Red Hat OpenStack Platform 16.1

업그레이드를 위한 프레임워크(13에서 16.1)

Red Hat OpenStack Platform 13에서 16.1로 즉각적 업그레이드

Last Updated: 2024-02-06

Red Hat OpenStack Platform 16.1 업그레이드를 위한 프레임워크(13에서 16.1)

Red Hat OpenStack Platform 13에서 16.1로 즉각적 업그레이드

OpenStack Team
rhos-docs@redhat.com

법적 공지

초록

이 가이드에는 롱라이프 버전에서 인플레이스 업그레이드를 위한 프레임워크에 대한 정보가 포함되어 있습니다. 이 프레임워크에는 OpenStack Platform 환경을 하나의 롱라이프 버전에서 다음 롱라이프 버전으로 업그레이드하는 도구가 포함되어 있습니다. 이 경우 가이드는 Red Hat OpenStack Platform 13(Queens)에서 16.1(Train)으로의 업그레이드에 중점을 둡니다.

차례

보다 포괄적 수용을 위한 오픈 소스 용어 교체

Red Hat은 코드, 문서, 웹 속성에서 문제가 있는 용어를 교체하기 위해 최선을 다하고 있습니다. 먼저 마스터(master), 슬레이브(slave), 블랙리스트(blacklist), 화이트리스트(whitelist) 등 네 가지 용어를 교체하고 있습니다. 이러한 변경 작업은 작업 범위가 크므로 향후 여러 릴리스에 걸쳐 점차 구현할 예정입니다. 자세한 내용은 CTO Chris Wright의 메시지 를 참조하십시오.

RED HAT 문서에 관한 피드백 제공

문서 개선을 위한 의견을 보내 주십시오. Red Hat이 어떻게 이를 개선하는지 알려주십시오.

DDF(직접 문서 피드백) 기능 사용

특정 문장, 단락 또는 코드 블록에 대한 직접 주석은 **피드백 추가** DDF 기능을 사용하십시오.

1. *다중 페이지 HTML* 형식으로 설명서를 봅니다.
2. 문서 오른쪽 상단에 **Feedback** (피드백) 버튼이 표시되는지 확인합니다.
3. 주석 처리하려는 텍스트 부분을 강조 표시합니다.
4. **피드백 추가**를 클릭합니다.
5. 주석을 사용하여 **Add Feedback** (피드백 추가) 필드를 작성합니다.
6. 선택 사항: 설명서 팀이 문제에 대한 자세한 내용을 문의할 수 있도록 이메일 주소를 추가하십시오.
7. **Submit(제출)**을 클릭합니다.

1장. 업그레이드를 위한 RED HAT OPENSTACK PLATFORM 프레임워크 정보

업그레이드를 위한 Red Hat OpenStack Platform 프레임워크는 Red Hat OpenStack Platform 환경을 긴 수명 버전에서 다음 장기 버전으로 업그레이드하는 워크플로입니다. 이 워크플로는 즉각적 솔루션이며 기존 환경 내에서 업그레이드가 수행됩니다.

1.1. 긴 라이프 버전을 위한 업그레이드 프레임워크

Red Hat OpenStack Platform **업그레이드 프레임워크를** 사용하여 여러 버전의 오버클라우드를 통해 즉각적 업그레이드 경로를 수행할 수 있습니다. 목표는 **수명이 긴 버전으로 간주되는 특정 OpenStack 버전을 유지하고 차후의 라이프사이클 버전을** 사용할 수 있을 때 업그레이드할 수 있는 기회를 제공하는 것입니다.

중요

Red Hat OpenStack Platform 13을 Red Hat OpenStack Platform 16.1로 업그레이드할 수 있습니다. 그러나 전체 제품 지원을 사용하려면 Red Hat OpenStack Platform 13에서 Red Hat OpenStack Platform 16.2 업그레이드 경로만 지원됩니다. 13에서 16.1로 업그레이드하려면 Support Exception을 가져와야 합니다.

Red Hat OpenStack Platform 16.2로 업그레이드하는 방법에 대한 자세한 내용은 Framework for upgrades 13 to 16.2 를 참조하십시오.

이 가이드에서는 다음 버전을 통해 업그레이드 프레임워크를 제공합니다.

현재 버전	대상 버전
Red Hat OpenStack Platform 13	Red Hat OpenStack Platform 16.1

1.2. 긴 라이프 사이클 버전에 대한 라이프 사이클 지원

Red Hat OpenStack Platform 라이프사이클 지원에 대한 자세한 지원 날짜 및 정보는 Red Hat OpenStack Platform 라이프 사이클 을 참조하십시오.

1.3. 롱라이프 릴리스의 업그레이드 경로

Red Hat은 다음 장기 릴리스로 환경을 업그레이드할 수 있는 두 가지 옵션을 제공합니다.

즉각적 업그레이드

기존 환경에서 서비스 업그레이드를 수행합니다. 이 가이드는 주로 이 옵션에 중점을 둡니다.

병렬 마이그레이션

새로운 Red Hat OpenStack Platform 16.1 환경을 생성하고 현재 환경에서 새 환경으로 워크로드를 마이그레이션합니다. Red Hat OpenStack Platform 병렬 마이그레이션에 대한 자세한 내용은 Red Hat Global Professional Services에 문의하십시오.

중요

이 표의 기간은 내부 테스트를 기반으로 하는 최소 추정치이며 모든 프로덕션 환경에 적용되지 않을 수 있습니다. 예를 들어 하드웨어에 사양이 낮거나 확장된 부팅 기간이 있는 경우 이러한 기간 동안 더 많은 시간을 할애할 수 있습니다. 각 작업의 업그레이드 기간을 정확하게 측정하려면 프로덕션 환경과 유사한 하드웨어를 사용하여 테스트 환경에서 다음 절차를 수행합니다.

표 1.1. 업그레이드 경로의 영향 및 기간

	즉각적 업그레이드	병렬 마이그레이션
언더클라우드의 업그레이드 기간	각 주요 작업의 예상 기간에는 다음이 포함됩니다. ● Leapp 업그레이드 명령의 경우 30분 ● Leapp을 재부팅하는 경우 30 분 ● director 업그레이드를 위한 40분	없음. 기존 언더클라우드 외에도 새 언더클라우드를 생성합니다.
오버클라우드 컨트롤 플레인 업그레이드 기간	각 컨트롤러 노드의 추정치: ● Leapp 업그레이드 및 재부팅을 위한 60분 ● 서비스 업그레이드를 위한 60분	없음. 기존 컨트롤 플레인 외에도 새 컨트롤 플레인을 생성합니다.
컨트롤 플레인의 중단 기간	부트스트랩 컨트롤러 노드의 서비스 업그레이드 기간은 약 60분입니다.	없음. 두 오버클라우드 모두 워크로드 마이그레이션 중에 작동합니다.
컨트롤 플레인 중단의 결과	중단하는 동안 OpenStack 작업을 수행할 수 없습니다.	중단 없음.
오버클라우드 데이터 플레인 업그레이드 기간	각 컴퓨팅 노드 및 Ceph Storage 노드의 추정치: ● Leapp 업그레이드 및 재부팅을 위한 60분 ● 서비스 업그레이드를 위한 30분	없음. 기존 데이터 플레인 외에도 새 데이터 플레인을 생성합니다.
데이터 플레인의 중단 기간	노드에서 노드로의 워크로드 마이그레이션으로 인해 중단이 최소화됩니다.	Overcloud에서 Overcloud로 워크로드 마이그레이션으로 인해 중단이 최소화됩니다.

	즉각적 업그레이드	병렬 마이그레이션
추가 하드웨어 요구 사항	추가 하드웨어는 필요하지 않습니다.	새로운 언더클라우드 및 Overcloud를 생성하려면 추가 하드웨어가 필요합니다.

2장. 즉각적 업그레이드 계획 및 준비

OpenStack Platform 환경을 즉각적으로 업그레이드하기 전에 업그레이드 계획을 생성하고 성공적인 업그레이드를 차단할 수 있는 잠재적인 장애물을 수용합니다.

2.1. RED HAT OPENSTACK PLATFORM 16.1 숙지

업그레이드를 수행하기 전에 Red Hat OpenStack Platform 16.1을 숙지하여 환경에 영향을 미칠 수 있는 잠재적인 버전 간 변경 사항을 파악할 수 있습니다. Red Hat OpenStack Platform 16.1을 숙지하려면 다음 권장 사항을 따르십시오.

- 업그레이드 경로의 모든 버전에 대한 릴리스 노트를 읽고 계획이 필요한 모든 잠재적인 측면을 확인합니다.
  - 새로운 기능이 포함된 구성 요소
  - 확인된 문제

  다음 링크를 사용하여 각 버전의 릴리스 노트를 엽니다.

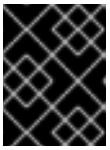
  - 현재 버전인 Red Hat OpenStack Platform 13
  - Red Hat OpenStack Platform 14
  - Red Hat OpenStack Platform 15
  - Red Hat OpenStack Platform 16.0
  - 대상 버전인 Red Hat OpenStack Platform 16.1
- 버전 16.1에 대한 Director 설치 및 사용 가이드를 읽고 이 가이드의 새로운 요구 사항과 프로세스를 숙지하십시오.
- 개념 증명 Red Hat OpenStack Platform 16.1 언더클라우드 및 오버클라우드 설치. 대상 OpenStack Platform 버전의 핸즈온 경험을 개발하고 대상 버전과 현재 버전 간의 잠재적인 차이점을 조사합니다.

2.2. RED HAT OPENSTACK PLATFORM 16.1의 상위 수준 변경

Red Hat OpenStack Platform 16.1로 업그레이드하는 동안 다음과 같은 고급 변경 사항이 발생합니다.

- OpenStack Platform director 16.1은 **config-download** 라는 Ansible 주도 방법을 사용하여 오버클라우드를 구성합니다. 이는 표준 heat 기반 구성 방법을 대체합니다. director는 heat를 사용하여 프로비저닝 작업을 오케스트레이션합니다.
- director 설치는 Overcloud 배포와 동일한 방법을 사용합니다. 따라서 언더클라우드에서는 **openstack-tripleo-heat-templates** 를 각 서비스를 설치하고 구성하는 데 청사진으로 사용합니다.
- Undercloud는 컨테이너에서 OpenStack 서비스를 실행합니다.
- Undercloud는 새로운 방법을 통해 컨테이너 이미지를 가져와서 저장합니다. Overcloud를 배포하기 전에 컨테이너 이미지를 가져오는 대신, Undercloud는 배포 프로세스 중에 관련 컨테이너 이미지를 모두 가져옵니다.

- 오버클라우드 배포 프로세스에는 노드를 등록할 고급 서브스크립션 관리 방법이 포함되어 있습니다. 이 메서드는 Ansible 역할을 통합하여 OpenStack Platform 노드를 등록합니다. 또한 새 방법은 필요한 경우 다른 노드 역할에 다른 서브스크립션을 적용합니다.
- 이제 오버클라우드에서 OVN(Open Virtual Network)을 기본 ML2 메커니즘 드라이버로 사용합니다. 성공적인 업그레이드가 완료된 후 수행하는 OVN(Open vSwitch) 서비스를 마이그레이션할 수 있습니다.
- 언더클라우드와 오버클라우드는 모두 Red Hat Enterprise Linux 8에서 실행됩니다.
- **openstack-tripleo-heat-templates** 에는 **배포** 디렉터리에 통합 구성 가능 서비스 템플릿 컬렉션이 포함되어 있습니다. 이 디렉터리에는 이제 컨테이너화된 서비스 및 Puppet 기반 구성 가능 서비스 템플릿의 병합된 콘텐츠가 포함된 템플릿이 포함되어 있습니다.
- OpenStack Data Processing 서비스(sahara)는 더 이상 지원되지 않습니다.



중요

Red Hat OpenStack Platform 13 환경에서 sahara를 활성화한 경우 이 업그레이드를 계속 진행하지 말고 Red Hat 글로벌 지원 서비스에 문의하십시오.

- OpenStack Telemetry 구성 요소는 더 이상 사용되지 않는 STF(Service Telemetry Framework)를 사용합니다.

2.3. RED HAT ENTERPRISE LINUX 8의 변경 사항

언더클라우드와 오버클라우드는 모두 Red Hat Enterprise Linux 8에서 실행됩니다. 여기에는 언더클라우드 및 오버클라우드와 관련된 새로운 툴과 기능이 포함됩니다.

- 언더클라우드 및 오버클라우드는 Red Hat Container Toolkit을 사용합니다. 컨테이너 라이프사이클을 빌드하고 제어하는 **docker** 대신 Red Hat Enterprise Linux 8에는 새 컨테이너 이미지를 빌드하는 **buildah** 와 컨테이너 관리를 위한 **podman** 이 포함되어 있습니다.
- Red Hat Enterprise Linux 8에는 **docker-distribution** 패키지가 포함되지 않습니다. 이제 언더클라우드에 컨테이너 이미지를 Overcloud 노드에 제공하는 프라이빗 HTTP 레지스트리가 포함됩니다.
- Red Hat Enterprise Linux 7에서 8로의 업그레이드 프로세스는 힙 **도구를** 사용합니다.
- Red Hat Enterprise Linux 8은 **ntp** 서비스를 사용하지 않습니다. 대신 Red Hat Enterprise Linux 8은 **chronyd** 를 사용합니다.
- Red Hat Enterprise Linux 8에는 새로운 버전의 고가용성 툴이 포함되어 있습니다.

Red Hat OpenStack Platform 16.1은 Red Hat Enterprise Linux 8.2를 기본 운영 체제로 사용합니다. 업그레이드 프로세스의 일환으로 노드의 기본 운영 체제를 Red Hat Enterprise Linux 8.2로 업그레이드합니다.

Red Hat Enterprise Linux 7과 8의 주요 차이점에 대한 자세한 내용은 RHEL 8 채택 시 고려 사항을 참조하십시오. Red Hat Enterprise Linux 8에 대한 일반 정보는 제품 설명서에서 Red Hat Enterprise Linux 8을 참조하십시오.

2.4. RED HAT OPENSTACK PLATFORM의 LEAPP 업그레이드 사용

Red Hat OpenStack Platform의 장기 업그레이드에는 Red Hat Enterprise Linux 7에서 Red Hat Enterprise Linux 8로 기본 운영 체제 업그레이드가 필요합니다. Red Hat Enterprise Linux 7은 Leapp 유틸

리티를 사용하여 Red Hat Enterprise Linux 8로의 업그레이드를 수행합니다. Leapp 및 해당 종속 항목을 사용할 수 있는지 확인하려면 다음 Red Hat Enterprise Linux 7 리포지토리가 활성화되어 있는지 확인합니다.

- Red Hat Enterprise Linux 7 Server RPMs x86_64 7Server 또는 Red Hat Enterprise Linux 7 Server RPMs x86_64 7.9

```
rhel-7-server-rpms
x86_64 7Server
or:
rhel-7-server-rpms
x86_64 7.9
```

- Red Hat Enterprise Linux 7 Server - Extras RPMs x86_64

```
rhel-7-server-extras-rpms
x86_64
```

자세한 내용은 업그레이드용 RHEL 7 시스템 준비를 참조하십시오.

Undercloud 및 오버클라우드는 운영 체제 업그레이드를 수행하기 위해 별도의 프로세스를 사용합니다.

언더클라우드 프로세스

openstack undercloud upgrade 명령을 실행하기 전에 업그레이드를 수동으로 실행합니다. 언더클라우드 업그레이드에는 sk **p 업그레이드를 수행하는 지침이** 포함되어 있습니다.

오버클라우드 프로세스

오버클라우드 업그레이드 프레임워크는 rep **업그레이드를 자동으로** 실행합니다.

제한

업그레이드에 영향을 줄 수 있는 잠재적인 제한 사항에 대한 자세한 내용은 *RHEL 7에서 RHEL 8로의 업그레이드* 가이드에서 다음 섹션을 참조하십시오.

- "업그레이드 계획"
- "알 수 있는 문제"

특히 LUKS 암호화 또는 파일 시스템 암호화와 같은 전체 디스크 또는 파티션의 암호화를 사용하는 노드에서 Leapp 업그레이드를 수행할 수 없습니다. 이 제한 사항은 **dmcrypt: true** 매개변수를 사용하여 구성한 Ceph OSD 노드에 영향을 미칩니다.

알려진 제한 사항이 환경에 영향을 주는 경우 Red Hat 기술 지원 팀의 조언을 요청하십시오 .

문제 해결

잠재적인 Leapp 문제 해결에 대한 자세한 내용은 *RHEL 7 에서 RHEL 8로의 업그레이드를* 참조하십시오.

2.5. 지원되는 업그레이드 시나리오

업그레이드를 진행하기 전에 오버클라우드가 지원되는지 확인합니다.

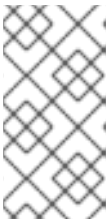
참고

해당 목록에 언급되지 않은 특정 시나리오가 지원되는지 여부가 확실하지 않은 경우 Red Hat 기술 지원 팀의 조언을 요청하십시오.

지원되는 시나리오

다음과 같은 즉각적 업그레이드 시나리오가 테스트 및 지원됩니다.

- 기본 역할 유형의 표준 환경: 컨트롤러, 계산 및 Ceph 스토리지 OSD
- split-Controller 구성 가능 역할
- Ceph Storage 구성 가능 역할
- Hyper-Converged Infrastructure: 동일한 노드의 Compute 및 Ceph Storage OSD 서비스
- NFV(Network Functions Virtualization) 기술을 사용하는 환경: SR-IOV(Single-root input/output virtualization) 및 DPDK(Data Plane Development Kit)
- 인스턴스 HA가 활성화된 환경



참고

업그레이드 프로세스 중에 nova 실시간 마이그레이션이 지원됩니다. 그러나 인스턴스 HA에서 시작한 비우기는 지원되지 않습니다. 컴퓨팅 노드를 업그레이드할 때 노드가 완전히 종료되고 노드에서 실행되는 모든 워크로드가 인스턴스 HA에서 자동으로 비우지 않습니다. 대신 실시간 마이그레이션을 수동으로 수행해야 합니다.

기술 프리뷰 시나리오

업그레이드를 위한 프레임워크는 이러한 기능과 함께 사용할 때 기술 프리뷰로 간주되므로 Red Hat에서 완전히 지원하지 않습니다. 개념 증명 환경에서만 이 시나리오를 테스트하고 프로덕션 환경에서는 업그레이드하지 않아야 합니다. 기술 프리뷰 기능에 대한 자세한 내용은 적용 범위 상세 정보를 참조하십시오.

- 엣지 및 DCN(Distributed Compute Node) 시나리오

2.6. 외부 CEPH 배포를 사용한 업그레이드 고려 사항

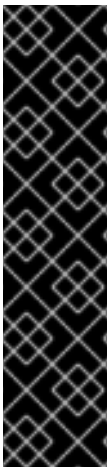
Red Hat Ceph Storage 시스템을 별도로 배포한 다음 director를 사용하여 OpenStack을 배포하고 구성하는 경우 Red Hat OpenStack Platform 프레임워크를 사용하여 외부 Ceph 배포에서 즉각적 업그레이드를 수행할 수 있습니다. 이 시나리오는 director를 사용하여 배포된 Ceph 클러스터를 업그레이드하는 것과는 다릅니다.

외부 Ceph 배포를 사용하여 즉각적 업그레이드를 계획하고 준비할 때 고려해야 하는 차이점은 다음과 같습니다.

1. Red Hat OpenStack Platform 배포를 버전 13에서 버전 16.1로 업그레이드하려면 Red Hat Ceph Storage 클러스터를 버전 3에서 버전 4로 업그레이드해야 합니다. 자세한 내용은 Red Hat Ceph Storage *4 설치 가이드의 Red Hat Ceph Storage 클러스터 업그레이드*를 참조하십시오.
2. Red Hat Ceph Storage 클러스터를 버전 3에서 버전 4로 업그레이드한 후에도 Red Hat OpenStack Platform 13은 RHCSv3 클라이언트 구성 요소를 계속 실행할 수 있지만 RHCSv4 클러스터와 호환됩니다.

3. Framework For Upgrades(13에서 16.1) 문서에 설명된 업그레이드 경로를 따를 수 있으며 해당하는 경우 이 특정 시나리오를 지원하는 조건부 단계를 완료해야 합니다. 조건부 단계는 다음 문으로 시작합니다. "외부 Ceph 배포로 업그레이드하는 경우".

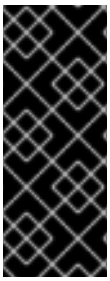
4. 외부 Ceph 배포로 업그레이드할 때 오버클라우드 업그레이드 프로세스의 일부로 RHCSv4 **ceph-ansible** 을 설치합니다. director를 사용하여 배포된 Ceph 클러스터를 업그레이드하면 오버클라우드 업그레이드 프로세스가 완료된 후 RHCSv4 **ceph-ansible** 을 설치합니다.



중요

지원되는 이전 버전에서 버전 4.2z2로 Red Hat Ceph Storage 클러스터를 업그레이드하면 업그레이드가 **HEALTH_WARN** 상태에서 스토리지 클러스터와 함께 완료 **되며 상태 모니터링이 안전하지 않은 global_id 회수를 허용합니다**. 이는 패치된 CVE(CVE-2021-20288)로 인해 RHCS 4.2z2 (이상)로 설치/업그레이드 후 Ceph HEALTH_WARN(mons이 안전하지 않은 global_id 회수 허용) 을 참조하십시오.

CVE로 인해 **HEALTH_WARN** 상태가 표시되므로 상태 경고를 일시적으로 음소거할 수 있습니다. 그러나 음소거 경고가 발생하면 클러스터에 연결된 이전 및 패키징되지 않은 잠재적인 클라이언트에 대한 가시성이 없다는 위험이 있습니다. 상태 경고 변경에 대한 자세한 내용은 Red Hat Ceph Storage 문서의 Upgrading a *Red Hat Ceph Storage* cluster 를 참조하십시오.



중요

모든 클라이언트가 업그레이드 및 패치되지 않으면 연결할 수 없을 때까지 수동으로 **global_id_reclaim** 을 비활성화하지 마십시오. 다음 명령을 **root** 사용자로 실행하여 클러스터에 연결된 패치되지 않은 클라이언트 목록을 볼 수 있습니다.

```
# ceph health detail
```

2.7. 업그레이드를 차단할 수 있는 알려진 문제

성공적인 업그레이드에 영향을 줄 수 있는 다음 알려진 문제를 검토합니다.

BZ#1997351 - (13octets16.1) 인스턴스는 부트스트랩 컨트롤러 업그레이드 후 액세스할 수 없습니다.

ML2-OVN을 사용하여 배포된 RHOSP(Red Hat OpenStack Platform) 13 환경을 업그레이드하면 컨트롤러 노드의 업그레이드 프로세스가 실패할 수 있습니다. Leapp 재부팅 후 SELinux 권한 거부로 인해 **ovn-dbs** 컨테이너가 시작되지 않을 수 있습니다. 버그 BZ#1997351 을 방지하는 방법에 대한 자세한 내용은 OSP-13 → OSP-16.1 FFU 동안 재부팅 후 Red Hat Knowledgebase 솔루션 OVN 이 구성되지 않음을 참조하십시오.

BZ#1902849 - 이전에 osp8, osp10에서 업그레이드된 클러스터에서 osp13-osp16.1 ffu가 실패합니다.

버전 RHOSP 10에서 이전에 업그레이드된 RHOSP(Red Hat OpenStack Platform) 환경에는 BZ#1902849 를 방지하려면 **python-docker** 패키지가 필요합니다. 자세한 내용은 Red Hat Knowledgebase 솔루션 osp13-osp16.1 ffu 실패를 참조하십시오.

BZ#1925078 - RHOSP13-16.1 FFU: 잘못된 ceph 이미지에 대한 참조가 실패한 후 오버클라우드 업그레이드가 컨트롤러에 중단됩니다.

UEFI 부팅 및 OSP13에서 UEFI 부트로더를 사용하는 시스템은 다음과 같은 UEFI 문제가 발생할 수 있습니다.

- **/etc/fstab** 이 업데이트되지 않음
- GRUB-install이 EFI 시스템에서 잘못 사용됨

자세한 내용은 Red Hat Knowledgebase 솔루션 FFU 13~16.1을 참조하십시오. LeApp은 UEFI 기반 시스템에서 커널을 업데이트하지 못하며 /etc/fstab에 EFI 파티션이 포함되지 않습니다..

시스템에서 UEFI를 사용하는 경우 Red Hat 기술 지원에 문의하십시오.

BZ#1895887 - ovs+dpdk가 장치 OvsDpdkHCI를 연결하지 못했습니다.

Leapp 유틸리티로 업그레이드한 후 OVS-DPDK 워크로드가 있는 컴퓨팅 노드가 제대로 작동하지 않습니다. 이 문제를 해결하려면 다음 단계 중 하나를 수행하십시오.
컴퓨팅 노드를 업그레이드하기 전에 **/etc/modules-load.d/vfio-pci.conf** 파일을 제거합니다.

또는

컴퓨팅 노드를 업그레이드한 후 컴퓨팅 노드에서 **ovs-vswitchd** 서비스를 다시 시작합니다.

이 문제는 RHOSP 16.1.3에 영향을 미칩니다. 자세한 내용은 HCI 컴퓨팅 노드의 OSP 13에서 16.1로 프레임워크 업그레이드 후 Red Hat Knowledgebase 솔루션 OVS-DPDK 오류를 참조하십시오.

BZ#1936419 - FFU 13-16.1 업그레이드: leap 매개변수로 ceph 노드에서 LeApp 업그레이드에 실패하여 Fast datapath 리포지터리를 활성화합니다.

Ceph 서브스크립션을 사용하고 director에서 Ceph 스토리지 노드에 **overcloud-minimal** 이미지를 사용하도록 구성한 경우 Leapp 제한으로 인해 Ceph 스토리지 노드용 운영 체제 업그레이드가 실패할 수 있습니다. 이 문제를 방지하려면 **system_upgrade** 실행 단계가 끝나면 Ceph 노드에 로그인하여 RHEL 마이너 릴리스 버전을 설정 해제하고 사용 가능한 최신 RHEL 마이너 릴리스 버전으로 업데이트하고 노드를 재부팅해야 합니다.
Red Hat Satellite Server를 사용하여 Leapp 업그레이드에 대한 RPM 콘텐츠를 호스팅하는 경우 사용하는 콘텐츠 뷰에 다음 8.2 리포지토리를 추가해야 합니다.

- Red Hat Enterprise Linux 8 for x86_64 - AppStream(RPM)

```
rhel-8-for-x86_64-appstream-rpms
x86_64 8.2
```

- Red Hat Enterprise Linux 8 for x86_64 - BaseOS(RPM)

```
rhel-8-for-x86_64-baseos-rpms
x86_64 8.2
```

이 가이드에는 이 문제를 방지하는 해결 방법이 포함되어 있습니다.

BZ#2016144 - FFU 13-16.1: Leapp 업그레이드 재부팅 중에 openvswitch가 오류시작 **ovsdb-server ovsdb-server: /var/run/openvswitch/ovsdb-server.pid.tmp: create failed (Permission denied)**

이전 버전에서 업그레이드된 RHOSP(Red Hat OpenStack Platform) 환경에는 **/etc/systemd/system/ovs*** 에 불필요한 파일이 포함될 수 있습니다. 오버클라우드 업그레이드 프로세스를 RHOSP 13에서 RHOSP 16.1로 시작하기 전에 이러한 파일을 삭제해야 합니다.

BZ#2008976 - Leapp의 종속 항목에서 Leapp 업그레이드 실패 후 Python2 패키지 정리

Leapp 버전 5.0.8-100.202109241452Z.1332835에서는 Leapp 패키지를 유지하는 DNF **제외** 옵션으로 인해 **python2** Leapp 패키지의 자동 제거가 실패했습니다.
환경 파일에 **UpgradeInitCommand** 매개변수를 포함하고 DNF 제외 문을 제거합니다.

```
parameter_defaults:
  UpgradeInitCommand: "sudo dnf config-manager --save --setopt exclude=""
```

자세한 내용은 업그레이드 환경 파일 생성을 참조하십시오.

BZ#1978228 - OSP13tekton16.2 Leapp 업그레이드가 TLSEverywhere에서 실패했습니다.

해당 환경에서 TLS-Everywhere를 사용하고 **authconfig** 에서 authselect 로 마이그레이션하려는 경우 **authselect _check.confirm** 매개변수를 **True** 로 설정합니다. 그렇지 않으면 이 값을 **False** 로 설정합니다. 자세한 내용은 업그레이드 환경 파일 생성을 참조하십시오.

BZ#2021525 - openstack overcloud upgrade run times out / HAProxy container fails to start

잘못된 SELinux 레이블로 인해 배포 단계 중에 RHOSP 13에서 RHOSP 16.1로의 업그레이드가 실패할 수 있습니다. 해결 방법 및 자세한 내용은 OSP13 - OSP16.x FFU 중에 Red Hat Knowledgebase 솔루션 Pacemaker 관리 서비스가 다시 시작되지 않을 수 있습니다.

BZ#2015325 - FFU: "Upgrade Mysql database from a temporary container" 단계에서 업그레이드에 실패했습니다.

Red Hat Enterprise Linux에는 **mariadb-server** 용 최신 RPM이 포함되어 있어 RHOSP(Red Hat OpenStack Platform)에서 컨테이너화된 mariadb의 업그레이드를 방해합니다. RHOSP 업그레이드를 수행하기 전에 컨트롤러 호스트에서 **mariadb-server** 패키지를 제거하십시오. 자세한 내용은 업그레이드 환경 파일 생성을 참조하십시오.

BZ#2024447 - placement 사용자의 Identity 서비스(keystone) 암호는 FFU RHOSP 13에서 16까지 NovaPassword로 재정의되었습니다.

Red Hat OpenStack Platform 13에서 16.1로 업그레이드하는 동안 **NovaPassword** 매개변수의 값을 정의하지만 **PlacementPassword** 매개변수는 배치 사용자의 OpenStack ID 서비스(keystone) 암호를 재정의합니다. ID 서비스 암호를 유지하려면 **parameter_defaults** 섹션에 **NovaPassword** 또는 **PlacementPassword** 를 설정하지 마십시오.
parameter_defaults 섹션에서 두 암호를 모두 설정하면 Compute 노드가 업그레이드될 때까지 컨트롤 플레인과 통신하지 못할 수 있습니다. 컴퓨팅 노드 업그레이드에 대한 자세한 내용은 Compute 노드 업그레이드를 참조하십시오. ???

또한 **NovaPassword**,**PlacementPassword** 또는 둘 다를 사용하여 RHOSP 13에 오버클라우드를 배포한 경우, RHOSP 16.1로 업그레이드하기 전에 해당 암호를 템플릿에서 제거하고 RHOSP 13에서 **openstack overcloud deploy** 명령을 실행해야 합니다.

BZ#2164396 - FFU: FFU(13~ 16.2)에 사용할 RedHat Satellite 툴 리포지토리

Satellite 버전 6.7을 사용하는 경우 Red Hat Satellite Tools for RHEL 8 Server RPMs x86_64 리포지토리를 활성화할 때 업그레이드가 실패합니다. 적절한 패키지를 설치할 수 없기 때문에 오류가 발생합니다. Red Hat 엔지니어링 팀은 이 문제에 대한 해결책을 조사하고 있습니다.

Red Hat Ceph Storage 문제

BZ#1855813 - Ceph 툴 리포지토리는 외부 업그레이드를 실행하기 전에 통합한 후에만 RHCS3에서 RHCS4로 전환해야 합니다.

언더클라우드의 **ceph-ansible** 플레이북 컬렉션은 오버클라우드에 Red Hat Ceph Storage 컨테이너를 배포합니다. 환경을 업그레이드하려면 업그레이드를 통해 Ceph Storage 3 컨테이너를 유지 관리하려면 Red Hat Ceph Storage 3 버전의 **ceph-ansible** 이 있어야 합니다. 이 가이드에서는 Ceph Storage 4로 업그레이드할 준비가 될 때까지 업그레이드 과정에서 **ceph-ansible** 버전 3을 유지하는 방법을 설명합니다. 13에서 16.1로의 업그레이드를 수행하기 전에 Red Hat OpenStack Platform 13 환경의 마이너 버전 업데이트를 수행하고 **ceph-ansible** 버전 3.2.46 이상이 있는지 확인해야 합니다.

2.8. 백업 및 복원

Red Hat OpenStack Platform 13 환경을 업그레이드하기 전에 언더클라우드 및 오버클라우드 컨트롤 플레인을 백업합니다. Relax-and-recover(ReaR) 유틸리티를 사용하여 노드를 백업하는 방법에 대한 자세한 내용은 *Undercloud 및 컨트롤 플레인 백업 및 복원* 가이드를 참조하십시오.

- 업그레이드를 수행하기 전에 노드를 백업합니다. 업그레이드하기 전에 노드 백업에 대한 자세한 내용은 Red Hat OpenStack Platform 13 Undercloud 및 컨트롤 플레인 백업 및 복원을 참조하십시오.
- 업그레이드한 후 각 노드를 백업할 수 있습니다. 업그레이드된 노드 백업에 대한 자세한 내용은 언더클라우드 및 컨트롤 플레인 노드 백업 및 복원을 참조하십시오.
- 언더클라우드 업그레이드 후 언더클라우드 노드에서 실행되는 데이터베이스를 백업한 후 오버클라우드 업그레이드를 수행할 수 있습니다. 언더클라우드 데이터베이스 백업에 대한 자세한 내용은 언더클라우드 및 *컨트롤 플레인 노드 지원 및 복원에서 언더클라우드 노드의 데이터베이스 백업 생성* 을 참조하십시오.

2.9. 마이너 버전 업데이트

Red Hat OpenStack Platform 환경을 업그레이드하기 전에 환경을 현재 릴리스의 최신 마이너 버전으로 업데이트합니다. 예를 들어 Red Hat OpenStack Platform 16.1로 업그레이드를 실행하기 전에 Red Hat OpenStack Platform 13 환경을 최신 13으로 업데이트합니다.

Red Hat OpenStack Platform 13에 대한 마이너 버전 업데이트 수행에 대한 자세한 내용은 Red Hat OpenStack Platform 업데이트 유지를 참조하십시오.

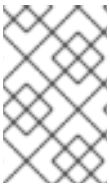
2.10. 프록시 설정

Red Hat OpenStack Platform 13 환경에서 프록시를 사용하는 경우 운영 체제 업그레이드 및 Red Hat OpenStack Platform 16.1 업그레이드 후에도 **/etc/environment** 파일의 프록시 구성이 유지됩니다.

- Red Hat OpenStack Platform 13의 프록시 구성에 대한 자세한 내용은 프록시를 사용하여 언더클라우드를 실행할 때 고려 사항을 참조하십시오.
- Red Hat OpenStack Platform 16.1의 프록시 구성에 대한 자세한 내용은 프록시를 사용하여 언더클라우드를 실행할 때 고려 사항을 참조하십시오.

2.11. 업그레이드 전 RED HAT OPENSTACK PLATFORM 13 검증

Red Hat OpenStack Platform 16.1로 업그레이드하기 전에 **tripleo-validations** 플레이북을 사용하여 언더클라우드 및 오버클라우드의 유효성을 검사합니다. Red Hat OpenStack Platform 13에서는 OpenStack Workflow 서비스(mistral)를 통해 이러한 플레이북을 실행합니다.



참고

CDN 또는 Satellite를 리포지토리 소스로 사용하는 경우 유효성 검사가 실패합니다. 이 문제를 해결하려면 Red Hat Knowledgebase 솔루션에서 SSL 인증서 오류로 인해 리포지토리 유효성 검사가 실패합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

3. **pre-upgrade-validations.sh** 라는 bash 스크립트를 생성하고 스크립트에 다음 내용을 포함합니다.

```
#!/bin/bash
for VALIDATION in $(openstack action execution run tripleo.validations.list_validations
'{"groups": ["pre-upgrade"]}' | jq ".result[] | .id")
do
  echo "=== Running validation: $VALIDATION ==="
  STACK_NAME=$(openstack stack list -f value -c 'Stack Name')
  ID=$(openstack workflow execution create -f value -c ID tripleo.validations.v1.run_validation
"{\"validation_name\": $VALIDATION, \"plan\": \"$STACK_NAME\"}")
  while [ $(openstack workflow execution show $ID -f value -c State) == "RUNNING" ]
  do
    sleep 1
  done
  echo ""
  openstack workflow execution output show $ID | jq -r ".stdout"
  echo ""
done
```

4. 스크립트를 실행할 권한을 추가합니다.

```
$ chmod +x pre-upgrade-validations.sh
```

5. 스크립트를 실행합니다.

```
$ ./pre-upgrade-validations.sh
```

스크립트 출력을 검토하여 검증이 성공하고 실패했는지 확인합니다.

```
=== Running validation: "check-ftype" ===

Success! The validation passed for all hosts:
* undercloud
```

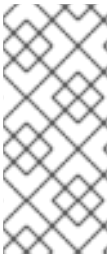
3장. 리포지토리

이 섹션에는 언더클라우드 및 오버클라우드용 리포지토리가 포함되어 있습니다. 특정 상황에서 리포지토리를 활성화해야 하는 경우 이 섹션을 참조하십시오.

- Red Hat 고객 포털에 등록할 때 리포지토리 활성화.
- Red Hat Satellite Server에 리포지토리를 활성화하고 동기화합니다.
- Red Hat Satellite Server에 등록할 때 리포지토리를 활성화합니다.

3.1. 언더클라우드 리포지토리

RHOSP(Red Hat OpenStack Platform) 16.1은 Red Hat Enterprise Linux 8.2에서 실행됩니다. 그러므로 해당 리포지토리의 콘텐츠를 해당하는 Red Hat Enterprise Linux 버전에 고정해야 합니다.



참고

Red Hat Satellite를 사용하여 리포지토리를 동기화하는 경우 특정 버전의 Red Hat Enterprise Linux 리포지토리를 활성화할 수 있습니다. 그러나 리포지토리 레이블은 선택한 버전에도 동일하게 유지됩니다. 예를 들어 8.2 버전의 BaseOS 리포지토리를 활성화하면 리포지토리 이름에 활성화된 특정 버전이 포함되지만 리포지토리 레이블은 여전히 **rhel-8-for-x86_64-baseos-eus-rpms** 입니다.

