래시 루프 상태에 있는 경우 비정상 **etcd** 멤버를 교체하는 방법을 설명합니다.*

사전 요구 사항

- *비정상적인 **etcd** 멤버를 식별했습니다.*
- ***etcd pod**가 크래시 루프 상태에 있는것으로 확인되었습니다.*
- ***cluster-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.*
- ***etcd** 백업이 수행되었습니다.*

중요

*문제가 발생할 경우 클러스터를 복원할 수 있도록 이 프로세스를 수행하기 전에 **etcd** 백업을 수행해야합니다.*

절차

1. *크래시 루프 상태에 있는 **etcd pod**를 중지합니다.*

   a. *크래시 루프 상태의 노드를 디버깅합니다.*

   *클러스터에 액세스할 수 있는 터미널에서 **cluster-admin** 사용자로 다음 명령을 실행합니다.*

```
$ oc debug node/ip-10-0-131-183.ec2.internal 1
```

1

이를 비정상 노드의 이름으로 변경합니다.

b.

루트 디렉토리를 /host 로 변경합니다.

```
sh-4.2# chroot /host
```

c.

kubelet 매니페스트 디렉토리에서 기존 etcd pod 파일을 이동합니다.

```
sh-4.2# mkdir /var/lib/etcd-backup
```

```
sh-4.2# mv /etc/kubernetes/manifests/etcd-pod.yaml /var/lib/etcd-backup/
```

d.

etcd 데이터 디렉토리를 다른 위치로 이동합니다.

```
sh-4.2# mv /var/lib/etcd/ /tmp
```

이제 노드 쉘을 종료할 수 있습니다.

2.

비정상적인 멤버를 제거합니다.

a.

영향을 받는 노드에 없는 pod를 선택합니다.

클러스터에 액세스할 수 있는 터미널에서 cluster-admin 사용자로 다음 명령을 실행합니다.

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

출력 예

```
etcd-ip-10-0-131-183.ec2.internal             2/3     Error       7          6h9m
etcd-ip-10-0-164-97.ec2.internal              3/3     Running     0          6h6m
etcd-ip-10-0-154-204.ec2.internal             3/3     Running     0          6h6m
```

b.

*실행중인 **etcd** 컨테이너에 연결하고 영향을 받는 노드에 없는 **pod** 이름을 전달합니다.*

*클러스터에 액세스할 수 있는 터미널에서 **cluster-admin** 사용자로 다음 명령을 실행합니다.*

```
$ oc rsh -n openshift-etcd etcd-ip-10-0-154-204.ec2.internal
```

c.

멤버 목록을 확인합니다.

```
sh-4.2# etcdctl member list -w table
```

출력 예

```
+------------------+---------+--------------------------+--------------------------+--------------
----------+
|        ID        | STATUS  |          NAME            |       PEER ADDRS         |      CLIENT
ADDRS       |
+------------------+---------+--------------------------+--------------------------+--------------
----------+
| 62bcf33650a7170a | started | ip-10-0-131-183.ec2.internal |
https://10.0.131.183:2380 | https://10.0.131.183:2379 |
| b78e2856655bc2eb | started |  ip-10-0-164-97.ec2.internal |
https://10.0.164.97:2380 |  https://10.0.164.97:2379 |
| d022e10b498760d5 | started | ip-10-0-154-204.ec2.internal |
https://10.0.154.204:2380 | https://10.0.154.204:2379 |
+------------------+---------+--------------------------+--------------------------+--------------
----------+
```

*이러한 값은 프로세스의 뒷부분에서 필요하므로 비정상 **etcd** 멤버의 **ID**와 이름을 기록해 두십시오.*

d. *etcdctl member remove 명령에 ID를 지정하여 비정상적인 etcd 멤버를 제거합니다.*

```
sh-4.2# etcdctl member remove 62bcf33650a7170a
```

출력 예

```
Member 62bcf33650a7170a removed from cluster ead669ce1fbfb346
```

e. *멤버 목록을 다시 표시하고 멤버가 제거되었는지 확인합니다.*

```
sh-4.2# etcdctl member list -w table
```

출력 예

```
+------------------+---------+--------------------------+--------------------------+------------
----------+
|        ID        | STATUS  |          NAME            |       PEER ADDRS         |      CLIENT
ADDRS       |
+------------------+---------+--------------------------+--------------------------+------------
----------+
| b78e2856655bc2eb | started |  ip-10-0-164-97.ec2.internal |
https://10.0.164.97:2380 |  https://10.0.164.97:2379 |
| d022e10b498760d5 | started | ip-10-0-154-204.ec2.internal |
https://10.0.154.204:2380 | https://10.0.154.204:2379 |
+------------------+---------+--------------------------+--------------------------+------------
----------+
```

이제 노드 쉘을 종료할 수 있습니다.

3. *다음 명령을 입력하여 쿼럼 보호기를 끄십시오.*

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides":
{"useUnsupportedUnsafeNonHANonProductionUnstableEtcd": true}}}'
```

*이 명령을 사용하면 보안을 다시 생성하고 정적 **Pod**를 롤아웃할 수 있습니다.*

4. *삭제된 비정상 **etcd** 멤버의 이전 암호를 제거합니다.*

a. *삭제된 비정상 **etcd** 멤버의 시크릿(**secrets**)을 나열합니다.*

```
$ oc get secrets -n openshift-etcd | grep ip-10-0-131-183.ec2.internal 1
```

1 *이 프로세스의 앞부분에서 기록한 비정상 **etcd** 멤버의 이름을 전달합니다.*

다음 출력에 표시된대로 피어, 서빙 및 메트릭 시크릿이 있습니다.

출력 예

```
etcd-peer-ip-10-0-131-183.ec2.internal             kubernetes.io/tls                     2
47m
etcd-serving-ip-10-0-131-183.ec2.internal          kubernetes.io/tls                     2
47m
etcd-serving-metrics-ip-10-0-131-183.ec2.internal  kubernetes.io/tls                     2
47m
```

b. *제거된 비정상 **etcd** 멤버의 시크릿을 삭제합니다.*

i. *피어 시크릿을 삭제합니다.*

```
$ oc delete secret -n openshift-etcd etcd-peer-ip-10-0-131-183.ec2.internal
```

ii. *서빙 시크릿을 삭제합니다.*

```
$ oc delete secret -n openshift-etcd etcd-serving-ip-10-0-131-183.ec2.internal
```

iii. 메트릭 시크릿을 삭제합니다.

```
$ oc delete secret -n openshift-etcd etcd-serving-metrics-ip-10-0-131-183.ec2.internal
```

5. etcd를 강제로 재배포합니다.

클러스터에 액세스할 수 있는 터미널에서 cluster-admin 사용자로 다음 명령을 실행합니다.

```
$ oc patch etcd cluster -p='{"spec": {"forceRedeploymentReason": "single-master-recovery-'"$( date --rfc-3339=ns )"'"}}' --type=merge 1
```

1 forceRedeploymentReason 값은 고유해야하므로 타임 스탬프가 추가됩니다.

etcd 클러스터 Operator가 재배포를 수행하면 모든 컨트롤 플레인 노드가 etcd pod가 작동하는지 확인합니다.

6. 다음 명령을 입력하여 쿼럼 보호기를 다시 켭니다.

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": null}}'
```

7. 다음 명령을 입력하여 unsupportedConfigOverrides 섹션이 오브젝트에서 제거되었는지 확인할 수 있습니다.

```
$ oc get etcd/cluster -oyaml
```

8. 단일 노드 OpenShift를 사용하는 경우 노드를 다시 시작합니다. 그렇지 않으면 etcd 클러스터 Operator에서 다음 오류가 발생할 수 있습니다.

출력 예

```
EtcdCertSignerControllerDegraded: [Operation cannot be fulfilled on secrets "etcd-peer-sno-0": the object has been modified; please apply your changes to the latest version and try again, Operation cannot be fulfilled on secrets "etcd-serving-sno-0":
```

```
the object has been modified; please apply your changes to the latest version and try
again, Operation cannot be fulfilled on secrets "etcd-serving-metrics-sno-0": the
object has been modified; please apply your changes to the latest version and try
again]
```

검증

- 새 멤버가 사용 가능하고 정상적인 상태에 있는지 확인합니다.

a. 실행중인 **etcd** 컨테이너에 다시 연결합니다.

cluster-admin 사용자로 클러스터에 액세스할 수 있는 터미널에서 다음 명령을 실행합니다.

```
$ oc rsh -n openshift-etcd etcd-ip-10-0-154-204.ec2.internal
```

b. 모든 멤버가 정상인지 확인합니다.

```
sh-4.2# etcdctl endpoint health
```

출력 예

```
https://10.0.131.183:2379 is healthy: successfully committed proposal: took =
16.671434ms
https://10.0.154.204:2379 is healthy: successfully committed proposal: took =
16.698331ms
https://10.0.164.97:2379 is healthy: successfully committed proposal: took =
16.621645ms
```

5.2.4.3. 시스템이 실행되고 있지 않거나 노드가 준비되지 않은 비정상적인 베어 메탈 etcd 멤버 교체

이 프로세스에서는 시스템이 실행되고 있지 않거나 노드가 준비되지 않았기 때문에 비정상 상태의 베어 메탈 **etcd** 멤버를 교체하는 단계를 자세히 설명합니다.