주의

여기에 지정된 리포지토리만 지원됩니다. 아래 표에 나열되지 않은 제품이나 리포지토리는 별도로 권장되지 않는 한 활성화하지 마십시오. 그러지 않으면 패키지 종속성 문제가 발생할 수 있습니다. EPEL(Extra Packages for Enterprise Linux)을 활성화하지 마십시오.

코어 리포지토리

다음 표에는 언더클라우드 설치에 사용되는 코어 리포지토리가 나와 있습니다.

이름	리포지토리	요구 사항 설명
Red Hat Enterprise Linux 8 for x86_64 - BaseOS(RPM) EUS(Extended Update Support)	**rhel-8-for-x86_64-baseos-eus-rpms**	x86_64 시스템용 기본 운영 체제 리포지토리입니다.
Red Hat Enterprise Linux 8 for x86_64 - AppStream(RPM)	**rhel-8-for-x86_64-appstream-eus-rpms**	Red Hat OpenStack Platform 종속 패키지를 포함합니다.
Red Hat Enterprise Linux 8 for x86_64 - High Availability (RPMs) Extended Update Support (EUS)	**rhel-8-for-x86_64-highavailability-eus-rpms**	Red Hat Enterprise Linux용 고가용성 툴입니다. 컨트롤러 노드 고가용성에 사용됩니다.

이름	리포지토리	요구 사항 설명
Red Hat Ansible Engine 2.9 for RHEL 8 x86_64(RPM)	**ansible-2.9-for-rhel-8-x86_64-rpms**	Ansible Engine for Red Hat Enterprise Linux입니다. 최신 버전의 Ansible을 제공하는 데 사용됩니다.
Advanced Virtualization for RHEL 8 x86_64(RPM)	**advanced-virt-for-rhel-8-x86_64-rpms**	OpenStack Platform용 가상화 패키지를 제공합니다.
Red Hat Satellite Tools for RHEL 8 Server RPMs x86_64	**satellite-tools-6.5-for-rhel-8-x86_64-rpms**	Red Hat Satellite 6으로 호스트를 관리하는 툴입니다.
Red Hat OpenStack Platform 16.1 for RHEL 8(RPM)	**openstack-16.1-for-rhel-8-x86_64-rpms**	Red Hat OpenStack Platform 코어 리포지토리로 Red Hat OpenStack Platform director용 패키지가 포함됩니다.
Red Hat Fast Datapath for RHEL 8(RPMS)	**fast-datapath-for-rhel-8-x86_64-rpms**	OpenStack Platform용 OVS(Open vSwitch) 패키지를 제공합니다.

Ceph 리포지토리

다음 표에는 언더클라우드의 Ceph Storage 관련 리포지토리가 나열되어 있습니다.

이름	리포지토리	요구 사항 설명
Red Hat Ceph Storage Tools 4 for RHEL 8 x86_64 (RPM)	**rhceph-4-tools-for-rhel-8-x86_64-rpms**	노드가 Ceph Storage 클러스터와 통신할 수 있는 툴을 제공합니다. 오버클라우드에서 Ceph Storage를 사용하거나 기존 Ceph Storage 클러스터와 통합하려는 경우 언더클라우드에 이 리포지토리의 **ceph-ansible** 패키지가 필요합니다.

IBM POWER 리포지토리

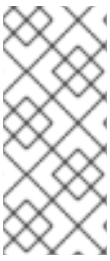
다음 표에는 POWER PC 아키텍처의 RHOSP용 리포지토리 목록이 나와 있습니다. 코어 리포지토리의 리포지토리 대신 이 리포지토리를 사용하십시오.

이름	리포지토리	요구 사항 설명
Red Hat Enterprise Linux for IBM Power, little endian - BaseOS(RPM)	**rhel-8-for-ppc64le-baseos-rpms**	ppc64le 시스템용 기본 운영 체제 리포지토리입니다.

이름	리포지토리	요구 사항 설명
Red Hat Enterprise Linux 8 for IBM Power, little endian - AppStream(RPM)	**rhel-8-for-ppc64le-appstream-rpms**	Red Hat OpenStack Platform 종속성을 포함합니다.
Red Hat Enterprise Linux 8 for IBM Power, little endian - High Availability(RPM)	**rhel-8-for-ppc64le-highavailability-rpms**	Red Hat Enterprise Linux용 고가용성 툴입니다. 컨트롤러 노드 고가용성에 사용됩니다.
Red Hat Fast Datapath for RHEL 8 IBM Power, little endian(RPMS)	**fast-datapath-for-rhel-8-ppc64le-rpms**	OpenStack Platform용 OVS(Open vSwitch) 패키지를 제공합니다.
Red Hat Ansible Engine 2.8 for RHEL 8 IBM Power, little endian(RPM)	**ansible-2.8-for-rhel-8-ppc64le-rpms**	Ansible Engine for Red Hat Enterprise Linux입니다. 최신 버전의 Ansible을 제공합니다.
Red Hat OpenStack Platform 16.1 for RHEL 8(RPM)	**openstack-16.1-for-rhel-8-ppc64le-rpms**	ppc64le 시스템용 Red Hat OpenStack Platform 코어 리포지토리입니다.

3.2. 오버클라우드 리포지토리

RHOSP(Red Hat OpenStack Platform) 16.1은 Red Hat Enterprise Linux 8.2에서 실행됩니다. 그러므로 해당 리포지토리의 콘텐츠를 해당하는 Red Hat Enterprise Linux 버전에 고정해야 합니다.



참고

Red Hat Satellite를 사용하여 리포지토리를 동기화하는 경우 특정 버전의 Red Hat Enterprise Linux 리포지토리를 활성화할 수 있습니다. 그러나 리포지토리 레이블은 선택한 버전에도 동일하게 유지됩니다. 예를 들어 8.2 버전의 BaseOS 리포지토리를 활성화하면 리포지토리 이름에 활성화된 특정 버전이 포함되지만 리포지토리 레이블은 여전히 **rhel-8-for-x86_64-baseos-eus-rpms** 입니다.

주의

여기에 지정된 리포지토리만 지원됩니다. 아래 표에 나열되지 않은 제품이나 리포지토리는 별도로 권장되지 않는 한 활성화하지 마십시오. 그러지 않으면 패키지 종속성 문제가 발생할 수 있습니다. EPEL(Extra Packages for Enterprise Linux)을 활성화하지 마십시오.

컨트롤러 노드 리포지토리

다음 표에는 오버클라우드에서 컨트롤러 노드를 위한 코어 리포지토리가 나와 있습니다.

이름	리포지토리	요구 사항 설명
Red Hat Enterprise Linux 8 for x86_64 - BaseOS(RPM) EUS(Extended Update Support)	**rhel-8-for-x86_64-baseos-eus-rpms**	x86_64 시스템용 기본 운영 체제 리포지토리입니다.
Red Hat Enterprise Linux 8 for x86_64 - AppStream(RPM)	**rhel-8-for-x86_64-appstream-eus-rpms**	Red Hat OpenStack Platform 종속성을 포함합니다.
Red Hat Enterprise Linux 8 for x86_64 - High Availability (RPMs) Extended Update Support (EUS)	**rhel-8-for-x86_64-highavailability-eus-rpms**	Red Hat Enterprise Linux용 고가용성 툴입니다.
Red Hat Ansible Engine 2.9 for RHEL 8 x86_64(RPM)	**ansible-2.9-for-rhel-8-x86_64-rpms**	Ansible Engine for Red Hat Enterprise Linux입니다. 최신 버전의 Ansible을 제공하는 데 사용됩니다.
Advanced Virtualization for RHEL 8 x86_64(RPM)	**advanced-virt-for-rhel-8-x86_64-rpms**	OpenStack Platform용 가상화 패키지를 제공합니다.
Red Hat OpenStack Platform 16.1 for RHEL 8(RPM)	**openstack-16.1-for-rhel-8-x86_64-rpms**	코어 Red Hat OpenStack Platform 리포지토리입니다.
Red Hat Fast Datapath for RHEL 8(RPMS)	**fast-datapath-for-rhel-8-x86_64-rpms**	OpenStack Platform용 OVS(Open vSwitch) 패키지를 제공합니다.
Red Hat Ceph Storage Tools 4 for RHEL 8 x86_64 (RPM)	**rhceph-4-tools-for-rhel-8-x86_64-rpms**	Red Hat Ceph Storage 4 for Red Hat Enterprise Linux 8용 툴입니다.
Red Hat Satellite Tools for RHEL 8 Server RPMs x86_64	**satellite-tools-6.5-for-rhel-8-x86_64-rpms**	Red Hat Satellite 6으로 호스트를 관리하는 툴입니다.

컴퓨팅 및 ComputeHCI 노드 리포지토리

다음 표에는 오버클라우드의 Compute 및 ComputeHCI 노드를 위한 코어 리포지토리가 나와 있습니다.

이름	리포지토리	요구 사항 설명
Red Hat Enterprise Linux 8 for x86_64 - BaseOS(RPM) EUS(Extended Update Support)	**rhel-8-for-x86_64-baseos-eus-rpms**	x86_64 시스템용 기본 운영 체제 리포지토리입니다.
Red Hat Enterprise Linux 8 for x86_64 - AppStream(RPM)	**rhel-8-for-x86_64-appstream-eus-rpms**	Red Hat OpenStack Platform 종속 패키지를 포함합니다.

이름	리포지토리	요구 사항 설명
Red Hat Enterprise Linux 8 for x86_64 - High Availability (RPMs) Extended Update Support (EUS)	**rhel-8-for-x86_64-highavailability-eus-rpms**	Red Hat Enterprise Linux용 고가용성 툴입니다.
Red Hat Ansible Engine 2.9 for RHEL 8 x86_64(RPM)	**ansible-2.9-for-rhel-8-x86_64-rpms**	Ansible Engine for Red Hat Enterprise Linux입니다. 최신 버전의 Ansible을 제공하는 데 사용됩니다.
Advanced Virtualization for RHEL 8 x86_64(RPM)	**advanced-virt-for-rhel-8-x86_64-rpms**	OpenStack Platform용 가상화 패키지를 제공합니다.
Red Hat OpenStack Platform 16.1 for RHEL 8(RPM)	**openstack-16.1-for-rhel-8-x86_64-rpms**	코어 Red Hat OpenStack Platform 리포지토리입니다.
Red Hat Fast Datapath for RHEL 8(RPMS)	**fast-datapath-for-rhel-8-x86_64-rpms**	OpenStack Platform용 OVS(Open vSwitch) 패키지를 제공합니다.
Red Hat Ceph Storage Tools 4 for RHEL 8 x86_64 (RPM)	**rhceph-4-tools-for-rhel-8-x86_64-rpms**	Red Hat Ceph Storage 4 for Red Hat Enterprise Linux 8용 툴입니다.
Red Hat Satellite Tools for RHEL 8 Server RPMs x86_64	**satellite-tools-6.5-for-rhel-8-x86_64-rpms**	Red Hat Satellite 6으로 호스트를 관리하는 툴입니다.

실시간 Compute 리포지토리

다음 표에는 RTC(Real Time Compute) 기능에 사용되는 리포지토리가 나와 있습니다.

이름	리포지토리	요구 사항 설명
Red Hat Enterprise Linux 8 for x86_64 - Real Time(RPM)	**rhel-8-for-x86_64-rt-rpms**	실시간 KVM(RT-KVM) 리포지토리로, 실시간 커널을 활성화하는 패키지가 포함되어 있습니다. RT-KVM을 대상으로 하는 모든 컴퓨팅 노드에 대해 이 리포지토리를 활성화합니다. 알림: 이 리포지토리에 액세스하려면 별도의 **Red Hat OpenStack Platform for Real Time** SKU 서브스크립션이 필요합니다.

이름	리포지토리	요구 사항 설명
Red Hat Enterprise Linux 8 for x86_64 - Real Time for NFV(RPM)	**rhel-8-for-x86_64-nfv-rpms**	NFV용 실시간 KVM(RT-KVM) 리포지토리로, 실시간 커널을 활성화하는 패키지가 포함되어 있습니다. RT-KVM을 대상으로 하는 모든 NFV 노드에 대해 이 리포지토리를 활성화합니다. 알림: 이 리포지토리에 액세스하려면 별도의 **Red Hat OpenStack Platform for Real Time** SKU 서브스크립션이 필요합니다.

Ceph Storage 노드 리포지토리

다음 표에는 오버클라우드의 Ceph Storage 관련 리포지토리가 나열되어 있습니다.

이름	리포지토리	요구 사항 설명
Red Hat Enterprise Linux 8 for x86_64 - BaseOS(RPM)	**rhel-8-for-x86_64-baseos-rpms**	x86_64 시스템용 기본 운영 체제 리포지토리입니다.
Red Hat Enterprise Linux 8 for x86_64 - AppStream(RPM)	**rhel-8-for-x86_64-appstream-rpms**	Red Hat OpenStack Platform 종속 패키지를 포함합니다.
Red Hat Enterprise Linux 8 for x86_64 - High Availability (RPMs) Extended Update Support (EUS)	**rhel-8-for-x86_64-highavailability-eus-rpms**	Red Hat Enterprise Linux용 고가용성 툴입니다. 알림: Ceph Storage 역할에 **overcloud-full** 이미지를 사용한 경우 이 리포지토리를 활성화해야 합니다. Ceph Storage 역할은 이 리포지토리가 필요하지 않은 **overcloud-minimal** 이미지를 사용해야 합니다.
Red Hat Ansible Engine 2.9 for RHEL 8 x86_64(RPM)	**ansible-2.9-for-rhel-8-x86_64-rpms**	Ansible Engine for Red Hat Enterprise Linux입니다. 최신 버전의 Ansible을 제공하는 데 사용됩니다.
Red Hat OpenStack Platform 16.1 Director Deployment Tools for RHEL 8 x86_64 (RPM)	**openstack-16.1-deployment-tools-for-rhel-8-x86_64-rpms**	director에서 Ceph Storage 노드를 설정하는 데 사용되는 패키지입니다. 이 리포지토리는 독립 실행형 Ceph Storage 서브스크립션에 포함되어 있습니다. 결합된 OpenStack Platform 및 Ceph Storage 서브스크립션을 사용하는 경우 **openstack-16.1-for-rhel-8-x86_64-rpms** 리포지토리를 사용합니다.

이름	리포지토리	요구 사항 설명
Red Hat OpenStack Platform 16.1 for RHEL 8(RPM)	**openstack-16.1-for-rhel-8-x86_64-rpms**	director에서 Ceph Storage 노드를 설정하는 데 사용되는 패키지입니다. 이 리포지토리는 OpenStack Platform 및 Ceph Storage 서브스크립션에 함께 포함되어 있습니다. 독립 실행형 Ceph Storage 서브스크립션을 사용하는 경우 **openstack-16.1-deployment-tools-for-rhel-8-x86_64-rpms** 리포지토리를 사용합니다.
Red Hat Ceph Storage Tools 4 for RHEL 8 x86_64 (RPM)	**rhceph-4-tools-for-rhel-8-x86_64-rpms**	노드가 Ceph Storage 클러스터와 통신할 수 있는 툴을 제공합니다.
Red Hat Fast Datapath for RHEL 8(RPMS)	**fast-datapath-for-rhel-8-x86_64-rpms**	OpenStack Platform용 OVS(Open vSwitch) 패키지를 제공합니다. Ceph Storage 노드에서 OVS를 사용하는 경우 이 리포지토리를 NIC(네트워크 인터페이스 구성) 템플릿에 추가합니다.

IBM POWER 리포지토리

다음 표에는 POWER PC 아키텍처의 RHOSP용 리포지토리가 나와 있습니다. 코어 리포지토리의 리포지토리 대신 이 리포지토리를 사용하십시오.

이름	리포지토리	요구 사항 설명
Red Hat Enterprise Linux for IBM Power, little endian - BaseOS(RPM)	**rhel-8-for-ppc64le-baseos-rpms**	ppc64le 시스템용 기본 운영 체제 리포지토리입니다.
Red Hat Enterprise Linux 8 for IBM Power, little endian - AppStream(RPM)	**rhel-8-for-ppc64le-appstream-rpms**	Red Hat OpenStack Platform 종속 패키지를 포함합니다.
Red Hat Enterprise Linux 8 for IBM Power, little endian - High Availability(RPM)	**rhel-8-for-ppc64le-highavailability-rpms**	Red Hat Enterprise Linux용 고가용성 툴입니다. 컨트롤러 노드 고가용성에 사용됩니다.
Red Hat Fast Datapath for RHEL 8 IBM Power, little endian(RPMS)	**fast-datapath-for-rhel-8-ppc64le-rpms**	OpenStack Platform용 OVS(Open vSwitch) 패키지를 제공합니다.

이름	리포지토리	요구 사항 설명
Red Hat Ansible Engine 2.8 for RHEL 8 IBM Power, little endian(RPM)	**ansible-2.8-for-rhel-8-ppc64le-rpms**	Ansible Engine for Red Hat Enterprise Linux입니다. 최신 버전의 Ansible을 제공하는 데 사용됩니다.
Red Hat OpenStack Platform 16.1 for RHEL 8(RPM)	**openstack-16.1-for-rhel-8-ppc64le-rpms**	ppc64le 시스템용 Red Hat OpenStack Platform 코어 리포지토리입니다.

3.3. RED HAT SATELLITE SERVER 6 고려 사항

Red Hat Satellite Server 6을 사용하여 Red Hat OpenStack Platform 환경의 RPM 및 컨테이너 이미지를 호스팅하는 경우 Satellite Server 6을 사용하여 Red Hat OpenStack Platform 16.1 업그레이드 중에 콘텐츠를 제공할 때 특정 고려 사항을 고려해야 합니다.

현재 환경에 대한 가정

- Satellite Server는 이미 Red Hat OpenStack Platform 13 RPM 및 컨테이너 이미지를 호스팅합니다.
- Red Hat OpenStack Platform 13 환경의 모든 노드를 Satellite Server에 이미 등록했습니다. 예를 들어 이전에 Red Hat OpenStack Platform 13 콘텐츠에 연결된 활성화 키를 사용하여 노드를 OpenStack Platform 13 콘텐츠에 등록했습니다.

Red Hat OpenStack Platform 업그레이드 권장 사항

- Red Hat OpenStack Platform 13 언더클라우드와 오버클라우드 모두에 필요한 RPM 리포지토리를 활성화하고 동기화합니다. 여기에는 필수 Red Hat Enterprise Linux 8.2 리포지토리가 포함됩니다.
- Satellite Server에서 사용자 지정 제품을 생성하여 다음 Red Hat OpenStack Platform 버전에 대한 컨테이너 이미지를 호스팅합니다.
  - Red Hat OpenStack Platform 16.1
  - Red Hat OpenStack Platform 15
- Red Hat OpenStack Platform 16.1 업그레이드에 대한 콘텐츠 뷰를 생성 및 승격하고 콘텐츠 뷰에 다음 콘텐츠를 포함합니다.
  - 다음 Red Hat Enterprise Linux 7 리포지토리:
    - Red Hat Enterprise Linux 7 Server RPMs x86_64 7Server 또는 Red Hat Enterprise Linux 7 Server RPMs x86_64 7.9

      ```
      rhel-7-server-rpms
      x86_64 7Server
      or:
      rhel-7-server-rpms
      x86_64 7.9
      ```

- Red Hat Enterprise Linux 7 Server - Extras RPMs x86_64

  ```
  rhel-7-server-extras-rpms
  x86_64
  ```

  - Red Hat Enterprise Linux 8.2 리포지토리를 포함한 모든 언더클라우드 및 오버클라우드 RPM 리포지토리 올바른 버전의 Red Hat Enterprise Linux 리포지토리가 8.2인지 확인하십시오. 올바른 버전을 포함하지 않는 경우 RHEL 8 리포지토리 활성화 관련 문제가 발생할 수 있습니다. 자세한 내용은 Red Hat Satellite 사용 시 RHEL 7에서 RHEL 8 LEAPP Upgrade Failing으로 Red Hat Knowledgebase 솔루션에서 참조하십시오.

  - Red Hat OpenStack Platform 16.1 컨테이너 이미지.

  - Red Hat OpenStack Platform 15 컨테이너 이미지.

- 활성화 키를 Red Hat OpenStack Platform 16.1 업그레이드를 위해 생성한 Red Hat OpenStack Platform 16.1 콘텐츠 뷰에 연결합니다.

- 노드에 **katello-host-tools-fact-plugin** 패키지가 설치되지 않았는지 확인합니다. Leapp 업그레이드는 이 패키지를 업그레이드하지 않고 이 패키지를 Red Hat Enterprise Linux 8.2 시스템에 남겨두어 **subscription-manager** 가 오류를 보고합니다.

- Red Hat OpenStack Platform 16.1 컨테이너 이미지를 호스팅하도록 Satellite Server를 구성할 수 있습니다. 자세한 내용은 *Director 설치 및 사용* 에서 컨테이너 이미지의 Satellite 서버 준비를 참조하십시오.

- Ceph 서브스크립션을 사용하고 Ceph 스토리지 노드에 **overcloud-minimal** 이미지를 사용하도록 director가 구성된 경우 Content View를 생성하고 다음 RHEL(Red Hat Enterprise Linux) 8.2 리포지토리를 추가해야 합니다.

  - Red Hat Enterprise Linux 8 for x86_64 - AppStream(RPM)

    ```
    rhel-8-for-x86_64-appstream-rpms
    x86_64 8.2
    ```

  - Red Hat Enterprise Linux 8 for x86_64 - BaseOS(RPM)

    ```
    rhel-8-for-x86_64-baseos-rpms
    x86_64 8.2
    ```

    자세한 내용은 Red Hat *Satellite Content* Management Guide의 Red Hat Content and Managing Content Views 를 참조하십시오.

4장. 언더클라우드 업그레이드 준비

언더클라우드 업그레이드를 수행하기 전에 언더클라우드 업그레이드가 성공적으로 실행되도록 몇 가지 준비 단계를 완료해야 합니다.

4.1. 외부 CEPH 사전 요구 사항으로 업그레이드

Red Hat OpenStack Platform 배포를 업그레이드하기 전에 외부 Ceph 배포를 사용하여 업그레이드하는 경우 Red Hat Ceph Storage 클러스터를 버전 3에서 버전 4로 업그레이드해야 합니다. 자세한 내용은 Red Hat Ceph Storage *4 설치 가이드의 Red Hat Ceph Storage 클러스터* 업그레이드를 참조하십시오.

4.2. 새 메모리 요구 사항

Red Hat OpenStack Platform 16.1에서 언더클라우드에는 새로운 메모리 요구 사항이 있습니다.

Red Hat OpenStack Platform 13	Red Hat OpenStack Platform 16.1
16GB RAM	24GB RAM

업그레이드를 진행하기 전에 언더클라우드가 이러한 새 요구 사항을 충족하는지 확인하십시오.

4.3. 언더클라우드 노드에 예측 가능한 NIC 이름 사용

언더클라우드 노드에서 Leapp 업그레이드를 실행하기 전에 일반적으로 **eth** 접두사가 포함된 커널 기반 NIC 이름을 확인해야 합니다. 일반적으로 이러한 NIC 이름은 NIC 할당 측면에서 예측할 수 없습니다.

playbook **-nics.yaml 플레이북을** 실행하여 **em** NIC 접두사를 사용하도록 NIC 이름 이름을 변경할 수 있습니다. 플레이북을 실행할 때 접두사 변수를 수정하여 다른 NIC **접두사를** 설정할 수도 있습니다. 그러나 NIC 변경은 Leapp 업그레이드 프로세스가 완료되고 노드가 재부팅된 후에만 적용됩니다.

절차

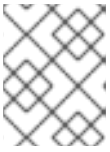
1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. **playbook-nics.yaml** 이라는 Ansible 플레이북을 생성하고 다음 콘텐츠를 플레이북에 복사합니다.

```
---
- name: Rename eth devices
  hosts: all
  become: yes
  vars:
    prefix: "em"
    undercloud_conf: "/home/stack/undercloud.conf"
    osnet_conf: "/etc/os-net-config/config.json"
  tasks:
    - set_fact:
        eth_interfaces: "{{ ansible_interfaces | select('match','eth.*') | list }}"
    - debug:
        msg: "{{ eth_interfaces }}"
    - name: Update udev rules
      lineinfile:
```

```
        line: "SUBSYSTEM==\"net\", ACTION==\"add\", DRIVERS==\"?*\", ATTR{address}==\"
{{ ansible_facts[item]['perm_macaddress'] | default(ansible_facts[item]['macaddress']) }}\",
NAME=\"{{ item|replace('eth',prefix) }}\""
        path: /etc/udev/rules.d/70-rhosp-persistent-net.rules
        create: True
      with_items: "{{ eth_interfaces }}"
    - name: Rename eth files
      block:
        - name: Check that eth files exists
          stat:
            path: /etc/sysconfig/network-scripts/ifcfg-{{ item }}
          register: nic_result
          with_items: "{{ eth_interfaces }}"
        - name: Copy nic files using the new prefix
          copy:
            remote_src: True
            src: "{{ item.stat.path }}"
            dest: "{{ item.stat.path|replace('eth',prefix) }}"
          with_items: "{{ nic_result.results }}"
          when: item.stat.exists
        - name: Edit NAME in new network-script files
          lineinfile:
            regexp: "^NAME=.*"
            line: "NAME={{ item.item|replace('eth',prefix) }}"
            path: "{{ item.stat.path|replace('eth',prefix) }}"
          with_items: "{{ nic_result.results }}"
          when: item.stat.exists
        - name: Edit DEVICE in new network-script files
          lineinfile:
            regexp: "^DEVICE=.*"
            line: "DEVICE={{ item.item|replace('eth',prefix) }}"
            path: "{{ item.stat.path|replace('eth',prefix) }}"
          with_items: "{{ nic_result.results }}"
          when: item.stat.exists
        - name: Backup old eth network-script files
          copy:
            remote_src: True
            src: "{{ item.stat.path }}"
            dest: "{{ item.stat.path }}.bak"
          with_items: "{{ nic_result.results }}"
          when: item.stat.exists
        - name: Remove old eth network-script files
          file:
            path: "{{ item.stat.path }}"
            state: absent
          with_items: "{{ nic_result.results }}"
          when: item.stat.exists
    - name: Rename route files
      block:
        - name: Check that route files exists
          stat:
            path: /etc/sysconfig/network-scripts/route-{{ item }}
          register: route_result
          with_items: "{{ eth_interfaces }}"
        - name: Copy route files using the new prefix
          copy:
```

```
        remote_src: True
        src: "{{ item.stat.path }}"
        dest: "{{ item.stat.path|replace('eth',prefix) }}"
      with_items: "{{ route_result.results }}"
      when: item.stat.exists
    - name: Update prefix in route files that use IP command arguments format
      replace:
        regexp: "eth"
        replace: "{{ prefix }}"
        path: "{{ item.stat.path|replace('eth',prefix) }}"
      with_items: "{{ route_result.results }}"
      when: item.stat.exists
    - name: Backup old route files
      copy:
        remote_src: True
        src: "{{ item.stat.path }}"
        dest: "{{ item.stat.path }}.bak"
      with_items: "{{ route_result.results }}"
      when: item.stat.exists
    - name: Remove old route files
      file:
        path: "{{ item.stat.path }}"
        state: absent
      with_items: "{{ route_result.results }}"
      when: item.stat.exists
- name: Perform a final regex for any remaining eth prefixes in ifcfg files
  block:
    - name: Get a list of all ifcfg files
      find:
        paths: /etc/sysconfig/network-scripts/
        patterns: 'ifcfg-*'
        excludes: '*.bak'
      register: ifcfg_files
    - name: Perform final regex on ifcfg files
      replace:
        path: "{{ item[0].path }}"
        regexp: "{{ item[1] }}"
        replace: "{{ item[1]|replace('eth',prefix) }}"
      with_nested:
        - "{{ ifcfg_files.files }}"
        - "{{ eth_interfaces }}"
- name: Replace interface name in files referencing old eth interface
  block:
    - name: Check if undercloud.conf exists
      stat:
        path: "{{ undercloud_conf }}"
      register: undercloud_conf_stat
    - name: Replace interface name in undercloud.conf
      replace:
        path: "{{ undercloud_conf }}"
        regexp: 'eth(\d+)'
        replace: "{{ prefix }}\\1"
      when: undercloud_conf_stat.stat.exists
    - name: Check if os-net-config's config.json exists
      stat:
        path: "{{ osnet_conf }}"
```

```
      register: osnet_conf_stat
    - name: Replace interface name in config.json
      replace:
        path: "{{ osnet_conf }}"
        regexp: 'eth(\d+)'
        replace: "{{ prefix }}\\1"
      when: osnet_conf_stat.stat.exists
  - name: Patch vlan devices
    block:
    - name: Check that vlan files exists
      stat:
        path: /etc/sysconfig/network-scripts/ifcfg-{{ item }}
      register: nic_result
      when: item.startswith("vlan")
      with_items: "{{ ansible_interfaces }}"
    - name: Backup old vlan network-script files
      copy:
        remote_src: True
        src: "{{ item.stat.path }}"
        dest: "{{ item.stat.path }}.bak"
      when: item.item.startswith("vlan") and item.stat.exists
      with_items: "{{ nic_result.results }}"
    - name: Edit PHYSDEV in new network-script files
      replace:
        path: "{{ item.stat.path }}"
        regexp: "^PHYSDEV=eth"
        replace: "PHYSDEV={{ prefix }}"
      when: item.item.startswith("vlan") and item.stat.exists
      with_items: "{{ nic_result.results }}"
```



참고

이 플레이북을 사용하여 업그레이드 프로세스의 이후 단계에서 오버클라우드 NIC의 이름을 변경합니다.

3. 언더클라우드에서 **playbook-nics.yaml** 플레이북을 실행합니다.

```
$ ansible-playbook -c local -i localhost, playbook-nics.yaml
```

플레이북은 새 NIC 접두사를 **em** 로 설정합니다. 다른 NIC 접두사를 설정하려면 플레이북을 실행할 때 **접두사** 변수를 설정합니다.

```
$ ansible-playbook -c local -i localhost, -e prefix="mynic" ~/playbook-nics.yaml
```

NIC 변경 사항은 Leapp 업그레이드 프로세스가 완료되고 노드가 재부팅된 후에만 적용됩니다.

Resources

- RHEL 7에서 커널 NIC 이름을 사용할 때 RHEL 8로 인플레이스 업그레이드를 수행하는 방법
- 일관된 네트워크 장치 이름 지정
- 예측 가능한 네트워크 인터페이스 장치 이름 이해

4.4. 언더클라우드에서 SSH 루트 권한 매개 변수 설정

Leapp 업그레이드는 **PermitRootLogin** 매개변수가 **/etc/ssh/sshd_config** 파일에 있는지 확인합니다. 이 매개변수를 **yes** 또는 **no** 로 명시적으로 설정해야 합니다.

보안을 위해 이 매개변수를 **no** 로 설정하여 언더클라우드의 root 사용자에 대한 SSH 액세스를 비활성화 합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. **PermitRootLogin** 매개변수가 있는지 **/etc/ssh/sshd_config** 파일을 확인합니다.

```
$ sudo grep PermitRootLogin /etc/ssh/sshd_config
```

3. 매개변수가 **/etc/ssh/sshd_config** 파일에 없는 경우 파일을 편집하고 **PermitRootLogin** 매개변수를 설정합니다.

```
PermitRootLogin no
```

4. 파일을 저장합니다.

4.5. 차세대 전원 관리 드라이버로 변환

Red Hat OpenStack Platform은 이제 기존 드라이버를 대체하는 *하드웨어 유형* 이라고도 하는 차세대 드라이버를 사용합니다.

다음 표는 이전 드라이버와 차세대 하드웨어 유형과 유사한 비교를 보여줍니다.

이전 드라이버	새 하드웨어 유형
pxe_ipmitool	**ipmi**
pxe_drac	**idrac**
pxe_ilo	**ilo**
pxe_irmc	**irmc**
VBMC(**pxe_ipmitool**)	**ipmi**
fake_pxe	**fake-hardware**

OpenStack Platform 15에서는 이러한 이전 드라이버가 제거되어 더 이상 액세스할 수 없습니다. OpenStack Platform 15로 업그레이드하기 **전에** 하드웨어 유형으로 변경해야 합니다.

절차

1. 현재 하드웨어 유형의 목록을 확인합니다.

```
$ source ~/stackrc
$ openstack baremetal driver list --type dynamic
```

2. 활성화되지 않은 하드웨어 유형 드라이버를 사용하는 경우 **undercloud.conf 파일에서 enabled_hardware_types 매개변수를 사용하여 드라이버를 활성화합니다.**

```
enabled_hardware_types = ipmi,redfish,idrac
```

3. 파일을 저장하고 언더클라우드를 새로 고칩니다.

```
$ openstack undercloud install
```

4. 전원 관리 유형에 대해 **OLDDRIVER 및 NEWDRIVER** 변수를 대체하여 다음 명령을 실행합니다.

```
$ source ~/stackrc
$ OLDDRIVER="pxe_ipmitool"
$ NEWDRIVER="ipmi"
$ for NODE in $(openstack baremetal node list --driver $OLDDRIVER -c UUID -f value) ; do
openstack baremetal node set $NODE --driver $NEWDRIVER; done
```

5장. 언더클라우드 운영 체제 업그레이드

director를 업그레이드하기 전에 언더클라우드 운영 체제를 Red Hat Enterprise Linux 7에서 Red Hat Enterprise Linux 8로 업그레이드해야 합니다. 이 운영 체제 업그레이드의 일환으로 Red Hat OpenStack Platform 13 패키지를 제거한 다음 Leapp 유틸리티를 실행하여 시스템 패키지를 업그레이드해야 합니다. 이 패키지 제거 및 운영 체제 업그레이드는 언더클라우드 데이터베이스에 영향을 미치지 않습니다. 운영 체제 업그레이드를 완료한 후 Red Hat OpenStack Platform 16.1 버전의 director 패키지를 다시 설치합니다.

5.1. RED HAT OPENSTACK PLATFORM DIRECTOR 패키지 제거

Leapp 유틸리티를 실행하기 전에 Red Hat Enterprise Linux 7에 연결된 Red Hat OpenStack Platform 13 패키지를 제거하십시오. 이러한 패키지 이름은 **el7ost** 릴리스 접미사를 사용합니다. 일부 **el7ost** 는 **subscription-manager** 및 Leapp 유틸리티에 대한 종속성으로 시스템에 남아 있습니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. 언더클라우드에서 기본 OpenStack 서비스를 비활성화합니다.

```
$ sudo systemctl stop 'openstack-*' httpd haproxy mariadb 'rabbitmq*' docker xinetd
```

3. OpenvSwitch 및 업그레이드에 필요한 특정 Python 2 패키지를 제외하고 언더클라우드에서 기본 OpenStack 서비스를 제거합니다.

```
$ sudo yum -y remove '*el7ost*' 'galera*' 'haproxy*' \
    httpd 'mysql*' 'pacemaker*' xinetd python-jsonpointer \
    qemu-kvm-common-rhev qemu-img-rhev 'rabbit*' \
    'redis*' \
    -- \
    -'*openvswitch*' -python-docker -python-PyMySQL \
    -python-pysocks -python2-asn1crypto -python2-babel \
    -python2-cffi -python2-cryptography -python2-dateutil \
    -python2-idna -python2-ipaddress -python2-jinja2 \
    -python2-jsonpatch -python2-markupsafe -python2-pyOpenSSL \
    -python2-requests -python2-six -python2-urllib3 \
    -python-httplib2 -python-passlib -python2-netaddr -ceph-ansible
```

4. **/etc/httpd 및 /var/ lib/docker** 디렉토리에서 콘텐츠를 제거합니다.

```
$ sudo rm -rf /etc/httpd /var/lib/docker
```

5.2. 언더클라우드에서 LEAPP 업그레이드 수행

Leapp 유틸리티를 설치하고 실행하여 운영 체제를 RHEL (Red Hat Enterprise Linux) 8로 업그레이드합니다.

사전 요구 사항

- Leapp을 설치하고 실행하기 전에 2.4절. “Red Hat OpenStack Platform의 LeApp 업그레이드 사용” 섹션을 숙지하십시오.

- Leapp 업그레이드를 수행하기 전에 4.3절. "언더클라우드 노드에 예측 가능한 NIC 이름 사용" 섹션을 완료해야 합니다. Leapp 업그레이드 프로세스를 수행하기 전에 네트워크 인터페이스 이름 이름을 변경하지 않으면 RHEL 8.2로 업그레이드한 후 인터페이스 이름이 변경될 수 있습니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. Leapp 유틸리티 및 jq를 설치합니다.

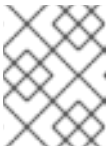
```
$ sudo yum install leapp
$ sudo yum install jq
```

3. Leapp 유틸리티에 필요한 추가 데이터 파일(RPM 패키지 변경 및 RPM 리포지토리 매핑)을 다운로드하여 RHEL 7에서 RHEL 8로의 인플레이스 유틸리티에 필요한 추가 데이터 파일(RPM 패키지 변경 및 RPM 리포지토리 매핑)을 다운로드하여 이러한 파일을 **/etc/leapp/files/** 디렉터리에 저장합니다.
4. Red Hat 서브스크립션을 업데이트합니다.
   - 언더클라우드에서 등록을 위해 Red Hat 고객 포털을 사용하는 경우 현재 서브스크립션을 업데이트하여 Red Hat Enterprise Linux 8.2 콘텐츠에 액세스할 수 있습니다.

```
$ sudo subscription-manager refresh
```

   - 언더클라우드에서 등록에 Red Hat Satellite Server를 사용하는 경우 RHOSP(Red Hat OpenStack Platform)16.1 활성화 키와 관련된 콘텐츠 뷰에 언더클라우드를 다시 등록합니다.

```
$ sudo subscription-manager register --force --org ORG --activationkey
ACTIVATION_KEY
```



참고

Red Hat OpenStack Platform 16.1에 대해 생성한 콘텐츠 뷰에는 Red Hat Enterprise Linux 8.2의 콘텐츠가 포함되어야 합니다.

5. Red Hat OpenStack Platform 16.1은 최신 버전의 Open vSwitch를 사용합니다. Open vSwitch 버전을 **to_remove 및 to_ install** 트랜잭션 파일을 통해 대체합니다.

```
$ echo 'openvswitch2.11' | sudo tee -a /etc/leapp/transaction/to_remove
$ echo 'openvswitch2.13' | sudo tee -a /etc/leapp/transaction/to_install
```

6. **to_keep** 트랜잭션 파일을 사용하여 업그레이드를 통해 Red Hat Ceph Storage 3 버전의 **ceph-anible** 을 유지합니다.

```
$ echo 'ceph-ansible' | sudo tee -a /etc/leapp/transaction/to_keep
```

7. RHEL 8에서 더 이상 지원되지 않는 커널 모듈을 조정합니다.

```
$ if [ -f /usr/share/leapp-
repository/repositories/system_upgrade/el7toel8/actors/kernel/checkkerneldrivers/files/removed
_drivers.txt ]; then
    for module in pata_acpi floppy; do
```

```
        sudo sed -i "/^${module}$/d" /usr/share/leapp-
repository/repositories/system_upgrade/el7toel8/actors/kernel/checkkerneldrivers/files/removed
_drivers.txt
    done
else
    for module in pata_acpi floppy; do
        jq ". | del(.data[] | select(.driver_name == \"${module}\"))"
/etc/leapp/files/device_driver_deprecation_data.json | sudo tee
/etc/leapp/files/device_driver_deprecation_data.json_modified
        mv /etc/leapp/files/device_driver_deprecation_data.json_modified
/etc/leapp/files/device_driver_deprecation_data.json
    done
fi
```

8. **leapp 응답** 명령을 실행하고 Leapp 응답을 지정하여 **pam_pkcs11** 모듈을 제거합니다.

```
$ sudo leapp answer --add --section remove_pam_pkcs11_module_check.confirm=True
```

9. 선택 사항: TLS-Everywhere 아키텍처와 함께 환경을 배포하고 더 이상 사용되지 않는 **authconfig** 유틸리티를 사용하여 시스템에서 인증을 구성하는 경우 **authselect** 유틸리티를 사용하여 RHEL 8 시스템을 구성합니다.

```
$ sudo leapp answer --add --section authselect_check.confirm=True
```

Leapp 업그레이드 프로세스 중에 인증 구성에 대한 자세한 내용은 *RHEL 7에서 RHEL 8로의 업그레이드에서* 알려진 문제를 참조하십시오.

10. **LEAPP_DEVEL_TARGET_RELEASE** 및 **LEAPP_UNSUPPORTED** 환경 변수를 설정하여 업그레이드할 RHEL 8 부 버전을 지정합니다. RHOSP 16.1의 경우 RHEL 8 부 버전을 **8.2** 로 설정해야 합니다.

```
$ export LEAPP_UNSUPPORTED=1
$ export LEAPP_DEVEL_TARGET_RELEASE=8.2
```

LEAPP_ DEVEL 접두사가 있는 환경 변수를 사용할 때마다 LEAPP_ UNSUPPORTED 환경 변수를 사용해야 합니다.

11. Leapp 프로세스에서 영구적인 네트워크 이름 행위자를 제거합니다.

참고

Leapp 업그레이드 프로세스를 수행하기 전에 네트워크 인터페이스 이름 이름을 변경하지 않으면 RHEL 8.2로 업그레이드한 후 인터페이스 이름이 변경될 수 있습니다. 네트워크 인터페이스 이름 이름을 바꾸는 방법에 대한 자세한 내용은 4.3절. "언더클라우드 노드에 예측 가능한 NIC 이름 사용" 을 참조하십시오.

```
$ sudo rm -f /usr/share/leapp-
repository/repositories/system_upgrade/el7toel8/actors/persistentnetnamesdisable/actor.py
```

12. Leapp 업그레이드 프로세스를 시작합니다.

```
$ sudo -E leapp upgrade --debug --enablerepo rhel-8-for-x86_64-baseos-eus-rpms --
enablerepo rhel-8-for-x86_64-appstream-eus-rpms --enablerepo fast-datapath-for-rhel-8-
x86_64-rpms --enablerepo ansible-2.9-for-rhel-8-x86_64-rpms
```

enable **repo** 옵션을 사용하여 Leapp 업그레이드 프로세스 중에 활성화하려는 리포지토리를 설정합니다. 특히 최신 버전의 Open vSwitch에서 Red Hat OpenStack Platform 16.1 전환을 용이하게 하려면 이러한 리포지토리를 포함해야 합니다.

13. **leapp upgrade** 명령이 성공적으로 완료될 때까지 기다립니다.

14. root 디렉토리에 empty **.autorelabel** 파일을 만듭니다.

```
$ sudo touch /.autorelabel
```

재부팅 후 SELinux는 이 파일을 감지하고 파일 시스템의 레이블을 자동으로 다시 지정합니다.

15. 언더클라우드를 재부팅합니다.

```
$ sudo reboot
```

16. DNF 구성에 정의된 트랜잭션 제외에서 Leapp 패키지를 제거합니다.

```
$ sudo dnf config-manager --save --setopt exclude=''
```

추가 리소스

- RHEL 7에서 RHEL 8으로 인플레이스 업그레이드에 필요한 Leapp 유틸리티에 필요한 데이터
- RHEL 7에서 커널 NIC 이름을 사용할 때 RHEL 8로 인플레이스 업그레이드를 수행하는 방법
- RHEL 7에서 RHEL 8로의 업그레이드 수행

6장. DIRECTOR 업그레이드

언더클라우드 운영 체제 업그레이드를 완료한 후 director를 업그레이드합니다. 이전 Red Hat OpenStack Platform 13 언더클라우드의 데이터베이스는 운영 체제 업그레이드 후에도 호스트에서 유지됩니다. 새로운 Red Hat OpenStack Platform 16.1 패키지를 설치하고 **openstack undercloud upgrade** 명령을 실행하기 전에 Red Hat OpenStack Platform 16.1 컨테이너 이미지의 새 소스를 구성합니다.

6.1. RED HAT ENTERPRISE LINUX 릴리스에 대한 환경 잠금

Red Hat OpenStack Platform 16.1은 Red Hat Enterprise Linux 8.2에서 지원됩니다. 업데이트를 수행하기 전에 운영 체제를 최신 마이너 릴리스로 업그레이드하지 않으려면 언더클라우드 리포지토리를 Red Hat Enterprise Linux 8.2 릴리스에 고정해야 합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. **subscription-manager release** 명령을 사용하여 언더클라우드를 특정 버전에 잠급니다.

```
$ sudo subscription-manager release --set=8.2
```

6.2. 언더클라우드용 리포지토리 활성화

언더클라우드에 필요한 리포지토리를 활성화하고 시스템 패키지를 최신 버전으로 업데이트합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. 기본 리포지토리를 모두 비활성화하고 필수 Red Hat Enterprise Linux 리포지토리를 활성화합니다.

```
[stack@director ~]$ sudo subscription-manager repos --disable=*
[stack@director ~]$ sudo subscription-manager repos --enable=rhel-8-for-x86_64-baseos-eus-rpms --enable=rhel-8-for-x86_64-appstream-eus-rpms --enable=rhel-8-for-x86_64-highavailability-eus-rpms --enable=ansible-2.9-for-rhel-8-x86_64-rpms --enable=openstack-16.1-for-rhel-8-x86_64-rpms --enable=fast-datapath-for-rhel-8-x86_64-rpms --enable=advanced-virt-for-rhel-8-x86_64-rpms
```

   이러한 리포지토리에는 director 설치에 필요한 패키지가 들어 있습니다.

3. **container-tools** 리포지터리 모듈을 버전 **2.0**으로 설정합니다.

```
[stack@director ~]$ sudo dnf module reset container-tools
[stack@director ~]$ sudo dnf module enable -y container-tools:2.0
```

4. 운영 체제를 동기화하여 시스템 패키지가 운영 체제 버전과 일치하는지 확인합니다.

```
[stack@director ~]$ sudo dnf distro-sync -y
[stack@director ~]$ sudo reboot
```

6.3. DIRECTOR 패키지 설치

Red Hat OpenStack Platform director와 관련된 패키지를 설치합니다.

절차

1. director 설치 및 설정에 필요한 명령행 툴을 설치합니다.

```
[stack@director ~]$ sudo dnf install -y python3-tripleoclient
```

6.4. 컨테이너 이미지 준비

언더클라우드 설치에는 컨테이너 이미지를 가져올 위치와 저장 방법을 결정하는 환경 파일이 필요합니다. 컨테이너 이미지를 준비하는 데 사용할 수 있는 이 환경 파일을 생성하고 사용자 지정하십시오.

참고

언더클라우드의 특정 컨테이너 이미지 버전을 구성해야 하는 경우 이미지를 특정 버전에 고정해야 합니다. 자세한 내용은 언더클라우드의 컨테이너 이미지 고정 을 참조하십시오.

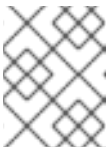
절차

1. **stack** 사용자로 언더클라우드 호스트에 로그인합니다.
2. 기본 컨테이너 이미지 준비 파일을 생성합니다.

```
$ sudo openstack tripleo container image prepare default \
  --local-push-destination \
  --output-env-file containers-prepare-parameter.yaml
```

이 명령은 다음과 같은 추가 옵션을 사용합니다.

- **--local-push-destination**은 언더클라우드의 레지스트리를 컨테이너 이미지의 위치로 설정합니다. 즉 director가 Red Hat Container Catalog에서 필요한 이미지를 가져와서 언더클라우드의 레지스트리로 푸시합니다. director는 이 레지스트리를 컨테이너 이미지 소스로 사용합니다. Red Hat Container Catalog에서 직접 가져오려면 이 옵션을 생략합니다.
- **--output-env-file**은 환경 파일 이름입니다. 이 파일 내용에는 컨테이너 이미지를 준비하는 데 필요한 매개변수가 포함되어 있습니다. 이 경우 파일 이름은 **containers-prepare-parameter.yaml**입니다.



참고

동일한 **containers-prepare-parameter.yaml** 파일을 사용하여 언더클라우드와 오버클라우드의 컨테이너 이미지 소스를 모두 정의할 수 있습니다.

3. **containers-prepare-parameter.yaml**을 요구 사항에 맞게 수정합니다.

6.5. 컨테이너 이미지 준비 매개변수

컨테이너를 준비하는 데 필요한 기본 파일(**containers-prepare-parameter.yaml**)에는 **ContainerImagePrepare** heat 매개변수가 포함되어 있습니다. 이 매개변수는 이미지 세트를 준비하기 위한 다양한 설정을 정의합니다.

```
parameter_defaults:
```

```
ContainerImagePrepare:
- (strategy one)
- (strategy two)
- (strategy three)
...
```

각각의 설정에서 하위 매개변수 세트를 통해 사용할 이미지와 해당 이미지의 사용 방법을 정의할 수 있습니다. 다음 표에는 각 **ContainerImagePrepare** 전략에 사용할 수 있는 하위 매개변수에 대한 정보가 나와 있습니다.

매개변수	설명
excludes	전략에서 이미지 이름을 제외하는 정규식 목록입니다.
includes	전략에 포함할 정규식 목록입니다. 기존 이미지와 일치하는 이미지 이름이 하나 이상 있어야 합니다. **includes**를 지정하면 **excludes**는 모두 무시됩니다.
modify_append_tag	대상 이미지 태그에 추가할 문자열입니다. 예를 들어 태그가 16.1.3-5.161인 이미지를 가져와서 **modify_append_tag** 를 **-hotfix** 로 설정하면 director는 최종 이미지에 16.1.3-5.161-hotfix로 태그를 지정합니다.
modify_only_with_labels	수정할 이미지를 필터링하는 이미지 레이블로 이루어진 사전입니다. 이미지가 정의된 레이블과 일치하면 director에서 그 이미지를 수정 프로세스에 포함합니다.
modify_role	이미지를 대상 레지스트리로 푸시하기 전 업로드 중에 실행할 ansible 역할 이름의 문자열입니다.
modify_vars	**modify_role**로 전달할 변수로 이루어진 사전입니다.

매개변수	설명
push_destination	업로드 프로세스 중 이미지를 푸시할 레지스트리의 네임스페이스를 정의합니다. ● **true**로 설정하면 **push_destination**이 해당 호스트 이름을 사용하는 언더클라우드 레지스트리 네임스페이스로 설정되며, 이것이 권장되는 방법입니다. ● **false**로 설정하면 로컬 레지스트리로 푸시되지 않고 노드가 소스에서 직접 이미지를 가져옵니다. ● 사용자 지정 값으로 설정하면 director가 이미지를 외부 로컬 레지스트리로 푸시합니다. Red Hat Container Catalog에서 직접 이미지를 가져오는 동안 프로덕션 환경에서 이 매개변수를 **false**로 설정하면 모든 오버클라우드 노드에서 외부 연결을 통해 Red Hat Container Catalog의 이미지를 동시에 가져옵니다. 이로 인해 대역폭 문제가 발생할 수 있습니다. 컨테이너 이미지를 호스팅하는 Red Hat Satellite 서버에서 이미지를 직접 가져올 때만 **false**를 사용하십시오. **push_destination** 매개변수가 정의되지 않았거나 **false**로 설정되었는데 원격 레지스트리에 인증이 필요한 경우, **ContainerImageRegistryLogin** 매개변수를 **true**로 설정하고 **ContainerImageRegistryCredentials** 매개변수가 있는 인증 정보를 포함하십시오.
pull_source	원본 컨테이너 이미지를 가져온 소스 레지스트리입니다.
set	초기 이미지를 가져올 위치를 정의하는 **key: value** 정의로 이루어진 사전입니다.
tag_from_label	지정된 컨테이너 이미지 메타데이터 레이블의 값을 사용하여 모든 이미지에 대한 태그를 생성하고 해당 태그가 지정된 이미지를 가져옵니다. 예를 들어 **tag_from_label: {version}-{release}**를 설정하면 director는 **version** 및 **release** 레이블을 사용하여 새 태그를 구성합니다. 한 컨테이너의 경우 **version** 을 16.1.3으로 설정하고 **릴리스가 5.161** 으로 설정될 수 있으므로 태그 16.1.3-5.161이 발생할 수 있습니다. director는 **set** 사전에 **tag**가 정의되지 않은 경우에만 이 매개변수를 사용합니다.

중요

이미지를 언더클라우드로 내보내는 경우 **push _destination 대신 push_destination: true 를 사용합니다. UNDERCLOUD_IP:PORT**. **push_destination: true** 방식은 IPv4 및 IPv6 주소 모두에서 일관성 수준을 제공합니다.

set 매개변수에 여러 **key: value** 정의를 사용할 수 있습니다.

키	설명
ceph_image	Ceph Storage 컨테이너 이미지의 이름입니다.
ceph_namespace	Ceph Storage 컨테이너 이미지의 네임스페이스입니다.
ceph_tag	Ceph Storage 컨테이너 이미지의 태그입니다.
ceph_alertmanager_image **ceph_alertmanager_namespace** **ceph_alertmanager_tag**	Ceph Storage Alert Manager 컨테이너 이미지의 이름, 네임스페이스 및 태그입니다.
ceph_grafana_image **ceph_grafana_namespace** **ceph_grafana_tag**	Ceph Storage Grafana 컨테이너 이미지의 이름, 네임스페이스 및 태그입니다.
ceph_node_exporter_image **ceph_node_exporter_namespace** **ceph_node_exporter_tag**	Ceph Storage 노드 내보내기 컨테이너 이미지의 이름, 네임스페이스 및 태그입니다.
ceph_prometheus_image **ceph_prometheus_namespace** **ceph_prometheus_tag**	Ceph Storage Prometheus 컨테이너 이미지의 이름, 네임스페이스 및 태그입니다.
name_prefix	각 OpenStack 서비스 이미지의 접두사입니다.
name_suffix	각 OpenStack 서비스 이미지의 접미사입니다.
namespace	각 OpenStack 서비스 이미지의 네임스페이스입니다.
neutron_driver	사용할 OpenStack Networking (Neutron) 컨테이너를 결정하는 데 사용할 드라이버입니다. null 값을 사용하여 표준 **neutron-server** 컨테이너를 설정합니다. OVN 기반 컨테이너를 사용하려면 **ovn**으로 설정합니다.

키	설명
tag	소스의 모든 이미지에 대한 특정 태그를 설정합니다. 정의되지 않은 경우 director는 Red Hat OpenStack Platform 버전 번호를 기본값으로 사용합니다. 이 매개변수가 **tag_from_label** 값보다 우선합니다.

참고

컨테이너 이미지는 Red Hat OpenStack Platform 버전을 기반으로 멀티 스트림 태그를 사용합니다. 즉, 더 이상 **latest** 태그가 없음을 의미합니다.

6.6. 컨테이너 이미지 태그 지침

Red Hat Container Registry는 특정 버전 형식을 사용하여 모든 Red Hat OpenStack Platform 컨테이너 이미지에 태그를 지정합니다. 이 형식은 **version-release**인 각 컨테이너의 레이블 메타데이터를 따릅니다.

버전

Red Hat OpenStack Platform의 주요 및 마이너 버전에 해당합니다. 이러한 버전은 하나 이상의 릴리스를 포함하는 스트림으로 작동합니다.

릴리스

버전 스트림 내 특정 컨테이너 이미지 버전의 릴리스에 해당합니다.

예를 들어 최신 버전의 Red Hat OpenStack Platform이 16.1.3이고 컨테이너 이미지의 릴리스가 **5.161** 인 경우 컨테이너 이미지의 결과 태그는 16.1.3-5.161입니다.

Red Hat Container Registry에서는 해당 컨테이너 이미지 버전의 최신 릴리스에 연결되는 메이저 및 마이너 **version** 버전 태그 세트도 사용합니다. 예를 들어 16.1 및 16.1.3 컨테이너 스트림의 최신 **릴리스에** 대한 링크입니다. 16.1의 새 마이너 릴리스가 발생하면 16.1.3 태그가 16.1.3 **스트림** 내의 최신 **릴리스에** 계속 연결됩니다.

ContainerImagePrepare 매개변수에는 다운로드할 컨테이너 이미지를 결정하는 데 사용할 두 개의 하위 매개변수가 포함되어 있습니다. 이러한 하위 매개변수는 **set** 사전 내의 **tag** 매개변수와 **tag_from_label** 매개변수입니다. 다음 지침을 사용하여 **tag** 또는 **tag_from_label**을 사용할지 여부를 결정합니다.

- **tag**의 기본값은 OpenStack Platform 버전의 주요 버전입니다. 이 버전의 경우 16.1입니다. 이는 항상 최신 마이너 버전 및 릴리스에 해당합니다.

```
parameter_defaults:
  ContainerImagePrepare:
  - set:
      ...
      tag: 16.1
      ...
```

- OpenStack Platform 컨테이너 이미지의 특정 마이너 버전으로 변경하려면 태그를 마이너 버전으로 설정합니다. 예를 들어 16.1.2로 변경하려면 **tag** 를 16.1.2로 설정합니다.

```
parameter_defaults:
  ContainerImagePrepare:
  - set:
```

```
    ...
    tag: 16.1.2
    ...
```

- **tag**를 설정하면 director는 설치 및 업데이트 중에 **tag**에 설정된 버전의 최신 컨테이너 이미지 **release**를 항상 다운로드합니다.
- **tag**를 설정하지 않으면 director는 최신 주요 버전과 함께 **tag_from_label** 값을 사용합니다.

```
parameter_defaults:
  ContainerImagePrepare:
  - set:
      ...
      # tag: 16.1
      ...
    tag_from_label: '{version}-{release}'
```

- **tag_from_label** 매개변수는 Red Hat Container Registry에서 검사하는 최신 컨테이너 이미지 릴리스의 레이블 메타데이터에서 태그를 생성합니다. 예를 들어 특정 컨테이너의 레이블에서 다음 **version** 및 **release** 메타데이터를 사용할 수 있습니다.

```
"Labels": {
  "release": "5.161",
  "version": "16.1.3",
  ...
}
```

- **tag_from_label**의 기본값은 **{version}-{release}**로, 각 컨테이너 이미지의 버전 및 릴리스 메타데이터 레이블에 해당합니다. 예를 들어 컨테이너 이미지에 **버전** 용으로 16.1.3이 설정되어 있고 **릴리스** 용으로 5.161이 설정된 경우 컨테이너 이미지의 결과 태그는 16.1.3-5.161입니다.
- **tag** 매개변수는 항상 **tag_from_label** 매개변수보다 우선합니다. **tag_from_label**을 사용하려면 컨테이너 준비 구성에서 **tag** 매개변수를 생략합니다.
- **tag**와 **tag_from_label**의 주요 차이점은 director가 **tag**를 사용하여 주요 또는 마이너 버전 태그를 기반으로만 이미지를 가져온다는 것입니다. 이 태그는 버전 스트림 내의 최신 이미지 릴리스에 대한 Red Hat Container Registry 링크인 반면 director는 **tag_from_label**을 사용하여 director가 태그를 생성하고 해당 이미지를 가져올 수 있도록 각 컨테이너 이미지의 메타데이터 검사를 수행합니다.

6.7. 개인 레지스트리에서 컨테이너 이미지 가져오기

registry.redhat.io 레지스트리에는 이미지에 액세스하여 가져오기 위한 인증이 필요합니다. **registry.redhat.io** 및 기타 개인 레지스트리로 인증하려면 **containers-prepare-parameter.yaml** 파일에 **ContainerImageRegistryCredentials** 및 **ContainerImageRegistryLogin** 매개변수를 포함합니다.

ContainerImageRegistryCredentials

일부 컨테이너 이미지 레지스트리는 이미지에 액세스하기 위해 인증이 필요합니다. 이 경우 **container-prepare-parameter.yaml** 환경 파일에서 **ContainerImageRegistryCredentials** 매개변수를 사용합니다. **ContainerImageRegistryCredentials** 매개변수는 개인 레지스트리 URL에 따라 여러 키를 사용합니다. 각 개인 레지스트리 URL은 고유한 키와 값 쌍을 사용하여 사용자 이름(키)과 암호(값)를 정의합니다. 이런 방법으로 여러 개인 레지스트리의 인증 정보를 지정할 수 있습니다.

```
parameter_defaults:
```

```
  ContainerImagePrepare:
  - push_destination: true
    set:
      namespace: registry.redhat.io/...
      ...
  ContainerImageRegistryCredentials:
    registry.redhat.io:
      my_username: my_password
```

예제에서 **my_username** 및 **my_password**를 사용자 인증 정보로 교체합니다. 개별 사용자 인증 정보를 사용하는 대신, 레지스트리 서비스 계정을 생성하고 해당 인증 정보를 사용하여 **registry.redhat.io** 콘텐츠에 액세스하는 것이 좋습니다.

여러 레지스트리에 대한 인증 세부 정보를 지정하려면 **ContainerImageRegistryCredentials**의 각 레지스트리에 대해 여러 개의 키 쌍 값을 설정합니다.

```
parameter_defaults:
  ContainerImagePrepare:
  - push_destination: true
    set:
      namespace: registry.redhat.io/...
      ...
  - push_destination: true
    set:
      namespace: registry.internalsite.com/...
      ...
  ...
  ContainerImageRegistryCredentials:
    registry.redhat.io:
      myuser: 'p@55w0rd!'
    registry.internalsite.com:
      myuser2: '0th3rp@55w0rd!'
    '192.0.2.1:8787':
      myuser3: '@n0th3rp@55w0rd!'
```

중요

기본 **ContainerImagePrepare** 매개변수는 인증이 필요한 **registry.redhat.io**에서 컨테이너 이미지를 가져옵니다.

자세한 내용은 Red Hat Container Registry Authentication 을 참조하십시오.

ContainerImageRegistryLogin

ContainerImageRegistryLogin 매개변수는 오버클라우드 노드 시스템이 컨테이너 이미지를 가져오기 위해 원격 레지스트리에 로그인해야 하는지 여부를 제어하는 데 사용합니다. 이러한 상황은 오버클라우드 노드에서 이미지를 호스팅하는 데 언더클라우드를 사용하는 대신 이미지를 직접 가져오도록 할 때 발생합니다.

지정된 설정에 대해 **push_destination**이 구성되지 않은 경우 **ContainerImageRegistryLogin**을 **true**로 설정해야 합니다.

```
parameter_defaults:
  ContainerImagePrepare:
  - push_destination: false
```

```
    set:
      namespace: registry.redhat.io/...
      ...
  ...
  ContainerImageRegistryCredentials:
    registry.redhat.io:
      myuser: 'p@55w0rd!'
  ContainerImageRegistryLogin: true
```

하지만 오버클라우드 노드에서 **ContainerImageRegistryCredentials**에 정의된 레지스트리 호스트에 네트워크로 연결할 수 없고 **ContainerImageRegistryLogin**을 **true**로 설정하는 경우, 로그인할 때 배포에 실패할 수 있습니다. 오버클라우드 노드에서 **ContainerImageRegistryCredentials**에 정의된 레지스트리 호스트에 네트워크로 연결할 수 없고 **push_destination**을 **true**로 설정하고 **ContainerImageRegistryLogin**을 **false**로 설정하여 오버클라우드 노드가 언더클라우드에서 이미지를 가져오도록 합니다.

```
parameter_defaults:
  ContainerImagePrepare:
  - push_destination: true
    set:
      namespace: registry.redhat.io/...
      ...
  ...
  ContainerImageRegistryCredentials:
    registry.redhat.io:
      myuser: 'p@55w0rd!'
  ContainerImageRegistryLogin: false
```

6.8. 업그레이드를 위해 전환 컨테이너 확보

업그레이드를 수행하려면 이전 버전의 Red Hat OpenStack Platform 및 Red Hat Ceph Storage의 컨테이너가 필요합니다. 이러한 컨테이너는 Red Hat OpenStack Platform 16.1로 전환하는 데 도움이 됩니다.

절차

1. **stack** 사용자로 언더클라우드 호스트에 로그인합니다.
2. **containers-prepare-parameter.yaml** 파일을 편집합니다.
3. **ContainerImagePrepare** 매개변수에 **설정** 할 transitional 컨테이너 매개변수를 추가합니다. 배포 유형에 따라 다음 방법 중 하나로 매개 변수를 설정합니다.
   - 배포에서 director를 사용하여 배포된 Red Hat Ceph Storage 클러스터를 사용하는 경우 다음 매개변수를 추가합니다.

     ```
     parameter_defaults:
       ContainerImagePrepare:
       - push_destination: true
         set:
           ...
           name_prefix_stein: openstack-
           name_suffix_stein: ''
           namespace_stein: registry.redhat.io/rhosp15-rhel8
           tag_stein: 15.0
           ceph3_namespace: registry.redhat.io/rhceph
     ```

```
    ceph3_tag: latest
    ceph3_image: rhceph-3-rhel7
    ...
```

- ***_automatic** 매개 변수는 업그레이드 프로세스에서 데이터베이스 마이그레이션에 사용하는 Red Hat OpenStack Platform 15의 컨테이너 이미지를 정의합니다.
- **ceph3_*** 매개 변수는 오버클라우드에서 사용하는 현재 Red Hat Ceph Storage 컨테이너 이미지를 정의합니다. 오버클라우드에는 Red Hat Ceph Storage 3에서 4로 전환하려면 **ceph3_* 및 ceph_*** 매개 변수가 필요합니다.
- 컨테이너 이미지 스토리지에 Red Hat Satellite Server를 사용하는 경우 네임스페이스를 Red Hat Satellite Server의 이미지 위치로 설정합니다.

- 배포에서 외부 Ceph Storage 클러스터를 사용하는 경우 다음 매개변수를 추가합니다.

```
parameter_defaults:
  ContainerImagePrepare:
  - push_destination: true
    set:
      ...
      name_prefix_stein: openstack-
      name_suffix_stein: ''
      namespace_stein: registry.redhat.io/rhosp15-rhel8
      tag_stein: 15.0
      ceph_namespace: registry.redhat.io/rhceph
      ceph_tag: latest
      ceph_image: rhceph-4-rhel8
      ...
```

- ***_automatic** 매개 변수는 업그레이드 프로세스에서 데이터베이스 마이그레이션에 사용하는 Red Hat OpenStack Platform 15의 컨테이너 이미지를 정의합니다.
- **ceph_*** 매개 변수는 오버클라우드에서 사용하는 현재 Red Hat Ceph Storage 4 컨테이너 이미지를 정의합니다.
- 컨테이너 이미지 스토리지에 Red Hat Satellite Server를 사용하는 경우 네임스페이스를 Red Hat Satellite Server의 이미지 위치로 설정합니다.

4. **neutron_driver** 매개 변수를 **openvswitch** 로 변경합니다.

```
parameter_defaults:
  ContainerImagePrepare:
  - push_destination: true
    set:
      ...
      neutron_driver: openvswitch
      ...
```

업그레이드는 프로세스 전체에서 Open vSwitch 호환성을 유지합니다. Red Hat OpenStack Platform 16.1로 업그레이드를 완료한 후 오버클라우드를 Open vSwitch에서 OVN(Open Virtual Network)으로 마이그레이션합니다.

5. **containers-prepare-parameter.yaml** 파일을 저장합니다.

6.9. UNDERCLOUD.CONF 파일 업데이트

Red Hat OpenStack Platform 13 환경에서 원본 **undercloud.conf** 파일을 계속 사용할 수 있지만 Red Hat OpenStack Platform 16.1과의 호환성을 유지하려면 파일을 수정해야 합니다.

절차

1. **stack** 사용자로 언더클라우드 호스트에 로그인합니다.
2. **undercloud.conf** 파일을 편집합니다.
3. 파일의 **DEFAULT** 섹션에 다음 매개변수를 추가합니다.

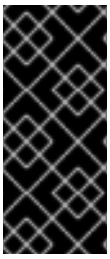
```
container_images_file = /home/stack/containers-prepare-parameter.yaml
```

   이 매개변수는 director가 올바른 위치에서 언더클라우드의 컨테이너 이미지를 가져오도록 **containers-prepare-parameter.yaml** 환경 파일의 위치를 정의합니다.

4. **generate_service_certificate** 매개 변수를 확인합니다. 이 매개변수의 기본값은 **false** 에서 **true** 로 변경되어 업그레이드 중에 언더클라우드에서 SSL/TLS를 활성화합니다.
5. 예측 가능한 NIC 명명 규칙으로 마이그레이션한 경우 **local_interface** 매개변수를 확인합니다.
6. Red Hat OpenStack Platform 13에서 **masquerade_network** 매개변수를 설정하는 경우 이 매개변수를 제거하고 각 서브넷에 **masquerade = true** 를 설정합니다.
7. 변경 사항이 있는지 파일의 다른 매개 변수를 모두 확인합니다.
8. 파일을 저장합니다.

6.10. DIRECTOR 설정 매개변수

다음 목록에는 **undercloud.conf** 파일을 설정하기 위한 매개변수 정보가 나와 있습니다. 오류를 방지하려면 모든 매개변수를 관련 섹션에 보관합니다.



중요

최소한 **container_images_file** 매개변수를 컨테이너 이미지 구성이 포함된 환경 파일로 설정해야 합니다. 이 매개변수를 적절한 파일로 올바르게 설정하지 않으면 director가 **ContainerImagePrepare** 매개변수에서 컨테이너 이미지 규칙 세트를 가져올 수 없거나 **ImageRegistryCredentials** 매개변수에서 컨테이너 레지스트리 인증 세부 정보를 가져올 수 없습니다.

기본값

다음 매개변수는 **undercloud.conf** 파일의 **[DEFAULT]** 섹션에 정의됩니다.

additional_architectures

오버클라우드에서 지원하는 추가(커널) 아키텍처 목록입니다. 현재 오버클라우드는 기본 **x86_64** 아키텍처 외에 **ppc64le** 아키텍처를 지원합니다.

certificate_generation_ca

요청된 인증서에 서명하는 CA의 **certmonger** 닉네임입니다. **generate_service_certificate** 매개변수를 설정한 경우에만 이 옵션을 사용합니다. **local** CA를 선택한 경우, certmonger는 로컬 CA 인증서를 **/etc/pki/ca-trust/source/anchors/cm-local-ca.pem**으로 추출하고 신뢰 체인에 인증서를 추가합니다.

clean_nodes

배포 중에 그리고 인트로스펙션 후에 하드 드라이브를 초기화할 것인지 여부를 정의합니다.

cleanup

임시 파일을 정리합니다. 배포 명령을 실행한 후 배포 중 사용한 임시 파일을 그대로 두려면 **False**로 설정하십시오. 이 매개변수는 생성된 파일을 디버깅하거나 오류가 발생한 경우에 유용합니다.

container_cli

컨테이너 관리를 위한 CLI 툴입니다. 이 매개변수를 **podman**으로 설정해 둡니다. Red Hat Enterprise Linux 8.2는 **podman**만 지원합니다.

container_healthcheck_disabled

컨테이너화된 서비스 상태 점검을 비활성화합니다. 상태 점검을 활성화하고 이 옵션을 **false**로 설정해 두는 것이 좋습니다.

container_images_file

컨테이너 이미지 정보가 포함된 heat 환경 파일입니다. 이 파일은 다음 항목을 포함할 수 있습니다.

- 필요한 모든 컨테이너 이미지에 대한 매개변수
- 필요한 이미지 준비를 수행하는 **ContainerImagePrepare** 매개변수. 일반적으로 이 매개변수가 포함된 파일의 이름은 **containers-prepare-parameter.yaml**입니다.

container_insecure_registries

podman이 사용할 수 있는 비보안 레지스트리 목록입니다. 개인 컨테이너 레지스트리와 같은 다른 소스에서 이미지를 가져오려는 경우 이 매개변수를 사용합니다. 대부분의 경우 언더클라우드가 Satellite에 등록되어 있으면, Red Hat Container Catalog 또는 Satellite 서버에서 컨테이너 이미지를 가져올 수 있는 인증서가 **podman**에 있습니다.

container_registry_mirror

podman에서 사용하도록 구성된 **registry-mirror**(선택 사항)입니다.

custom_env_files

언더클라우드 설치에 추가할 추가 환경 파일입니다.

deployment_user

언더클라우드를 설치하는 사용자입니다. 현재 기본 사용자인 **stack**을 사용하려면 이 매개변수를 설정되지 않은 상태로 두십시오.

discovery_default_driver

자동으로 등록된 노드의 기본 드라이버를 설정합니다. **enable_node_discovery** 매개변수를 활성화해야 하며, 드라이버를 **enabled_hardware_types** 목록에 포함해야 합니다.

enable_ironic; enable_ironic_inspector; enable_mistral; enable_nova; enable_tempest; enable_validations; enable_zaqar

director를 위해 활성화할 코어 서비스를 정의합니다. 이 매개변수를 **true**로 설정된 상태로 두십시오.

enable_node_discovery

인트로스펙션 램디스크를 PXE 부팅하는 알려지지 않은 노드를 자동으로 등록합니다. 새로운 노드는 **fake** 드라이버를 기본값으로 사용하지만 덮어쓸 **discovery_default_driver**를 설정할 수 있습니다. 또한 introspection 규칙을 사용하여 새로 등록된 노드의 드라이버 정보를 지정할 수 있습니다.

enable_novajoin

언더클라우드에서 **novajoin** 메타데이터 서비스 설치 여부를 정의합니다.

enable_routed_networks

라우팅된 컨트롤 플레인 네트워크에 대한 지원을 활성화할지 여부를 정의합니다.

enable_swift_encryption

유휴 시 Swift 암호화를 활성화할지 여부를 정의합니다.

enable_telemetry

언더클라우드에 OpenStack Telemetry 서비스(gnocchi, aodh, panko)를 설치할지 여부를 정의합니다. Telemetry 서비스를 자동으로 설치하고 구성하려면 **enable_telemetry** 매개변수를 **true**로 설정합니다. 기본값은 **false**로, 언더클라우드에서 Telemetry를 비활성화합니다. 이 매개변수는 메트릭 데이터를 사용하는 Red Hat CloudForms 같은 제품을 사용할 때 필요합니다.

주의

RBAC는 모든 구성 요소에서 지원되지 않습니다. 알람 서비스(aodh) 및 Gnocchi는 보안 RBAC 규칙을 고려하지 않습니다.

enabled_hardware_types

언더클라우드를 위해 활성화할 하드웨어 유형 목록입니다.

generate_service_certificate

언더클라우드 설치 중에 **undercloud_service_certificate** 매개변수에 사용되는 SSL/TLS 인증서를 생성할지 여부를 정의합니다. 언더클라우드 설치에서 생성된 인증서 **/etc/pki/tls/certs/undercloud-[undercloud_public_vip].pem**을 저장합니다. **certificate_generation_ca** 매개변수에 정의된 CA가 이 인증서에 서명합니다.

heat_container_image

사용할 Heat 컨테이너 이미지의 URL입니다. 설정되지 않은 상태로 두십시오.

heat_native

heat-all을 사용하여 호스트 기반 언더클라우드 설정을 실행합니다. **true**로 두십시오.

hieradata_override

director에서 Puppet hieradata를 구성하여 **undercloud.conf** 매개변수 이외의 사용자 지정 설정을 서비스에 제공하는 **hieradata** 오버라이드 파일의 경로입니다. 설정한 경우 언더클라우드 설치 시 이 파일이 **/etc/puppet/hieradata** 디렉터리에 복사되고 계층에서 첫 번째 파일로 설정됩니다. 이 기능 사용에 관한 자세한 내용은 언더클라우드에서 hieradata 구성을 참조하십시오.

inspection_extras

검사 프로세스 중에 추가 하드웨어 컬렉션의 활성화 여부를 정의합니다. 이 매개변수를 사용하려면 인트로스펙션 이미지에 **python-hardware** 또는 **python-hardware-detect** 패키지가 필요합니다.

inspection_interface

director에서 노드 인트로스펙션에 사용하는 브릿지입니다. 이 브릿지는 director 구성으로 생성되는 사용자 지정 브릿지입니다. **LOCAL_INTERFACE**가 이 브릿지에 연결됩니다. 이 브릿지를 기본값 **br-ctlplane**으로 두십시오.

inspection_runbench

노드 인트로스펙션 중 벤치마크 집합을 실행합니다. 벤치마크를 활성화하려면 이 매개변수를 **true**로 설정합니다. 등록된 노드의 하드웨어를 검사할 때 벤치마크 분석을 수행하려는 경우 이 옵션이 필요합니다.

ipa_otp

언더클라우드 노드를 IPA 서버에 등록할 때 사용할 일회성 암호를 정의합니다. 이 암호는 **enable_novajoin**이 활성화된 경우 필요합니다.

ipv6_address_mode

언더클라우드 프로비저닝 네트워크의 IPv6 주소 구성 모드입니다. 다음 목록에 이 매개변수에 사용할 수 있는 값이 나와 있습니다.