*설치 관리자 프로비저닝 인프라를 실행 중이거나 **Machine API**를 사용하여 머신을 생성한 경우 다음 단계를 따르십시오. 그렇지 않으면 원래 생성하는 데 사용된 방법과 동일한 방법으로 새 컨트롤 플레인 노드를 생성해야 합니다.*

사전 요구 사항

- *비정상적인 베어 메탈 **etcd** 멤버를 식별했습니다.*
- *시스템이 실행되고 있지 않거나 노드가 준비되지 않았음을 확인했습니다.*
- ***cluster-admin** 역할의 사용자로 클러스터에 액세스할 수 있습니다.*
- ***etcd** 백업이 수행되었습니다.*

중요

*문제가 발생할 경우 클러스터를 복원할 수 있도록 이 단계를 수행하기 전에 **etcd** 백업을 수행해야합니다.*

절차

1. *비정상 멤버를 확인하고 제거합니다.*

   a. *영향을 받는 노드에 없는 **pod**를 선택합니다.*

   *클러스터에 액세스할 수 있는 터미널에서 **cluster-admin** 사용자로 다음 명령을 실행합니다.*

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd -o wide
```

   출력 예

```
etcd-openshift-control-plane-0   5/5   Running   11   3h56m   192.168.10.9
openshift-control-plane-0  <none>          <none>
etcd-openshift-control-plane-1   5/5   Running   0    3h54m   192.168.10.10
```

```
openshift-control-plane-1   <none>          <none>
etcd-openshift-control-plane-2   5/5   Running   0   3h58m   192.168.10.11
openshift-control-plane-2   <none>          <none>
```

b.

*실행중인 **etcd** 컨테이너에 연결하고 영향을 받는 노드에 없는 **pod** 이름을 전달합니다.*

*클러스터에 액세스할 수 있는 터미널에서 **cluster-admin** 사용자로 다음 명령을 실행합니다.*

```
$ oc rsh -n openshift-etcd etcd-openshift-control-plane-0
```

c.

멤버 목록을 확인합니다.

```
sh-4.2# etcdctl member list -w table
```

출력 예

```
+-----------------+---------+------------------+-------------------------+-------------------------
-+--------------------+
| ID              | STATUS  | NAME             | PEER ADDRS              | CLIENT
ADDRS              | IS LEARNER |
+-----------------+---------+------------------+-------------------------+-------------------------
-+--------------------+
| 7a8197040a5126c8 | started | openshift-control-plane-2 |
https://192.168.10.11:2380/ | https://192.168.10.11:2379/ | false |
| 8d5abe9669a39192 | started | openshift-control-plane-1 |
https://192.168.10.10:2380/ | https://192.168.10.10:2379/ | false |
| cc3830a72fc357f9 | started | openshift-control-plane-0 | https://192.168.10.9:2380/ |
https://192.168.10.9:2379/   | false |
+-----------------+---------+------------------+-------------------------+-------------------------
-+--------------------+
```

*이러한 값은 프로세스의 뒷부분에서 필요하므로 비정상 **etcd** 멤버의 **ID**와 이름을 기록해 두십시오. **etcdctl endpoint health** 명령은 교체 절차가 완료되고 새 멤버가 추가될 때까지 제거된 멤버를 나열합니다.*

.

d. *etcdctl member remove 명령에 ID를 지정하여 비정상적인 etcd 멤버를 제거합니다.*

> **주의**
>
> *올바른 etcd 멤버를 제거하십시오. etcd 멤버를 제거하면 쿼럼이 손실될 수 있습니다.*

```
sh-4.2# etcdctl member remove 7a8197040a5126c8
```

출력 예

```
Member 7a8197040a5126c8 removed from cluster b23536c33f2cdd1b
```

e. *멤버 목록을 다시 표시하고 멤버가 제거되었는지 확인합니다.*

```
sh-4.2# etcdctl member list -w table
```

출력 예

```
+------------------+---------+------------------+--------------------------+--------------------------
-+-----------------------+
| ID               | STATUS  | NAME             | PEER ADDRS               | CLIENT
ADDRS                | IS LEARNER |
+------------------+---------+------------------+--------------------------+--------------------------
-+-----------------------+
| 7a8197040a5126c8 | started | openshift-control-plane-2 |
https://192.168.10.11:2380/ | https://192.168.10.11:2379/ | false |
| 8d5abe9669a39192 | started | openshift-control-plane-1 |
https://192.168.10.10:2380/ | https://192.168.10.10:2379/ | false |
+------------------+---------+------------------+--------------------------+--------------------------
-+-----------------------+
```

이제 노드 쉘을 종료할 수 있습니다.

중요

*멤버를 제거한 후 나머지 **etcd** 인스턴스가 재부팅되는 동안 잠시 동안 클러스터에 연결할 수 없습니다.*

2. *다음 명령을 입력하여 쿼럼 보호기를 끄십시오.*

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides":
{"useUnsupportedUnsafeNonHANonProductionUnstableEtcd": true}}}'
```

*이 명령을 사용하면 보안을 다시 생성하고 정적 **Pod**를 롤아웃할 수 있습니다.*

3. *다음 명령을 실행하여 제거된 비정상 **etcd** 멤버의 이전 암호를 제거합니다.*

   a. *삭제된 비정상 **etcd** 멤버의 시크릿(**secrets**)을 나열합니다.*

```
$ oc get secrets -n openshift-etcd | grep openshift-control-plane-2
```

*이 프로세스의 앞부분에서 기록한 비정상 **etcd** 멤버의 이름을 전달합니다.*

다음 출력에 표시된대로 피어, 서빙 및 메트릭 시크릿이 있습니다.

```
etcd-peer-openshift-control-plane-2             kubernetes.io/tls   2   134m
etcd-serving-metrics-openshift-control-plane-2  kubernetes.io/tls   2   134m
etcd-serving-openshift-control-plane-2          kubernetes.io/tls   2   134m
```

   b. *제거된 비정상 **etcd** 멤버의 시크릿을 삭제합니다.*

      i. *피어 시크릿을 삭제합니다.*

```
$ oc delete secret etcd-peer-openshift-control-plane-2 -n openshift-etcd

secret "etcd-peer-openshift-control-plane-2" deleted
```

ii. 서빙 시크릿을 삭제합니다.

```
$ oc delete secret etcd-serving-metrics-openshift-control-plane-2 -n openshift-etcd

secret "etcd-serving-metrics-openshift-control-plane-2" deleted
```

iii. 메트릭 시크릿을 삭제합니다.

```
$ oc delete secret etcd-serving-openshift-control-plane-2 -n openshift-etcd

secret "etcd-serving-openshift-control-plane-2" deleted
```

4. 컨트롤 플레인 시스템을 삭제합니다.

설치 프로그램에서 제공한 인프라를 실행 중이거나 **Machine API**를 사용하여 컴퓨터를 만든 경우 다음 단계를 수행합니다. 그렇지 않으면 원래 생성하는 데 사용된 방법과 동일한 방법으로 새 컨트롤 플레인 노드를 생성해야 합니다.

a. 비정상 멤버의 컴퓨터를 가져옵니다.

클러스터에 액세스할 수 있는 터미널에서 **cluster-admin** 사용자로 다음 명령을 실행합니다.

```
$ oc get machines -n openshift-machine-api -o wide
```

출력 예

```
NAME                        PHASE     TYPE   REGION   ZONE   AGE     NODE
PROVIDERID                                                        STATE
examplecluster-control-plane-0   Running                 3h11m   openshift-
control-plane-0   baremetalhost:///openshift-machine-api/openshift-control-plane-
0/da1ebe11-3ff2-41c5-b099-0aa41222964e   externally provisioned (1)
examplecluster-control-plane-1   Running                 3h11m   openshift-
control-plane-1   baremetalhost:///openshift-machine-api/openshift-control-plane-
1/d9f9acbc-329c-475e-8d81-03b20280a3e1   externally provisioned
examplecluster-control-plane-2   Running                 3h11m   openshift-
```

```
control-plane-2   baremetalhost:///openshift-machine-api/openshift-control-plane-
2/3354bdac-61d8-410f-be5b-6a395b056135   externally provisioned
examplecluster-compute-0        Running                 165m    openshift-
compute-0       baremetalhost:///openshift-machine-api/openshift-compute-
0/3d685b81-7410-4bb3-80ec-13a31858241f       provisioned
examplecluster-compute-1        Running                 165m    openshift-
compute-1       baremetalhost:///openshift-machine-api/openshift-compute-
1/0fdae6eb-2066-4241-91dc-e7ea72ab13b9       provisioned
```

1

비정상 노드의 컨트롤 플레인 시스템 예**cluster-control-plane-2.**

b.

시스템 설정을 파일 시스템의 파일에 저장합니다.

```
$ oc get machine examplecluster-control-plane-2 \ 1
  -n openshift-machine-api \
  -o yaml \
  > new-master-machine.yaml
```

1

비정상 노드의 컨트롤 플레인 시스템의 이름을 지정합니다.

c.

이전 단계에서 만든 **new-master-machine.yaml** 파일을 편집하여 새 이름을 할당하고 불필요한 필드를 제거합니다.

i.