- dhcpv6-stateless - RA(라우터 알림)를 사용한 주소 구성 및 DHCPv6을 사용한 선택적 정보입니다.
- dhcpv6-stateful - DHCPv6을 사용한 주소 설정 및 선택적 정보입니다.

ipxe_enabled

iPXE 또는 표준 PXE 사용 여부를 정의합니다. 기본값은 **true**이며, iPXE를 활성화합니다. 표준 PXE를 사용하려면 이 매개변수를 **false**로 설정하십시오.

local_interface

director 프로비저닝 NIC용으로 선택한 인터페이스로, director에서 해당 DHCP 및 PXE 부팅 서비스에 사용하는 장치이기도 합니다. 이 값을 원하는 장치로 변경하십시오. 연결된 장치를 확인하려면 **ip addr** 명령을 사용합니다. 예를 들면 다음은 **ip addr** 명령을 실행한 결과입니다.

```
2: em0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen
1000
    link/ether 52:54:00:75:24:09 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.178/24 brd 192.168.122.255 scope global dynamic em0
       valid_lft 3462sec preferred_lft 3462sec
    inet6 fe80::5054:ff:fe75:2409/64 scope link
       valid_lft forever preferred_lft forever
3: em1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noop state DOWN
    link/ether 42:0b:c2:a5:c1:26 brd ff:ff:ff:ff:ff:ff
```

이 예제에서 외부 NIC는 **em0**을 사용하고, 프로비저닝 NIC는 현재 구성되지 않은 **em1**을 사용합니다. 이 경우에는 **local_interface**를 **em1**로 설정합니다. 설정 스크립트는 이 인터페이스를 **inspection_interface** 매개변수로 정의된 사용자 브릿지에 연결합니다.

local_ip

director 프로비저닝 NIC에 대해 정의된 IP 주소입니다. director에서 DHCP 및 PXE 부팅 서비스에 사용하는 IP 주소이기도 합니다. 프로비저닝 네트워크에 다른 서브넷을 사용하지 않는 경우, 예를 들어 이 IP 주소가 해당 환경의 기존 IP 주소 또는 서브넷과 충돌하는 경우 이 값을 기본값인 **192.168.24.1/24**로 유지하십시오.
IPv6의 경우 상태 저장 및 상태 비저장 연결을 모두 지원하려면 로컬 IP 주소 접두사 길이는 **/64** 여야 합니다.

local_mtu

local_interface에 사용할 MTU(최대 전송 단위)입니다. 언더클라우드의 경우 1500을 초과하지 마십시오.

local_subnet

PXE 부팅 및 DHCP 인터페이스에 사용할 로컬 서브넷입니다. **local_ip** 주소는 이 서브넷에 존재해야 합니다. 기본값은 **ctlplane-subnet**입니다.

net_config_override

네트워크 구성 덮어쓰기 템플릿의 경로입니다. 이 매개변수를 설정하면 언더클라우드는 JSON 또는 YAML 포멧 템플릿을 사용하여 **os-net-config**로 네트워킹을 구성하고 **undercloud.conf**에 설정된 네트워크 매개변수를 무시합니다. 본딩을 구성하거나 인터페이스에 옵션을 추가하려는 경우 이 매개변수를 사용합니다. 언더클라우드 네트워크 인터페이스 사용자 지정에 대한 자세한 내용은 언더클라우드 네트워크 인터페이스 구성을 참조하십시오.

networks_file

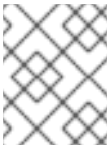
heat에 대해 오버라이드할 네트워크 파일입니다.

output_dir

상태, 처리된 heat 템플릿, Ansible 배포 파일을 출력할 디렉터리입니다.

overcloud_domain_name

오버클라우드를 배포할 때 사용할 DNS 도메인 이름입니다.



참고

오버클라우드를 구성할 때 **CloudDomain** 매개변수를 일치하는 값으로 설정해야 합니다. 오버클라우드 구성 시 환경 파일에서 이 매개변수를 설정하십시오.

roles_file

언더클라우드 설치를 위한 기본 역할 파일을 덮어쓰는 데 사용할 역할 파일입니다. director 설치에 기본 역할 파일이 사용되도록 이 매개변수를 설정되지 않은 상태로 두는 것이 좋습니다.

scheduler_max_attempts

스케줄러가 인스턴스 배포를 시도하는 최대 횟수입니다. 이 값은 스케줄링할 때 잠재적인 경합 조건을 피하기 위해 즉시 배포해야 하는 베어 메탈 노드 수보다 크거나 같아야 합니다.

service_principal

인증서를 사용하는 서비스에 대한 Kerberos 사용자입니다. FreeIPA와 같이 CA에 Kerberos 사용자가 필요한 경우에만 이 매개변수를 사용합니다.

subnets

프로비저닝 및 인트로스펙션에 사용되는 라우팅된 네트워크 서브넷 목록입니다. 기본값은 **ctlplane-subnet** 서브넷만 포함합니다. 자세한 내용은 서브넷 의 내용을 참조하십시오.

templates

재정의할 heat 템플릿 파일입니다.

undercloud_admin_host

SSL/TLS를 통한 director Admin API 엔드포인트에 대해 정의된 IP 주소 또는 호스트 이름입니다. director 설정에 따라 **/32** 넷마스크를 사용하는 라우팅된 IP 주소로 IP 주소를 director 소프트웨어 브릿지에 연결합니다.
undercloud_admin_host 가 **local_ip** 와 동일한 IP 네트워크에 없는 경우 **ControlVirtualInterface** 매개변수를 언더클라우드의 관리자 API를 수신 대기할 인터페이스로 설정해야 합니다. 기본적으로 관리 API는 **br-ctlplane** 인터페이스에서 수신 대기합니다. 사용자 지정 환경 파일에서 **ControlVirtualInterface** 매개변수를 설정하고 **custom_env_files** 매개변수를 구성하여 **undercloud.conf** 파일에 사용자 지정 환경 파일을 포함합니다.

언더클라우드 네트워크 인터페이스 사용자 지정에 대한 자세한 내용은 언더클라우드 네트워크 인터페이스 구성을 참조하십시오.

undercloud_debug

언더클라우드 서비스의 로그 수준을 **DEBUG**로 설정합니다. **DEBUG** 로그 수준을 활성화하려면 이 값을 **true**로 설정합니다.

undercloud_enable_selinux

배포 중 SELinux를 사용 또는 사용 안 함으로 설정합니다. 문제를 디버깅하지 않는 경우 이 값을 **true**로 설정된 상태로 두는 것이 좋습니다.

undercloud_hostname

언더클라우드에 대해 정규화된 호스트 이름을 정의합니다. 설정되어 있는 경우 언더클라우드 설치 시 모든 시스템 호스트 이름 설정이 구성됩니다. 설정되어 있지 않은 경우 언더클라우드에서 현재 호스트 이름을 사용하지만 사용자가 모든 시스템 호스트 이름 설정을 적절하게 구성해야 합니다.

undercloud_log_file

언더클라우드 설치 및 업그레이드 로그를 저장할 로그 파일 경로입니다. 기본적으로 로그 파일은 홈 디렉터리에 있는 **install-undercloud.log**입니다. 예를 들면 **/home/stack/install-undercloud.log**입니다.

undercloud_nameservers

언더클라우드 호스트 이름 확인에 사용할 DNS 이름 서버 목록입니다.

undercloud_ntp_servers

언더클라우드의 날짜 및 시간을 동기화하는 데 사용되는 네트워크 시간 프로토콜 서버 목록입니다.

undercloud_public_host

SSL/TLS를 통한 director Public API 엔드포인트에 대해 정의된 IP 주소 또는 호스트 이름입니다. director 설정에 따라 **/32** 넷마스크를 사용하는 라우팅된 IP 주소로 IP 주소를 director 소프트웨어 브릿지에 연결합니다.
undercloud_public_host 가 **local_ip** 와 동일한 IP 네트워크에 없는 경우 **PublicVirtualInterface** 매개 변수를 언더클라우드의 공용 API를 수신 대기할 공용용 인터페이스로 설정해야 합니다. 기본적으로 공용 API는 **br-ctlplane** 인터페이스에서 수신 대기합니다. 사용자 지정 환경 파일에 **PublicVirtualInterface** 매개변수를 설정하고 **custom_env_files** 매개변수를 구성하여 **undercloud.conf** 파일에 사용자 지정 환경 파일을 포함합니다.

언더클라우드 네트워크 인터페이스 사용자 지정에 대한 자세한 내용은 언더클라우드 네트워크 인터페이스 구성을 참조하십시오.

undercloud_service_certificate

OpenStack SSL/TLS 통신을 위한 인증서 위치 및 파일 이름입니다. 이 인증서를 신뢰할 수 있는 인증 기관에서 가져오는 것이 가장 좋습니다. 또는 자체 서명된 고유 인증서를 생성합니다.

undercloud_timezone

언더클라우드의 호스트 시간대입니다. 시간대를 지정하지 않으면 director는 기존의 표준 시간대 설정을 사용합니다.

undercloud_update_packages

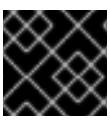
언더클라우드 설치 중 패키지 업데이트 여부를 정의합니다.

서브넷

각 프로비저닝 서브넷은 **undercloud.conf** 파일에서 이름이 지정된 섹션입니다. 예를 들어 **ctlplane-subnet**이라는 서브넷을 생성하려면 **undercloud.conf** 파일에서 다음 샘플을 사용합니다.

```
[ctlplane-subnet]
cidr = 192.168.24.0/24
dhcp_start = 192.168.24.5
dhcp_end = 192.168.24.24
inspection_iprange = 192.168.24.100,192.168.24.120
gateway = 192.168.24.1
masquerade = true
```

환경에 따라 필요한 만큼의 프로비저닝 네트워크를 지정할 수 있습니다.



중요

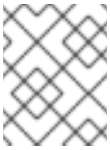
director가 서브넷을 생성한 후에는 director가 서브넷의 IP 주소를 변경할 수 없습니다.

cidr

director에서 오버클라우드 인스턴스를 관리하는 데 사용하는 네트워크입니다. 이 네트워크는 언더클라우드의 **neutron** 서비스에서 관리하는 프로비저닝 네트워크입니다. 프로비저닝 네트워크에 다른 서브넷을 사용하지 않는 경우 기본값 **192.168.24.0/24**로 두십시오.

masquerade

외부 액세스를 위해 **cidr**에 정의된 네트워크를 마스커레이드할지 여부를 정의합니다. 그러면 프로비저닝 네트워크에 일정 수준의 NAT(네트워크 주소 변환)가 제공되어 director를 통해 프로비저닝 네트워크에서 외부 액세스가 가능합니다.



참고

director 구성은 적절한 **sysctl** 커널 매개변수를 사용하여 IP 포워딩을 자동으로 활성화합니다.

dhcp_start; dhcp_end

오버클라우드 노드의 DHCP 할당 범위 시작과 끝 값입니다. 이 범위에 노드를 할당하기에 충분한 IP 주소가 포함되어 있는지 확인하십시오.

dhcp_exclude

DHCP 할당 범위에서 제외할 IP 주소입니다.

dns_nameservers

서브넷과 관련된 DNS 네임 서버입니다. 서브넷의 네임 서버가 정의되지 않은 경우 서브넷에서는 **undercloud_nameservers** 매개변수에 정의된 네임 서버를 사용합니다.

gateway

오버클라우드 인스턴스의 게이트웨이입니다. 트래픽을 외부 네트워크로 전달하는 언더클라우드 호스트입니다. director에 다른 IP 주소를 사용하거나 외부 게이트웨이를 직접 사용하려는 경우를 제외하고 기본값 **192.168.24.1**로 두십시오.

host_routes

이 네트워크의 오버클라우드 인스턴스에 대한 Neutron 관리 서브넷의 호스트 경로입니다. 언더클라우드 **local_subnet**의 호스트 경로도 구성합니다.

inspection_iprange

이 네트워크에서 노드가 검사 프로세스 중에 사용할 임시 IP 범위입니다. 이 범위는 **dhcp_start** 및 **dhcp_end**에 정의된 범위와 겹치지 않아야 하며, 동일한 IP 서브넷에 있어야 합니다.

6.11. DIRECTOR 업그레이드 실행

director를 업그레이드하려면 다음 단계를 완료합니다.

절차

1. 다음 명령을 실행하여 언더클라우드에서 director를 업그레이드합니다.

```
$ openstack undercloud upgrade
```

이 명령은 director 설정 스크립트를 실행합니다. director가 패키지를 업그레이드하고 **undercloud.conf** 의 설정에 맞게 해당 서비스를 구성합니다. 이 스크립트를 완료하는 데 몇 분이 걸립니다.

참고

계속하기 전에 director 구성 스크립트에서 확인 메시지를 표시합니다. **-y** 옵션을 사용하여 이 확인을 바이패스합니다.

```
$ openstack undercloud upgrade -y
```

2. 이 스크립트는 언더클라우드의 모든 OpenStack Platform 서비스 컨테이너를 자동으로 시작합니다. **systemd** 리소스를 통해 각 서비스를 관리합니다. **systemd** 리소스를 확인합니다.

```
$ sudo systemctl list-units "tripleo_*"
```

각 **systemd** 서비스는 컨테이너를 제어합니다. 다음 명령을 사용하여 활성화된 컨테이너를 확인합니다.

```
$ sudo podman ps
```

3. 이 스크립트는 **stack** 사용자를 **docker** 그룹에 추가하여 **stack** 사용자가 컨테이너 관리 명령에 액세스할 수 있는지 확인합니다. 다음 명령을 사용하여 **stack** 사용자 권한을 새로 고칩니다.

```
$ exec su -l stack
```

명령을 실행하면 다시 로그인하라는 메시지가 표시됩니다. stack 사용자 암호를 입력합니다.

4. **stack** 사용자를 초기화하여 명령줄 툴을 사용하려면 다음 명령을 실행합니다.

```
$ source ~/stackrc
```

프롬프트 메세지에 OpenStack 명령이 언더클라우드를 인증 및 실행될 수 있음을 나타냅니다.

```
(undercloud) $
```

director 업그레이드가 완료되었습니다.

7장. 오버클라우드 준비의 초기 단계

오버클라우드 업그레이드를 준비하려면 몇 가지 초기 단계를 완료해야 합니다.

7.1. 오버클라우드 서비스 다운타임 준비

오버클라우드 업그레이드 프로세스에서는 주요 시점에서 기본 컨트롤 플레인 서비스를 비활성화합니다. 이러한 키 지점에 도달할 때 오버클라우드 서비스를 사용하여 새 리소스를 생성할 수 없습니다. 오버클라우드에서 실행 중인 워크로드는 업그레이드 프로세스 중에 활성 상태로 유지되므로 컨트롤 플레인을 업그레이드하는 동안 인스턴스가 계속 실행됩니다. 컴퓨팅 노드를 업그레이드하는 동안 이러한 워크로드를 이미 업그레이드한 컴퓨팅 노드로 실시간 마이그레이션할 수 있습니다.

사용자가 업그레이드하는 동안 오버클라우드 서비스에 액세스할 수 없도록 유지 관리 기간을 계획하는 것이 중요합니다.

오버클라우드 업그레이드의 영향을 받습니다.

- OpenStack Platform 서비스

오버클라우드 업그레이드의 영향을 받지 않음

- 업그레이드 중에 실행되는 인스턴스
- Ceph Storage OSD(인스턴스용 백엔드 스토리지)
- Linux 네트워킹
- Open vSwitch 네트워킹
- 언더클라우드

7.2. 업그레이드 테스트용 컴퓨팅 노드 선택

오버클라우드 업그레이드 프로세스를 통해 다음 중 하나를 수행할 수 있습니다.

- 역할의 모든 노드 업그레이드
- 개별 노드 별

오버클라우드 업그레이드 프로세스를 원활하게 수행하려면 모든 컴퓨팅 노드를 업그레이드하기 전에 사용자 환경의 몇 가지 개별 컴퓨팅 노드에서 업그레이드를 테스트하는 것이 좋습니다. 이렇게 하면 워크로드에 대한 다운타임을 최소화하면서 업그레이드하는 동안 주요 문제가 발생하지 않습니다.

다음 권장 사항을 사용하여 업그레이드할 테스트 노드를 선택하는 데 도움이 됩니다.

- 업그레이드 테스팅을 위해 두 개 또는 세 개의 컴퓨팅 노드 선택
- 중요한 인스턴스가 실행되지 않은 노드 선택
- 필요한 경우 선택한 테스트 컴퓨팅 노드에서 다른 컴퓨팅 노드로 중요한 인스턴스를 마이그레이션합니다.

7.3. 오버클라우드 인벤토리 파일 생성

tripleo-ansible-inventory 명령을 사용하여 환경에 있는 모든 노드의 Ansible 인벤토리 파일을 생성합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

3. 모든 노드의 정적 인벤토리 파일을 생성합니다.

```
$ tripleo-ansible-inventory --static-yaml-inventory ~/inventory.yaml --stack STACK_NAME
```

기본 **오버클라우드** 스택 이름을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack *STACK NAME*** 옵션으로 스택 이름을 설정합니다.

4. 환경에서 Ansible 플레이북을 실행하려면 **ansible-playbook** 명령을 실행하고 **-i** 옵션을 사용하여 동적 인벤토리 툴의 전체 경로를 포함합니다. 예를 들면 다음과 같습니다.

```
(undercloud) $ ansible-playbook -i ~/inventory.yaml PLAYBOOK
```

7.4. 사전 업그레이드 요구 사항 검증

pre-upgrade 검증 그룹을 실행하여 사전 업그레이드 요구 사항을 확인합니다.

RHOSP(Red Hat OpenStack Platform) 검증 프레임워크에 대한 자세한 내용은 *Director 설치 및 사용 가이드의 검증 프레임워크 사용*을 참조하십시오.

절차

1. **stackrc** 파일을 소싱합니다.

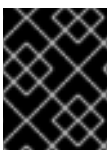
```
$ source ~/stackrc
```

2. **--group pre-upgrade** 옵션을 사용하여 **openstack tripleo validator run 명령을 실행하고 /usr/libexec/platform-python** python 런타임 환경을 포함합니다.

```
$ openstack tripleo validator run --group pre-upgrade --python-interpreter
/usr/libexec/platform-python -i inventory.yaml
```

3. 검증 보고서 결과를 확인하십시오. 특정 검증의 자세한 출력을 보려면 보고서의 특정 검증 UUID에 대해 **openstack tripleo validator show run --full** 명령을 실행합니다.

```
$ openstack tripleo validator show run  --full <UUID>
```



중요

검증 **FAILED** 로 인해 RHOSP를 배포하거나 실행할 수 없습니다. 그러나 **FAILED** 검증 결과는 프로덕션 환경에서 잠재적으로 문제가 발행할 수 있다는 것을 의미합니다.

7.5. 오버클라우드에서 펜싱 비활성화

오버클라우드를 업그레이드하기 전에 펜싱이 비활성화되어 있는지 확인합니다.

오버클라우드를 업그레이드하는 경우 각 컨트롤러 노드를 개별적으로 업그레이드하여 고가용성 기능을 유지합니다. 펜싱이 환경에 배포된 경우 오버클라우드는 특정 노드를 비활성화된 것으로 탐지하고 펜싱 작업을 시도하여 의도하지 않은 결과를 초래할 수 있습니다.

오버클라우드에서 펜싱을 활성화한 경우 의도하지 않은 결과를 피하려면 업그레이드 기간 동안 펜싱을 일시적으로 비활성화해야 합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

3. 컨트롤러 노드에 로그인하고 Pacemaker 명령을 실행하여 펜싱을 비활성화합니다.

```
$ ssh heat-admin@CONTROLLER_IP "sudo pcs property set stonith-enabled=false"
```

4. **fencing.yaml** 환경 파일에서 업그레이드 프로세스 중에 펜싱을 계속 비활성화하도록 **EnableFencing** 매개변수를 **false** 로 설정합니다.

추가 리소스

- STONITH를 사용하여 컨트롤러 노드 펜싱

7.6. 언더클라우드 노드 데이터베이스 백업

backup-and-restore Ansible 역할을 사용하여 언더클라우드 노드에서 실행되는 데이터베이스 백업을 만들고 해당 백업을 사용하여 손상된 이벤트의 데이터베이스 상태를 복구할 수 있습니다. 언더클라우드 데이터베이스 백업에 대한 자세한 내용은 언더클라우드 및 *컨트롤 플레인 노드 지원 및 복원에서 언더클라우드 노드의 데이터베이스 백업 생성* 을 참조하십시오.

8장. LEAPP 업그레이드를 위한 오버클라우드 구성

수명이 긴 RHOSP(Red Hat OpenStack Platform) 업그레이드에는 Red Hat Enterprise Linux 7에서 Red Hat Enterprise Linux 8으로 기본 운영 체제를 업그레이드해야 합니다. Red Hat Enterprise Linux 7은 Leapp 유틸리티를 사용하여 Red Hat Enterprise Linux 8로의 업그레이드를 수행합니다. Leapp 및 해당 종속 항목에 대한 자세한 내용은 업그레이드용 RHEL 7 시스템 준비를 참조하십시오.

오버클라우드 업그레이드 프레임워크는 rep 업그레이드를 자동으로 실행합니다. RHOSP 업그레이드에 성공하려면 사전 업그레이드 보고서를 수동으로 실행하여 잠재적인 문제를 파악하고 해결하는 것이 좋습니다. 각 Compute, Controller, Ceph Storage 역할 중 하나 이상의 호스트에 대해 사전 업그레이드 보고서를 실행합니다. Leapp 사전 업그레이드 보고서에 대한 자세한 내용은 사전 업그레이드 보고서 검토를 참조하십시오.

8.1. 업그레이드 환경 파일 생성

업그레이드 프로세스에서는 환경 파일을 사용하여 업그레이드 프로세스를 활성화하고 특정 업그레이드 매개변수를 구성합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. **templates** 디렉터리에 upgrade **-environment.yaml** 이라는 환경 파일을 생성합니다.

```
$ touch templates/upgrades-environment.yaml
```

3. 파일을 편집하고 다음 필수 콘텐츠를 추가합니다.

```
parameter_defaults:
  UpgradeLeappDevelSkip: "LEAPP_UNSUPPORTED=1
LEAPP_DEVEL_TARGET_RELEASE=8.2"
  LeappInitCommand: |
    for module in pata_acpi floppy; do sudo sed -i "/^${module}$/d" /usr/share/leapp-
repository/repositories/system_upgrade/el7toel8/actors/kernel/checkkerneldrivers/files/removed
_drivers.txt; done
    sudo rm -f /usr/share/leapp-
repository/repositories/system_upgrade/el7toel8/actors/persistentnetnamesdisable/actor.py
    sudo yum -y remove mariadb-server* || true
  UpgradeInitCommand: "sudo dnf config-manager --save --setopt exclude=""
```

   - **UpgradeLeappDevelSkip** 은 Leapp 점검을 건너뛰고 Leapp을 실행할 때 환경 변수를 설정합니다.
   - **LeappInitCommand** 는 각 오버클라우드 노드에서 실행되는 명령 또는 스크립트 조각을 전달하고 Leapp 업그레이드를 위해 노드를 준비합니다.
   - **UpgradeInitCommand** 는 각 오버클라우드 노드에서 실행되는 명령 또는 스크립트 조각을 전달합니다. **dnf config-manager --save --setopt exclude="** 명령은 DNF 제외에서 Leapp 패키지를 제거하여 **python2** 패키지의 자동 제거를 성공적으로 수행합니다.

4. 선택 사항: 해당 환경에서 TLS-Everywhere를 사용하고 **authconfig** 에서 **authselect** 로 마이그레이션하려는 경우 BZ#1978228 - OSP13octets16.2 Leapp 업그레이드가 TLS everywhere :로 실패하지 않도록 **authselect_check.confirm** 매개변수를 **True** 로 설정합니다.

```
parameter_defaults:
  LeappInitCommand: |
    sudo leapp answer --section authselect_check.confirm=True --add
```

그렇지 않으면 값을 **False** 로 설정합니다.

참고

| 구문을 사용하여 **LeappInitCommand** 매개변수에 여러 명령을 전달합니다.

```
parameter-defaults:
  LeappInitCommand: |
    <command_1>
    <command_2>
```

5. upgrade **-environment.yaml** 파일을 저장합니다.

8.2. 업그레이드 매개변수

매개변수	설명
UpgradeInitCommand	업그레이드 프로세스를 초기화하기 위해 모든 Overcloud 노드에서 실행할 명령 또는 스크립트 코드 조각입니다. 예를 들어 리포지터리 스위치는 다음과 같습니다.
UpgradeInitCommonCommand	업그레이드 프로세스에 필요한 일반적인 명령. 이는 일반적으로 운영자가 수정하지 않아야 하며 major-upgrade-composable-steps.yaml 및 major-upgrade-converge.yaml 환경 파일에 설정 및 설정 해제됩니다.
UpgradeLeappCommandOptions	Leapp 명령에 추가할 추가 명령줄 옵션입니다.
UpgradeLeappDebug	Leapp을 실행할 때 디버깅 출력을 출력합니다. 기본값은 **True** 입니다.
UpgradeLeappDevelSkip	개발/테스트에서 Leapp을 실행할 때 env 변수를 설정하여 Leapp 확인을 건너뜁니다. 예를 들면 LEAPP_DEVEL_SKIP_RHSM=1입니다.
UpgradeLeappEnabled	운영 체제 업그레이드에는 Leapp을 사용합니다. 기본값은 **False** 입니다.
UpgradeLeappPostRebootDelay	시스템이 재부팅될 때까지 대기하고 테스트 명령에 응답하는 최대(초)입니다. 기본값은 **120** 입니다.
UpgradeLeappRebootTimeout	Leapp을 통해 OS 업그레이드 단계의 시간 제한(초). 기본값은 **3600** 입니다.

매개변수	설명
UpgradeLeappToInstall	Leapp 업그레이드 후 설치할 패키지 목록입니다.
UpgradeLeappToRemove	Leapp 업그레이드 중에 제거할 패키지 목록.

8.3. 오버클라우드 노드에 LEAPP 데이터 복사

각 오버클라우드 노드에는 Leapp 데이터 파일이 필요합니다. 언더클라우드의 **/etc/leapp/files** 디렉터리에 있는 데이터 파일을 각 Overcloud 노드의 동일한 위치로 복사합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

3. 환경에 있는 모든 노드의 정적 인벤토리 파일을 생성합니다.

```
$ tripleo-ansible-inventory --static-yaml-inventory ~/inventory.yaml --stack STACK_NAME
```

기본 **오버클라우드** 스택 이름을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack *STACK NAME*** 옵션으로 스택 이름을 설정합니다.

4. 빠른 데이터를 오버클라우드 노드에 복사하려면 다음 **동기화** Ansible 명령을 실행합니다.

```
$ ansible -i ~/inventory.yaml --become -m synchronize -a "src=/etc/leapp/files
dest=/etc/leapp/" overcloud
```

8.4. 오버클라우드 노드에 예측 가능한 NIC 이름 사용

오버클라우드 노드에서 Leapp 업그레이드를 실행하기 전에 일반적으로 **eth** 접두사가 포함된 커널 기반 NIC 이름을 확인해야 합니다. 일반적으로 이러한 NIC 이름은 NIC 할당 측면에서 예측할 수 없습니다.

playbook **-nics.yaml 플레이북을** 실행하여 **em** NIC 접두사를 사용하도록 NIC 이름 이름을 변경할 수 있습니다. 플레이북을 실행할 때 접두사 변수를 수정하여 다른 NIC **접두사를** 설정할 수도 있습니다. 그러나 NIC 변경은 Leapp 업그레이드 프로세스가 완료되고 노드가 재부팅된 후에만 적용됩니다.

사전 요구 사항

- Undercloud 준비 프로세스 중에 생성된 **playbook-nics.yaml** 플레이북입니다. **playbook-nics.yaml** 플레이북은 이더넷 장치, 브리지 및 Linux 본딩을 사용하는 대부분의 오버클라우드 네트워킹 시나리오를 수용합니다. 환경에 이러한 장치 유형 이외의 추가 구성이 필요한 경우 계속하기 전에 다음 권장 사항을 따르십시오.
  - 오버클라우드 노드와 유사한 네트워킹 구성을 사용하여 별도의 시스템에서 플레이북을 테스트합니다.
  - 다른 장치 유형의 구성 내에서 **eth** 접두사의 이름을 변경할 수 있도록 플레이북을 수정합니다.

- 이 절차를 완료한 후 오버클라우드 노드의 네트워킹 구성을 확인합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. 모든 오버클라우드 노드에서 **playbook-nics.yaml** 플레이북을 실행합니다.

```
$ ansible-playbook -i ~/inventory.yaml playbook-nics.yaml
```

플레이북은 새 NIC 접두사를 **em** 로 설정합니다. 다른 NIC 접두사를 설정하려면 플레이북을 실행할 때 **접두사** 변수를 설정합니다.

```
$ ansible-playbook -i ~/inventory.yaml -e prefix="mynic" playbook-nics.yaml
```

NIC 변경 사항은 Leapp 업그레이드 프로세스가 완료되고 노드가 재부팅된 후에만 적용됩니다.

Resources

- RHEL 7에서 커널 NIC 이름을 사용할 때 RHEL 8로 인플레이스 업그레이드를 수행하는 방법
- 일관된 네트워크 장치 이름 지정
- 예측 가능한 네트워크 인터페이스 장치 이름 이해

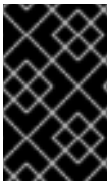
9장. 구성 가능 서비스 및 매개변수 업데이트

오버클라우드 업그레이드를 준비하려면 구성 가능 서비스 구성을 완료해야 합니다.

9.1. 사용자 정의 ROLES_DATA 파일에서 구성 가능 서비스 업데이트

이 섹션에는 새 구성 및 사용되지 않는 구성 가능 서비스에 대한 정보가 포함되어 있습니다.

- 기본 **roles_data** 파일을 사용하는 경우 이러한 서비스가 자동으로 포함됩니다.
- 사용자 지정 **roles_data** 파일을 사용하는 경우 새 서비스를 추가하고 관련 역할에 대해 더 이상 사용되지 않는 서비스를 제거합니다.

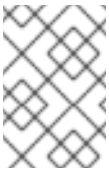


중요

배포의 오버클라우드 노드가 전용 Object Storage(swift) 노드인 경우 기본 **roles_data.yaml** 파일을 복사하고 **ObjectStorage** 를 편집하여 다음 행을 삭제합니다. **deprecated_server_resource_name: 'SwiftStorage'**

컨트롤러 노드

다음 서비스는 컨트롤러 노드에 대해 더 이상 사용되지 않습니다. 컨트롤러 역할에서 제거합니다.

Service	이유
OS::TripleO::Services::AodhApi **OS::TripleO::Services::AodhEvaluator** **OS::TripleO::Services::AodhListener** **OS::TripleO::Services::AodhNotifier**	OpenStack Telemetry 서비스는 지표 및 모니터링을 위해 STF(Service Telemetry Framework)를 대신하여 더 이상 사용되지 않습니다. 레거시 원격 분석 서비스는 STF로 쉽게 전환할 수 있도록 RHOSP 16.1에서만 사용할 수 있으며 향후 RHOSP 버전에서 제거됩니다.  **참고** 자동 스케일링을 사용하는 경우 컨트롤러 노드에서 해당 서비스를 제거하지 마십시오.
OS::TripleO::Services::PankoApi	OpenStack Telemetry 서비스는 지표 및 모니터링을 위해 STF(Service Telemetry Framework)를 대신하여 더 이상 사용되지 않습니다. 레거시 원격 분석 서비스는 STF로 쉽게 전환할 수 있도록 RHOSP 16.1에서만 사용할 수 있으며 향후 RHOSP 버전에서 제거됩니다.  **참고** CloudForms를 사용하는 경우 컨트롤러 노드에서 해당 서비스를 제거하지 마십시오.

Service	이유
OS::TripleO::Services::CeilometerApi **OS::TripleO::Services::CeilometerCollector** **OS::TripleO::Services::CeilometerExpirer** **OS::TripleO::Services::MongoDb**	이러한 서비스는 더 이상 지원되지 않습니다. 컨트롤러 노드에서 이러한 서비스를 제거합니다.
OS::TripleO::Services::Congress	카나리아 제도는 더 이상 지원되지 않습니다.
OS::TripleO::Services::GlanceRegistry	이 서비스는 OpenStack Platform Image Service(glance) API v2로 인해 더 이상 지원되지 않습니다.
OS::TripleO::Services::NeutronCorePluginPlumgrid **OS::TripleO::Services::NeutronCorePluginMidonet**	OpenStack Networking(neutron)용 더 이상 사용되지 않는 플러그인.
OS::TripleO::Services::NeutronLbaasv2Agent **OS::TripleO::Services::NeutronLbaasv2Api**	Octavia를 대표하여 더 이상 사용되지 않는 서비스로 OpenStack Networking(neutron) 로드 밸런싱을 수행합니다.
OS::TripleO::Services::NovaConsoleauth	이 서비스는 제거되었습니다.
OS::TripleO::Services::NovaPlacement	**OS::TripleO::Services::PlacementApi** 대신 더 이상 사용되지 않습니다.
OS::TripleO::Services::OpenDaylightApi **OS::TripleO::Services::OpenDaylightOvs**	OpenDaylight는 더 이상 지원되지 않습니다.
OS::TripleO::Services::RabbitMQ	이 서비스는 두 개의 새 서비스로 대체되었습니다. **OS::TripleO::Services::OsloMessagingRpc** **OS::TripleO::Services::OsloMessagingNotify**
OS::TripleO::Services::SkydiveAgent **OS::TripleO::Services::SkydiveAnalyzer**	Skydive는 더 이상 지원되지 않습니다.
OS::TripleO::Services::Tacker	타커는 더 이상 지원되지 않습니다.

다음 서비스는 컨트롤러 노드에 대한 새로운 서비스입니다. 이를 컨트롤러 역할에 추가합니다.

Service	이유
OS::TripleO::Services::CephGrafana	Ceph 대시보드 서비스를 활성화하는 작업.
OS::TripleO::Services::CinderBackendDellEMCPowermax **OS::TripleO::Services::CinderBackendDellEMCSc** **OS::TripleO::Services::CinderBackendNVMeOF**	블록 스토리지용 새 백엔드(cinder).
OS::TripleO::Services::ContainerImagePrepare	명령을 실행하여 오버클라우드의 서비스와 관련된 컨테이너 이미지를 자동으로 가져오고 준비합니다.
OS::TripleO::Services::DesignateApi **OS::TripleO::Services::DesignateCentral** **OS::TripleO::Services::DesignateProducer** **OS::TripleO::Services::DesignateWorker** **OS::TripleO::Services::DesignateMDNS** **OS::TripleO::Services::DesignateSink**	DNS-as-a-Service 서비스(designate).
OS::TripleO::Services::IronicInspector	Overcloud에 대한 베어 메탈 인트로스펙션을 위한 서비스.
OS::TripleO::Services::IronicNeutronAgent	OpenStack Bare Metal(ironic)용 네트워킹 에이전트입니다.
OS::TripleO::Services::NeutronAgentsIBConfig	Mellanox InfiniBand용 OpenStack Networking(neutron) 에이전트.
OS::TripleO::Services::OpenStackClients	Red Hat OpenStack Platform 명령줄 툴을 설치하기 위한 서비스입니다.
OS::TripleO::Services::OsloMessagingRpc **OS::TripleO::Services::OsloMessagingNotify**	**OS::TripleO::Services::RabbitMQ** 서비스의 교체 서비스.
OS::TripleO::Services::PlacementApi	배치 API 서비스.

컴퓨팅 노드

다음 서비스는 Compute 노드에 대해 더 이상 사용되지 않습니다. 컴퓨팅 역할에서 제거합니다.

Service	이유
OS::TripleO::Services::OpenDaylightOvs	OpenDaylight는 더 이상 지원되지 않습니다.
OS::TripleO::Services::SkydiveAgent	Skydive는 더 이상 지원되지 않습니다.

다음 서비스는 Compute 노드에 대한 새로운 서비스입니다. Compute 역할에 추가합니다.

Service	이유
OS::TripleO::Services::NovaAZConfig	OpenStack Compute(nova)에서 호스트 집계 및 가용 영역을 구성하는 서비스입니다.

모든 노드

다음 서비스는 모든 노드에서 더 이상 사용되지 않습니다. 모든 역할에서 제거합니다.

Service	이유
OS::TripleO::Services::Docker	Podman으로 대체.
OS::TripleO::Services::Fluentd	**OS::TripleO::Services::Rsyslog** 를 선호하는 경우 더 이상 사용되지 않습니다.
OS::TripleO::Services::Ntp	**OS::TripleO::Services::Timesync** 를 선호하는 경우 더 이상 사용되지 않습니다.
OS::TripleO::Services::SensuClient	더 이상 사용되지 않는 서비스.
OS::TripleO::Services::Ptp	**OS::TripleO::Services::Timesync** 를 선호하는 경우 더 이상 사용되지 않습니다.

다음 서비스는 모든 노드에 대해 새로운 서비스입니다. 모든 역할에 추가합니다.

Service	이유
OS::TripleO::Services::BootParams	커널 인수, tuned 프로필 및 CPU 격리를 설정하는 서비스입니다.
OS::TripleO::Services::Collectd	수집을 구성하는 서비스.
OS::TripleO::Services::Multipathd	기본적으로 비활성화된 Multipathd 서비스 제공
OS::TripleO::Services::Podman	Podman을 설치하고 활성화할 서비스.

Service	이유
OS::TripleO::Services::Rear	Relax-and- recovery(ReaR) 백업 및 복원 도구를 설치하고 활성화하는 서비스입니다.
OS::TripleO::Services::Rsyslog	중앙 집중식 로그 컬렉션을 구성하는 서비스입니다.
OS::TripleO::Services::Timesync	시간 동기화 방법을 활성화하는 서비스는 기본적으로 Chronyd입니다.

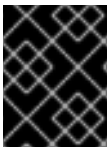
9.2. 사용자 지정 환경 파일에서 구성 가능 서비스 업데이트

resource **_registry** 섹션이 있는 사용자 지정 환경 파일이 있는 경우 **resource_registry** 섹션에서 구성 가능한 서비스 템플릿 매핑의 변경 사항이 있는지 확인합니다. Red Hat OpenStack Platform 16.1용 구성 가능 서비스 파일은 **/usr/share/openstack-tripleo-heat-templates/** 내의 새 위치에 있습니다.