전체 **status** 섹션을 삭제합니다.

```
status:
  addresses:
  - address: ""
    type: InternalIP
  - address: fe80::4adf:37ff:feb0:8aa1%ens1f1.373
    type: InternalDNS
  - address: fe80::4adf:37ff:feb0:8aa1%ens1f1.371
    type: Hostname
  lastUpdated: "2020-04-20T17:44:29Z"
  nodeRef:
    kind: Machine
    name: fe80::4adf:37ff:feb0:8aa1%ens1f1.372
    uid: acca4411-af0d-4387-b73e-52b2484295ad
```

```
  phase: Running
  providerStatus:
    apiVersion: machine.openshift.io/v1beta1
    conditions:
    - lastProbeTime: "2020-04-20T16:53:50Z"
      lastTransitionTime: "2020-04-20T16:53:50Z"
      message: machine successfully created
      reason: MachineCreationSucceeded
      status: "True"
      type: MachineCreation
    instanceId: i-0fdb85790d76d0c3f
    instanceState: stopped
    kind: Machine
```

5. *metadata.name 필드를 새 이름으로 변경합니다.*

   이전 시스템과 동일한 기본 이름을 유지하고 마지막 번호를 사용 가능한 다음 번호로 변경하는 것이 좋습니다. 이 예에서 examplecluster-control-plane-2 는 examplecluster-control-plane-3 으로 변경되었습니다.

   예를 들어 다음과 같습니다.

```
apiVersion: machine.openshift.io/v1beta1
kind: Machine
metadata:
  ...
  name: examplecluster-control-plane-3
  ...
```

   a. *spec.providerID 필드를 삭제합니다.*

```
  providerID: baremetalhost:///openshift-machine-api/openshift-control-plane-2/3354bdac-61d8-410f-be5b-6a395b056135
```

   b. *metadata.annotations 및 metadata.generation 필드를 제거합니다.*

```
annotations:
  machine.openshift.io/instance-state: externally provisioned
...
generation: 2
```

   c. *spec.conditions,spec.lastUpdated,spec.nodeRef 및 spec.phase 필드를 제거합니다.*

```
  lastTransitionTime: "2022-08-03T08:40:36Z"
message: 'Drain operation currently blocked by: [{Name:EtcdQuorumOperator
Owner:clusteroperator/etcd}]'
reason: HookPresent
severity: Warning
status: "False"

type: Drainable
lastTransitionTime: "2022-08-03T08:39:55Z"
status: "True"
type: InstanceExists

lastTransitionTime: "2022-08-03T08:36:37Z"
status: "True"
type: Terminable
lastUpdated: "2022-08-03T08:40:36Z"
nodeRef:
kind: Node
name: openshift-control-plane-2
uid: 788df282-6507-4ea2-9a43-24f237ccbc3c
phase: Running
```

6. 다음 명령을 실행하여 **Bare Metal Operator**를 사용할 수 있는지 확인합니다.

```
$ oc get clusteroperator baremetal
```

출력 예

```
NAME        VERSION   AVAILABLE   PROGRESSING   DEGRADED   SINCE   MESSAGE
baremetal   4.10.x    True        False         False      3d15h
```

7. 다음 명령을 실행하여 이전 **BareMetalHost** 오브젝트를 제거합니다.

```
$ oc delete bmh openshift-control-plane-2 -n openshift-machine-api
```

출력 예

```
baremetalhost.metal3.io "openshift-control-plane-2" deleted
```

8.

다음 명령을 실행하여 비정상 멤버의 시스템을 삭제합니다.

```
$ oc delete machine -n openshift-machine-api examplecluster-control-plane-2
```

BareMetalHost 및 **Machine** 오브젝트를 제거한 후 머신 컨트롤러에서 **Node** 오브젝트를 자동으로 삭제합니다.

어떤 이유로든 머신 삭제가 지연되거나 명령이 차단되고 지연되면 **machine object finalizer** 필드를 제거하여 강제로 삭제할 수 있습니다.

중요

Ctrl+c 를 눌러 머신 삭제를 중단하지 마십시오. 명령이 완료될 수 있도록 허용해야 합니다. 새 터미널 창을 열어 편집하여 종료자 필드를 삭제합니다.

a.

다음 명령을 실행하여 머신 구성을 편집합니다.

```
$ oc edit machine -n openshift-machine-api examplecluster-control-plane-2
```

b.

Machine 사용자 정의 리소스에서 다음 필드를 삭제한 다음 업데이트된 파일을 저장합니다.

```
finalizers:
- machine.machine.openshift.io
```

출력 예

```
machine.machine.openshift.io/examplecluster-control-plane-2 edited
```

9. 다음 명령을 실행하여 시스템이 삭제되었는지 확인합니다.

```
$ oc get machines -n openshift-machine-api -o wide
```

출력 예

```
NAME                        PHASE     TYPE   REGION   ZONE   AGE     NODE
PROVIDERID                                                    STATE
examplecluster-control-plane-0    Running                      3h11m   openshift-control-
plane-0   baremetalhost:///openshift-machine-api/openshift-control-plane-0/da1ebe11-
3ff2-41c5-b099-0aa41222964e   externally provisioned
examplecluster-control-plane-1    Running                      3h11m   openshift-control-
plane-1   baremetalhost:///openshift-machine-api/openshift-control-plane-1/d9f9acbc-
329c-475e-8d81-03b20280a3e1   externally provisioned
examplecluster-compute-0          Running                      165m    openshift-compute-0
baremetalhost:///openshift-machine-api/openshift-compute-0/3d685b81-7410-4bb3-
80ec-13a31858241f         provisioned
examplecluster-compute-1          Running                      165m    openshift-compute-1
baremetalhost:///openshift-machine-api/openshift-compute-1/0fdae6eb-2066-4241-
91dc-e7ea72ab13b9         provisioned
```

10. 다음 명령을 실행하여 노드가 삭제되었는지 확인합니다.

```
$ oc get nodes

NAME                 STATUS ROLES   AGE   VERSION
openshift-control-plane-0 Ready master 3h24m v1.24.0+9546431
openshift-control-plane-1 Ready master 3h24m v1.24.0+9546431
openshift-compute-0       Ready worker 176m v1.24.0+9546431
openshift-compute-1       Ready worker 176m v1.24.0+9546431
```

11. 새 **BareMetalHost** 오브젝트와 시크릿을 생성하여 **BMC** 자격 증명을 저장합니다.

```
$ cat <<EOF | oc apply -f -
apiVersion: v1
kind: Secret
metadata:
  name: openshift-control-plane-2-bmc-secret
  namespace: openshift-machine-api
data:
  password: <password>
  username: <username>
```

```
type: Opaque
---
apiVersion: metal3.io/v1alpha1
kind: BareMetalHost
metadata:
 name: openshift-control-plane-2
 namespace: openshift-machine-api
spec:
 automatedCleaningMode: disabled
 bmc:
  address: redfish://10.46.61.18:443/redfish/v1/Systems/1
  credentialsName: openshift-control-plane-2-bmc-secret
  disableCertificateVerification: true
 bootMACAddress: 48:df:37:b0:8a:a0
 bootMode: UEFI
 externallyProvisioned: false
 online: true
 rootDeviceHints:
  deviceName: /dev/sda
 userData:
  name: master-user-data-managed
  namespace: openshift-machine-api
EOF
```

참고

*사용자 이름과 암호는 다른 베어 메탈 호스트의 시크릿에서 찾을 수 있습니다. **bmc:address** 에 사용할 프로토콜은 다른 **bmh** 개체에서 가져올 수 있습니다.*

중요

*기존 컨트롤 플레인 호스트에서 **BareMetalHost** 오브젝트 정의를 재사용하는 경우 **external Provisioned** 필드를 **true** 로 설정하지 마십시오.*

*기존 컨트롤 플레인 **BareMetalHost** 오브젝트는 **OpenShift Container Platform** 설치 프로그램에서 프로비저닝한 경우 외부**Provisioned** 플래그를 **true** 로 설정할 수 있습니다.*

*검사가 완료되면 **BareMetalHost** 오브젝트가 생성되고 프로비저닝할 수 있습니다.*

12. *사용 가능한 **BareMetalHost** 오브젝트를 사용하여 생성 프로세스를 확인합니다.*

```
$ oc get bmh -n openshift-machine-api

NAME                 STATE                CONSUMER                 ONLINE ERROR   AGE
```

```
openshift-control-plane-0 externally provisioned examplecluster-control-plane-0 true
4h48m
openshift-control-plane-1 externally provisioned examplecluster-control-plane-1 true
4h48m
openshift-control-plane-2 available           examplecluster-control-plane-3 true
47m
openshift-compute-0       provisioned         examplecluster-compute-0       true
4h48m
openshift-compute-1       provisioned         examplecluster-compute-1       true
4h48m
```

a. *new-master-machine.yaml 파일을 사용하여 새 컨트롤 플레인 시스템을 생성합니다.*

```
$ oc apply -f new-master-machine.yaml
```

b. *새 시스템이 생성되었는지 확인합니다.*

```
$ oc get machines -n openshift-machine-api -o wide
```

출력 예

```
NAME                              PHASE     TYPE   REGION   ZONE   AGE     NODE
PROVIDERID                                                        STATE
examplecluster-control-plane-0    Running                          3h11m   openshift-
control-plane-0   baremetalhost:///openshift-machine-api/openshift-control-plane-
0/da1ebe11-3ff2-41c5-b099-0aa41222964e   externally provisioned   (1)
examplecluster-control-plane-1    Running                          3h11m   openshift-
control-plane-1   baremetalhost:///openshift-machine-api/openshift-control-plane-
1/d9f9acbc-329c-475e-8d81-03b20280a3e1   externally provisioned
examplecluster-control-plane-2    Running                          3h11m   openshift-
control-plane-2   baremetalhost:///openshift-machine-api/openshift-control-plane-
2/3354bdac-61d8-410f-be5b-6a395b056135   externally provisioned
examplecluster-compute-0          Running                          165m    openshift-
compute-0         baremetalhost:///openshift-machine-api/openshift-compute-
0/3d685b81-7410-4bb3-80ec-13a31858241f         provisioned
examplecluster-compute-1          Running                          165m    openshift-
compute-1         baremetalhost:///openshift-machine-api/openshift-compute-
1/0fdae6eb-2066-4241-91dc-e7ea72ab13b9         provisioned
```

(1) *새 시스템 clustername-8qw5l-master-3 이 생성되고 단계가 Provisioning 에서 Running 으로 변경된 후 준비 상태가 됩니다.*

새 시스템을 생성하는 데 몇 분이 걸릴 수 있습니다. etcd 클러스터 Operator는 머신 또는 노드가 정상 상태로 돌아 오면 자동으로 동기화됩니다.

c.

베어 메탈 호스트가 프로비저닝되고 다음 명령을 실행하여 오류가 보고되지 않았는지 확인합니다.

```
$ oc get bmh -n openshift-machine-api
```

출력 예

```
$ oc get bmh -n openshift-machine-api
NAME                 STATE                CONSUMER                      ONLINE ERROR AGE
openshift-control-plane-0 externally provisioned examplecluster-control-plane-0
true        4h48m
openshift-control-plane-1 externally provisioned examplecluster-control-plane-1
true        4h48m
openshift-control-plane-2 provisioned          examplecluster-control-plane-3 true
47m
openshift-compute-0      provisioned          examplecluster-compute-0      true
4h48m
openshift-compute-1      provisioned          examplecluster-compute-1      true
4h48m
```

d.

다음 명령을 실행하여 새 노드가 추가되고 준비 상태에 있는지 확인합니다.

```
$ oc get nodes
```

출력 예

```
$ oc get nodes
NAME                 STATUS ROLES   AGE   VERSION
openshift-control-plane-0 Ready master 4h26m v1.24.0+9546431
openshift-control-plane-1 Ready master 4h26m v1.24.0+9546431
openshift-control-plane-2 Ready master 12m   v1.24.0+9546431
openshift-compute-0       Ready worker 3h58m v1.24.0+9546431
openshift-compute-1       Ready worker 3h58m v1.24.0+9546431
```

13. *다음 명령을 입력하여 쿼럼 보호기를 다시 켭니다.*

```
$ oc patch etcd/cluster --type=merge -p '{"spec": {"unsupportedConfigOverrides": null}}'
```

14. *다음 명령을 입력하여 **unsupportedConfigOverrides** 섹션이 오브젝트에서 제거되었는지 확인할 수 있습니다.*

```
$ oc get etcd/cluster -oyaml
```

15. *단일 노드 **OpenShift**를 사용하는 경우 노드를 다시 시작합니다. 그렇지 않으면 **etcd** 클러스터 **Operator**에서 다음 오류가 발생할 수 있습니다.*

출력 예

```
EtcdCertSignerControllerDegraded: [Operation cannot be fulfilled on secrets "etcd-peer-sno-0": the object has been modified; please apply your changes to the latest version and try again, Operation cannot be fulfilled on secrets "etcd-serving-sno-0": the object has been modified; please apply your changes to the latest version and try again, Operation cannot be fulfilled on secrets "etcd-serving-metrics-sno-0": the object has been modified; please apply your changes to the latest version and try again]
```

검증

1. *모든 **etcd pod**가 올바르게 실행되고 있는지 확인합니다.*

*클러스터에 액세스할 수 있는 터미널에서 **cluster-admin** 사용자로 다음 명령을 실행합니다.*

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd -o wide
```

출력 예

```
etcd-openshift-control-plane-0     5/5     Running     0     105m
etcd-openshift-control-plane-1     5/5     Running     0     107m
etcd-openshift-control-plane-2     5/5     Running     0     103m
```

이전 명령의 출력에 두 개의 **pod**만 나열되는 경우 수동으로 **etcd** 재배포를 강제 수행할 수 있습니다. 클러스터에 액세스할 수 있는 터미널에서 **cluster-admin** 사용자로 다음 명령을 실행합니다.

```
$ oc patch etcd cluster -p='{"spec": {"forceRedeploymentReason": "recovery-"'$( date --rfc-3339=ns )'"}}' --type=merge 1
```

1

forceRedeploymentReason 값은 고유해야하므로 타임 스탬프가 추가됩니다.

정확히 세 개의 **etcd** 멤버가 있는지 확인하려면 실행중인 **etcd** 컨테이너에 연결하고 영향을 받는 노드에 없는 **pod** 이름을 전달합니다. 클러스터에 액세스할 수 있는 터미널에서 **cluster-admin** 사용자로 다음 명령을 실행합니다.

```
$ oc rsh -n openshift-etcd etcd-openshift-control-plane-0
```

2. 멤버 목록을 확인합니다.

```
sh-4.2# etcdctl member list -w table
```

출력 예

```
+------------------+---------+------------------+--------------------------+--------------------------+---
--------------+
|        ID        | STATUS  |       NAME       |       PEER ADDRS         |       CLIENT ADDRS
|   IS LEARNER     |
+------------------+---------+------------------+--------------------------+--------------------------+---
--------------+
| 7a8197040a5126c8 | started | openshift-control-plane-2 | https://192.168.10.11:2380 |
https://192.168.10.11:2379 |   false |
| 8d5abe9669a39192 | started | openshift-control-plane-1 | https://192.168.10.10:2380 |
https://192.168.10.10:2379 |   false |
| cc3830a72fc357f9 | started | openshift-control-plane-0 | https://192.168.10.9:2380 |
```

```
https://192.168.10.9:2379 |     false |
+-----------------+---------+-----------------+-------------------------+-------------------------+---
--------------+
```

참고

이전 명령의 출력에 세 개 이상의 **etcd** 멤버가 나열된 경우 원하지 않는 멤버를 신중하게 제거해야 합니다.

3. 다음 명령을 실행하여 모든 **etcd** 멤버가 정상인지 확인합니다.

```
# etcdctl endpoint health --cluster
```

출력 예

```
https://192.168.10.10:2379 is healthy: successfully committed proposal: took =
8.973065ms
https://192.168.10.9:2379 is healthy: successfully committed proposal: took =
11.559829ms
https://192.168.10.11:2379 is healthy: successfully committed proposal: took =
11.665203ms
```

4. 다음 명령을 실행하여 모든 노드가 최신 버전인지 확인합니다.