Red Hat OpenStack Platform 13	Red Hat OpenStack Platform 16.1
docker/services/	**Deployment**

배포 디렉터리에는 구성 가능한 서비스를 그룹화할 수 있는 하위 디렉터리 집합이 포함되어 있습니다. 예를 들어 **keystone 하위 디렉터리에** 는 OpenStack ID(keystone)를 위한 구성 가능 서비스 템플릿이 포함되어 있습니다.

사용자 지정 환경 파일에서 구성 가능한 서비스를 다시 매핑하려면 현재 서비스 매핑의 템플릿 위치를 확인하고 새 위치로 매핑을 편집합니다. 이 절차에서는 예제로 **ceph-mgr.yaml** 을 사용합니다.



중요

이 절차는 구성 가능 서비스를 다시 매핑하는 데만 도움이 됩니다. 매핑이 확실하지 않은 경우 Red Hat 에 문의하고 올바른 매핑에 대한 조언을 요청하십시오.

절차

1. 구성 가능한 서비스를 사용하는 사용자 지정 환경 파일을 검색합니다. 일반적으로 사용자 지정 환경 파일을 **/home/stack/templates** 디렉터리에 저장합니다.

```
$ cd ~/templates/
$ grep "OS::TripleO::Services" *
```

이 시나리오에서는 파일 중 하나에서 오래된 매핑을 보여줍니다.

```
  OS::TripleO::Services::CephMgr: /usr/share/openstack-tripleo-heat-
templates/docker/services/ceph-ansible/ceph-mgr.yaml
```

2. **/usr/share/openstack -tripleo-heat-templates/에서 새 ceph- mgr.yaml** 위치를 식별합니다. 이제 이 파일은 'deployment/ceph-ansible' 디렉토리에 있습니다.

```
$ find /usr/share/openstack-tripleo-heat-templates/ -name ceph-mgr.yaml
/usr/share/openstack-tripleo-heat-templates/deployment/ceph-ansible/ceph-mgr.yaml
```

3. 사용자 지정 환경 파일에서 서비스를 편집합니다.

```
resource_registry:
  OS::TripleO::Services::CephMgr: /usr/share/openstack-tripleo-heat-
templates/deployment/ceph-ansible/ceph-mgr.yaml
```

파일을 저장합니다.

9.3. 언더클라우드 레지스트리에 대한 액세스 구성

언더클라우드 레지스트리에 대한 액세스를 구성하려면 프로비저닝 네트워크에서 언더클라우드의 컨트롤 플레인 호스트 이름과 언더클라우드의 IP 주소를 **DockerInsecureRegistryAddress** 매개변수에 추가합니다. container **-prepare-parameter.yaml** 파일에 이 매개변수를 배치하여 향후 오버클라우드 배포에 매개 변수가 포함되어 있는지 확인합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. 언더클라우드에서 컨트롤 플레인 호스트 이름을 가져옵니다.

```
$ sudo hiera container_image_prepare_node_names
["undercloud.ctlplane.localdomain"]
```

3. **containers-prepare-parameter.yaml** 파일을 편집하고 언더클라우드의 컨트롤 플레인 호스트 이름과 프로비저닝 네트워크에서 언더클라우드의 IP 주소가 포함된 YAML 목록을 사용하여 **DockerInsecureRegistryAddress** 매개변수를 추가합니다.

```
parameter_defaults:
  DockerInsecureRegistryAddress:
  - undercloud.ctlplane.localdomain:8787
  - 192.168.24.1:8787
  ...
```

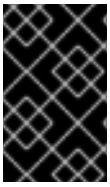
또한 오버클라우드 레지스트리의 포트 번호를 호스트 이름 및 IP 주소 값에 추가해야 합니다. 포트 번호는 **8787** 입니다.

9.4. NOVASCHEDULERDEFAULTFILTERS 매개변수에 더 이상 사용되지 않거나 삭제된 필터

환경에서 사용자 지정 **NovaSchedulerDefaultFilters** 매개변수를 사용하는 경우 매개변수를 편집하여 더 이상 사용되지 않고 제거된 다음 필터를 제거합니다.

필터	상태
AggregateCoreFilter	더 이상 사용되지 않음
AggregateRamFilter	더 이상 사용되지 않음
AggregateDiskFilter	더 이상 사용되지 않음

필터	상태
ExactCoreFilter	삭제됨
ExactRamFilter	삭제됨
ExactDiskFilter	삭제됨
CoreFilter	삭제됨
RamFilter	삭제됨
DiskFilter	삭제됨
RetryFilter	더 이상 사용되지 않음



중요

더 이상 사용되지 않는 필터를 사용하지 마십시오. Red Hat OpenStack Platform 16.1에는 더 이상 사용되지 않는 필터가 포함되지만 향후 버전의 Red Hat OpenStack Platform에는 이러한 필터가 포함되지 않습니다.

9.5. 컴퓨팅 이름 형식 설정

Red Hat OpenStack Platform 13에서는 **%stackname%-compute-%index%** 를 Compute 노드의 기본 명명 형식으로 사용합니다. Red Hat OpenStack Platform 16.1은 **%stackname%-novacompute-%index%** 를 Compute 노드의 기본 명명 형식으로 사용합니다. 원래 Red Hat OpenStack Platform 13 명명 형식을 유지하도록 기본 명명 형식을 변경합니다. 원래 명명 형식을 사용하지 않는 경우 director는 새 이름 지정 형식으로 새 OpenStack Compute(nova) 에이전트를 구성하고 기존 OpenStack Compute(nova) 에이전트를 이전 명명 형식으로 분리된 서비스로 유지합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. 컴퓨팅 이름 지정 형식을 설정합니다.
   - 사용자 지정 **roles_data** 파일을 사용하는 경우 사용자 지정 **roles_data** 파일을 편집하고 **Compute** 역할의 **HostnameFormatDefault** 매개변수를 설정합니다.

     ```
     - name: Compute
       …
       HostnameFormatDefault: '%stackname%-compute-%index%'
       …
     ```

     사용자 지정 **roles_data** 파일을 저장합니다.

   - **openstack-tripleo-heat-templates** 에서 기본 **roles_data** 파일을 사용하는 경우 환경 파일에서 이름 지정 형식을 설정합니다. 노드 수와 플레이버(일반적으로 **node-info.yaml**)로 환경 파일을 편집합니다. **ComputeHostnameFormat** 매개변수를 **parameter_defaults** 섹션에 추가합니다.

```
parameter_defaults:
  …
  ComputeHostnameFormat: '%stackname%-compute-%index%'
  …
```

node-info.yaml 파일을 저장합니다.

9.6. 인스턴스 일련 번호 구성

Red Hat OpenStack Platform 13에서 호스트 시스템의 가상 BIOS에 저장된 인스턴스의 일련 번호는 호스트의 일련 번호를 기반으로 합니다.

Red Hat OpenStack Platform 16.1에서 호스트 시스템의 가상 BIOS에 저장된 인스턴스의 일련 번호는 기본적으로 인스턴스의 UUID를 기반으로 합니다.

Red Hat OpenStack Platform 16.1로 업그레이드할 때 Red Hat OpenStack Platform 13 배포 동작을 유지하려면 **[libvirt]sysinfo_serial** 을 구성해야 합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. **stackrc** 파일을 소싱합니다.

```
[stack@director ~]$ source ~/stackrc
```

3. 환경 파일을 엽니다.
4. 환경 파일에 다음 구성을 추가하여 호스트 일련 번호를 기반으로 인스턴스 일련 번호를 지정합니다.

```
parameter_defaults:
  <Role>ExtraConfig:
    nova::config::nova_config:
      libvirt/sysinfo_serial:
        value: auto
```

5. 환경 파일에 업데이트를 저장하고 파일을 오버클라우드 업그레이드 및 배포 명령에 추가합니다.

9.7. SSL/TLS 구성 업데이트

resource_registry 에서 **NodeTLSData** 리소스를 제거하여 SSL/TLS 구성을 업데이트합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. **stackrc** 파일을 소싱합니다.

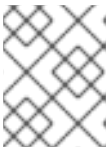
```
$ source ~/stackrc
```

3. 사용자 지정 오버클라우드 SSL/TLS 공용 엔드포인트 파일을 편집합니다. 이 파일은 일반적으로 **~/templates/enable-tls.yaml** 입니다.

4. 'resource_registry'에서 **NodeTLSData** 리소스를 제거합니다.

```
resource_registry:
  OS::TripleO::NodeTLSData: /usr/share/openstack-tripleo-heat-
templates/puppet/extraconfig/tls/tls-cert-inject.yaml
  …
```

Overcloud 배포에서는 HAProxy의 새 서비스를 사용하여 SSL/TLS가 활성화되어 있는지 확인합니다.



참고

enable-tls.yaml 파일의 **resource_registry** 섹션에 있는 유일한 리소스인 경우 전체 **resource_registry** 섹션을 제거합니다.

5. SSL/TLS 공용 엔드포인트 파일 파일을 저장합니다.

9.8. 사후 구성 템플릿 업데이트

OS::TripleO::NodeExtraConfigPost 리소스를 사용하여 구성 템플릿을 등록하고 실행하는 경우 템플릿에 **EndpointMap** 매개변수를 추가해야 합니다.

절차

1. 구성 후 템플릿을 편집합니다.
2. **parameters** 섹션에서 **EndpointMap** 매개변수 및 하위 매개변수를 추가합니다.

```
parameters:
  servers:
    type: json
  nameserver_ip:
    type: string
  DeployIdentifier:
    type: string
  EndpointMap:
    default: {}
    type: json
```

3. 템플릿을 저장합니다.

9.9. 레거시 TELEMETRY 서비스 유지 시 고려 사항

RHOSP(Red Hat OpenStack Platform) 16.1에서 OpenStack Telemetry 구성 요소는 STF(Service Telemetry Framework) 대신 더 이상 사용되지 않으므로 업그레이드 후 기존 Telemetry 구성 요소가 활성화되지 않습니다.

자동 확장 또는 CloudForms 서비스를 사용하는 경우 레거시 원격 분석 서비스를 유지해야 합니다.

기존 RHOSP 13 원격 분석 서비스를 유지하려면 openstack overcloud upgrade **prepare 및 openstack overcloud upgrade converge** 명령을 실행할 때 **/usr/share/openstack-tripleo-heat-templates/environments/enable-legacy-telemetry.yaml** 환경 파일을 포함합니다.

업그레이드 후 오버클라우드를 업데이트할 때마다 **enable-legacy-telemetry.yaml** 환경 파일도 포함해야 합니다.

레거시 원격 분석 서비스는 STF로 쉽게 전환할 수 있도록만 사용할 수 있으며 향후 RHOSP 버전에서 제거됩니다.

10장. 오버클라우드 등록을 RED HAT 고객 포털로 업데이트

Red Hat OpenStack Platform 16.1에서는 Ansible 기반 방법을 사용하여 오버클라우드 노드를 Red Hat 고객 포털에 등록합니다.

10.1. RHSM(RED HAT SUBSCRIPTION MANAGER) 구성 가능 서비스

rhsm 구성 가능 서비스를 사용하여 Ansible을 통해 오버클라우드 노드를 등록할 수 있습니다. 기본 **roles_data** 파일의 각 역할에는 기본적으로 비활성화된 **OS::TripleO::Services::Rhsm** 리소스가 포함되어 있습니다. 서비스를 활성화하려면 리소스를 **rhsm** 구성 가능 서비스 파일에 등록합니다.

```
resource_registry:
  OS::TripleO::Services::Rhsm: /usr/share/openstack-tripleo-heat-templates/deployment/rhsm/rhsm-baremetal-ansible.yaml
```

The **rhsm** 구성 가능 서비스는 **RhsmVars** 매개변수를 사용할 수 있으며, 등록과 관련된 여러 하위 매개변수를 정의하는 데 사용할 수 있습니다.

```
parameter_defaults:
  RhsmVars:
    rhsm_repos:
      - rhel-8-for-x86_64-baseos-eus-rpms
      - rhel-8-for-x86_64-appstream-eus-rpms
      - rhel-8-for-x86_64-highavailability-eus-rpms
      …
    rhsm_username: "myusername"
    rhsm_password: "p@55w0rd!"
    rhsm_org_id: "1234567"
    rhsm_release: 8.2
```

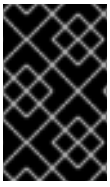
역할별 매개변수(예: **ControllerParameter s)와 함께 RhsmVar s** 매개변수를 사용하여 다른 노드 유형에 특정 리포지토리를 활성화할 때 유연성을 제공할 수도 있습니다.

10.2. RHSMVARS 하위 매개변수

구성 가능 서비스를 구성할 때 다음 하위 매개 변수를 **RhsmVars** 매개변수 **의** 일부로 사용합니다. 사용 가능한 Ansible 매개변수에 대한 자세한 내용은 역할 설명서 를 참조하십시오.

rhsm	설명
rhsm_method	등록 방법을 선택합니다. **portal**,**satellite** 또는 **disable** 중 하나입니다.
rhsm_org_id	등록에 사용하려는 조직입니다. 이 ID를 찾으려면 언더클라우드 노드에서 **sudo subscription-manager orgs** 를 실행합니다. 프롬프트에 Red Hat 자격 증명을 입력하고 결과로 생성된 **Key** (키) 값을 사용합니다. 조직 ID에 대한 자세한 내용은 Red Hat Subscription Management Organization ID 이해를 참조하십시오.

rhsm	설명
rhsm_pool_ids	사용하려는 서브스크립션 풀 ID입니다. 서브스크립션을 자동 첨부하지 않으려면 이 매개변수를 사용합니다. 이 ID를 찾으려면 언더클라우드 노드에서 **sudo subscription-manager list --available --all --matches="Red Hat OpenStack"** 을 실행하고 결과 **풀 ID** 값을 사용합니다.
rhsm_activation_key	등록에 사용할 활성화 키입니다.
rhsm_autosubscribe	이 매개변수를 사용하여 호환 가능한 서브스크립션을 이 시스템에 자동으로 첨부합니다. 이 기능을 활성화하려면 값을 **true** 로 설정합니다.
rhsm_baseurl	콘텐츠를 가져오는 기본 URL입니다. 기본 URL은 Red Hat Content Delivery Network입니다. Satellite 서버를 사용하는 경우 이 값을 Satellite 서버 콘텐츠 리포지토리의 기본 URL로 변경합니다.
rhsm_server_hostname	등록을 위한 서브스크립션 관리 서비스의 호스트 이름입니다. 기본값은 Red Hat 서브스크립션 관리 호스트 이름입니다. Satellite 서버를 사용하는 경우 이 값을 Satellite 서버 호스트 이름으로 변경합니다.
rhsm_repos	활성화하려는 리포지토리 목록입니다.
rhsm_username	등록할 사용자 이름입니다. 가능한 경우 등록에 활성화 키를 사용합니다.
rhsm_password	등록할 암호입니다. 가능한 경우 등록에 활성화 키를 사용합니다.
rhsm_release	리포지토리 고정을 위한 Red Hat Enterprise Linux 릴리스. Red Hat OpenStack Platform의 경우 8.2로 설정됩니다.
rhsm_rhsm_proxy_hostname	HTTP 프록시의 호스트 이름입니다. 예: **proxy.example.com**
rhsm_rhsm_proxy_port	HTTP 프록시 통신용 포트입니다. 예를 들면 다음과 같습니다. **8080**.
rhsm_rhsm_proxy_user	HTTP 프록시에 액세스할 사용자 이름입니다.
rhsm_rhsm_proxy_password	HTTP 프록시에 액세스할 암호입니다.



중요

rhsm_ method가 portal 로 설정된 경우에만 use rhsm_activation_key 및rhsm_ repos 를 함께 사용할 수 있습니다. If **rhsm_method** 가 'satellite'로 설정되면 either **rhsm_activation_key or rhsm_ repos** 만 사용할 수 있습니다.

10.3. RHSM 구성 가능 서비스로 전환

이전 **rhel-registration** 메서드는 Overcloud 등록을 처리하는 bash 스크립트를 실행합니다. 이 메서드의 스크립트 및 환경 파일은 **/usr/share/openstack-tripleo-heat-templates/extraconfig/pre_deploy/rhel-registration/** 의 코어 heat 템플릿 컬렉션에 있습니다.

rhel-registration 방법에서 구성 가능 서비스로 전환하려면 **다음** 단계를 완료합니다.

절차

1. **rhel-registration** 환경 파일을 향후 배포 작업에서 제외합니다. 대부분의 경우 다음 파일을 제외합니다.
   - **rhel-registration/environment-rhel-registration.yaml**
   - **rhel-registration/rhel-registration-resource-registry.yaml**
2. 사용자 지정 **roles_data** 파일을 사용하는 경우 **roles_data** 파일의 각 역할에 **OS::TripleO::Services::Rhsm** 구성 가능 서비스가 포함되어 있는지 확인합니다. 예를 들면 다음과 같습니다.

```
- name: Controller
  description: |
    Controller role that has all the controller services loaded and handles
    Database, Messaging and Network functions.
  CountDefault: 1
  ...
  ServicesDefault:
    ...
    - OS::TripleO::Services::Rhsm
    ...
```

3. 향후 배포 작업에 환경 파일 for **rhsm** composable 서비스 매개변수를 추가합니다.

이 메서드는 **rhel-registration** 매개변수를 **rhsm** 서비스 매개변수로 교체하고 서비스를 활성화하는 heat 리소스를 변경합니다.

```
resource_registry:
  OS::TripleO::NodeExtraConfig: rhel-registration.yaml
```

다음으로 변경합니다.

```
resource_registry:
  OS::TripleO::Services::Rhsm: /usr/share/openstack-tripleo-heat-templates/deployment/rhsm/rhsm-baremetal-ansible.yaml
```

배포를 통해 **/usr/share/openstack-tripleo-heat-templates/environments/rhsm.yaml** 환경 파일을 배포하여 서비스를 활성화할 수도 있습니다.

10.4. RHEL-REGISTRATION TO RHSM MAPPINGS

rhel-registration 방법에서 the **rhsm** 메서드로 세부 정보를 전환하는 데 도움이 되도록 다음 표를 사용하여 매개변수와 값을 매핑합니다.

rhel-registration	rhsm / RhsmVars
rhel_reg_method	**rhsm_method**
rhel_reg_org	**rhsm_org_id**
rhel_reg_pool_id	**rhsm_pool_ids**
rhel_reg_activation_key	**rhsm_activation_key**
rhel_reg_auto_attach	**rhsm_autosubscribe**
rhel_reg_sat_url	**rhsm_satellite_url**
rhel_reg_repos	**rhsm_repos**
rhel_reg_user	**rhsm_username**
rhel_reg_password	**rhsm_password**
rhel_reg_release	**rhsm_release**
rhel_reg_http_proxy_host	**rhsm_rhsm_proxy_hostname**
rhel_reg_http_proxy_port	**rhsm_rhsm_proxy_port**
rhel_reg_http_proxy_username	**rhsm_rhsm_proxy_user**
rhel_reg_http_proxy_password	**rhsm_rhsm_proxy_password**

10.5. RHSM 구성 가능 서비스를 사용하여 오버클라우드 등록

rhsm 구성 가능 서비스를 활성화하고 구성하는 환경 파일을 생성합니다. director는 이 환경 파일을 사용하여 노드를 등록하고 서브스크립션합니다.

절차

1. **templates/rhsm.yml** 이라는 환경 파일을 만들어 구성을 저장합니다.
2. 환경 파일에 구성을 포함합니다. 예를 들면 다음과 같습니다.

```
resource_registry:
  OS::TripleO::Services::Rhsm: /usr/share/openstack-tripleo-heat-
templates/deployment/rhsm/rhsm-baremetal-ansible.yaml
parameter_defaults:
  RhsmVars:
    rhsm_repos:
      - rhel-8-for-x86_64-baseos-eus-rpms
```

```
    - rhel-8-for-x86_64-appstream-eus-rpms
    - rhel-8-for-x86_64-highavailability-eus-rpms
    …
  rhsm_username: "myusername"
  rhsm_password: "p@55w0rd!"
  rhsm_org_id: "1234567"
  rhsm_pool_ids: "1a85f9223e3d5e43013e3d6e8ff506fd"
  rhsm_method: "portal"
  rhsm_release: 8.2
```

- **resource_registry** 섹션은 각 역할에서 사용할 수 있는 **OS::TripleO::Services::Rhsm** 리소스와 **rhsm** 구성 가능 서비스를 연결합니다.
- **RhsmVars** 변수는 Red Hat 등록을 구성하기 위해 매개 변수를 Ansible에 전달합니다.

3. 환경 파일을 저장합니다.

10.6. 다른 역할에 RHSM 구성 가능 서비스 적용

역할별로 the **rhsm** 구성 가능 서비스를 적용할 수 있습니다. 예를 들어 컨트롤러 노드, 컴퓨팅 노드 및 Ceph Storage 노드에 다양한 구성 세트를 적용할 수 있습니다.

절차

1. **templates/rhsm.yml** 이라는 환경 파일을 만들어 구성을 저장합니다.
2. 환경 파일에 구성을 포함합니다. 예를 들면 다음과 같습니다.

```
resource_registry:
  OS::TripleO::Services::Rhsm: /usr/share/openstack-tripleo-heat-
templates/deployment/rhsm/rhsm-baremetal-ansible.yaml
parameter_defaults:
  ControllerParameters:
    RhsmVars:
      rhsm_repos:
        - rhel-8-for-x86_64-baseos-eus-rpms
        - rhel-8-for-x86_64-appstream-eus-rpms
        - rhel-8-for-x86_64-highavailability-eus-rpms
        - ansible-2.9-for-rhel-8-x86_64-rpms
        - advanced-virt-for-rhel-8-x86_64-rpms
        - openstack-16.1-for-rhel-8-x86_64-rpms
        - fast-datapath-for-rhel-8-x86_64-rpms
      rhsm_username: "myusername"
      rhsm_password: "p@55w0rd!"
      rhsm_org_id: "1234567"
      rhsm_pool_ids: "55d251f1490556f3e75aa37e89e10ce5"
      rhsm_method: "portal"
      rhsm_release: 8.2
  ComputeParameters:
    RhsmVars:
      rhsm_repos:
        - rhel-8-for-x86_64-baseos-eus-rpms
        - rhel-8-for-x86_64-appstream-eus-rpms
        - rhel-8-for-x86_64-highavailability-eus-rpms
        - ansible-2.9-for-rhel-8-x86_64-rpms
```

```
      - advanced-virt-for-rhel-8-x86_64-rpms
      - openstack-16.1-for-rhel-8-x86_64-rpms
      - fast-datapath-for-rhel-8-x86_64-rpms
    rhsm_username: "myusername"
    rhsm_password: "p@55w0rd!"
    rhsm_org_id: "1234567"
    rhsm_pool_ids: "55d251f1490556f3e75aa37e89e10ce5"
    rhsm_method: "portal"
    rhsm_release: 8.2
CephStorageParameters:
  RhsmVars:
    rhsm_repos:
      - rhel-8-for-x86_64-baseos-rpms
      - rhel-8-for-x86_64-appstream-rpms
      - rhel-8-for-x86_64-highavailability-rpms
      - ansible-2.9-for-rhel-8-x86_64-rpms
      - openstack-16.1-deployment-tools-for-rhel-8-x86_64-rpms
    rhsm_username: "myusername"
    rhsm_password: "p@55w0rd!"
    rhsm_org_id: "1234567"
    rhsm_pool_ids: "68790a7aa2dc9dc50a9bc39fabc55e0d"
    rhsm_method: "portal"
    rhsm_release: 8.2
```

resource_registry 는 각 역할에서 사용할 수 있는 **OS::TripleO::Services::Rhsm** 리소스와 **rhsm** 구성 가능 서비스를 연결합니다.

ControllerParameters,**ComputeParameters** 및 **CephStorageParameters** 매개 변수는 각각 별도의 **RhsmVars** 매개변수를 사용하여 서브스크립션 세부 정보를 해당 역할에 전달합니다.

참고

CephStorageParameter s 매개변수 내에서 RhsmVar s 매개변수를 설정하여 Red Hat Ceph Storage 서브스크립션 및 Ceph Storage 관련 리포지토리를 사용합니다. **rhsm_repos** 매개변수에 컨트롤러 및 컴퓨팅 노드에 필요한 EUS(Extended Update Support) 리포지토리 대신 표준 Red Hat Enterprise Linux 리포지토리가 포함되어 있는지 확인합니다.

3. 환경 파일을 저장합니다.

11장. 오버클라우드를 RED HAT SATELLITE SERVER에 등록 업데이트

Red Hat OpenStack Platform 16.1에서는 Ansible 기반 방법을 사용하여 오버클라우드 노드를 Red Hat Satellite Server 6에 등록합니다.

11.1. RHSM(RED HAT SUBSCRIPTION MANAGER) 구성 가능 서비스

rhsm 구성 가능 서비스를 사용하여 Ansible을 통해 오버클라우드 노드를 등록할 수 있습니다. 기본 **roles_data** 파일의 각 역할에는 기본적으로 비활성화된 **OS::TripleO::Services::Rhsm** 리소스가 포함되어 있습니다. 서비스를 활성화하려면 리소스를 **rhsm** 구성 가능 서비스 파일에 등록합니다.

```
resource_registry:
  OS::TripleO::Services::Rhsm: /usr/share/openstack-tripleo-heat-templates/deployment/rhsm/rhsm-baremetal-ansible.yaml
```

The **rhsm** 구성 가능 서비스는 **RhsmVars** 매개변수를 사용할 수 있으며, 등록과 관련된 여러 하위 매개변수를 정의하는 데 사용할 수 있습니다.

```
parameter_defaults:
  RhsmVars:
    rhsm_repos:
      - rhel-8-for-x86_64-baseos-eus-rpms
      - rhel-8-for-x86_64-appstream-eus-rpms
      - rhel-8-for-x86_64-highavailability-eus-rpms
      …
    rhsm_username: "myusername"
    rhsm_password: "p@55w0rd!"
    rhsm_org_id: "1234567"
    rhsm_release: 8.2
```

역할별 매개변수(예: **ControllerParameter s)와 함께 RhsmVar s** 매개변수를 사용하여 다른 노드 유형에 특정 리포지토리를 활성화할 때 유연성을 제공할 수도 있습니다.

11.2. RHSMVARS 하위 매개변수

구성 가능 서비스를 구성할 때 다음 하위 매개 변수를 **RhsmVars** 매개변수 **의** 일부로 사용합니다. 사용 가능한 Ansible 매개변수에 대한 자세한 내용은 역할 설명서 를 참조하십시오.

rhsm	설명
rhsm_method	등록 방법을 선택합니다. **portal**,**satellite** 또는 **disable** 중 하나입니다.
rhsm_org_id	등록에 사용하려는 조직입니다. 이 ID를 찾으려면 언더클라우드 노드에서 **sudo subscription-manager orgs** 를 실행합니다. 프롬프트에 Red Hat 자격 증명을 입력하고 결과로 생성된 **Key** (키) 값을 사용합니다. 조직 ID에 대한 자세한 내용은 Red Hat Subscription Management Organization ID 이해를 참조하십시오.

rhsm	설명
rhsm_pool_ids	사용하려는 서브스크립션 풀 ID입니다. 서브스크립션을 자동 첨부하지 않으려면 이 매개변수를 사용합니다. 이 ID를 찾으려면 언더클라우드 노드에서 **sudo subscription-manager list --available --all --matches="Red Hat OpenStack"** 을 실행하고 결과 **풀 ID** 값을 사용합니다.
rhsm_activation_key	등록에 사용할 활성화 키입니다.
rhsm_autosubscribe	이 매개변수를 사용하여 호환 가능한 서브스크립션을 이 시스템에 자동으로 첨부합니다. 이 기능을 활성화하려면 값을 **true** 로 설정합니다.
rhsm_baseurl	콘텐츠를 가져오는 기본 URL입니다. 기본 URL은 Red Hat Content Delivery Network입니다. Satellite 서버를 사용하는 경우 이 값을 Satellite 서버 콘텐츠 리포지토리의 기본 URL로 변경합니다.
rhsm_server_hostname	등록을 위한 서브스크립션 관리 서비스의 호스트 이름입니다. 기본값은 Red Hat 서브스크립션 관리 호스트 이름입니다. Satellite 서버를 사용하는 경우 이 값을 Satellite 서버 호스트 이름으로 변경합니다.
rhsm_repos	활성화하려는 리포지토리 목록입니다.
rhsm_username	등록할 사용자 이름입니다. 가능한 경우 등록에 활성화 키를 사용합니다.
rhsm_password	등록할 암호입니다. 가능한 경우 등록에 활성화 키를 사용합니다.
rhsm_release	리포지토리 고정을 위한 Red Hat Enterprise Linux 릴리스. Red Hat OpenStack Platform의 경우 8.2로 설정됩니다.
rhsm_rhsm_proxy_hostname	HTTP 프록시의 호스트 이름입니다. 예: **proxy.example.com**
rhsm_rhsm_proxy_port	HTTP 프록시 통신용 포트입니다. 예를 들면 다음과 같습니다. **8080**.
rhsm_rhsm_proxy_user	HTTP 프록시에 액세스할 사용자 이름입니다.
rhsm_rhsm_proxy_password	HTTP 프록시에 액세스할 암호입니다.

중요

rhsm_ method가 portal 로 설정된 경우에만 use rhsm_activation_key 및rhsm_ repos 를 함께 사용할 수 있습니다. If **rhsm_method** 가 'satellite'로 설정되면 either **rhsm_activation_key or rhsm_ repos** 만 사용할 수 있습니다.

11.3. RHSM 구성 가능 서비스로 전환

이전 **rhel-registration** 메서드는 Overcloud 등록을 처리하는 bash 스크립트를 실행합니다. 이 메서드의 스크립트 및 환경 파일은 **/usr/share/openstack-tripleo-heat-templates/extraconfig/pre_deploy/rhel-registration/** 의 코어 heat 템플릿 컬렉션에 있습니다.

rhel-registration 방법에서 구성 가능 서비스로 전환하려면 **다음** 단계를 완료합니다.

절차

1. **rhel-registration** 환경 파일을 향후 배포 작업에서 제외합니다. 대부분의 경우 다음 파일을 제외합니다.
   - **rhel-registration/environment-rhel-registration.yaml**
   - **rhel-registration/rhel-registration-resource-registry.yaml**
2. 사용자 지정 **roles_data** 파일을 사용하는 경우 **roles_data** 파일의 각 역할에 **OS::TripleO::Services::Rhsm** 구성 가능 서비스가 포함되어 있는지 확인합니다. 예를 들면 다음과 같습니다.

```
- name: Controller
  description: |
    Controller role that has all the controller services loaded and handles
    Database, Messaging and Network functions.
  CountDefault: 1
  ...
  ServicesDefault:
    ...
    - OS::TripleO::Services::Rhsm
    ...
```

3. 향후 배포 작업에 환경 파일 for **rhsm** composable 서비스 매개변수를 추가합니다.

이 메서드는 **rhel-registration** 매개변수를 **rhsm** 서비스 매개변수로 교체하고 서비스를 활성화하는 heat 리소스를 변경합니다.

```
resource_registry:
  OS::TripleO::NodeExtraConfig: rhel-registration.yaml
```

다음으로 변경합니다.

```
resource_registry:
  OS::TripleO::Services::Rhsm: /usr/share/openstack-tripleo-heat-templates/deployment/rhsm/rhsm-baremetal-ansible.yaml
```

배포를 통해 **/usr/share/openstack-tripleo-heat-templates/environments/rhsm.yaml** 환경 파일을 배포하여 서비스를 활성화할 수도 있습니다.

11.4. RHEL-REGISTRATION TO RHSM MAPPINGS

rhel-registration 방법에서 the **rhsm** 메서드로 세부 정보를 전환하는 데 도움이 되도록 다음 표를 사용하여 매개변수와 값을 매핑합니다.

rhel-registration	rhsm / RhsmVars
rhel_reg_method	**rhsm_method**
rhel_reg_org	**rhsm_org_id**
rhel_reg_pool_id	**rhsm_pool_ids**
rhel_reg_activation_key	**rhsm_activation_key**
rhel_reg_auto_attach	**rhsm_autosubscribe**
rhel_reg_sat_url	**rhsm_satellite_url**
rhel_reg_repos	**rhsm_repos**
rhel_reg_user	**rhsm_username**
rhel_reg_password	**rhsm_password**
rhel_reg_release	**rhsm_release**
rhel_reg_http_proxy_host	**rhsm_rhsm_proxy_hostname**
rhel_reg_http_proxy_port	**rhsm_rhsm_proxy_port**
rhel_reg_http_proxy_username	**rhsm_rhsm_proxy_user**
rhel_reg_http_proxy_password	**rhsm_rhsm_proxy_password**

11.5. 오버클라우드를 RED HAT SATELLITE SERVER에 등록

Red Hat 고객 포털 대신 Red Hat Satellite에 노드를 등록하도록 **rhsm** 구성 가능 서비스를 활성화하고 구성하는 환경 파일을 생성합니다.

절차

1. **templates/rhsm.yml** 이라는 환경 파일을 만들어 구성을 저장합니다.
2. 환경 파일에 구성을 포함합니다. 예를 들면 다음과 같습니다.

```
resource_registry:
  OS::TripleO::Services::Rhsm: /usr/share/openstack-tripleo-heat-
templates/deployment/rhsm/rhsm-baremetal-ansible.yaml
parameter_defaults:
  RhsmVars:
    rhsm_activation_key: "myactivationkey"
    rhsm_method: "satellite"
```

```
    rhsm_org_id: "ACME"
    rhsm_server_hostname: "satellite.example.com"
    rhsm_baseurl: "https://satellite.example.com/pulp/repos"
    rhsm_release: 8.2
```

resource_registry 는 각 역할에서 사용할 수 있는 **OS::TripleO::Services::Rhsm** 리소스와 **rhsm** 구성 가능 서비스를 연결합니다.

RhsmVars 변수는 Red Hat 등록을 구성하기 위해 매개 변수를 Ansible에 전달합니다.

3. 환경 파일을 저장합니다.

11.6. SATELLITE SERVER 사용을 위한 LEAPP 준비

Satellite Server 6을 사용하여 RPM 콘텐츠를 호스팅하는 경우 다음 준비 단계를 완료하여 Satellite를 사용하여 Leapp 업그레이드를 성공적으로 수행합니다.

사전 요구 사항

- Red Hat OpenStack Platform 16.1 및 Red Hat Enterprise Linux 8.2의 리포지토리에 연결된 Satellite Server 활성화 키를 만듭니다.
- 오버클라우드 노드에 대한 Ansible 인벤토리 파일을 생성합니다.
- Satellite Server에서 Red Hat OpenStack Platform 16.1 업그레이드에 대한 콘텐츠 뷰를 생성 및 승격하고 업그레이드에 필요한 리포지토리를 포함합니다. 자세한 내용은 Red Hat Satellite Server 6 고려 사항을 참조하십시오.
- Ceph 서브스크립션을 사용하고 Ceph 스토리지 노드에 **overcloud-minimal** 이미지를 사용하도록 director가 구성된 경우 Content View를 생성하고 다음 RHEL(Red Hat Enterprise Linux) 8.2 리포지토리를 추가해야 합니다.
  - Red Hat Enterprise Linux 8 for x86_64 - AppStream(RPM)

    ```
    rhel-8-for-x86_64-appstream-rpms
    x86_64 8.2
    ```

  - Red Hat Enterprise Linux 8 for x86_64 - BaseOS(RPM)

    ```
    rhel-8-for-x86_64-baseos-rpms
    x86_64 8.2
    ```

    자세한 내용은 Red Hat *Satellite Content* Management Guide의 Red Hat Content and Managing Content Views 를 참조하십시오.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. **playbook-satellite.yaml** 이라는 파일을 생성하고 파일에 다음 콘텐츠를 붙여넣습니다.

```
- name: Pre-install leapp
  hosts: overcloud
  become: yes
  tasks:
```

```
- name: Pre-install leapp
  yum:
    name:
      - leapp
      - leapp-repository
    state: installed
- name: Remove katello-host-tools-fact-plugin
  yum:
    name:
      - katello-host-tools-fact-plugin
    state: removed
- name: Register system
  redhat_subscription:
    activationkey: "osp16-key"
    org_id: "ACME"
    server_hostname: "satellite.example.com"
    rhsm_baseurl: "https://satellite.example.com/pulp/repos"
    force_register: True
```

Satellite 서버에 맞게 **redhat_subscription** 변수를 수정합니다.

3. 플레이북을 실행합니다.

```
$ ansible-playbook -i ~/inventory.yaml playbook-satellite.yaml
```

12장. DIRECTOR 배포 CEPH STORAGE 업그레이드 준비

배포 시 director가 배포된 Red Hat Ceph Storage 클러스터를 사용하는 경우 이 섹션에 포함된 절차를 완료해야 합니다.

중요

RHOSP 16.1은 RHEL 8.2에서 지원됩니다. 그러나 Ceph Storage 역할 업데이트에 매핑된 호스트는 최신 주요 RHEL 릴리스로 업데이트합니다. 자세한 내용은 Red Hat Ceph Storage를 참조하십시오. 지원되는 구성.

참고

외부 Ceph 배포를 사용하여 업그레이드하는 경우 이 섹션에 포함된 절차를 건너뛰고 13장. *외부 Ceph 배포를 사용하여 업그레이드 준비* 계속 진행해야 합니다.

업그레이드 프로세스에서는 Red Hat OpenStack Platform 16.1로 업그레이드하는 동안 Red Hat Ceph Storage 3 컨테이너화된 서비스를 사용합니다. Red Hat OpenStack Platform 16.1 업그레이드를 완료한 후 Ceph Storage 서비스를 Red Hat Ceph Storage 4로 업그레이드합니다.

Red Hat OpenStack Platform 16.1 업그레이드 및 Ceph Storage 서비스가 Red Hat Ceph Storage 4로 업그레이드될 때까지 Shared File Systems 서비스(manila)로 새 공유를 프로비저닝할 수 없습니다.

12.1. 높은 수준의 CEPH STORAGE 노드 업그레이드 프로세스 이해

오버클라우드 업그레이드 프로세스 중에 director가 배포한 Ceph Storage 노드는 Red Hat Ceph Storage 3 컨테이너를 계속 사용합니다. 업그레이드 프로세스 중에 Ceph Storage 노드 및 서비스에 미치는 영향을 이해하려면 Ceph Storage 업그레이드 프로세스의 각 측면에 대해 다음 요약을 읽어보십시오.