```
$ oc get etcd -o=jsonpath='{range.items[0].status.conditions[?
(@.type=="NodeInstallerProgressing")]}{.reason}{"\n"}{.message}{"\n"}'
```

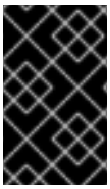
```
AllNodesAtLatestRevision
```

5.3. 재해 복구

5.3.1. 재해 복구 정보

재해 복구 문서에서는 관리자에게 **OpenShift Container Platform** 클러스터에서 발생할 수있는 여러

재해 상황을 복구하는 방법에 대한 정보를 제공합니다. 관리자는 클러스터를 작동 상태로 복원하려면 다음 절차 중 하나 이상을 수행해야합니다.



중요

재해 복구를 위해서는 하나 이상의 정상 컨트롤 플레인 호스트가 있어야 합니다.

이전 클러스터 상태로 복원

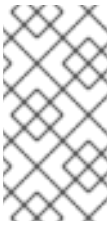
클러스터를 이전 상태로 복원하려는 경우 (예: 관리자가 일부 주요 정보를 삭제한 경우) 이 솔루션을 사용할 수 있습니다. 이에는 대부분의 컨트롤 플레인 호스트가 손실되고 **etcd** 쿼럼이 손실되고 클러스터가 오프라인인 상태에서도 사용할 수 있습니다. **etcd** 백업을 수행한 경우 이 절차에 따라 클러스터를 이전 상태로 복원할 수 있습니다.

해당하는 경우 만료된 컨트롤 플레인 인증서를 복구해야 할 수도 있습니다.

주의

이전 클러스터 상태로 복원하는 것은 실행 중인 클러스터에서 수행하기에 위험하고 불안정한 작업입니다. 이 절차는 마지막 수단으로만 사용해야 합니다.

복원을 수행하기 전에 클러스터에 미치는 영향에 대한 자세한 내용은 클러스터 상태 복원 을 참조하십시오.



참고

대다수의 마스터를 계속 사용할 수 있고 **etcd** 쿼럼이 있는 경우 절차에 따라 비정상적인 단일 **etcd** 멤버 교체 를 수행합니다.

만료된 컨트롤 플레인 인증서 복구

컨트롤 플레인 인증서가 만료된 경우 이 솔루션을 사용할 수 있습니다. 예를 들어, 설치 후 **24** 시간 내에 발생하는 첫 번째 인증서 교체 전에 클러스터를 종료하면 인증서가 교체되지 않고 만료됩니다. 다음 단계에 따라 만료된 컨트롤 플레인 인증서를 복구할 수 있습니다.

5.3.2. 이전 클러스터 상태로 복원

클러스터를 이전 상태로 복원하려면 스냅샷을 작성하여 etcd 데이터를 백업 해야 합니다. 이 스냅샷을 사용하여 클러스터 상태를 복구합니다.

5.3.2.1. 클러스터 상태 복원 정보

etcd 백업을 사용하여 클러스터를 이전 상태로 복원할 수 있습니다. 이를 사용하여 다음과 같은 상황에서 복구할 수 있습니다.

- 클러스터에서 대부분의 컨트롤 플레인 호스트가 손실되었습니다(쿼럼 손실).
- 관리자가 중요한 것을 삭제했으며 클러스터를 복구하려면 복원해야 합니다.

주의

이전 클러스터 상태로 복원하는 것은 실행 중인 클러스터에서 수행하기에 위험하고 불안정한 작업입니다. 이는 마지막 수단으로만 사용해야 합니다.

Kubernetes API 서버를 사용하여 데이터를 검색할 수 있는 경우 etcd를 사용할 수 있으며 etcd 백업을 사용하여 복원할 수 없습니다.

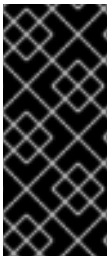
etcd를 복원하려면 클러스터를 효율적으로 복원하는 데 시간이 걸리며 모든 클라이언트가 충돌하는 병렬 기록이 발생합니다. 이는 kubelets, Kubernetes 컨트롤러 관리자, SDN 컨트롤러 및 영구 볼륨 컨트롤러와 같은 구성 요소 모니터링 동작에 영향을 줄 수 있습니다.

이로 인해 etcd의 콘텐츠가 디스크의 실제 콘텐츠와 일치하지 않을 때 Operator가 문제가 발생하여 디스크의 파일이 etcd의 콘텐츠와 충돌할 때 Kubernetes API 서버, Kubernetes 컨트롤러 관리자, Kubernetes 스케줄러 및 etcd의 Operator가 중단될 수 있습니다. 여기에는 문제를 해결하기 위해 수동 작업이 필요할 수 있습니다.

극단적인 경우 클러스터에서 영구 볼륨 추적을 손실하고, 더 이상 존재하지 않는 중요한 워크로드를 삭제하고, 시스템을 다시 이미지화하고, 만료된 인증서로 CA 번들을 다시 작성할 수 있습니다.

5.3.2.2. 이전 클러스터 상태로 복원

저장된 **etcd** 백업을 사용하여 이전 클러스터 상태를 복원하거나 대부분의 컨트롤 플레인 호스트가 손실된 클러스터를 복원할 수 있습니다.



중요

클러스터를 복원할 때 동일한 **z-stream** 릴리스에서 가져온 **etcd** 백업을 사용해야 합니다. 예를 들어 **OpenShift Container Platform 4.7.2** 클러스터는 **4.7.2**에서 가져온 **etcd** 백업을 사용해야 합니다.

사전 요구 사항

- **cluster-admin** 역할의 사용자로 클러스터에 액세스할 수 있어야 합니다.
- 복구 호스트로 사용할 정상적인 컨트롤 플레인 호스트가 있어야 합니다.
- 컨트롤 플레인 호스트에 대한 **SSH** 액세스.
- 동일한 백업에서 가져온 **etcd** 스냅샷과 정적 **pod** 리소스가 모두 포함된 백업 디렉토리입니다. 디렉토리의 파일 이름은 **snapshot_<datetimestamp>.db** 및 **static_kuberesources_<datetimestamp>.tar.gz** 형식이어야합니다.

중요

복구되지 않은 컨트롤 플레인 노드의 경우 **SSH** 연결을 설정하거나 고정 **Pod**를 중지할 필요가 없습니다. 다른 비 복구, 컨트롤 플레인 시스템을 삭제하고 하나씩 다시 생성할 수 있습니다.

절차

1. 복구 호스트로 사용할 컨트롤 플레인 호스트를 선택합니다. 이는 복구 작업을 실행할 호스트입니다.
2. 복구 호스트를 포함하여 각 컨트롤 플레인 노드에 **SSH** 연결을 설정합니다.

복구 프로세스가 시작된 후에는 **Kubernetes API** 서버에 액세스할 수 없으므로 컨트롤 플레

*인 노드에 액세스할 수 없습니다. 따라서 다른 터미널에서 각 컨트롤 플레인 호스트에 대한 **SSH** 연결을 설정하는 것이 좋습니다.*

중요

이 단계를 완료하지 않으면 컨트롤 플레인 호스트에 액세스하여 복구 프로세스를 완료할 수 없으며 이 상태에서 클러스터를 복구할 수 없습니다.

3. ***etcd** 백업 디렉토리를 복구 컨트롤 플레인 호스트에 복사합니다.*

*이 단계에서는 **etcd** 스냅샷 및 정적 **pod**의 리소스가 포함된 **backup** 디렉터리를 복구 컨트롤 플레인 호스트의 **/home/core/** 디렉터리에 복사하는 것을 전제로하고 있습니다.*

4. *다른 컨트롤 플레인 노드에서 고정 **Pod**를 중지합니다.*

참고

*복구 호스트에서 **pod**를 수동으로 중지할 필요는 없습니다. 복구 스크립트는 복구 호스트에서 **pod**를 중지합니다.*

a. *복구 호스트가 아닌 컨트롤 플레인 호스트에 액세스합니다.*

b. ***kubelet** 매니페스트 디렉토리에서 기존 **etcd pod** 파일을 이동합니다.*

```
$ sudo mv /etc/kubernetes/manifests/etcd-pod.yaml /tmp
```

c. ***etcd pod**가 중지되었는지 확인합니다.*

```
$ sudo crictl ps | grep etcd | grep -v operator
```

이 명령의 출력은 비어 있어야합니다. 비어 있지 않은 경우 몇 분 기다렸다가 다시 확인하십시오.

d. ***kubelet** 매니페스트 디렉토리에서 기존 **Kubernetes API** 서버 **pod** 파일을 이동합니다.*

```
$ sudo mv /etc/kubernetes/manifests/kube-apiserver-pod.yaml /tmp
```

e. *Kubernetes API 서버 pod가 중지되었는지 확인합니다.*

```
$ sudo crictl ps | grep kube-apiserver | grep -v operator
```

이 명령의 출력은 비어 있어야합니다. 비어 있지 않은 경우 몇 분 기다렸다가 다시 확인하십시오.

f. *etcd 데이터 디렉토리를 다른 위치로 이동합니다.*

```
$ sudo mv /var/lib/etcd/ /tmp
```

g. *복구 호스트가 아닌 다른 컨트롤 플레인 호스트에서 이 단계를 반복합니다.*

5. *복구 컨트롤 플레인 호스트에 액세스합니다.*

6. *클러스터 전체의 프록시가 활성화되어 있는 경우 NO_PROXY, HTTP_PROXY 및 https_proxy 환경 변수를 내보내고 있는지 확인합니다.*

작은 정보

oc get proxy cluster -o yaml의 출력을 확인하여 프록시가 사용 가능한지 여부를 확인할 수 있습니다. httpProxy, httpsProxy 및 noProxy 필드에 값이 설정되어 있으면 프록시가 사용됩니다.

7. *복구 컨트롤 플레인 호스트에서 복원 스크립트를 실행하고 etcd 백업 디렉터리에 경로를 전달합니다.*

```
$ sudo -E /usr/local/bin/cluster-restore.sh /home/core/backup
```

스크립트 출력 예

```
...stopping kube-scheduler-pod.yaml
...stopping kube-controller-manager-pod.yaml
...stopping etcd-pod.yaml
...stopping kube-apiserver-pod.yaml
Waiting for container etcd to stop
.complete
Waiting for container etcdctl to stop
.....complete
Waiting for container etcd-metrics to stop
complete
Waiting for container kube-controller-manager to stop
complete
Waiting for container kube-apiserver to stop
.....complete
Waiting for container kube-scheduler to stop
complete
Moving etcd data-dir /var/lib/etcd/member to /var/lib/etcd-backup
starting restore-etcd static pod
starting kube-apiserver-pod.yaml
static-pod-resources/kube-apiserver-pod-7/kube-apiserver-pod.yaml
starting kube-controller-manager-pod.yaml
static-pod-resources/kube-controller-manager-pod-7/kube-controller-manager-
pod.yaml
starting kube-scheduler-pod.yaml
static-pod-resources/kube-scheduler-pod-8/kube-scheduler-pod.yaml
```

참고

마지막 **etcd** 백업 후 노드 인증서가 업데이트된 경우 복원 프로세스를 통해 노드가 **NotReady** 상태를 입력할 수 있습니다.

8. 노드가 **Ready** 상태에 있는지 확인합니다.

   a. 다음 명령을 실행합니다.

```
$ oc get nodes -w
```

샘플 출력

```
NAME              STATUS  ROLES        AGE     VERSION
host-172-25-75-28   Ready   master       3d20h   v1.23.3+e419edf
host-172-25-75-38   Ready   infra,worker   3d20h   v1.23.3+e419edf
```

```
host-172-25-75-40   Ready   master        3d20h   v1.23.3+e419edf
host-172-25-75-65   Ready   master        3d20h   v1.23.3+e419edf
host-172-25-75-74   Ready   infra,worker   3d20h   v1.23.3+e419edf
host-172-25-75-79   Ready   worker        3d20h   v1.23.3+e419edf
host-172-25-75-86   Ready   worker        3d20h   v1.23.3+e419edf
host-172-25-75-98   Ready   infra,worker   3d20h   v1.23.3+e419edf
```

모든 노드가 상태를 보고하는 데 몇 분이 걸릴 수 있습니다.

b. ***NotReady** 상태인 노드가 있는 경우 노드에 로그인하고 각 노드의 **/var/lib/kubelet/pki** 디렉터리에서 모든 **PEM** 파일을 삭제합니다. 노드에 **SSH**를 사용하거나 웹 콘솔에서 터미널 창을 사용할 수 있습니다.*

```
$  ssh -i <ssh-key-path> core@<master-hostname>
```

***pki** 디렉터리 샘플*

```
sh-4.4# pwd
/var/lib/kubelet/pki
sh-4.4# ls
kubelet-client-2022-04-28-11-24-09.pem  kubelet-server-2022-04-28-11-24-15.pem
kubelet-client-current.pem              kubelet-server-current.pem
```

9. *모든 컨트롤 플레인 호스트에서 **kubelet** 서비스를 다시 시작합니다.*

a. *복구 호스트에서 다음 명령을 실행합니다.*

```
$ sudo systemctl restart kubelet.service
```

b. *다른 모든 컨트롤 플레인 호스트에서 이 단계를 반복합니다.*

10. *보류 중인 **CSR**을 승인합니다.*

a. *현재 CSR의 목록을 가져옵니다.*

```
$ oc get csr
```

출력 예

```
NAME        AGE    SIGNERNAME                                    REQUESTOR
CONDITION
csr-2s94x   8m3s   kubernetes.io/kubelet-serving                 system:node:<node_name>
Pending (1)
csr-4bd6t   8m3s   kubernetes.io/kubelet-serving                 system:node:<node_name>
Pending (2)
csr-4hl85   13m    kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
(3)
csr-zhhhp   3m8s   kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
(4)
...
```

(1) (2) *보류 중인 kubelet 서비스 CSR(사용자 프로비저닝 설치용)입니다.*

(3) (4) *보류 중인 node-bootstrapper CSR입니다.*

b. *CSR의 세부 사항을 검토하여 CSR이 유효한지 확인합니다.*

```
$ oc describe csr <csr_name> (1)
```

(1) *<csr_name>은 현재 CSR 목록에 있는 CSR의 이름입니다.*

c. *각각의 유효한 node-bootstrapper CSR을 승인합니다.*

```
$ oc adm certificate approve <csr_name>
```

d. 사용자 프로비저닝 설치의 경우 각 유효한 **kubelet** 서비스 **CSR**을 승인합니다.

```
$ oc adm certificate approve <csr_name>
```

11. 단일 멤버 컨트롤 플레인이 제대로 시작되었는지 확인합니다.

a. 복구 호스트에서 **etcd** 컨테이너가 실행 중인지 확인합니다.

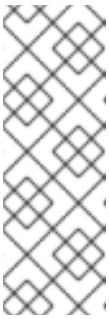
```
$ sudo crictl ps | grep etcd | grep -v operator
```

출력 예

```
3ad41b7908e32
36f86e2eeaaffe662df0d21041eb22b8198e0e58abeeae8c743c3e6e977e8009
About a minute ago   Running             etcd                                0
7c05f8af362f0
```

b. 복구 호스트에서 **etcd pod**가 실행 중인지 확인합니다.

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```



참고

이 명령을 실행하기 전에 **oc login**을 실행하여 다음 오류가 발생하면 인증 컨트롤러가 시작될 때까지 잠시 기다렸다가 다시 시도하십시오.

```
Unable to connect to the server: EOF
```

출력 예

```
NAME                                 READY   STATUS    RESTARTS   AGE
etcd-ip-10-0-143-125.ec2.internal    1/1     Running   1          2m47s
```

Pending 상태에 있거나 출력에 여러 실행중인 etcd pod가 나열되어 있는 경우 몇 분 기다렸다가 다시 확인합니다.

참고

OVNKubernetes CNI(Container Network Interface) 플러그인을 사용하는 경우에만 다음 단계를 수행합니다.

12. *모든 호스트에서 OVN(Open Virtual Network) Kubernetes Pod를 재시작합니다.*

a. *northbound 데이터베이스(nbdb) 및 southbound 데이터베이스(sbdb)를 제거합니다. SSH(Secure Shell)를 사용하여 복구 호스트 및 나머지 컨트롤 플레인 노드에 액세스하고 다음 명령을 실행합니다.*

```
$ sudo rm -f /var/lib/ovn/etc/*.db
```

b. *다음 명령을 실행하여 모든 OVN-Kubernetes 컨트롤 플레인 Pod를 삭제합니다.*

```
$ oc delete pods -l app=ovnkube-master -n openshift-ovn-kubernetes
```

c. *모든 OVN-Kubernetes 컨트롤 플레인 Pod가 다시 배포되어 다음 명령을 실행하여 Running 상태인지 확인합니다.*

```
$ oc get pods -l app=ovnkube-master -n openshift-ovn-kubernetes
```

출력 예

```
NAME                   READY   STATUS    RESTARTS   AGE
ovnkube-master-nb24h   4/4     Running   0          48s
ovnkube-master-rm8kw   4/4     Running   0          47s
ovnkube-master-zbqnh   4/4     Running   0          56s
```

d. 다음 명령을 실행하여 **ovnkube-node Pod**를 모두 삭제합니다.

```
$ oc get pods -n openshift-ovn-kubernetes -o name | grep ovnkube-node | while read p ; do oc delete $p -n openshift-ovn-kubernetes ; done
```

e. 모든 **ovnkube-node Pod**가 다시 배포되어 다음 명령을 실행하여 **Running** 상태인지 확인합니다.

```
$ oc get  pods -n openshift-ovn-kubernetes | grep ovnkube-node
```

13. 복구되지 않는 다른 컨트롤 플레인 시스템을 삭제하고 하나씩 다시 생성합니다. 머신을 다시 생성한 후 새 버전이 강제되고 **etcd**가 자동으로 확장됩니다.

- 사용자가 프로비저닝한 베어 메탈 설치를 사용하는 경우 원래 사용했던 것과 동일한 방법을 사용하여 컨트롤 플레인 시스템을 다시 생성할 수 있습니다. 자세한 내용은 " 베어 메탈에 사용자 프로비저닝 클러스터 설치"를 참조하십시오.

> **주의**
>
> 복구 호스트의 시스템을 삭제하고 다시 생성하지 마십시오.

- 설치 관리자 프로비저닝 인프라를 실행 중이거나 **Machine API**를 사용하여 머신을 생성한 경우 다음 단계를 따르십시오.

주의

복구 호스트의 시스템을 삭제하고 다시 생성하지 마십시오.

설치 관리자 프로비저닝 인프라에 베어 메탈 설치의 경우 컨트롤 플레인 머신이 다시 생성되지 않습니다. 자세한 내용은 " 베어 메탈 컨트롤 플레인 노드 교체"를 참조하십시오.

a.

손실된 컨트롤 플레인 호스트 중 하나에 대한 시스템을 가져옵니다.

***cluster-admin** 사용자로 클러스터에 액세스할 수 있는 터미널에서 다음 명령을 실행합니다.*

```
$ oc get machines -n openshift-machine-api -o wide
```

출력 예:

```
NAME                                PHASE     TYPE        REGION      ZONE         AGE
NODE                                PROVIDERID                              STATE
clustername-8qw5l-master-0          Running   m4.xlarge   us-east-1   us-east-1a   3h37m   ip-10-0-131-183.ec2.internal   aws:///us-east-1a/i-0ec2782f8287dfb7e   stopped   1
clustername-8qw5l-master-1          Running   m4.xlarge   us-east-1   us-east-1b   3h37m   ip-10-0-143-125.ec2.internal   aws:///us-east-1b/i-096c349b700a19631   running
clustername-8qw5l-master-2          Running   m4.xlarge   us-east-1   us-east-1c   3h37m   ip-10-0-154-194.ec2.internal   aws:///us-east-1c/i-02626f1dba9ed5bba   running
clustername-8qw5l-worker-us-east-1a-wbtgd   Running   m4.large   us-east-1   us-east-1a   3h28m   ip-10-0-129-226.ec2.internal   aws:///us-east-1a/i-010ef6279b4662ced   running
clustername-8qw5l-worker-us-east-1b-lrdxb   Running   m4.large   us-east-1   us-east-1b   3h28m   ip-10-0-144-248.ec2.internal   aws:///us-east-1b/i-0cb45ac45a166173b   running
clustername-8qw5l-worker-us-east-1c-pkg26   Running   m4.large   us-east-1   us-east-1c   3h28m   ip-10-0-170-181.ec2.internal   aws:///us-east-1c/i-06861c00007751b0a   running
```

1

b.

시스템 설정을 파일 시스템의 파일에 저장합니다.