Ceph-ansible

Ceph-ansible 은 director가 Ceph Storage 서비스를 설치, 유지 관리 및 업그레이드하는 데 사용하는 역할 및 플레이북 컬렉션입니다. 언더클라우드를 업그레이드할 때 Red Hat Enterprise Linux 8.2로 전환한 후 **ceph-ansible** 이 최신 버전 3 컬렉션에 남아 있는지 확인하는 특정 명령을 실행했습니다. **ceph-ansible** 버전 3은 오버클라우드 업그레이드 기간 동안 컨테이너화된 Ceph Storage 서비스를 버전 3에 유지합니다. 업그레이드가 완료되면 Red Hat Ceph Storage에서 **RHEL 8용 Red Hat Ceph Storage Tools 4** 리포지토리를 업데이트하고 **ceph-ansible** 을 버전 4로 업데이트할 수 있습니다.

Podman으로 마이그레이션

오버클라우드 업그레이드 중에 Docker 대신 Podman을 사용하도록 Ceph Storage 컨테이너화된 서비스를 제어하는 **systemd** 서비스를 변경하려면 **openstack overcloud external-upgrade run --tags ceph_systemd** 명령을 실행해야 합니다. Ceph Storage 컨테이너화된 서비스가 포함된 노드에서 운영 체제 업그레이드를 수행하기 전에 이 명령을 실행합니다.

노드에서 Podman을 사용하도록 **systemd** 서비스를 변경한 후 운영 체제 업그레이드 및 OpenStack Platform 서비스 업그레이드를 수행합니다. OpenStack Platform 서비스를 업그레이드한 후 해당 노드의 Ceph Storage 컨테이너가 다시 실행됩니다.

Ceph Storage 운영 체제 업그레이드

일반적으로 오버클라우드 노드에서 수행하는 것과 동일한 워크플로를 Ceph Storage 노드에서 따릅니다. Ceph Storage 노드에 대해 **openstack overcloud upgrade run --tags system_upgrade** 명령을 실행하면 director가 Ceph Storage 노드에서 Leapp을 실행하고 운영 체제를 Red Hat Enterprise Linux 8.2로 업

그레이드합니다. 그런 다음 다음 컨테이너를 실행하는 Ceph Storage 노드에 대해 태그되지 않은 **openstack overcloud upgrade run** 명령을 실행합니다.

- Red Hat Ceph Storage 3 컨테이너 서비스
- Red Hat OpenStack Platform 16.1 컨테이너화된 서비스

Red Hat Ceph Storage 4로 업그레이드

Leapp 업그레이드 및 Red Hat OpenStack Platform 업그레이드를 완료한 후에도 Ceph Storage 컨테이너화된 서비스에서 버전 3 컨테이너를 사용합니다. 이때 **ceph-ansible** 을 버전 4로 업그레이드한 다음 모든 노드에서 모든 Red Hat Ceph Storage 서비스를 버전 4로 업그레이드하는 **openstack overcloud external-upgrade run --tags ceph** 명령을 실행해야 합니다.

Ceph Storage 워크플로 요약

다음 목록은 Red Hat Ceph Storage 업그레이드를 위한 높은 수준의 워크플로입니다. 이 워크플로는 일반적인 Red Hat OpenStack Platform 워크플로우에 통합되며 이 워크플로에서 작업을 수행하기 위해 언더클라우드에서 업그레이드 프레임워크 명령을 실행합니다.

1. 언더클라우드 업그레이드를 하지만 **ceph-ansible**버전 3 유지
2. 오버클라우드 업그레이드 시작
3. Ceph Storage 컨테이너화된 서비스를 호스팅하는 각 노드에 대해 다음 작업을 수행합니다.
    a. Ceph Storage 컨테이너화된 서비스를 Podman으로 마이그레이션
    b. 운영 체제 업그레이드
    c. Ceph Storage 버전 3 컨테이너화된 서비스를 다시 시작하는 OpenStack Platform 서비스를 업그레이드합니다.
4. 오버클라우드 업그레이드 완료
5. 언더클라우드의 **ceph-ansible** 을 버전 4로 업그레이드
6. 오버클라우드에서 Red Hat Ceph Storage 4로 업그레이드

참고

이 목록은 전체 Red Hat OpenStack Platform 16.1 업그레이드 프로세스의 모든 단계를 포착하지는 않지만 Red Hat Ceph Storage와 관련된 측면에만 중점을 두고 업그레이드 프로세스 중에 Ceph Storage 서비스에 발생하는 사항을 설명합니다.

12.2. CEPH-ANIBLE 버전 확인

언더클라우드 업그레이드 중에 Ceph Storage 3 버전의 **ceph-ansible** 패키지가 유지되었습니다. 이를 통해 Ceph Storage 노드에서 Ceph Storage 3 컨테이너의 호환성을 유지할 수 있습니다. 이 패키지가 언더클라우드에 남아 있는지 확인합니다.

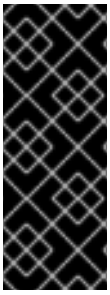
절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. **dnf** 명령을 실행하여 **ceph-ansible** 패키지 버전을 확인합니다.

```
$ sudo dnf info ceph-ansible
```

명령 출력에는 **ceph-ansible** 패키지의 버전 3이 표시됩니다.

```
Installed Packages
Name         : ceph-ansible
Version      : 3.xx.xx.xx
...
```



중요

ceph-ansible 패키지가 없거나 버전 3 패키지가 없는 경우 Red Hat Package Browser 에서 최신 버전 3 패키지를 다운로드하여 언더클라우드에 패키지를 수동으로 설치합니다. **ceph-ansible** 버전 3 패키지는 Red Hat Enterprise Linux 7 리포지토리에서만 사용할 수 있으며 Red Hat Enterprise Linux 8 리포지토리에서는 사용할 수 없습니다. **ceph-ansible** 버전 3은 업그레이드를 위해 Red Hat OpenStack Platform 프레임워크의 컨텍스트 외부의 Red Hat Enterprise Linux 8에서 지원되지 않습니다.

12.3. CEPH-ANSIBLE 리포지토리 설정

Red Hat OpenStack Platform 16.1 검증 프레임워크는 director가 오버클라우드를 Red Hat Ceph Storage 4로 업그레이드하기 전에 **ceph-anible** 이 올바르게 설치되었는지 테스트합니다. 프레임워크는 **CephAnsibleRepo** 매개변수를 사용하여 올바른 리포지토리에서 **ceph-anible** 을 설치했는지 확인합니다. **openstack overcloud upgrade prepare** 명령을 실행한 후 director는 테스트를 비활성화하고 이 테스트는 Red Hat OpenStack Platform 16.1 오버클라우드 업그레이드 기간 동안 비활성화된 상태로 유지됩니다. **openstack overcloud upgrade converge** 명령을 실행한 후 director가 이 테스트를 다시 활성화합니다. 그러나 이 검증을 준비하려면 **CephAnsibleRepo** 매개변수를 **RHEL 8 리포지토리용 Red Hat Ceph Storage Tools 4** 로 설정해야 합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. 오버클라우드 Ceph Storage 구성이 포함된 환경 파일을 편집합니다. 일반적으로 이 파일의 이름은 **ceph-config.yaml** 로 지정되며 **templates** 디렉터리에서 찾을 수 있습니다.

   ```
   $ vi /home/stack/templates/ceph-config.yaml
   ```

3. **CephAnsibleRepo** 매개변수를 **parameter_defaults** 섹션에 추가합니다.

   ```
   parameter_defaults:
     ...
     CephAnsibleRepo: rhceph-4-tools-for-rhel-8-x86_64-rpms
     ...
   ```

   CephAnsibleRepo 는 **ceph-anible** 을 포함하는 리포지토리를 설정합니다. 검증 프레임워크에서는 이 매개변수를 사용하여 언더클라우드에 **ceph-anible** 이 설치되어 있는지 확인합니다.

4. **ceph-config.yaml** 파일을 저장합니다.

12.4. 업그레이드하기 전에 CEPH 클러스터 상태 확인

오버클라우드 업그레이드를 진행하려면 Ceph 클러스터가 예상대로 작동하는지 확인해야 합니다.

절차

1. **ceph-mon** 서비스를 실행 중인 노드에 로그인합니다. 이 노드는 일반적으로 컨트롤러 노드 또는 독립 실행형 Ceph 모니터 노드입니다.

2. Ceph 클러스터의 상태를 보려면 다음 명령을 입력합니다.

```
$ docker exec ceph-mon-$HOSTNAME ceph -s
```

3. 클러스터 상태가 **HEALTH_OK** 이고 모든 OSD가 **작동** 중인지 확인합니다.

13장. 외부 CEPH 배포를 사용하여 업그레이드 준비

외부 Ceph 배포를 사용하여 업그레이드하는 경우 이 섹션에 포함된 절차를 완료해야 합니다.

참고

배포에서 외부 Ceph Storage 클러스터를 사용하지 않는 경우 이 섹션에 포함된 절차를 건너뛰고 다음 섹션을 계속 진행해야 합니다.

13.1. CEPH-ANIBLE 설치

외부 Ceph 배포를 사용하여 업그레이드하는 경우 다음 절차를 완료해야 합니다.

Red Hat OpenStack Platform에서 Ceph Storage를 사용하는 경우 **ceph-ansible** 패키지가 필요합니다.

절차

1. 다음과 같이 Ceph 툴 리포지토리를 활성화합니다.

```
[stack@director ~]$ sudo subscription-manager repos --enable=rhceph-4-tools-for-rhel-8-x86_64-rpms
```

2. **ceph-ansible** 패키지를 설치합니다.

```
[stack@director ~]$ sudo dnf install -y ceph-ansible
```

13.2. CEPH-ANSIBLE 리포지토리 설정

Red Hat OpenStack Platform 16.1 검증 프레임워크는 director가 오버클라우드를 Red Hat Ceph Storage 4로 업그레이드하기 전에 **ceph-anible** 이 올바르게 설치되었는지 테스트합니다. 프레임워크는 **CephAnsibleRepo** 매개변수를 사용하여 올바른 리포지토리에서 **ceph-anible** 을 설치했는지 확인합니다. **openstack overcloud upgrade prepare** 명령을 실행한 후 director는 테스트를 비활성화하고 이 테스트는 Red Hat OpenStack Platform 16.1 오버클라우드 업그레이드 기간 동안 비활성화된 상태로 유지됩니다. **openstack overcloud upgrade converge** 명령을 실행한 후 director가 이 테스트를 다시 활성화합니다. 그러나 이 검증을 준비하려면 **CephAnsibleRepo** 매개변수를 **RHEL 8 리포지토리용 Red Hat Ceph Storage Tools 4** 로 설정해야 합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.

2. 오버클라우드 Ceph Storage 구성이 포함된 환경 파일을 편집합니다. 일반적으로 이 파일의 이름은 **ceph-config.yaml** 로 지정되며 **templates** 디렉터리에서 찾을 수 있습니다.

```
$ vi /home/stack/templates/ceph-config.yaml
```

3. **CephAnsibleRepo** 매개변수를 **parameter_defaults** 섹션에 추가합니다.

```
parameter_defaults:
  ...
  CephAnsibleRepo: rhceph-4-tools-for-rhel-8-x86_64-rpms
  ...
```

CephAnsibleRepo 는 **ceph-anible** 을 포함하는 리포지토리를 설정합니다. 검증 프레임워크에서는 이 매개변수를 사용하여 언더클라우드에 **ceph-anible** 이 설치되어 있는지 확인합니다.

4. **ceph-config.yaml** 파일을 저장합니다.

14장. 네트워크 구성 업데이트

오버클라우드 업그레이드를 준비하려면 일부 네트워크 구성을 완료해야 합니다.

14.1. 네트워크 인터페이스 템플릿 업데이트

Red Hat OpenStack Platform에는 누락된 매개 변수를 NIC 템플릿 파일에 자동으로 추가하는 스크립트가 포함되어 있습니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

3. 언더클라우드에서 **update-nic-templates.sh** 라는 파일을 생성하고 파일에 다음 내용을 포함합니다.

```
#!/bin/bash
STACK_NAME="overcloud"
ROLES_DATA="/usr/share/openstack-tripleo-heat-templates/roles_data.yaml"
NETWORK_DATA="/usr/share/openstack-tripleo-heat-templates/network_data.yaml"
NIC_CONFIG_LINES=$(openstack stack environment show $STACK_NAME | grep
"::Net::SoftwareConfig" | sed -E 's/ *OS::TripleO::// ; s/::Net::SoftwareConfig:// ; s/ http.*user-
files/ /')
echo "$NIC_CONFIG_LINES" | while read LINE; do
    ROLE=$(echo "$LINE" | awk '{print $1;}')
    NIC_CONFIG=$(echo "$LINE" | awk '{print $2;}')

    if [ -f "$NIC_CONFIG" ]; then
        echo "Updating template for $ROLE role."
        python3 /usr/share/openstack-tripleo-heat-templates/tools/merge-new-params-nic-
config-script.py \
            --tht-dir /usr/share/openstack-tripleo-heat-templates \
            --roles-data $ROLES_DATA \
            --network-data $NETWORK_DATA \
            --role-name "$ROLE" \
            --discard-comments yes \
            --template "$NIC_CONFIG"
    else
        echo "No NIC template detected for $ROLE role. Skipping $ROLE role."
    fi
done
```

- 사용자 지정 오버클라우드 이름을 사용하는 경우 **STACK_NAME** 변수를 오버클라우드 이름으로 설정합니다. 오버클라우드 스택의 기본 이름은 **overcloud** 입니다.
- 사용자 지정 **roles_data** 파일을 사용하는 경우 **ROLES_DATA** 변수를 사용자 지정 파일의 위치로 설정합니다. 기본 **roles_data** 파일을 사용하는 경우 변수를 **/usr/share/openstack-tripleo-heat-templates/roles_data.yaml로 둡니다.**

- 사용자 지정 **network_data** 파일을 사용하는 경우 **NETWORK_DATA** 변수를 사용자 지정 파일의 위치로 설정합니다. 기본 **network_data** 파일을 사용하는 경우 변수를 **/usr/share/openstack-tripleo-heat-templates/network_data.yaml로 둡니다.**

- **/usr/share/openstack-tripleo-heat-templates/tools/merge-new-params-nic-config-script.py -h** 를 실행하여 스크립트에 추가할 옵션 목록을 확인합니다.

4. 스크립트에 실행 가능 권한을 추가합니다.

```
$ chmod +x update-nic-templates.sh
```

5. 선택 사항: RHOSP 환경에 스파인-리프형 네트워크 토폴로지를 사용하는 경우 **roles_data.yaml** 파일을 확인하고 배포에 NIC 템플릿에 올바른 역할 이름을 사용하는지 확인합니다. 이 스크립트는 **roles_data.yaml 파일에서 deprecated_nic_config_ name** 매개 변수 값을 사용합니다.

6. 스크립트를 실행합니다.

```
$ ./update-nic-templates.sh
```

이 스크립트는 각 사용자 지정 NIC 템플릿의 사본을 저장하고 누락된 매개 변수로 각 템플릿을 업데이트합니다. 또한 스크립트는 사용자 지정 템플릿이 없는 모든 역할을 건너뜁니다.

```
No NIC template detected for BlockStorage role. Skipping BlockStorage role.
Updating template for CephStorage role.
The original template was saved as: /home/stack/templates/custom-nics/ceph-storage.yaml.20200903144835
The update template was saved as: /home/stack/templates/custom-nics/ceph-storage.yaml
Updating template for Compute role.
The original template was saved as: /home/stack/templates/custom-nics/compute.yaml.20200903144838
The update template was saved as: /home/stack/templates/custom-nics/compute.yaml
Updating template for Controller role.
The original template was saved as: /home/stack/templates/custom-nics/controller.yaml.20200903144841
The update template was saved as: /home/stack/templates/custom-nics/controller.yaml
No NIC template detected for ObjectStorage role. Skipping ObjectStorage role.
```

14.2. 업그레이드 중 OPEN VSWITCH 호환성 유지

Red Hat OpenStack Platform 13에서는neutron(Open vSwitch)을 OpenStack Networking(neutron)의 기본 ML2 백엔드로 사용합니다. 최신 버전의 Red Hat OpenStack Platform에서는 OVS 기능을 기반으로 확장되는 OVN(Open Virtual Network)을 사용합니다. 그러나 안정적인 업그레이드를 보장하려면 업그레이드 기간 동안 OVS 기능을 유지 관리한 다음 업그레이드를 완료한 후 OVN으로 마이그레이션해야 합니다.

업그레이드 중에 OVS 호환성을 유지하려면 환경 파일 컬렉션의 일부로 다음 환경 파일을 포함합니다.

- **/usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml**

  참고

  neutron-ovs.yaml 환경 파일을 포함하는 경우 **neutron-ovs-dvr.yaml** 환경 파일이 환경 파일 컬렉션에 포함되어 있는지 확인합니다. 업그레이드 중에 오류가 발생하지 않도록 **neutron-ovs -dvr.yaml 파일 앞에 neutron-ovs.yaml 환경** 파일을 포함해야 합니다.

OVN으로 마이그레이션을 완료해야 이 파일을 배포의 일부로 처리합니다. 모든 오버클라우드 업그레이드 및 배포 명령을 사용하여 파일을 포함합니다.

- **OpenStack overcloud 업그레이드 준비**
- **OpenStack 오버클라우드 업그레이드 통합**
- **OpenStack overcloud deploy**
- **OpenStack overcloud 업데이트 준비**
- **OpenStack 오버클라우드 업데이트 통합**
- 환경 파일을 사용하는 기타 명령.

OVS 호환성 문제 해결

neutron-ovs.yaml 파일에 정의된 매개변수가 neutron- ovs-dvr.yaml 에 정의된 매개변수를 덮어쓰므로 업그레이드 프로세스가 실패하는 경우 이러한 파일을 포함하는 순서를 변경하고 openstack overcloud upgrade **prepare 및 openstack overcloud upgrade** 을 다시 실행하고 영향을 받는 노드에서 **openstack overcloud upgrade을 다시 실행합니다**. 영향을 받는 노드 중 하나가 컴퓨팅 노드인 경우 해당 노드에서 **openstack-neutron*** 패키지를 제거합니다.

14.3. 업그레이드 중에 구성 가능한 네트워크 호환성 유지

Red Hat OpenStack Platform 16.1의 **network_data** 파일 형식에는 네트워크 내의 추가 서브넷 및 경로를 정의하는 데 사용할 수 있는 새로운 섹션이 포함되어 있습니다. 그러나 사용자 지정 **network_data** 파일을 사용하는 경우 Red Hat OpenStack Platform 13의 **network_data** 파일 형식을 계속 사용할 수 있습니다.

- Red Hat OpenStack Platform 13에서 16.1로 업그레이드하는 경우 업그레이드 중 또는 이후에 Red Hat OpenStack Platform 13 **network_data** 파일 형식을 사용하십시오. Red Hat OpenStack Platform 13 구성 가능 네트워크 구문에 대한 자세한 내용은 사용자 지정 구성 가능 네트워크를 참조하십시오.
- Red Hat OpenStack Platform 16.1에서 새 오버클라우드를 생성하는 경우 Red Hat OpenStack Platform 16.1 **network_data** 파일 형식을 사용합니다. Red Hat OpenStack Platform 16.1 구성 가능 네트워크 구문에 대한 자세한 내용은 사용자 정의 구성 가능 네트워크를 참조하십시오.

15장. NFV(네트워크 기능 가상화) 준비

NFV(네트워크 기능 가상화)를 사용하는 경우 오버클라우드 업그레이드를 위한 몇 가지 준비를 완료해야 합니다.

15.1. NFV(네트워크 기능 가상화) 환경 파일

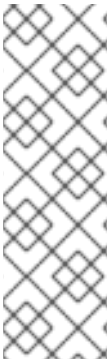
일반적인 NFV 기반 환경에서는 다음과 같은 서비스를 활성화할 수 있습니다.

- SR-IOV(Single-root input/output virtualization)
- DPDK(Data Plane Development Kit)

Red Hat OpenStack Platform 16.1로의 업그레이드를 수용하기 위해 해당 서비스에 대한 특정 재구성이 필요하지 않습니다. 그러나 NFV 기능을 활성화하는 환경 파일이 다음 요구 사항을 충족하는지 확인합니다.

- NFV 기능을 활성화하기 위한 기본 환경 파일은 Red Hat OpenStack Platform 16.1 **openstack-tripleo-heat-templates** 컬렉션의 environment **/services** 디렉터리에 있습니다. Red Hat OpenStack Platform 13 배포를 통해 **openstack-tripleo-heat-templates** 의 기본 NFV 환경 파일을 포함하는 경우 Red Hat OpenStack Platform 16.1의 각 기능에 대한 올바른 환경 파일 위치를 확인합니다.
  - OVS(Open vSwitch) 네트워킹 및 SR-IOV: **/usr/share/openstack-tripleo-heat-templates/environments/services/neutron-sriov.yaml**
  - OVS(Open vSwitch) 네트워킹 및 DPDK: **/usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs-dpdk.yaml**
- Red Hat OpenStack Platform 13에서 Red Hat OpenStack Platform 16.1로 업그레이드하는 동안 OVS 호환성을 유지하려면 **/usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml** 환경 파일을 포함해야 합니다. 환경 파일과 관련된 배포 및 업그레이드 명령을 실행하는 경우 **neutron-ovs.yaml** 파일 다음에 NFV 관련 환경 파일을 포함해야 합니다. 예를 들어 OVS 및 NFV 환경 파일을 사용하여 **openstack overcloud upgrade prepare** 를 실행하는 경우 파일을 다음 순서로 포함합니다.
- OVS 환경 파일
- SR-IOV 환경 파일
- DPDK 환경 파일

```
$ openstack overcloud upgrade prepare \
  ...
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-sriov.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs-dpdk.yaml \
  ...
```



참고

업그레이드하는 동안 RHOSP 13 Compute 노드가 **하이브리드** 상태에 있는 경우에만 RHOSP 13 및 RHOSP 16.1.x 컴퓨팅 노드 간에 인스턴스를 마이그레이션할 수 있습니다. 자세한 내용은 *Configuring the Compute Service for Instance Creation* 가이드의 마이그레이션 제약 조건을 참조하십시오.

NFV 워크로드에는 추가 마이그레이션 제한 조건이 있습니다. 업그레이드 중에 OVS-DPDK 컴퓨팅 노드에서 인스턴스를 실시간 마이그레이션할 수 없습니다. 또는 업그레이드 중에 OVS-DPDK 컴퓨팅 노드에서 인스턴스를 콜드 마이그레이션할 수 있습니다.

16장. 업그레이드 전 최종 검토

업그레이드를 시작하기 전에 모든 준비 단계의 최종 점검을 완료합니다.

16.1. 배포에 포함할 사용자 지정 파일

배포의 오버클라우드 노드가 전용 Object Storage(swift) 노드인 경우 기본 **roles_data.yaml** 파일을 복사하고 **ObjectStorage** 를 편집하여 **deprecated_server_resource_name을 제거해야 합니다. 'SwiftStorage'**. 그런 다음 **--roles-file** 옵션을 사용하여 파일을 **openstack overcloud upgrade prepare** 또는 **openstack overcloud upgrade converge** 명령에 전달합니다.

16.2. 배포에 포함할 새 환경 파일

일반 오버클라우드 환경 파일 외에도 RHOSP(Red Hat OpenStack Platform) 16.1로 쉽게 업그레이드할 수 있도록 새 환경 파일을 포함해야 합니다.

파일	참고
/home/stack/templates/upgrades-environment.yaml	이 파일에는 업그레이드와 관련된 매개변수가 포함되어 있습니다. 이 파일은 업그레이드 기간에만 필요합니다. **openstack overcloud upgrade converge를 실행한 후 이 파일을 폐기합니다**.
/home/stack/templates/rhsm.yaml	이 파일에는 오버클라우드에 대한 등록 및 서브스크립션 정보가 포함되어 있습니다. 이 파일은 시스템을 Red Hat 고객 포털 또는 Red Hat Satellite Server에 등록합니다.
/home/stack/containers-prepare-parameter.yaml	소스 및 준비 단계가 포함된 파일입니다. 이 파일은 언더클라우드 업그레이드에 사용하는 것과 같습니다.
/usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml	OpenStack Platform 16.1에서는 OVN(Open Virtual Network)을 기본 네트워킹 백엔드로 사용합니다. 그러나 OpenStack Platform 13에서는 OVS(Open vSwitch)를 사용했습니다. 업그레이드 중에 OVS 호환성을 유지하기 위해 이 파일을 포함합니다. OpenStack Platform 16.1 설명서에는 업그레이드 후 OVS에서 OVN으로 마이그레이션하는 방법이 포함되어 있습니다.

다음 명령을 실행할 때 환경 파일 목록의 끝에 해당 파일을 추가합니다.

- **OpenStack overcloud 업그레이드 준비**
- **OpenStack 오버클라우드 업그레이드 통합**
- **OpenStack overcloud deploy**

16.3. 배포에서 제거할 환경 파일

OpenStack Platform Red Hat OpenStack Platform 13과 관련된 환경 파일을 제거합니다.

- Red Hat OpenStack Platform 13 컨테이너 이미지 목록
- Red Hat OpenStack Platform 13 Customer Portal 또는 Satellite **rhel-registration** 스크립트

다음 명령을 실행할 때 포함하는 환경 파일 목록에서 해당 파일을 제거합니다.

- **OpenStack overcloud 업그레이드 준비**
- **OpenStack 오버클라우드 업그레이드 통합**
- **OpenStack overcloud deploy**

16.4. 업그레이드 체크리스트

다음 체크리스트를 사용하여 오버클라우드를 업그레이드할 준비 상태를 확인합니다.

항목	완료
작동 중인 Overcloud의 유효성을 검사합니다.	Y / N
오버클라우드 컨트롤 플레인의 ReaR(Relax-and- recovery) 백업을 수행합니다. 자세한 내용은 Red Hat OpenStack Platform 13 Undercloud 및 컨트롤 플레인 백업 및 복원을참조하십시오.	Y / N
언더클라우드 노드에서 실행되는 데이터베이스 백업을 생성했습니다. 자세한 내용은 *언더클라우드 및 컨트롤 플레인 노드 가이드의 Red Hat OpenStack Platform 16.1 백업 및 복원에서 언더클라우드 노드의* 데이터베이스 백업 생성 을 참조하십시오.	Y / N
다음을 포함하여 Leapp에 대한 모든 준비 구현: • LeApp 환경 파일 • 오버클라우드 노드에 복사된 LeApp 데이터 • 모든 오버클라우드 노드는 이제 예측 가능한 NIC 이름을 사용합니다. • 모든 오버클라우드 노드에 **PermitRootLogin** 이 설정되어 있습니다.	Y / N
등록 세부 정보를 Red Hat OpenStack Platform 16.1 리포지토리에 업데이트하고 Ansible 기반 방법을 사용하도록 환경 파일을 공동 구성했습니다.	Y / N
네트워크 구성 템플릿을 업데이트했습니다.	Y / N
Red Hat OpenStack Platform 16.1의 새 환경 파일로 환경 파일 목록을 업데이트했습니다.	Y / N
선택 사항: 배포에 전용 Object Storage(swift) 노드가 포함된 경우 다음을 수행합니다. **roles_data.yaml** 파일을 복사하고**deprecated_server_resource_name이 제거되었습니다. 'SwiftStorage'**, 파일을**openstack overcloud upgrade prepare** 또는 **openstack overcloud upgrade converge** 명령에 전달했습니다.	Y / N
이전 Red Hat 등록 및 컨테이너 이미지 위치 파일과 같은 Red Hat OpenStack Platform 13과 관련된 이전 환경 파일만 제거했습니다.	Y / N

17장. 업그레이드 명령 개요

업그레이드 프로세스에는 특정 프로세스 단계에서 실행되는 다양한 명령이 포함됩니다.

중요

이 섹션에는 각 명령에 대한 정보만 포함되어 있습니다. 이러한 명령을 특정 순서로 실행하고 오버클라우드와 관련된 옵션을 제공해야 합니다. 적절한 단계에서 이러한 명령을 실행할 지침이 수신될 때까지 기다립니다.

17.1. OPENSTACK OVERCLOUD 업그레이드 준비

이 명령은 오버클라우드 업그레이드를 위한 초기 준비 단계를 수행합니다. 여기에는 언더클라우드의 현재 오버클라우드 플랜을 새 OpenStack Platform 16.1 Overcloud 계획 및 업데이트된 환경 파일로 교체하는 작업이 포함됩니다. 이 명령은 **openstack overcloud deploy** 명령과 유사하게 작동하며 동일한 여러 옵션을 사용합니다.

17.2. OPENSTACK OVERCLOUD 업그레이드 실행

이 명령은 업그레이드 프로세스를 수행합니다. director는 새로운 OpenStack Platform 16.1 Overcloud 계획에 따라 일련의 Ansible 플레이북을 생성하고 전체 오버클라우드에서 빠른 전달 작업을 실행합니다. 여기에는 13에서 16.1로 각 OpenStack Platform 버전을 통해 업그레이드 프로세스를 실행하는 작업이 포함됩니다.

이 명령은 표준 업그레이드 프로세스 외에도 오버클라우드 노드에서 운영 체제의 Leapp 업그레이드를 수행할 수 있습니다. 이러한 작업을 **--tags** 옵션을 사용하여 실행합니다.

Leapp의 작업 태그 업그레이드

system_upgrade

system_upgrade_ **prepare, system_upgrade_ run 및 system_upgrade_ reboot** 의 작업을 결합하는 작업.

system_upgrade_prepare

Leapp을 사용하여 운영 체제 업그레이드를 준비하는 작업.

system_upgrade_run

Leapp을 실행하고 운영 체제를 업그레이드하는 작업.

system_upgrade_reboot

시스템을 재부팅하고 운영 체제 업그레이드를 완료하는 작업.

워크로드 마이그레이션을 위한 업그레이드 작업 태그

nova_hybrid_state

업그레이드 중에 워크로드 마이그레이션을 용이하게 하기 위해 컴퓨팅 노드에 임시 OpenStack Platform 16.1 컨테이너를 설정하는 작업입니다.

17.3. OPENSTACK OVERCLOUD EXTERNAL-UPGRADE RUN

이 명령은 표준 업그레이드 프로세스 외부에서 업그레이드 작업을 수행합니다. director는 새로운 OpenStack Platform 16.1 오버클라우드 계획에 따라 일련의 Ansible 플레이북을 생성하고 **--tags** 옵션을 사용하여 특정 작업을 실행합니다.

컨테이너 관리를 위한 외부 작업 태그

container_image_prepare

컨테이너 이미지를 언더클라우드 레지스트리로 가져와 Overcloud에서 사용할 이미지를 준비하는 작업입니다.

Ceph Storage 업그레이드를 위한 외부 작업 태그

- 배포 시 director를 사용하여 배포된 Red Hat Ceph Storage 클러스터를 사용하는 경우 다음 태그를 사용할 수 있습니다.

  Ceph

  ceph-anible 플레이북을 사용하여 Red Hat Ceph Storage를 설치하는 작업.

  ceph_systemd

  podman 관리를 사용하도록 Red Hat Ceph Storage systemd 장치 파일을 변환하는 작업.

- 외부 Ceph 배포를 사용하여 업그레이드하는 경우 **ceph 및 ceph _systemd** 태그를 사용하는 작업을 건너뛸 수 있습니다.

데이터베이스 전송을 위한 외부 작업 태그

system_upgrade_cleanup

system_upgrade_transfer_data 작업과 관련된 스토리지 디렉터리를 정리하는 작업.

system_upgrade_stop_services

모든 서비스를 종료하는 작업.

system_upgrade_transfer_data

모든 서비스를 종료하고 부트스트랩 노드로 데이터베이스 전송을 수행하는 작업입니다.

17.4. OPENSTACK 오버클라우드 업그레이드 통합

이 명령은 오버클라우드 업그레이드의 최종 단계를 수행합니다. 이 마지막 단계에서는 오버클라우드 heat 스택을 OpenStack Platform 16.1 Overcloud 계획 및 업데이트된 환경 파일과 동기화합니다. 이 프로세스를 통해 생성된 오버클라우드가 새 OpenStack Platform 16.1 오버클라우드 구성과 일치하는지 확인합니다. 이 명령은 **openstack overcloud deploy** 명령과 유사하며 동일한 많은 옵션을 사용합니다.

17.5. 오버클라우드 노드 업그레이드 워크플로

각 오버클라우드 노드에서 업그레이드를 수행하는 경우 다음 측면을 고려하여 업그레이드의 관련 단계에서 실행할 올바른 명령을 결정해야 합니다.

컨트롤러 서비스

- **노드에 Pacemaker 서비스가 포함되어 있습니까?** 먼저 데이터베이스 전송을 시작하고 Red Hat OpenStack 13에서 16.1로 쉽게 마이그레이션하는 임시 컨테이너를 시작하려면 부트스트랩 노드를 업그레이드해야 합니다. 부트스트랩 컨트롤러 노드 업그레이드 프로세스 중에 새 Pacemaker 클러스터가 생성되고 노드에서 새로운 Red Hat OpenStack 16.1 컨테이너가 시작되지만 나머지 컨트롤러 노드는 여전히 Red Hat OpenStack 13에서 실행됩니다. 부트스트랩 노드를 업그레이드한 후에는 Pacemaker 서비스를 사용하여 각 추가 노드를 업그레이드하고 각 노드가 부트스트랩 노드로 시작되는 새 Pacemaker 클러스터에 참여하는지 확인해야 합니다. Pacemaker 없이 분할 서비스 컨트롤러 노드를 업그레이드하는 프로세스에는 이러한 추가 단계가 필요하지 않습니다.

Compute 서비스

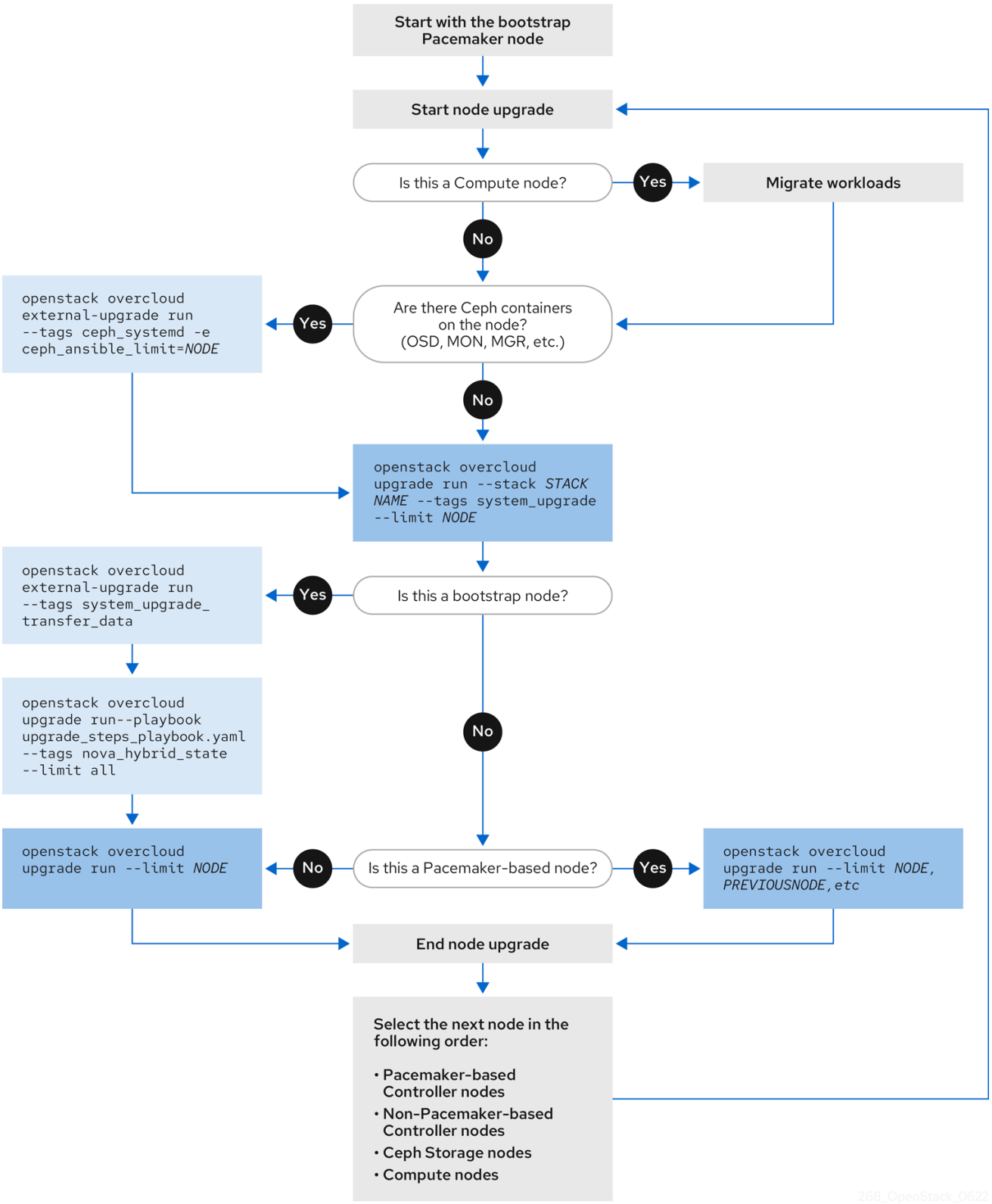
- **노드가 컴퓨팅 노드입니까?** 노드에 컴퓨팅 서비스가 포함된 경우 가용성을 극대화하려면 노드에서 가상 머신을 마이그레이션해야 합니다. 이 경우 컴퓨팅 노드에는 가상 머신을 호스팅하도록 설계된 모든 노드가 포함됩니다. 이 정의에는 다음과 같은 컴퓨팅 노드 유형이 포함됩니다.
  - 일반 컴퓨팅 노드
  - HCI(Hyper-Converged Infrastructure)가 있는 컴퓨팅 노드
  - DPDK(Data Plane Development Kit) 또는 SR-IOV(Single Root Input/Output Virtualization)와 같은 네트워크 기능 가상화 기술이 있는 컴퓨팅 노드
  - 실시간 컴퓨팅 노드

Ceph Storage 서비스

- **노드에 Ceph Storage 서비스가 포함되어 있습니까?docker** 대신 **podman** 을 사용하려면 노드의 컨테이너화된 Ceph Storage 서비스의 **systemd** 장치 파일을 변환해야 합니다. 이는 다음 노드 유형에 적용됩니다.
  - Ceph Storage OSD 노드
  - Ceph MON 서비스가 포함된 컨트롤러 노드
  - split-Controller Ceph MON 노드
  - HCI(Hyper-Converged Infrastructure)가 있는 컴퓨팅 노드

워크플로

다음 워크플로 다이어그램을 사용하여 특정 노드의 올바른 업데이트 경로를 식별합니다.


Start with the bootstrap Pacemaker node
Start node upgrade
Is this a Compute node?
Yes
Migrate workloads
No
Are there Ceph containers on the node? (OSD, MON, MGR, etc.)
Yes
openstack overcloud external-upgrade run --tags ceph_systemd -e ceph_ansible_limit=NODE
No
openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit NODE
Is this a bootstrap node?
Yes
openstack overcloud external-upgrade run --tags system_upgrade_ transfer_data
openstack overcloud upgrade run--playbook upgrade_steps_playbook.yaml --tags nova_hybrid_state --limit all
No
Is this a Pacemaker-based node?
Yes
openstack overcloud upgrade run --limit NODE, PREVIOUSNODE,etc
No
openstack overcloud upgrade run --limit NODE
End node upgrade
Select the next node in the following order:
• Pacemaker-based Controller nodes
• Non-Pacemaker-based Controller nodes
• Ceph Storage nodes
• Compute nodes
268_OpenStack_0622

18장. 표준 오버클라우드 업그레이드

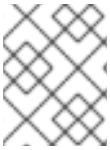
이 시나리오에는 다음 노드 유형을 포함하는 표준 오버클라우드 환경의 업그레이드 프로세스 예가 포함되어 있습니다.

- 컨트롤러 노드 세 개
- Ceph Storage 노드 세 개
- 여러 컴퓨팅 노드

18.1. 오버클라우드 업그레이드 준비 실행

업그레이드를 수행하려면 다음 작업을 수행하는 **openstack overcloud upgrade prepare** 명령을 실행해야 합니다.

- 오버클라우드 플랜을 OpenStack Platform 16.1로 업데이트
- 업그레이드할 노드를 준비합니다.



참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack** ***STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. 업그레이드 준비 명령을 실행합니다.

```
$ openstack overcloud upgrade prepare \
  --stack STACK NAME\
  --templates \
  -e ENVIRONMENT FILE
  …
  -e /home/stack/templates/upgrades-environment.yaml \
  -e /home/stack/templates/rhsm.yaml \
  -e /home/stack/containers-prepare-parameter.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
  …
```

환경과 관련된 다음 옵션을 포함합니다.

- **업그레이드별 매개 변수(-e)가 있는 환경 파일(upgrade-environment.yaml**)입니다.
- 등록 및 서브스크립션 매개 변수(**-e**)가 있는 환경 파일(**rhsm.yaml**).
- 새 컨테이너 이미지 위치(**-e**)가 있는 환경 파일(**containers-prepare-parameter.yaml**)입니다. 대부분의 경우 언더클라우드에서 사용하는 것과 동일한 환경 파일입니다.
- OVS 호환성을 유지 관리하기 위한 환경 파일(**neutron-ovs.yaml**)입니다.

- 배포와 관련된 모든 사용자 지정 구성 환경 파일(**-e**)입니다.
- 해당하는 경우 **--roles-file** 을 사용하는 사용자 지정 **역할(roles_data**) 파일을 사용합니다.
- 해당하는 경우 **--networks-file** 을 사용하여 구성 가능한 **네트워크(network_data**) 파일입니다.
- 사용자 지정 스택 이름을 사용하는 경우 **--stack** 옵션으로 이름을 전달합니다.

3. 업그레이드 준비가 완료될 때까지 기다립니다.
4. 컨테이너 이미지를 다운로드합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
container_image_prepare
```

18.2. DIRECTOR가 배포한 CEPH STORAGE로 컨트롤러 노드 업그레이드

배포 시 director를 사용하여 배포된 Red Hat Ceph Storage 클러스터를 사용하는 경우 다음 절차를 완료해야 합니다.

모든 컨트롤러 노드를 OpenStack Platform 16.1로 업그레이드하려면 부트스트랩 컨트롤러 노드부터 각 컨트롤러 노드를 업그레이드해야 합니다.

부트스트랩 컨트롤러 노드 업그레이드 프로세스 중에 새 Pacemaker 클러스터가 생성되고 노드에서 새로운 Red Hat OpenStack 16.1 컨테이너가 시작되지만 나머지 컨트롤러 노드는 여전히 Red Hat OpenStack 13에서 실행됩니다.

부트스트랩 노드를 업그레이드한 후에는 Pacemaker 서비스를 사용하여 각 추가 노드를 업그레이드하고 각 노드가 부트스트랩 노드로 시작되는 새 Pacemaker 클러스터에 참여하는지 확인해야 합니다. 자세한 내용은 Overcloud 노드 업그레이드 워크플로를 참조하십시오.

이 예에서 컨트롤러 노드의 이름은 기본 **overcloud-controller-*NODEID*** 규칙을 사용하여 이름이 지정됩니다. 여기에는 다음과 같은 세 가지 컨트롤러 노드가 포함됩니다.

- **overcloud-controller-0**
- **overcloud-controller-1**
- **overcloud-controller-2**

해당하는 경우 고유한 노드 이름으로 이 값을 바꿉니다.

참고

오버클라우드 기본 스택 이름을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack *STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. 언더클라우드 노드에서 다음 명령을 실행하여 부트스트랩 컨트롤러 노드를 식별합니다.

```
$ tripleo-ansible-inventory --list --stack overcloud |jq .overcloud_Controller.hosts[0]
```

3. 부트스트랩 컨트롤러 노드를 업그레이드합니다.

   a. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud external-upgrade run --stack <stack_name> --tags ceph_systemd
   -e ceph_ansible_limit=overcloud-controller-0
   ```

   & **lt;stack_name** >을 스택 이름으로 바꿉니다.

   이 명령은 다음 기능을 수행합니다.

   - Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
   - **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 컨트롤러 노드로 제한합니다.

   이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

   b. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
   overcloud-controller-0
   ```

   이 명령은 다음 작업을 수행합니다.

   - 운영 체제의 Leapp 업그레이드를 수행합니다.
   - Leapp 업그레이드의 일부로 재부팅을 수행합니다.

   중요

   다음 명령을 실행하면 컨트롤 플레인이 중단됩니다. 다음 몇 단계에서 오버클라우드에서 표준 작업을 수행할 수 없습니다.

   c. **system_upgrade_transfer_data** 태그를 사용하여 외부 upgrade 명령을 실행합니다.

   ```
   $ openstack overcloud external-upgrade run --stack STACK NAME --tags
   system_upgrade_transfer_data
   ```

   이 명령은 기존 노드의 최신 데이터베이스 버전을 부트스트랩 노드로 복사합니다.

   d. **nova_hybrid_state 태그로 upgrade 명령을 실행하고 upgrade_ steps_playbook.yaml 플레이북만 실행합니다.**

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --playbook
   upgrade_steps_playbook.yaml --tags nova_hybrid_state --limit all
   ```

   이 명령은 이후 단계에서 컴퓨팅 노드를 업그레이드할 때 워크로드 마이그레이션을 용이하게 하기 위해 컴퓨팅 노드에서 임시 16.1 컨테이너를 시작합니다.

   e. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0
```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다.

중요

이 명령이 완료되면 컨트롤 플레인이 활성화됩니다. 오버클라우드에서 표준 작업을 다시 수행할 수 있습니다.

f. 업그레이드 후 새 Pacemaker 클러스터가 시작되고 galera, rabbit, haproxy, redis와 같은 컨트롤 플레인 서비스가 실행 중인지 확인합니다.

```
$ sudo pcs status
```

4. 다음 컨트롤러 노드를 업그레이드합니다.

a. 이전 클러스터가 더 이상 실행되지 않는지 확인합니다.

```
$ sudo pcs status
```

클러스터가 실행 중이 아닌 경우 다음과 유사한 오류가 표시됩니다.

```
Error: cluster is not currently running on this node
```

b. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd
-e ceph_ansible_limit=overcloud-controller-1
```

이 명령은 다음 기능을 수행합니다.

- Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
- **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 컨트롤러 노드로 제한합니다.

이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

c. 다음 컨트롤러 노드에서 **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-controller-1
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

d. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-
0,overcloud-controller-1
```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다. 이 노드 외에도 **--limit** 옵션에 이전에 업그레이드한 부트스트랩 노드를 포함합니다.

5. 최종 컨트롤러 노드를 업그레이드합니다.

   a. 이전 클러스터가 더 이상 실행되지 않는지 확인합니다.

   ```
   $ sudo pcs status
   ```

   클러스터가 실행 중이 아닌 경우 다음과 유사한 오류가 표시됩니다.

   ```
   Error: cluster is not currently running on this node
   ```

   b. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd
   -e ceph_ansible_limit=overcloud-controller-2
   ```

   이 명령은 다음 기능을 수행합니다.

   - Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
   - **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 컨트롤러 노드로 제한합니다.

   이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

   c. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
   overcloud-controller-2
   ```

   이 명령은 다음 작업을 수행합니다.

   - 운영 체제의 Leapp 업그레이드를 수행합니다.
   - Leapp 업그레이드의 일부로 재부팅을 수행합니다.

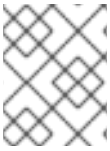
   d. 태그 없이 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-
   0,overcloud-controller-1,overcloud-controller-2
   ```

   이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다. 모든 컨트롤러 노드를 **--limit** 옵션에 포함합니다.

18.3. CEPH STORAGE 노드의 운영 체제 업그레이드

배포 시 director를 사용하여 배포된 Red Hat Ceph Storage 클러스터를 사용하는 경우 각 Ceph Storage 노드의 운영 체제를 업그레이드해야 합니다.



참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack** ***STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. Ceph Storage 노드를 선택하고 운영 체제를 업그레이드합니다.

   a. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd
-e ceph_ansible_limit=overcloud-cephstorage-0
```

   이 명령은 다음 기능을 수행합니다.

   - Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
   - **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 노드로 제한합니다.

   이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

   b. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-cephstorage-0
```

   이 명령은 다음 작업을 수행합니다.

   - 운영 체제의 Leapp 업그레이드를 수행합니다.
   - Leapp 업그레이드의 일부로 재부팅을 수행합니다.

   c. 선택 사항: Ceph 서브스크립션을 사용하고 Ceph 스토리지 노드에 **overcloud-minimal** 이미지를 사용하도록 director가 구성된 경우 다음 단계를 완료해야 합니다.

      i. 노드에 로그인하고 RHEL (Red Hat Enterprise Linux) 마이너 릴리스 버전을 설정 해제합니다.

```
$ sudo subscription-manager release --unset
```

      ii. 노드에서 시스템 업데이트를 수행합니다.

```
$ sudo dnf -y update
```

      iii. 노드를 재부팅합니다.

```
$ sudo reboot
```

   d. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-cephstorage-0
```

이 명령은 **config-download** 플레이북을 실행하고 Ceph Storage 노드에서 구성 가능한 서비스를 구성합니다. 이 단계에서는 Ceph Storage 노드를 Red Hat Ceph Storage 4로 업그레이드하지 않습니다. Red Hat Ceph Storage 4 업그레이드는 이후 절차에서 수행됩니다.

3. 다음 Ceph Storage 노드를 선택하고 운영 체제를 업그레이드합니다.

   a. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e ceph_ansible_limit=overcloud-cephstorage-1
   ```

   이 명령은 다음 기능을 수행합니다.

   - Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
   - **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 노드로 제한합니다.

   이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

   b. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-cephstorage-1
   ```

   이 명령은 다음 작업을 수행합니다.

   - 운영 체제의 Leapp 업그레이드를 수행합니다.
   - Leapp 업그레이드의 일부로 재부팅을 수행합니다.

   c. 태그 없이 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-cephstorage-1
   ```

   이 명령은 **config-download** 플레이북을 실행하고 Ceph Storage 노드에서 구성 가능한 서비스를 구성합니다. 이 단계에서는 Ceph Storage 노드를 Red Hat Ceph Storage 4로 업그레이드하지 않습니다. Red Hat Ceph Storage 4 업그레이드는 이후 절차에서 수행됩니다.

4. 최종 Ceph Storage 노드를 선택하고 운영 체제를 업그레이드합니다.

   a. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e ceph_ansible_limit=overcloud-cephstorage-2
   ```

   이 명령은 다음 기능을 수행합니다.

   - Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
   - **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 노드로 제한합니다.

이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

b. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-cephstorage-2
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

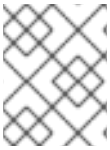
c. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-cephstorage-
2
```

이 명령은 **config-download** 플레이북을 실행하고 Ceph Storage 노드에서 구성 가능한 서비스를 구성합니다. 이 단계에서는 Ceph Storage 노드를 Red Hat Ceph Storage 4로 업그레이드하지 않습니다. Red Hat Ceph Storage 4 업그레이드는 이후 절차에서 수행됩니다.

18.4. 컴퓨팅 노드 업그레이드

모든 컴퓨팅 노드를 OpenStack Platform 16.1로 업그레이드합니다.



참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack *STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. 인스턴스를 마이그레이션합니다. 마이그레이션 전략에 대한 자세한 내용은 컴퓨팅 노드 간 가상 머신 마이그레이션을 참조하십시오.

3. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-compute-0
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

4. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-compute-0
```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다.

5. 여러 컴퓨팅 노드를 병렬로 업그레이드하려면 **--limit** 옵션을 업그레이드할 노드의 쉼표로 구분된 목록으로 설정합니다. 먼저 **system_upgrade** 작업을 수행합니다.

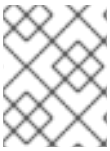
```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-compute-0,overcloud-compute-1,overcloud-compute-2
```

그런 다음 표준 OpenStack 서비스 업그레이드를 수행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME  --limit overcloud-compute-
0,overcloud-compute-1,overcloud-compute-2
```

18.5. 오버클라우드 스택 동기화

업그레이드를 위해서는 스택 리소스 구조 및 매개변수가 OpenStack Platform 16.1의 새로운 배포에 맞게 조정되도록 오버클라우드 스택을 업데이트해야 합니다.



참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack *STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. **containers-prepare-parameter.yaml** 파일을 편집하고 다음 매개변수와 해당 값을 제거합니다.
   - **ceph3_namespace**
   - **ceph3_tag**
   - **ceph3_image**
   - **name_prefix_stein**
   - **name_suffix_stein**
   - **namespace_stein**
   - **tag_stein**
3. 오버클라우드에서 펜싱을 다시 활성화하려면 **fencing.yaml** 환경 파일에서 **EnableFencing** 매개변수를 **true** 로 설정합니다.
4. 업그레이드 종료 명령을 실행합니다.

```
$ openstack overcloud upgrade converge \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE
```

```
…
-e /home/stack/templates/upgrades-environment.yaml \
-e /home/stack/templates/rhsm.yaml \
-e /home/stack/containers-prepare-parameter.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
…
```

환경과 관련된 다음 옵션을 포함합니다.

- **업그레이드별 매개 변수(-e)가 있는 환경 파일(upgrade-environment.yaml**)입니다.
- **EnableFencing** 매개변수가 **true** 로 설정된 환경 파일(**fencing.yaml**)입니다.
- 등록 및 서브스크립션 매개 변수(**-e**)가 있는 환경 파일(**rhsm.yaml**).
- 새 컨테이너 이미지 위치(**-e**)가 있는 환경 파일(**containers-prepare-parameter.yaml**)입니다. 대부분의 경우 언더클라우드에서 사용하는 것과 동일한 환경 파일입니다.
- OVS 호환성을 유지 관리하기 위한 환경 파일(**neutron-ovs.yaml**)입니다.
- 배포와 관련된 모든 사용자 지정 구성 환경 파일(**-e**)입니다.
- 해당하는 경우 **--roles-file** 을 사용하는 사용자 지정 **역할(roles_data**) 파일을 사용합니다.
- 해당하는 경우 **--networks-file** 을 사용하여 구성 가능한 **네트워크(network_data**) 파일입니다.
- 사용자 지정 스택 이름을 사용하는 경우 **--stack** 옵션으로 이름을 전달합니다.

5. 스택 동기화가 완료될 때까지 기다립니다.

중요

추가 배포 작업에는 **upgrade-environment.yaml** 파일이 필요하지 않습니다.

19장. 외부 CEPH 배포로 오버클라우드 업그레이드

이 시나리오에는 다음 노드 유형을 포함하는 외부 Ceph 배포가 있는 오버클라우드 환경의 업그레이드 프로세스 예가 포함되어 있습니다.

- 컨트롤러 노드 세 개
- 외부 Ceph Storage 클러스터
- 여러 컴퓨팅 노드

19.1. 오버클라우드 업그레이드 준비 실행

업그레이드를 수행하려면 다음 작업을 수행하는 **openstack overcloud upgrade prepare** 명령을 실행해야 합니다.

- 오버클라우드 플랜을 OpenStack Platform 16.1로 업데이트
- 업그레이드할 노드를 준비합니다.

참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack** ***STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. 업그레이드 준비 명령을 실행합니다.

```
$ openstack overcloud upgrade prepare \
  --stack STACK NAME\
  --templates \
  -e ENVIRONMENT FILE
  …
  -e /home/stack/templates/upgrades-environment.yaml \
  -e /home/stack/templates/rhsm.yaml \
  -e /home/stack/containers-prepare-parameter.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
  …
```

환경과 관련된 다음 옵션을 포함합니다.

- **업그레이드별 매개 변수(-e)가 있는 환경 파일(upgrade-environment.yaml**)입니다.
- 등록 및 서브스크립션 매개 변수(**-e**)가 있는 환경 파일(**rhsm.yaml**).
- 새 컨테이너 이미지 위치(**-e**)가 있는 환경 파일(**containers-prepare-parameter.yaml**)입니다. 대부분의 경우 언더클라우드에서 사용하는 것과 동일한 환경 파일입니다.
- OVS 호환성을 유지 관리하기 위한 환경 파일(**neutron-ovs.yaml**)입니다.

- 배포와 관련된 모든 사용자 지정 구성 환경 파일(**-e**)입니다.
- 해당하는 경우 **--roles-file** 을 사용하는 사용자 지정 **역할(roles_data**) 파일을 사용합니다.
- 해당하는 경우 **--networks-file** 을 사용하여 구성 가능한 **네트워크(network_data**) 파일입니다.
- 사용자 지정 스택 이름을 사용하는 경우 **--stack** 옵션으로 이름을 전달합니다.

3. 업그레이드 준비가 완료될 때까지 기다립니다.
4. 컨테이너 이미지를 다운로드합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
container_image_prepare
```

19.2. 외부 CEPH 배포로 컨트롤러 노드 업그레이드

외부 Ceph 배포를 사용하여 업그레이드하는 경우 다음 절차를 완료해야 합니다.

모든 컨트롤러 노드를 OpenStack Platform 16.1로 업그레이드하려면 부트스트랩 컨트롤러 노드부터 각 컨트롤러 노드를 업그레이드해야 합니다.

부트스트랩 컨트롤러 노드 업그레이드 프로세스 중에 새 Pacemaker 클러스터가 생성되고 노드에서 새로운 Red Hat OpenStack 16.1 컨테이너가 시작되지만 나머지 컨트롤러 노드는 여전히 Red Hat OpenStack 13에서 실행됩니다.

부트스트랩 노드를 업그레이드한 후에는 Pacemaker 서비스를 사용하여 각 추가 노드를 업그레이드하고 각 노드가 부트스트랩 노드로 시작되는 새 Pacemaker 클러스터에 참여하는지 확인해야 합니다. 자세한 내용은 Overcloud 노드 업그레이드 워크플로를 참조하십시오.

이 예에서 컨트롤러 노드의 이름은 기본 **overcloud-controller-*NODEID*** 규칙을 사용하여 이름이 지정됩니다. 여기에는 다음과 같은 세 가지 컨트롤러 노드가 포함됩니다.

- **overcloud-controller-0**
- **overcloud-controller-1**
- **overcloud-controller-2**

해당하는 경우 고유한 노드 이름으로 이 값을 바꿉니다.

참고

오버클라우드 기본 스택 이름을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack *STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. 언더클라우드 노드에서 다음 명령을 실행하여 부트스트랩 컨트롤러 노드를 식별합니다.

```
$ tripleo-ansible-inventory --list --stack overcloud |jq .overcloud_Controller.hosts[0]
```

3. 부트스트랩 컨트롤러 노드를 업그레이드합니다.

   a. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
   overcloud-controller-0
   ```

   이 명령은 다음 작업을 수행합니다.

   - 운영 체제의 Leapp 업그레이드를 수행합니다.
   - Leapp 업그레이드의 일부로 재부팅을 수행합니다.

   중요

   다음 명령을 실행하면 컨트롤 플레인이 중단됩니다. 다음 몇 단계에서 오버클라우드에서 표준 작업을 수행할 수 없습니다.

   b. **system_upgrade_transfer_data** 태그를 사용하여 외부 upgrade 명령을 실행합니다.

   ```
   $ openstack overcloud external-upgrade run --stack STACK NAME --tags
   system_upgrade_transfer_data
   ```

   이 명령은 기존 노드의 최신 데이터베이스 버전을 부트스트랩 노드로 복사합니다.

   c. **nova_hybrid_state 태그로 upgrade 명령을 실행하고 upgrade_ steps_playbook.yaml 플레이북만 실행합니다.**

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --playbook
   upgrade_steps_playbook.yaml --tags nova_hybrid_state --limit all
   ```

   이 명령은 이후 단계에서 컴퓨팅 노드를 업그레이드할 때 워크로드 마이그레이션을 용이하게 하기 위해 컴퓨팅 노드에서 임시 16.1 컨테이너를 시작합니다.

   d. 태그 없이 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0
   ```

   이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다.

   중요

   이 명령이 완료되면 컨트롤 플레인이 활성화됩니다. 오버클라우드에서 표준 작업을 다시 수행할 수 있습니다.

   e. 업그레이드 후 새 Pacemaker 클러스터가 시작되고 galera, rabbit, haproxy, redis와 같은 컨트롤 플레인 서비스가 실행 중인지 확인합니다.

   ```
   $ sudo pcs status
   ```

4. 다음 컨트롤러 노드를 업그레이드합니다.

a. 이전 클러스터가 더 이상 실행되지 않는지 확인합니다.

```
$ sudo pcs status
```

클러스터가 실행 중이 아닌 경우 다음과 유사한 오류가 표시됩니다.

```
Error: cluster is not currently running on this node
```

b. 다음 컨트롤러 노드에서 **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-controller-1
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

c. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-
0,overcloud-controller-1
```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다. 이 노드 외에도 **--limit** 옵션에 이전에 업그레이드한 부트스트랩 노드를 포함합니다.

5. 최종 컨트롤러 노드를 업그레이드합니다.

a. 이전 클러스터가 더 이상 실행되지 않는지 확인합니다.

```
$ sudo pcs status
```

클러스터가 실행 중이 아닌 경우 다음과 유사한 오류가 표시됩니다.

```
Error: cluster is not currently running on this node
```

b. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-controller-2
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

c. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-
0,overcloud-controller-1,overcloud-controller-2
```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다. 모든 컨트롤러 노드를 **--limit** 옵션에 포함합니다.

19.3. 컴퓨팅 노드 업그레이드

모든 컴퓨팅 노드를 OpenStack Platform 16.1로 업그레이드합니다.

참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack** ***STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. 인스턴스를 마이그레이션합니다. 마이그레이션 전략에 대한 자세한 내용은 컴퓨팅 노드 간 가상 머신 마이그레이션을 참조하십시오.

3. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-compute-0
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

4. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-compute-0
```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다.

5. 여러 컴퓨팅 노드를 병렬로 업그레이드하려면 **--limit** 옵션을 업그레이드할 노드의 쉼표로 구분된 목록으로 설정합니다. 먼저 **system_upgrade** 작업을 수행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-compute-0,overcloud-compute-1,overcloud-compute-2
```

그런 다음 표준 OpenStack 서비스 업그레이드를 수행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-compute-
0,overcloud-compute-1,overcloud-compute-2
```

19.4. 오버클라우드 스택 동기화

업그레이드를 위해서는 스택 리소스 구조 및 매개변수가 OpenStack Platform 16.1의 새로운 배포에 맞게 조정되도록 오버클라우드 스택을 업데이트해야 합니다.

참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack** ***STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. **containers-prepare-parameter.yaml** 파일을 편집하고 다음 매개변수와 해당 값을 제거합니다.
   - **ceph3_namespace**
   - **ceph3_tag**
   - **ceph3_image**
   - **name_prefix_stein**
   - **name_suffix_stein**
   - **namespace_stein**
   - **tag_stein**
3. 오버클라우드에서 펜싱을 다시 활성화하려면 **fencing.yaml** 환경 파일에서 **EnableFencing** 매개변수를 **true** 로 설정합니다.
4. 업그레이드 종료 명령을 실행합니다.

```
$ openstack overcloud upgrade converge \
  --stack STACK NAME\
  --templates \
  -e ENVIRONMENT FILE
  …
  -e /home/stack/templates/upgrades-environment.yaml \
  -e /home/stack/templates/rhsm.yaml \
  -e /home/stack/containers-prepare-parameter.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
  …
```

환경과 관련된 다음 옵션을 포함합니다.

- **업그레이드별 매개 변수(-e)가 있는 환경 파일(upgrade-environment.yaml**)입니다.
- **EnableFencing** 매개변수가 **true** 로 설정된 환경 파일(**fencing.yaml**)입니다.
- 등록 및 서브스크립션 매개 변수(**-e**)가 있는 환경 파일(**rhsm.yaml**).
- 새 컨테이너 이미지 위치(**-e**)가 있는 환경 파일(**containers-prepare-parameter.yaml**)입니다. 대부분의 경우 언더클라우드에서 사용하는 것과 동일한 환경 파일입니다.
- OVS 호환성을 유지 관리하기 위한 환경 파일(**neutron-ovs.yaml**)입니다.

- 배포와 관련된 모든 사용자 지정 구성 환경 파일(**-e**)입니다.
- 해당하는 경우 **--roles-file** 을 사용하는 사용자 지정 **역할(roles_data**) 파일을 사용합니다.
- 해당하는 경우 **--networks-file** 을 사용하여 구성 가능한 **네트워크(network_data**) 파일입니다.
- 사용자 지정 스택 이름을 사용하는 경우 **--stack** 옵션으로 이름을 전달합니다.

5. 스택 동기화가 완료될 때까지 기다립니다.

중요

추가 배포 작업에는 **upgrade-environment.yaml** 파일이 필요하지 않습니다.

20장. 오버클라우드 업그레이드 가속화

오버클라우드 업그레이드 프로세스의 속도를 높이기 위해 부트스트랩 노드부터 한 번에 컨트롤 플레인의 1/3을 업그레이드할 수 있습니다.

컨트롤 플레인의 첫 번째 1/3 업그레이드가 완료된 후 컨트롤 플레인 API가 실행되고 클라우드가 작동하는 혼합 모드로 환경을 이동할 수 있습니다. 고가용성 운영 성능은 전체 컨트롤 플레인이 업그레이드된 후에만 다시 시작할 수 있습니다.

성능을 향상시키기 위해 다수의 컴퓨팅 노드를 업그레이드하는 경우 20개의 노드 그룹에서 **--limit Compute** 옵션으로 **openstack overcloud upgrade run 명령을 실행할** 수 있습니다. 각 작업에서 20개의 노드 그룹을 별도의 그룹을 업그레이드하는 백그라운드에서 여러 업그레이드 작업을 실행할 수 있습니다.

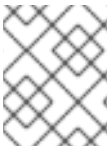
이 시나리오에는 구성 가능 역할과 함께 다음 노드 유형을 포함하는 오버클라우드 환경의 업그레이드 프로세스 예가 포함되어 있습니다.

- 컨트롤러 노드 세 개
- 데이터베이스 노드 세 개
- 네트워크 노드 세 개
- Ceph Storage 노드 세 개
- 여러 컴퓨팅 노드

20.1. 오버클라우드 업그레이드 준비 실행

업그레이드를 수행하려면 다음 작업을 수행하는 **openstack overcloud upgrade prepare** 명령을 실행해야 합니다.

- 오버클라우드 플랜을 OpenStack Platform 16.1로 업데이트
- 업그레이드할 노드를 준비합니다.



참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack *STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. 업그레이드 준비 명령을 실행합니다.

```
$ openstack overcloud upgrade prepare \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE
  …
  -e /home/stack/templates/upgrades-environment.yaml \
  -e /home/stack/templates/rhsm.yaml \
```

```
-e /home/stack/containers-prepare-parameter.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
…
```

환경과 관련된 다음 옵션을 포함합니다.

- **업그레이드별 매개 변수(-e)가 있는 환경 파일(upgrade-environment.yaml**)입니다.
- 등록 및 서브스크립션 매개 변수(**-e**)가 있는 환경 파일(**rhsm.yaml**).
- 새 컨테이너 이미지 위치(**-e**)가 있는 환경 파일(**containers-prepare-parameter.yaml**)입니다. 대부분의 경우 언더클라우드에서 사용하는 것과 동일한 환경 파일입니다.
- OVS 호환성을 유지 관리하기 위한 환경 파일(**neutron-ovs.yaml**)입니다.
- 배포와 관련된 모든 사용자 지정 구성 환경 파일(**-e**)입니다.
- 해당하는 경우 **--roles-file** 을 사용하는 사용자 지정 **역할(roles_data**) 파일을 사용합니다.
- 해당하는 경우 **--networks-file** 을 사용하여 구성 가능한 **네트워크(network_data**) 파일입니다.
- 사용자 지정 스택 이름을 사용하는 경우 **--stack** 옵션으로 이름을 전달합니다.

3. 업그레이드 준비가 완료될 때까지 기다립니다.
4. 컨테이너 이미지를 다운로드합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
container_image_prepare
```

20.2. 컨트롤 플레인 노드 업그레이드

환경의 컨트롤 플레인 노드를 OpenStack Platform 16.1로 업그레이드하려면 부트스트랩 노드를 시작하여 한 번에 컨트롤 플레인 노드의 1/3을 업그레이드해야 합니다.

부트스트랩 컨트롤러 노드 업그레이드 프로세스 중에 새 Pacemaker 클러스터가 생성되고 노드에서 새로운 Red Hat OpenStack 16.1 컨테이너가 시작되고 나머지 컨트롤러 노드는 Red Hat OpenStack 13에서 계속 실행됩니다.

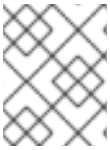
이 예에서 컨트롤 플레인 노드의 이름은 기본 **overcloud-*ROLE-NODEID*** 규칙을 사용하여 이름이 지정됩니다. 여기에는 구성 가능한 역할이 있는 다음 노드 유형이 포함됩니다.

- **overcloud-controller-0**
- **overcloud-controller-1**
- **overcloud-controller-2**
- **overcloud-database-0**
- **overcloud-database-1**
- **overcloud-database-2**
- **overcloud-networker-0**

- **overcloud-networker-1**
- **overcloud-networker-2**
- **overcloud-ceph-0**
- **overcloud-ceph-1**
- **overcloud-ceph-2**

해당하는 경우 고유한 노드 이름으로 이 값을 바꿉니다.

overcloud-controller-0, overcloud-database-0,overcloud- networker-0 및 **overcloud-ceph-0** 부트스트랩 노드를 업그레이드한 후 컨트롤 플레인 노드의 첫 1/3을 구성하는 overcloud-ceph-0 부트스트랩 노드는 Pacemaker 서비스를 사용하여 노드의 추가 1/3을 업그레이드하고 각 노드가 부트스트랩 노드로 시작하는 새 Pacemaker 클러스터에 참여해야 합니다. 따라서 **overcloud-controller-2, overcloud - database-2, overcloud - ceph -2를 업그레이드하기 전에 overcloud -controller-1, overcloud-database-1, overcloud-networker-1, overcloud - ceph-1 을 업그레이드해야 합니다.**



참고

기본 스택 이름 **오버클라우드** 를 사용하지 않는 경우 **--stack *STACK NAME*** 옵션을 사용하여 스택 이름을 설정하고 ***STACK NAME*** 을 스택 이름으로 바꿉니다.

절차

1. 언더클라우드 호스트에 **stack** 사용자로 로그인합니다.
2. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

3. 언더클라우드 노드에서 다음 명령을 실행하여 부트스트랩 컨트롤러 노드를 확인합니다.

```
$ tripleo-ansible-inventory --list --stack overcloud |jq .overcloud_Controller.hosts[0]
```

4. **overcloud-controller-0**,**overcloud-database-0**,**overcloud-networker-0** 및 **overcloud-ceph-0** 컨트롤 플레인 노드를 업그레이드합니다.
   a. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

```
$ openstack overcloud external-upgrade run --stack <stack_name> --tags ceph_systemd
-e ceph_ansible_limit=overcloud-controller-0,overcloud-database-0,overcloud-networker-
0,overcloud-ceph-0
```

   & **lt;stack_name** >을 스택 이름으로 바꿉니다.

   이 명령은 다음 작업을 수행합니다.

   - Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
   - **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 노드로 제한합니다.

   이 단계는 빠른 **업그레이드를** 위해 Ceph Storage 서비스를 준비하기 위한 예비 조치입니다.

b. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-controller-0 &
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-database-0 &
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-networker-0 &
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-ceph-0 &
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

중요

다음 명령을 실행하면 컨트롤 플레인이 중단됩니다. 다음 몇 단계에서 오버클라우드에서 표준 작업을 수행할 수 없습니다.

c. **system_upgrade_transfer_data** 태그를 사용하여 외부 upgrade 명령을 실행합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
system_upgrade_transfer_data
```

이 명령은 기존 노드의 최신 데이터베이스 버전을 부트스트랩 노드로 복사합니다.

d. **nova_hybrid_state 태그로 upgrade 명령을 실행하고 upgrade_ steps_playbook.yaml 플레이북만 실행합니다.**

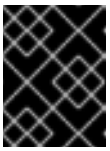
```
$ openstack overcloud upgrade run --stack STACK NAME --playbook
upgrade_steps_playbook.yaml --tags nova_hybrid_state --limit all
```

이 명령은 이후 단계에서 컴퓨팅 노드를 업그레이드할 때 워크로드 마이그레이션을 용이하게 하기 위해 컴퓨팅 노드에서 임시 16.1 컨테이너를 시작합니다.

e. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-
0,overcloud-database-0,overcloud-networker-0,overcloud-ceph-0 --playbook all
```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다.



중요

이 명령이 완료되면 컨트롤 플레인이 활성화됩니다. 오버클라우드에서 표준 작업을 다시 수행할 수 있습니다.

f. 선택 사항: 부트스트랩 Contoller 노드에서 업그레이드 후 새 Pacemaker 클러스터가 시작되고 galera, rabbit, haproxy, redis와 같은 컨트롤 플레인 서비스가 실행 중인지 확인합니다.

```
$ sudo pcs status
```

5. **overcloud-controller-1, overcloud -database-1, overcloud -networker-1** 및 **overcloud-ceph-1** 컨트롤 플레인 노드를 업그레이드합니다.

   a. **overcloud-controller-1** 노드에 로그인하고 이전 클러스터가 더 이상 실행되지 않는지 확인합니다.

   ```
   $ sudo pcs status
   ```

   클러스터가 실행 중이 아닌 경우 다음과 유사한 오류가 표시됩니다.

   ```
   Error: cluster is not currently running on this node
   ```

   b. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd
   -e ceph_ansible_limit=overcloud-controller-1,overcloud-database-1,overcloud-networker-
   1,overcloud-ceph-1
   ```

   이 명령은 다음 기능을 수행합니다.

   - Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
   - **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 노드로 제한합니다.

   이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

   c. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
   overcloud-controller-1,overcloud-database-1,overcloud-networker-1,overcloud-ceph-1
   ```

   이 명령은 다음 작업을 수행합니다.

   - 운영 체제의 Leapp 업그레이드를 수행합니다.
   - Leapp 업그레이드의 일부로 재부팅을 수행합니다.

   d. 태그 없이 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-
   0,overcloud-controller-1,overcloud-database-0,overcloud-database-1,overcloud-
   networker-0,overcloud-networker-1,overcloud-ceph-0,overcloud-ceph-1
   ```

   이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다. 이 노드 외에도 **--limit** 옵션에 이전에 업그레이드한 부트스트랩 노드를 포함합니다.

6. **overcloud-controller-2, overcloud -database-2, overcloud -networker-2** 및 **overcloud-ceph-2** 컨트롤 플레인 노드를 업그레이드합니다.

   a. **overcloud-controller-2** 노드에 로그인하고 이전 클러스터가 더 이상 실행되지 않는지 확인합니다.

   ```
   $ sudo pcs status
   ```

클러스터가 실행 중이 아닌 경우 다음과 유사한 오류가 표시됩니다.

```
Error: cluster is not currently running on this node
```

b. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd
-e ceph_ansible_limit=overcloud-controller-2,overcloud-database-2,overcloud-networker-
2,overcloud-ceph-2
```

이 명령은 다음 기능을 수행합니다.

- Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
- **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 노드로 제한합니다.

이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

c. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-controller-2,overcloud-database-2,overcloud-networker-2,overcloud-ceph-2
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

d. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-
0,overcloud-controller-1,overcloud-controller-2,overcloud-database-0,overcloud-
database-1,overcloud-database-2,overcloud-networker-0,overcloud-networker-
1,overcloud-networker-2,overcloud-ceph-0,overcloud-ceph-1,overcloud-ceph-2
```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다. 모든 컨트롤 플레인 노드를 **--limit** 옵션에 포함합니다.

20.3. 컴퓨팅 노드를 병렬로 업그레이드

다수의 컴퓨팅 노드를 OpenStack Platform 16.1로 업그레이드하려면 20개의 노드 그룹에서 **--limit Compute** 옵션을 사용하여 **openstack overcloud upgrade run 명령을 실행할** 수 있습니다.

각 작업에서 20개의 노드 그룹을 별도의 그룹을 업그레이드하는 백그라운드에서 여러 업그레이드 작업을 실행할 수 있습니다. 이 방법을 사용하여 컴퓨팅 노드를 병렬로 업그레이드하는 경우 업그레이드할 노드를 선택할 수 없습니다. 노드 선택은 **tripleo-ansible-inventory** 명령을 실행할 때 생성하는 인벤토리 파일을 기반으로 합니다. 예를 들어 배포에 80개의 컴퓨팅 노드가 있는 경우 다음 명령을 실행하여 컴퓨팅 노드를 병렬로 업데이트할 수 있습니다.