```
$ oc get machine clustername-8qw5l-master-0 \ 1
  -n openshift-machine-api \
  -o yaml \
  > new-master-machine.yaml
```

1

손실된 컨트롤 플레인 호스트의 컨트롤 플레인 시스템의 이름을 지정합니다.

c.

*이전 단계에서 만든 **new-master-machine.yaml** 파일을 편집하여 새 이름을 할당하고 불필요한 필드를 제거합니다.*

i.

*전체 **status** 섹션을 삭제합니다.*

```
status:
  addresses:
  - address: 10.0.131.183
    type: InternalIP
  - address: ip-10-0-131-183.ec2.internal
    type: InternalDNS
  - address: ip-10-0-131-183.ec2.internal
    type: Hostname
  lastUpdated: "2020-04-20T17:44:29Z"
  nodeRef:
    kind: Node
    name: ip-10-0-131-183.ec2.internal
    uid: acca4411-af0d-4387-b73e-52b2484295ad
  phase: Running
  providerStatus:
    apiVersion: awsproviderconfig.openshift.io/v1beta1
    conditions:
    - lastProbeTime: "2020-04-20T16:53:50Z"
      lastTransitionTime: "2020-04-20T16:53:50Z"
      message: machine successfully created
      reason: MachineCreationSucceeded
      status: "True"
      type: MachineCreation
    instanceId: i-0fdb85790d76d0c3f
    instanceState: stopped
    kind: AWSMachineProviderStatus
```

ii. *metadata.name 필드를 새 이름으로 변경합니다.*

이전 시스템과 동일한 기본 이름을 유지하고 마지막 번호를 사용 가능한 다음 번호로 변경하는 것이 좋습니다. 이 예에서 clustername-8qw5l-master-0 은 clustername-8qw5l-master-3 으로 변경되었습니다.

```
apiVersion: machine.openshift.io/v1beta1
kind: Machine
metadata:
 ...
 name: clustername-8qw5l-master-3
 ...
```

iii. *spec.providerID 필드를 삭제합니다.*

```
providerID: aws:///us-east-1a/i-0fdb85790d76d0c3f
```

iv. *metadata.annotations 및 metadata.generation 필드를 제거합니다.*

```
annotations:
  machine.openshift.io/instance-state: running
...
generation: 2
```

v. *metadata.resourceVersion 및 metadata.uid 필드를 제거합니다.*

```
resourceVersion: "13291"
uid: a282eb70-40a2-4e89-8009-d05dd420d31a
```

d. *손실된 컨트롤 플레인 호스트의 시스템을 삭제합니다.*

```
$ oc delete machine -n openshift-machine-api clustername-8qw5l-master-0 1
```

1 *손실된 컨트롤 플레인 호스트의 컨트롤 플레인 시스템의 이름을 지정합니다.*

e. *시스템이 삭제되었는지 확인합니다.*

```
$ oc get machines -n openshift-machine-api -o wide
```

출력 예:

```
NAME                                 PHASE     TYPE        REGION      ZONE         AGE
NODE                           PROVIDERID                          STATE
clustername-8qw5l-master-1           Running   m4.xlarge   us-east-1   us-east-1b   3h37m   ip-10-0-143-125.ec2.internal   aws:///us-east-1b/i-096c349b700a19631   running
clustername-8qw5l-master-2           Running   m4.xlarge   us-east-1   us-east-1c   3h37m   ip-10-0-154-194.ec2.internal   aws:///us-east-1c/i-02626f1dba9ed5bba   running
clustername-8qw5l-worker-us-east-1a-wbtgd   Running   m4.large    us-east-1   us-east-1a   3h28m   ip-10-0-129-226.ec2.internal   aws:///us-east-1a/i-010ef6279b4662ced   running
clustername-8qw5l-worker-us-east-1b-lrdxb   Running   m4.large    us-east-1   us-east-1b   3h28m   ip-10-0-144-248.ec2.internal   aws:///us-east-1b/i-0cb45ac45a166173b   running
clustername-8qw5l-worker-us-east-1c-pkg26   Running   m4.large    us-east-1   us-east-1c   3h28m   ip-10-0-170-181.ec2.internal   aws:///us-east-1c/i-06861c00007751b0a   running
```

f. *new-master-machine.yaml 파일을 사용하여 시스템을 생성합니다.*

```
$ oc apply -f new-master-machine.yaml
```

g. *새 시스템이 생성되었는지 확인합니다.*

```
$ oc get machines -n openshift-machine-api -o wide
```

출력 예:

```
NAME                                 PHASE     TYPE        REGION      ZONE         AGE    NODE                           PROVIDERID                          STATE
clustername-8qw5l-master-1           Running   m4.xlarge   us-east-1   us-east-1b   3h37m   ip-10-0-143-125.ec2.internal   aws:///us-east-1b/i-096c349b700a19631   running
clustername-8qw5l-master-2           Running   m4.xlarge   us-east-1   us-east-1c   3h37m   ip-10-0-154-194.ec2.internal   aws:///us-east-1c/i-02626f1dba9ed5bba   running
clustername-8qw5l-master-3           Provisioning   m4.xlarge   us-east-1   us-east-1a   85s   ip-10-0-173-171.ec2.internal   aws:///us-east-1a/i-015b0888fe17bc2c8   running (1)
clustername-8qw5l-worker-us-east-1a-wbtgd   Running   m4.large    us-east-1   us-east-1a   3h28m   ip-10-0-129-226.ec2.internal   aws:///us-east-1a/i-010ef6279b4662ced   running
```

```
clustername-8qw5l-worker-us-east-1b-lrdxb   Running     m4.large   us-east-1
us-east-1b   3h28m   ip-10-0-144-248.ec2.internal   aws:///us-east-1b/i-
0cb45ac45a166173b   running
clustername-8qw5l-worker-us-east-1c-pkg26   Running     m4.large   us-east-
1   us-east-1c   3h28m   ip-10-0-170-181.ec2.internal   aws:///us-east-1c/i-
06861c00007751b0a   running
```

(1)

새 시스템 clustername-8qw5l-master-3 이 생성되고 단계가 Provisioning 에서 Running 으로 변경된 후 준비 상태가 됩니다.

새 시스템을 만드는 데 몇 분이 소요될 수 있습니다. etcd 클러스터 Operator는 머신 또는 노드가 정상 상태로 돌아 오면 자동으로 동기화됩니다.

h. 복구 호스트가 아닌 각 손실된 컨트롤 플레인 호스트에 대해 이 단계를 반복합니다.

14. 별도의 터미널 창에서 다음 명령을 입력하여 cluster-admin 역할의 사용자로 클러스터에 로그인합니다.

```
$ oc login -u <cluster_admin> (1)
```

(1) <cluster_admin>은 cluster-admin 역할을 사용하여 사용자 이름을 지정합니다.

15. etcd를 강제로 재배포합니다.

클러스터에 액세스할 수 있는 터미널에서 cluster-admin 사용자로 다음 명령을 실행합니다.

```
$ oc patch etcd cluster -p='{"spec": {"forceRedeploymentReason": "recovery-"$( date
--rfc-3339=ns )""}}' --type=merge (1)
```

(1) forceRedeploymentReason 값은 고유해야하므로 타임 스탬프가 추가됩니다.

etcd 클러스터 Operator가 재배포를 실행하면 기존 노드가 초기 부트 스트랩 확장과 유사한 새 pod를 사용하기 시작합니다.

16. 모든 노드가 최신 버전으로 업데이트되었는지 확인합니다.

클러스터에 액세스할 수 있는 터미널에서 **cluster-admin** 사용자로 다음 명령을 실행합니다.

```
$ oc get etcd -o=jsonpath='{range .items[0].status.conditions[?
(@.type=="NodeInstallerProgressing")]}{.reason}{"\n"}{.message}{"\n"}'
```

etcd의 **NodeInstallerProgressing** 상태 조건을 확인하고 모든 노드가 최신 버전인지 확인합니다. 업데이트가 성공적으로 실행되면 출력에 **AllNodesAtLatestRevision**이 표시됩니다.

```
AllNodesAtLatestRevision
3 nodes are at revision 7 (1)
```

(1) 이 예에서 최신 버전 번호는 **7**입니다.

출력에 **2 nodes are at revision 6; 1 nodes are at revision 7**와 같은 여러 버전 번호가 표시되면 이는 업데이트가 아직 진행 중임을 의미합니다. 몇 분 기다린 후 다시 시도합니다.

17. **etcd**를 재배포한 후 컨트롤 플레인에 새 롤아웃을 강제 실행합니다. **kubelet**이 내부 로드 밸런서를 사용하여 **API** 서버에 연결되어 있으므로 **Kubernetes API** 서버는 다른 노드에 다시 설치됩니다.

cluster-admin 사용자로 클러스터에 액세스할 수있는 터미널에서 다음 명령을 실행합니다.

a. **Kubernetes API** 서버에 대해 새 롤아웃을 강제 적용합니다.

```
$ oc patch kubeapiserver cluster -p='{"spec": {"forceRedeploymentReason":
"recovery-"$( date --rfc-3339=ns )""}}' --type=merge
```

모든 노드가 최신 버전으로 업데이트되었는지 확인합니다.

```
$ oc get kubeapiserver -o=jsonpath='{range .items[0].status.conditions[?
(@.type=="NodeInstallerProgressing")]}{.reason}{"\n"}{.message}{"\n"}'
```

NodeInstallerProgressing 상태 조건을 확인하고 모든 노드가 최신 버전인지 확인합니다. 업데이트가 성공적으로 실행되면 출력에 AllNodesAtLatestRevision이 표시됩니다.

```
AllNodesAtLatestRevision
3 nodes are at revision 7 (1)
```

(1) *이 예에서 최신 버전 번호는 7입니다.*

출력에 2 nodes are at revision 6; 1 nodes are at revision 7와 같은 여러 버전 번호가 표시되면 이는 업데이트가 아직 진행 중임을 의미합니다. 몇 분 기다린 후 다시 시도합니다.

b. *Kubernetes 컨트롤러 관리자의 새 롤아웃을 강제 적용합니다.*

```
$ oc patch kubecontrollermanager cluster -p='{"spec":
{"forceRedeploymentReason": "recovery-"'$( date --rfc-3339=ns )'"}}' --
type=merge
```

모든 노드가 최신 버전으로 업데이트되었는지 확인합니다.

```
$ oc get kubecontrollermanager -o=jsonpath='{range .items[0].status.conditions[?
(@.type=="NodeInstallerProgressing")]}{.reason}{"\n"}{.message}{"\n"}'
```

NodeInstallerProgressing 상태 조건을 확인하고 모든 노드가 최신 버전인지 확인합니다. 업데이트가 성공적으로 실행되면 출력에 AllNodesAtLatestRevision이 표시됩니다.

```
AllNodesAtLatestRevision
3 nodes are at revision 7 (1)
```

(1) *이 예에서 최신 버전 번호는 7입니다.*

출력에 2 nodes are at revision 6; 1 nodes are at revision 7와 같은 여러 버전 번호가 표시되면 이는 업데이트가 아직 진행 중임을 의미합니다. 몇 분 기다린 후 다시 시도합니다.

c.

Kubernetes 스케줄러에 새 롤아웃을 강제 적용합니다.

```
$ oc patch kubescheduler cluster -p='{"spec": {"forceRedeploymentReason":
"recovery-'"$( date --rfc-3339=ns )"'"}}' --type=merge
```

모든 노드가 최신 버전으로 업데이트되었는지 확인합니다.

```
$ oc get kubescheduler -o=jsonpath='{range .items[0].status.conditions[?
(@.type=="NodeInstallerProgressing")]}{.reason}{"\n"}{.message}{"\n"}'
```

NodeInstallerProgressing 상태 조건을 확인하고 모든 노드가 최신 버전인지 확인합니다. 업데이트가 성공적으로 실행되면 출력에 AllNodesAtLatestRevision이 표시됩니다.

```
AllNodesAtLatestRevision
3 nodes are at revision 7 (1)
```

(1)

이 예에서 최신 버전 번호는 7입니다.

출력에 2 nodes are at revision 6; 1 nodes are at revision 7와 같은 여러 버전 번호가 표시되면 이는 업데이트가 아직 진행 중임을 의미합니다. 몇 분 기다린 후 다시 시도합니다.

18.

모든 컨트롤 플레인 호스트가 클러스터를 시작하여 참여하고 있는지 확인합니다.

클러스터에 액세스할 수 있는 터미널에서 cluster-admin 사용자로 다음 명령을 실행합니다.

```
$ oc -n openshift-etcd get pods -l k8s-app=etcd
```

출력 예

```
etcd-ip-10-0-143-125.ec2.internal        2/2     Running     0          9h
etcd-ip-10-0-154-194.ec2.internal        2/2     Running     0          9h
etcd-ip-10-0-173-171.ec2.internal        2/2     Running     0          9h
```

복구 프로시저에 따라 모든 워크로드가 정상 작업으로 돌아가도록 하려면 **Kubernetes API** 정보를 저장하는 각 **Pod**를 다시 시작합니다. 여기에는 라우터, **Operator** 및 타사 구성 요소와 같은 **OpenShift Container Platform** 구성 요소가 포함됩니다.

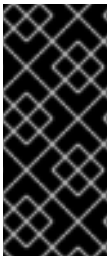
이 프로세스를 완료한 후 모든 서비스를 복구하는데 몇 분 정도 걸릴 수 있습니다. 예를 들어, **OAuth** 서버 **pod**가 다시 시작될 때까지 **oc login**을 사용한 인증이 즉시 작동하지 않을 수 있습니다.

5.3.2.3. 추가 리소스

- 베어 메탈에 사용자 프로비저닝 클러스터 설치
- **SSH**를 사용하여 **OpenShift Container Platform** 인스턴스 및 컨트롤 플레인 노드에 액세스하기 위한 **bastion** 호스트 생성
- 베어 메탈 컨트롤 플레인 노드 교체

5.3.2.4. 영구 스토리지 상태 복원을 위한 문제 및 해결 방법

OpenShift Container Platform 클러스터에서 모든 형식의 영구저장장치를 사용하는 경우 일반적으로 클러스터의 상태가 **etcd** 외부에 저장됩니다. **StatefulSet** 오브젝트에서 실행 중인 **Pod** 또는 데이터베이스에서 실행 중인 **Elasticsearch** 클러스터일 수 있습니다. **etcd** 백업에서 복원하면 **OpenShift Container Platform**의 워크로드 상태도 복원됩니다. 그러나 **etcd** 스냅샷이 오래된 경우 상태가 유효하지 않거나 오래되었을 수 있습니다.



중요

PV(영구 볼륨)의 내용은 **etcd** 스냅샷의 일부가 아닙니다. **etcd** 스냅샷에서 **OpenShift Container Platform** 클러스터를 복원할 때 중요하지 않은 워크로드가 중요한 데이터에 액세스할 수 있으며 그 반대의 경우로도 할 수 있습니다.

다음은 사용되지 않는 상태를 생성하는 몇 가지 예제 시나리오입니다.

- **MySQL** 데이터베이스는 **PV** 오브젝트에서 지원하는 **pod**에서 실행됩니다. **etcd** 스냅샷에서 **OpenShift Container Platform**을 복원해도 스토리지 공급자의 볼륨을 다시 가져오지 않으며

pod를 반복적으로 시작하려고 하지만 실행 중인 MySQL pod는 생성되지 않습니다. 스토리지 공급자에서 볼륨을 복원한 다음 새 볼륨을 가리키도록 PV를 편집하여 이 Pod를 수동으로 복원해야 합니다.

- *Pod P1에서는 노드 X에 연결된 볼륨 A를 사용합니다. 다른 pod가 노드 Y에서 동일한 볼륨을 사용하는 동안 etcd 스냅샷을 가져오는 경우 etcd 복원이 수행되면 해당 볼륨이 여전히 Y 노드에 연결되어 있으므로 Pod P1이 제대로 시작되지 않을 수 있습니다. OpenShift Container Platform은 연결을 인식하지 못하고 자동으로 연결을 분리하지 않습니다. 이 경우 볼륨이 노드 X에 연결된 다음 Pod P1이 시작될 수 있도록 노드 Y에서 볼륨을 수동으로 분리해야 합니다.*

- *etcd 스냅샷을 만든 후 클라우드 공급자 또는 스토리지 공급자 인증 정보가 업데이트되었습니다. 이로 인해 해당 인증 정보를 사용하는 CSI 드라이버 또는 Operator가 작동하지 않습니다. 해당 드라이버 또는 Operator에 필요한 인증 정보를 수동으로 업데이트해야 할 수 있습니다.*

- *etcd 스냅샷을 만든 후 OpenShift Container Platform 노드에서 장치가 제거되거나 이름이 변경됩니다. Local Storage Operator는 /dev/disk/by-id 또는 /dev 디렉터리에서 관리하는 각 PV에 대한 심볼릭 링크를 생성합니다. 이 경우 로컬 PV가 더 이상 존재하지 않는 장치를 참조할 수 있습니다.*

  이 문제를 해결하려면 관리자가 다음을 수행해야 합니다.

  1. *잘못된 장치가 있는 PV를 수동으로 제거합니다.*
  2. *각 노드에서 심볼릭 링크를 제거합니다.*
  3. *LocalVolume 또는 LocalVolumeSet 오브젝트를 삭제합니다 (스토리지 → 영구 스토리지 구성 → 로컬 볼륨을 사용하는 영구 스토리지 → Local Storage Operator 리소스 삭제참조).*

5.3.3. 만료된 컨트롤 플레인 인증서 복구

5.3.3.1. 만료된 컨트롤 플레인 인증서 복구

클러스터는 만료된 컨트롤 플레인 인증서에서 자동으로 복구될 수 있습니다.

그러나 kubelet 인증서를 복구하려면 대기 중인 node-bootstrapper 인증서 서명 요청(CSR)을 수동으로 승인해야 합니다. 사용자 프로비저닝 설치의 경우 보류 중인 kubelet 서비스 CSR을 승인해야 할 수

도 있습니다.

*보류중인 **CSR**을 승인하려면 다음 단계를 수행합니다.*

절차

1. *현재 **CSR**의 목록을 가져옵니다.*

```
$ oc get csr
```

출력 예

```
NAME        AGE    SIGNERNAME                                    REQUESTOR
CONDITION
csr-2s94x   8m3s   kubernetes.io/kubelet-serving                 system:node:<node_name>
Pending (1)
csr-4bd6t   8m3s   kubernetes.io/kubelet-serving                 system:node:<node_name>
Pending (2)
csr-4hl85   13m    kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending (3)
csr-zhhhp   3m8s   kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending (4)
...
```

(1) (2) *보류 중인 **kubelet** 서비스 **CSR**(사용자 프로비저닝 설치용)입니다.*

(3) (4) *보류 중인 **node-bootstrapper CSR**입니다.*

2. ***CSR**의 세부 사항을 검토하여 **CSR**이 유효한지 확인합니다.*

```
$ oc describe csr <csr_name> (1)
```

(1)

3. *각각의 유효한 **node-bootstrapper CSR**을 승인합니다.*

```
$ oc adm certificate approve <csr_name>
```

4. *사용자 프로비저닝 설치의 경우 **CSR**을 제공하는 각 유효한 **kubelet**을 승인합니다.*

```
$ oc adm certificate approve <csr_name>
```