```
$ openstack overcloud upgrade run -y --limit 'Compute[0:19]' > upgrade-compute-00-19.log 2>&1 &
$ openstack overcloud upgrade run -y --limit 'Compute[20:29]' > upgrade-compute-20-29.log 2>&1 &
$ openstack overcloud upgrade run -y --limit 'Compute[40:59]' > update-compute-40-59.log 2>&1 &
```

```
$ openstack overcloud upgrade run -y --limit 'Compute[60:79]' > update-compute-60-79.log 2>&1 &
```

특정 컴퓨팅 노드를 업그레이드하려면 쉼표로 구분된 노드 목록을 사용합니다.

```
$ openstack overcloud upgrade run --limit <Compute0>,<Compute1>,<Compute2>,<Compute3>
```

참고

기본 스택 이름 **오버클라우드** 를 사용하지 않는 경우 **--stack *STACK NAME*** 옵션을 사용하고 ***STACK NAME*** 을 스택 이름으로 교체합니다.

절차

1. 언더클라우드 호스트에 **stack** 사용자로 로그인합니다.
2. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

3. 인스턴스를 마이그레이션합니다. 마이그레이션 전략에 대한 자세한 내용은 컴퓨팅 노드 간 가상 머신 마이그레이션을 참조하십시오.
4. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run -y --stack STACK NAME --tags system_upgrade --limit
'Compute[0:19]' > upgrade-compute-00-19.log 2>&1 &
$ openstack overcloud upgrade run -y --stack STACK NAME --tags system_upgrade --limit
'Compute[20:29]' > upgrade-compute-20-29.log 2>&1 &
$ openstack overcloud upgrade run -y --stack STACK NAME --tags system_upgrade --limit
'Compute[40:59]' > update-compute-40-59.log 2>&1 &
$ openstack overcloud upgrade run -y --stack STACK NAME --tags system_upgrade --limit
'Compute[60:79]' > update-compute-60-79.log 2>&1 &
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

5. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run -y --stack STACK NAME --limit 'Compute[0:19]' >
upgrade-compute-00-19.log 2>&1 &
$ openstack overcloud upgrade run -y --stack STACK NAME --limit 'Compute[20:29]' >
upgrade-compute-20-29.log 2>&1 &
$ openstack overcloud upgrade run -y --stack STACK NAME --limit 'Compute[40:59]' >
update-compute-40-59.log 2>&1 &
$ openstack overcloud upgrade run -y --stack STACK NAME --limit 'Compute[60:79]' >
update-compute-60-79.log 2>&1 &
```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다.

6. 선택 사항: 선택한 컴퓨팅 노드를 업그레이드하려면 업그레이드할 쉼표로 구분된 노드 목록과 함께 **--limit** 옵션을 사용합니다. 다음 예제에서는 **overcloud-compute-0**,**overcloud-compute-1,overcloud-compute-2** 노드를 병렬로 업그레이드합니다.

   a. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-compute-0,overcloud-compute-1,overcloud-compute-2
   ```

   b. 태그 없이 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME  --limit overcloud-compute-0,overcloud-compute-1,overcloud-compute-2
   ```

20.4. 오버클라우드 스택 동기화

업그레이드를 위해서는 스택 리소스 구조 및 매개변수가 OpenStack Platform 16.1의 새로운 배포에 맞게 조정되도록 오버클라우드 스택을 업데이트해야 합니다.

참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack** ***STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

   ```
   $ source ~/stackrc
   ```

2. **containers-prepare-parameter.yaml** 파일을 편집하고 다음 매개변수와 해당 값을 제거합니다.

   - **ceph3_namespace**
   - **ceph3_tag**
   - **ceph3_image**
   - **name_prefix_stein**
   - **name_suffix_stein**
   - **namespace_stein**
   - **tag_stein**

3. 오버클라우드에서 펜싱을 다시 활성화하려면 **fencing.yaml** 환경 파일에서 **EnableFencing** 매개변수를 **true** 로 설정합니다.

4. 업그레이드 종료 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade converge \
     --stack STACK NAME \
     --templates \
     -e ENVIRONMENT FILE
   ```

```
…
-e /home/stack/templates/upgrades-environment.yaml \
-e /home/stack/templates/rhsm.yaml \
-e /home/stack/containers-prepare-parameter.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
…
```

환경과 관련된 다음 옵션을 포함합니다.

- **업그레이드별 매개 변수(-e)가 있는 환경 파일(upgrade-environment.yaml**)입니다.
- **EnableFencing** 매개변수가 **true** 로 설정된 환경 파일(**fencing.yaml**)입니다.
- 등록 및 서브스크립션 매개 변수(**-e**)가 있는 환경 파일(**rhsm.yaml**).
- 새 컨테이너 이미지 위치(**-e**)가 있는 환경 파일(**containers-prepare-parameter.yaml**)입니다. 대부분의 경우 언더클라우드에서 사용하는 것과 동일한 환경 파일입니다.
- OVS 호환성을 유지 관리하기 위한 환경 파일(**neutron-ovs.yaml**)입니다.
- 배포와 관련된 모든 사용자 지정 구성 환경 파일(**-e**)입니다.
- 해당하는 경우 **--roles-file** 을 사용하는 사용자 지정 **역할(roles_data**) 파일을 사용합니다.
- 해당하는 경우 **--networks-file** 을 사용하여 구성 가능한 **네트워크(network_data**) 파일입니다.
- 사용자 지정 스택 이름을 사용하는 경우 **--stack** 옵션으로 이름을 전달합니다.

5. 스택 동기화가 완료될 때까지 기다립니다.

중요

추가 배포 작업에는 **upgrade-environment.yaml** 파일이 필요하지 않습니다.

21장. 분할된 컨트롤러 오버클라우드 업그레이드

이 시나리오에는 컨트롤러 노드 서비스가 있는 오버클라우드의 업그레이드 프로세스가 여러 노드로 분할된 예제가 포함되어 있습니다. 여기에는 다음과 같은 노드 유형이 포함됩니다.

- Pacemaker를 사용하여 여러 개의 개별 고가용성 서비스
- 여러 분할 컨트롤러 서비스
- Ceph MON 노드 세 개
- Ceph Storage 노드 세 개
- 여러 컴퓨팅 노드

21.1. 오버클라우드 업그레이드 준비 실행

업그레이드를 수행하려면 다음 작업을 수행하는 **openstack overcloud upgrade prepare** 명령을 실행해야 합니다.

- 오버클라우드 플랜을 OpenStack Platform 16.1로 업데이트
- 업그레이드할 노드를 준비합니다.

참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack** ***STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. 업그레이드 준비 명령을 실행합니다.

```
$ openstack overcloud upgrade prepare \
  --stack STACK NAME\
  --templates \
  -e ENVIRONMENT FILE
  …
  -e /home/stack/templates/upgrades-environment.yaml \
  -e /home/stack/templates/rhsm.yaml \
  -e /home/stack/containers-prepare-parameter.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
  …
```

환경과 관련된 다음 옵션을 포함합니다.

- **업그레이드별 매개 변수(-e)가 있는 환경 파일(upgrade-environment.yaml**)입니다.
- 등록 및 서브스크립션 매개 변수(**-e**)가 있는 환경 파일(**rhsm.yaml**).

- 새 컨테이너 이미지 위치(**-e**)가 있는 환경 파일(**containers-prepare-parameter.yaml**)입니다. 대부분의 경우 언더클라우드에서 사용하는 것과 동일한 환경 파일입니다.
- OVS 호환성을 유지 관리하기 위한 환경 파일(**neutron-ovs.yaml**)입니다.
- 배포와 관련된 모든 사용자 지정 구성 환경 파일(**-e**)입니다.
- 해당하는 경우 **--roles-file** 을 사용하는 사용자 지정 **역할(roles_data**) 파일을 사용합니다.
- 해당하는 경우 **--networks-file** 을 사용하여 구성 가능한 **네트워크(network_data**) 파일입니다.
- 사용자 지정 스택 이름을 사용하는 경우 **--stack** 옵션으로 이름을 전달합니다.

3. 업그레이드 준비가 완료될 때까지 기다립니다.
4. 컨테이너 이미지를 다운로드합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
container_image_prepare
```

21.2. PACEMAKER 기반 노드 업그레이드

Pacemaker 서비스를 호스팅하는 모든 노드를 OpenStack Platform 16.1로 업그레이드합니다. 다음 역할에는 Pacemaker 기반 서비스가 포함됩니다.

- 컨트롤러
- 데이터베이스 (MySQL, Galera)
- 메시징 (RabbitMQ)
- 로드 밸런싱 (HAProxy)
- 다음 서비스를 포함하는 다른 모든 역할:
  - **OS::TripleO::Services::Pacemaker**
  - **OS::TripleO::Services::PacemakerRemote**

이 프로세스에는 부트스트랩 노드에서 시작하는 각 노드를 업그레이드해야 합니다.

참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack *STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. 언더클라우드 노드에서 다음 명령을 실행하여 부트스트랩 노드를 식별합니다.

```
$ tripleo-ansible-inventory --list --stack overcloud |jq .overcloud_Controller.hosts[0]
```

3. 부트스트랩 노드를 업그레이드합니다.

   a. 노드에 Ceph Storage 컨테이너가 포함된 경우 **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud external-upgrade run --stack <stack_name> --tags ceph_systemd -e ceph_ansible_limit=overcloud-controller-0
   ```

   & **lt;stack_name** >을 스택 이름으로 바꿉니다.

   이 명령은 다음 기능을 수행합니다.

   - Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
   - **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 노드로 제한합니다.

   이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

   b. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-controller-0
   ```

   이 명령은 다음 작업을 수행합니다.

   - 운영 체제의 Leapp 업그레이드를 수행합니다.
   - Leapp 업그레이드의 일부로 재부팅을 수행합니다.

   c. **system_upgrade_transfer_data** 태그를 사용하여 외부 upgrade 명령을 실행합니다.

   ```
   $ openstack overcloud external-upgrade run --stack STACK NAME --tags system_upgrade_transfer_data
   ```

   이 명령은 기존 노드의 최신 데이터베이스 버전을 부트스트랩 노드로 복사합니다.

   d. **nova_hybrid_state 태그로 upgrade 명령을 실행하고 upgrade_ steps_playbook.yaml 플레이북만 실행합니다.**

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --playbook upgrade_steps_playbook.yaml --tags nova_hybrid_state --limit all
   ```

   이 명령은 이후 단계에서 컴퓨팅 노드를 업그레이드할 때 워크로드 마이그레이션을 용이하게 하기 위해 컴퓨팅 노드에서 임시 16.1 컨테이너를 시작합니다.

   e. 태그 없이 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0
   ```

   이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다.

4. 각 Pacemaker 기반 노드를 업그레이드합니다.

a. 노드에 Ceph Storage 컨테이너가 포함된 경우 **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd
-e ceph_ansible_limit=overcloud-database-0
```

이 명령은 다음 기능을 수행합니다.

- Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
- **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 노드로 제한합니다.

이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

b. 다음 노드에서 **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-database-0
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

c. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-
0,overcloud-database-0
```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다. 이 노드 외에도 이전에 업그레이드한 노드를 **--limit** 옵션에 포함합니다.

5. 모든 Pacemaker 기반 노드를 업그레이드할 때까지 각 Pacemaker 기반 노드에서 업그레이드 프로세스를 반복합니다.

21.3. PACEMAKER 이외의 컨트롤러 노드 업그레이드

Pacemaker 기반 서비스가 없는 모든 노드를 OpenStack Platform 16.1로 업그레이드합니다. 이러한 노드에는 일반적으로 특정 OpenStack 서비스가 포함됩니다. Pacemaker 기반 서비스가 없는 역할의 예는 다음과 같습니다.

- Networker
- Ironic Conductor
- 오브젝트 스토리지
- 표준 컨트롤러 노드에서 분리 또는 확장된 서비스가 있는 모든 사용자 정의 역할

이 그룹화에는 다음 노드를 포함하지 마십시오.

- 모든 컴퓨팅 노드
- 모든 Ceph Storage 노드

이 프로세스에는 각 노드를 업그레이드해야 합니다.

참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack** ***STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-networker-0
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

3. 태그 없이 업그레이드 명령을 실행합니다.

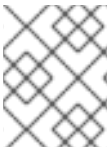
```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-networker-0
```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다.

4. 모든 컨트롤러 기반 노드를 업그레이드할 때까지 각 노드에서 업그레이드 프로세스를 반복합니다.

21.4. CEPH MON 노드의 운영 체제 업그레이드

각 Ceph MON 노드의 운영 체제를 업그레이드합니다. 노드 간에 쿼럼을 유지하려면 각 Ceph MON 노드를 개별적으로 업그레이드하는 것이 좋습니다.



참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack** ***STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. Ceph MON 노드를 선택하고 운영 체제를 업그레이드합니다.

   a. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd
-e ceph_ansible_limit=overcloud-cephmon-0
```

이 명령은 다음 기능을 수행합니다.

- Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
- **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 노드로 제한합니다.

이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

b. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-cephmon-0
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

c. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-cephmon-0
```

이 명령은 **config-download** 플레이북을 실행하고 Ceph MON 노드에서 구성 가능한 서비스를 구성합니다. 이 단계에서는 Ceph MON 노드를 Red Hat Ceph Storage 4로 업그레이드하지 않습니다. Red Hat Ceph Storage 4 업그레이드는 이후 절차에서 수행됩니다.

3. 다음 Ceph MON 노드를 선택하고 운영 체제를 업그레이드합니다.

a. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd
-e ceph_ansible_limit=overcloud-cephmon-1
```

이 명령은 다음 기능을 수행합니다.

- Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
- **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 노드로 제한합니다.

이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

b. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-cephmon-1
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.

- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

c. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-cephmon-1
```

이 명령은 **config-download** 플레이북을 실행하고 Ceph MON 노드에서 구성 가능한 서비스를 구성합니다. 이 단계에서는 Ceph MON 노드를 Red Hat Ceph Storage 4로 업그레이드하지 않습니다. Red Hat Ceph Storage 4 업그레이드는 이후 절차에서 수행됩니다.

4. 최종 Ceph MON 노드를 선택하고 운영 체제를 업그레이드합니다.

a. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd
-e ceph_ansible_limit=overcloud-cephmon-2
```

이 명령은 다음 기능을 수행합니다.

- Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
- **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 노드로 제한합니다.

이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

b. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-cephmon-2
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

c. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-cephmon-2
```

이 명령은 **config-download** 플레이북을 실행하고 Ceph MON 노드에서 구성 가능한 서비스를 구성합니다. 이 단계에서는 Ceph MON 노드를 Red Hat Ceph Storage 4로 업그레이드하지 않습니다. Red Hat Ceph Storage 4 업그레이드는 이후 절차에서 수행됩니다.

21.5. CEPH STORAGE 노드의 운영 체제 업그레이드

배포 시 director를 사용하여 배포된 Red Hat Ceph Storage 클러스터를 사용하는 경우 각 Ceph Storage 노드의 운영 체제를 업그레이드해야 합니다.

참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack *STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. Ceph Storage 노드를 선택하고 운영 체제를 업그레이드합니다.

   a. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd
-e ceph_ansible_limit=overcloud-cephstorage-0
```

   이 명령은 다음 기능을 수행합니다.

   - Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
   - **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 노드로 제한합니다.

   이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

   b. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-cephstorage-0
```

   이 명령은 다음 작업을 수행합니다.

   - 운영 체제의 Leapp 업그레이드를 수행합니다.
   - Leapp 업그레이드의 일부로 재부팅을 수행합니다.

   c. 선택 사항: Ceph 서브스크립션을 사용하고 Ceph 스토리지 노드에 **overcloud-minimal** 이미지를 사용하도록 director가 구성된 경우 다음 단계를 완료해야 합니다.

      i. 노드에 로그인하고 RHEL (Red Hat Enterprise Linux) 마이너 릴리스 버전을 설정 해제합니다.

```
$ sudo subscription-manager release --unset
```

      ii. 노드에서 시스템 업데이트를 수행합니다.

```
$ sudo dnf -y update
```

      iii. 노드를 재부팅합니다.

```
$ sudo reboot
```

   d. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-cephstorage-
0
```

이 명령은 **config-download** 플레이북을 실행하고 Ceph Storage 노드에서 구성 가능한 서비스를 구성합니다. 이 단계에서는 Ceph Storage 노드를 Red Hat Ceph Storage 4로 업그레이드하지 않습니다. Red Hat Ceph Storage 4 업그레이드는 이후 절차에서 수행됩니다.

3. 다음 Ceph Storage 노드를 선택하고 운영 체제를 업그레이드합니다.

   a. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e ceph_ansible_limit=overcloud-cephstorage-1
   ```

   이 명령은 다음 기능을 수행합니다.

   - Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
   - **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 노드로 제한합니다.

   이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

   b. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit overcloud-cephstorage-1
   ```

   이 명령은 다음 작업을 수행합니다.

   - 운영 체제의 Leapp 업그레이드를 수행합니다.
   - Leapp 업그레이드의 일부로 재부팅을 수행합니다.

   c. 태그 없이 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-cephstorage-1
   ```

   이 명령은 **config-download** 플레이북을 실행하고 Ceph Storage 노드에서 구성 가능한 서비스를 구성합니다. 이 단계에서는 Ceph Storage 노드를 Red Hat Ceph Storage 4로 업그레이드하지 않습니다. Red Hat Ceph Storage 4 업그레이드는 이후 절차에서 수행됩니다.

4. 최종 Ceph Storage 노드를 선택하고 운영 체제를 업그레이드합니다.

   a. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e ceph_ansible_limit=overcloud-cephstorage-2
   ```

   이 명령은 다음 기능을 수행합니다.

   - Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
   - **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 노드로 제한합니다.

   이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

   b. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-cephstorage-2
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

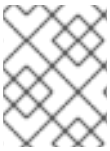
c. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-cephstorage-
2
```

이 명령은 **config-download** 플레이북을 실행하고 Ceph Storage 노드에서 구성 가능한 서비스를 구성합니다. 이 단계에서는 Ceph Storage 노드를 Red Hat Ceph Storage 4로 업그레이드하지 않습니다. Red Hat Ceph Storage 4 업그레이드는 이후 절차에서 수행됩니다.

21.6. 컴퓨팅 노드 업그레이드

모든 컴퓨팅 노드를 OpenStack Platform 16.1로 업그레이드합니다.



참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack *STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. 인스턴스를 마이그레이션합니다. 마이그레이션 전략에 대한 자세한 내용은 컴퓨팅 노드 간 가상 머신 마이그레이션을 참조하십시오.

3. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-compute-0
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

4. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-compute-0
```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다.

5. 여러 컴퓨팅 노드를 병렬로 업그레이드하려면 **--limit** 옵션을 업그레이드할 노드의 쉼표로 구분된 목록으로 설정합니다. 먼저 **system_upgrade** 작업을 수행합니다.

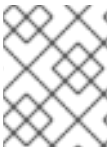
```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-compute-0,overcloud-compute-1,overcloud-compute-2
```

그런 다음 표준 OpenStack 서비스 업그레이드를 수행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME  --limit overcloud-compute-
0,overcloud-compute-1,overcloud-compute-2
```

21.7. 오버클라우드 스택 동기화

업그레이드를 위해서는 스택 리소스 구조 및 매개변수가 OpenStack Platform 16.1의 새로운 배포에 맞게 조정되도록 오버클라우드 스택을 업데이트해야 합니다.



참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack *STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. **containers-prepare-parameter.yaml** 파일을 편집하고 다음 매개변수와 해당 값을 제거합니다.
   - **ceph3_namespace**
   - **ceph3_tag**
   - **ceph3_image**
   - **name_prefix_stein**
   - **name_suffix_stein**
   - **namespace_stein**
   - **tag_stein**
3. 오버클라우드에서 펜싱을 다시 활성화하려면 **fencing.yaml** 환경 파일에서 **EnableFencing** 매개변수를 **true** 로 설정합니다.
4. 업그레이드 종료 명령을 실행합니다.

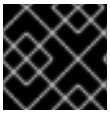
```
$ openstack overcloud upgrade converge \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE
  …
  -e /home/stack/templates/upgrades-environment.yaml \
```

```
-e /home/stack/templates/rhsm.yaml \
-e /home/stack/containers-prepare-parameter.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
…
```

환경과 관련된 다음 옵션을 포함합니다.

- **업그레이드별 매개 변수(-e)가 있는 환경 파일(upgrade-environment.yaml**)입니다.
- **EnableFencing** 매개변수가 **true** 로 설정된 환경 파일(**fencing.yaml**)입니다.
- 등록 및 서브스크립션 매개 변수(**-e**)가 있는 환경 파일(**rhsm.yaml**).
- 새 컨테이너 이미지 위치(**-e**)가 있는 환경 파일(**containers-prepare-parameter.yaml**)입니다. 대부분의 경우 언더클라우드에서 사용하는 것과 동일한 환경 파일입니다.
- OVS 호환성을 유지 관리하기 위한 환경 파일(**neutron-ovs.yaml**)입니다.
- 배포와 관련된 모든 사용자 지정 구성 환경 파일(**-e**)입니다.
- 해당하는 경우 **--roles-file** 을 사용하는 사용자 지정 **역할(roles_data**) 파일을 사용합니다.
- 해당하는 경우 **--networks-file** 을 사용하여 구성 가능한 **네트워크(network_data**) 파일입니다.
- 사용자 지정 스택 이름을 사용하는 경우 **--stack** 옵션으로 이름을 전달합니다.

5. 스택 동기화가 완료될 때까지 기다립니다.



중요

추가 배포 작업에는 **upgrade-environment.yaml** 파일이 필요하지 않습니다.

22장. 하이퍼 컨버지드 인프라를 사용하여 오버클라우드 업그레이드

이 시나리오에는 다음 노드 유형을 포함하는 HCI(하이퍼 컨버지드 인프라)가 있는 오버클라우드의 업그레이드 프로세스 예가 포함되어 있습니다.

- 컨트롤러 노드 세 개
- 결합된 Compute 및 Ceph OSD 서비스가 포함된 여러 HCI 컴퓨팅 노드

22.1. 오버클라우드 업그레이드 준비 실행

업그레이드를 수행하려면 다음 작업을 수행하는 **openstack overcloud upgrade prepare** 명령을 실행해야 합니다.

- 오버클라우드 플랜을 OpenStack Platform 16.1로 업데이트
- 업그레이드할 노드를 준비합니다.

참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack** ***STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. 업그레이드 준비 명령을 실행합니다.

```
$ openstack overcloud upgrade prepare \
  --stack STACK NAME\
  --templates \
  -e ENVIRONMENT FILE
  …
  -e /home/stack/templates/upgrades-environment.yaml \
  -e /home/stack/templates/rhsm.yaml \
  -e /home/stack/containers-prepare-parameter.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
  …
```

환경과 관련된 다음 옵션을 포함합니다.

- **업그레이드별 매개 변수(-e)가 있는 환경 파일(upgrade-environment.yaml**)입니다.
- 등록 및 서브스크립션 매개 변수(**-e**)가 있는 환경 파일(**rhsm.yaml**).
- 새 컨테이너 이미지 위치(**-e**)가 있는 환경 파일(**containers-prepare-parameter.yaml**)입니다. 대부분의 경우 언더클라우드에서 사용하는 것과 동일한 환경 파일입니다.
- OVS 호환성을 유지 관리하기 위한 환경 파일(**neutron-ovs.yaml**)입니다.
- 배포와 관련된 모든 사용자 지정 구성 환경 파일(**-e**)입니다.

- 해당하는 경우 **--roles-file** 을 사용하는 사용자 지정 **역할(roles_data**) 파일을 사용합니다.
- 해당하는 경우 **--networks-file** 을 사용하여 구성 가능한 **네트워크(network_data**) 파일입니다.
- 사용자 지정 스택 이름을 사용하는 경우 **--stack** 옵션으로 이름을 전달합니다.

3. 업그레이드 준비가 완료될 때까지 기다립니다.
4. 컨테이너 이미지를 다운로드합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags
container_image_prepare
```

22.2. DIRECTOR가 배포한 CEPH STORAGE로 컨트롤러 노드 업그레이드

배포 시 director를 사용하여 배포된 Red Hat Ceph Storage 클러스터를 사용하는 경우 다음 절차를 완료해야 합니다.

모든 컨트롤러 노드를 OpenStack Platform 16.1로 업그레이드하려면 부트스트랩 컨트롤러 노드부터 각 컨트롤러 노드를 업그레이드해야 합니다.

부트스트랩 컨트롤러 노드 업그레이드 프로세스 중에 새 Pacemaker 클러스터가 생성되고 노드에서 새로운 Red Hat OpenStack 16.1 컨테이너가 시작되지만 나머지 컨트롤러 노드는 여전히 Red Hat OpenStack 13에서 실행됩니다.

부트스트랩 노드를 업그레이드한 후에는 Pacemaker 서비스를 사용하여 각 추가 노드를 업그레이드하고 각 노드가 부트스트랩 노드로 시작되는 새 Pacemaker 클러스터에 참여하는지 확인해야 합니다. 자세한 내용은 Overcloud 노드 업그레이드 워크플로를 참조하십시오.

이 예에서 컨트롤러 노드의 이름은 기본 **overcloud-controller-*NODEID*** 규칙을 사용하여 이름이 지정됩니다. 여기에는 다음과 같은 세 가지 컨트롤러 노드가 포함됩니다.

- **overcloud-controller-0**
- **overcloud-controller-1**
- **overcloud-controller-2**

해당하는 경우 고유한 노드 이름으로 이 값을 바꿉니다.

참고

오버클라우드 기본 스택 이름을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack *STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. 언더클라우드 노드에서 다음 명령을 실행하여 부트스트랩 컨트롤러 노드를 식별합니다.

```
$ tripleo-ansible-inventory --list --stack overcloud |jq .overcloud_Controller.hosts[0]
```

-

3. 부트스트랩 컨트롤러 노드를 업그레이드합니다.

   a. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud external-upgrade run --stack <stack_name> --tags ceph_systemd
   -e ceph_ansible_limit=overcloud-controller-0
   ```

   & **lt;stack_name** >을 스택 이름으로 바꿉니다.

   이 명령은 다음 기능을 수행합니다.

   - Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
   - **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 컨트롤러 노드로 제한합니다.

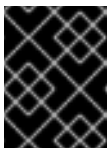
   이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

   b. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
   overcloud-controller-0
   ```

   이 명령은 다음 작업을 수행합니다.

   - 운영 체제의 Leapp 업그레이드를 수행합니다.
   - Leapp 업그레이드의 일부로 재부팅을 수행합니다.

   중요

   다음 명령을 실행하면 컨트롤 플레인이 중단됩니다. 다음 몇 단계에서 오버클라우드에서 표준 작업을 수행할 수 없습니다.

   c. **system_upgrade_transfer_data** 태그를 사용하여 외부 upgrade 명령을 실행합니다.

   ```
   $ openstack overcloud external-upgrade run --stack STACK NAME --tags
   system_upgrade_transfer_data
   ```

   이 명령은 기존 노드의 최신 데이터베이스 버전을 부트스트랩 노드로 복사합니다.

   d. **nova_hybrid_state 태그로 upgrade 명령을 실행하고 upgrade_ steps_playbook.yaml 플레이북만 실행합니다.**

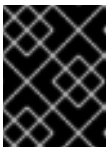
   ```
   $ openstack overcloud upgrade run --stack STACK NAME --playbook
   upgrade_steps_playbook.yaml --tags nova_hybrid_state --limit all
   ```

   이 명령은 이후 단계에서 컴퓨팅 노드를 업그레이드할 때 워크로드 마이그레이션을 용이하게 하기 위해 컴퓨팅 노드에서 임시 16.1 컨테이너를 시작합니다.

   e. 태그 없이 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-0
   ```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다.



중요

이 명령이 완료되면 컨트롤 플레인이 활성화됩니다. 오버클라우드에서 표준 작업을 다시 수행할 수 있습니다.

f. 업그레이드 후 새 Pacemaker 클러스터가 시작되고 galera, rabbit, haproxy, redis와 같은 컨트롤 플레인 서비스가 실행 중인지 확인합니다.

```
$ sudo pcs status
```

4. 다음 컨트롤러 노드를 업그레이드합니다.

a. 이전 클러스터가 더 이상 실행되지 않는지 확인합니다.

```
$ sudo pcs status
```

클러스터가 실행 중이 아닌 경우 다음과 유사한 오류가 표시됩니다.

```
Error: cluster is not currently running on this node
```

b. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd
-e ceph_ansible_limit=overcloud-controller-1
```

이 명령은 다음 기능을 수행합니다.

- Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
- **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 컨트롤러 노드로 제한합니다.

이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

c. 다음 컨트롤러 노드에서 **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-controller-1
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

d. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-
0,overcloud-controller-1
```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다. 이 노드 외에도 **--limit** 옵션에 이전에 업그레이드한 부트스트랩 노드를 포함합니다.

5. 최종 컨트롤러 노드를 업그레이드합니다.

   a. 이전 클러스터가 더 이상 실행되지 않는지 확인합니다.

   ```
   $ sudo pcs status
   ```

   클러스터가 실행 중이 아닌 경우 다음과 유사한 오류가 표시됩니다.

   ```
   Error: cluster is not currently running on this node
   ```

   b. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd
   -e ceph_ansible_limit=overcloud-controller-2
   ```

   이 명령은 다음 기능을 수행합니다.

   - Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
   - **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 컨트롤러 노드로 제한합니다.

   이 단계는 Ceph Storage 서비스를 준비하여 **업그레이드를** 준비하기 위한 예비 조치입니다.

   c. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
   overcloud-controller-2
   ```

   이 명령은 다음 작업을 수행합니다.

   - 운영 체제의 Leapp 업그레이드를 수행합니다.
   - Leapp 업그레이드의 일부로 재부팅을 수행합니다.

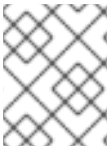
   d. 태그 없이 업그레이드 명령을 실행합니다.

   ```
   $ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-controller-
   0,overcloud-controller-1,overcloud-controller-2
   ```

   이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다. 모든 컨트롤러 노드를 **--limit** 옵션에 포함합니다.

22.3. HCI(HYPER-CONVERGED INFRASTRUCTURE)로 컴퓨팅 노드 업그레이드

HCI 계산 노드를 OpenStack Platform 16.1로 업그레이드.



참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack *STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. 인스턴스를 마이그레이션합니다. 마이그레이션 전략에 대한 자세한 내용은 컴퓨팅 노드 간 가상 머신 마이그레이션을 참조하십시오.

3. Ceph MON 서비스를 사용하여 노드에서 로그아웃하고 Undercloud로 돌아갑니다.

4. **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e
ceph_ansible_limit=overcloud-computehci-0
```

이 명령은 다음 기능을 수행합니다.

- Podman 관리를 사용하도록 Ceph Storage 컨테이너를 제어하는 systemd 장치를 변경합니다.
- **ceph_ansible_limit** 변수를 사용하여 작업을 선택한 Ceph Storage 노드로 제한합니다.

이 단계는 빠른 **업그레이드를** 위해 Ceph Storage 서비스를 준비하기 위한 예비 조치입니다.

5. **system_upgrade** 태그로 upgrade 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-computehci-0
```

이 명령은 다음 작업을 수행합니다.

- 운영 체제의 Leapp 업그레이드를 수행합니다.
- Leapp 업그레이드의 일부로 재부팅을 수행합니다.

6. 태그 없이 업그레이드 명령을 실행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-computehci-0
```

이 명령은 Red Hat OpenStack Platform 업그레이드를 수행합니다.

7. 여러 컴퓨팅 노드를 병렬로 업그레이드하려면 **--limit** 옵션을 업그레이드할 노드의 쉼표로 구분된 목록으로 설정합니다. 먼저 **ceph_systemd** 태그를 사용하여 외부 업그레이드 명령을 실행합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph_systemd -e
ceph_ansible_limit=overcloud-computehci-0,overcloud-computehci-1,overcloud-computehci-
2
```

그런 다음 **system_upgrade** 작업을 수행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --tags system_upgrade --limit
overcloud-computehci-0,overcloud-computehci-1,overcloud-computehci-2
```

그런 다음 표준 OpenStack 서비스 업그레이드를 수행합니다.

```
$ openstack overcloud upgrade run --stack STACK NAME --limit overcloud-computehci-
0,overcloud-computehci-1,overcloud-computehci-2
```

22.4. 오버클라우드 스택 동기화

업그레이드를 위해서는 스택 리소스 구조 및 매개변수가 OpenStack Platform 16.1의 새로운 배포에 맞게 조정되도록 오버클라우드 스택을 업데이트해야 합니다.

참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack *STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. **containers-prepare-parameter.yaml** 파일을 편집하고 다음 매개변수와 해당 값을 제거합니다.
   - **ceph3_namespace**
   - **ceph3_tag**
   - **ceph3_image**
   - **name_prefix_stein**
   - **name_suffix_stein**
   - **namespace_stein**
   - **tag_stein**
3. 오버클라우드에서 펜싱을 다시 활성화하려면 **fencing.yaml** 환경 파일에서 **EnableFencing** 매개변수를 **true** 로 설정합니다.
4. 업그레이드 종료 명령을 실행합니다.

```
$ openstack overcloud upgrade converge \
  --stack STACK NAME \
  --templates \
  -e ENVIRONMENT FILE
  …
  -e /home/stack/templates/upgrades-environment.yaml \
  -e /home/stack/templates/rhsm.yaml \
  -e /home/stack/containers-prepare-parameter.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/neutron-ovs.yaml \
  …
```

환경과 관련된 다음 옵션을 포함합니다.

- **업그레이드별 매개 변수(-e)가 있는 환경 파일(upgrade-environment.yaml**)입니다.
- **EnableFencing** 매개변수가 **true** 로 설정된 환경 파일(**fencing.yaml**)입니다.
- 등록 및 서브스크립션 매개 변수(**-e**)가 있는 환경 파일(**rhsm.yaml**).
- 새 컨테이너 이미지 위치(**-e**)가 있는 환경 파일(**containers-prepare-parameter.yaml**)입니다. 대부분의 경우 언더클라우드에서 사용하는 것과 동일한 환경 파일입니다.
- OVS 호환성을 유지 관리하기 위한 환경 파일(**neutron-ovs.yaml**)입니다.
- 배포와 관련된 모든 사용자 지정 구성 환경 파일(**-e**)입니다.
- 해당하는 경우 **--roles-file** 을 사용하는 사용자 지정 **역할(roles_data**) 파일을 사용합니다.
- 해당하는 경우 **--networks-file** 을 사용하여 구성 가능한 **네트워크(network_data**) 파일입니다.
- 사용자 지정 스택 이름을 사용하는 경우 **--stack** 옵션으로 이름을 전달합니다.

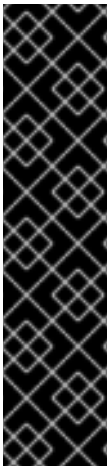
5. 스택 동기화가 완료될 때까지 기다립니다.

중요

추가 배포 작업에는 **upgrade-environment.yaml** 파일이 필요하지 않습니다.

23장. RED HAT CEPH STORAGE 4로 DIRECTOR가 배포된 CEPH STORAGE 클러스터 업그레이드

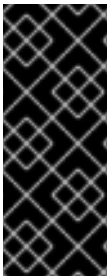
배포 시 director를 사용하여 배포된 Red Hat Ceph Storage 클러스터를 사용하는 경우 이 섹션에 포함된 절차를 완료해야 합니다.



중요

지원되는 이전 버전에서 버전 4.2z2로 Red Hat Ceph Storage 클러스터를 업그레이드하면 업그레이드가 **HEALTH_WARN** 상태에서 스토리지 클러스터와 함께 완료 **되며 상태 모니터링이 안전하지 않은 global_id 회수를 허용합니다**. 이는 패치된 CVE(CVE-2021-20288)로 인해 RHCS 4.2z2 (이상)로 설치/업그레이드 후 Ceph HEALTH_WARN(mons이 안전하지 않은 global_id 회수 허용) 을 참조하십시오.

CVE로 인해 **HEALTH_WARN** 상태가 표시되므로 상태 경고를 일시적으로 음소거할 수 있습니다. 그러나 음소거 경고가 발생하면 클러스터에 연결된 이전 및 패키징되지 않은 잠재적인 클라이언트에 대한 가시성이 없다는 위험이 있습니다. 상태 경고 변경에 대한 자세한 내용은 Red Hat Ceph Storage 문서의 Upgrading a *Red Hat Ceph Storage* cluster 를 참조하십시오.



중요

모든 클라이언트가 업그레이드 및 패치되지 않으면 연결할 수 없을 때까지 수동으로 **global_id_reclaim** 을 비활성화하지 마십시오. 다음 명령을 **root** 사용자로 실행하여 클러스터에 연결된 패치되지 않은 클라이언트 목록을 볼 수 있습니다.

```
# ceph health detail
```

오버클라우드를 업그레이드한 후 director에서 배포한 Ceph Storage 클러스터를 Red Hat Ceph Storage 클러스터 4로 업그레이드합니다.

23.1. CEPH-ANIBLE 설치

배포 시 director를 사용하여 배포된 Red Hat Ceph Storage 클러스터를 사용하는 경우 다음 절차를 완료해야 합니다.

Red Hat OpenStack Platform에서 Ceph Storage를 사용하는 경우 **ceph-ansible** 패키지가 필요합니다.

절차

1. 다음과 같이 Ceph 툴 리포지토리를 활성화합니다.

```
[stack@director ~]$ sudo subscription-manager repos --enable=rhceph-4-tools-for-rhel-8-x86_64-rpms
```

2. **ceph-ansible** 패키지를 설치합니다.

```
[stack@director ~]$ sudo dnf install -y ceph-ansible
```

23.2. CEPH STORAGE 4로 업그레이드

Ceph Storage 노드를 버전 3에서 버전 4로 업그레이드합니다.

참고

기본 스택 이름(**Overcloud**)을 사용하지 않는 경우 STACK NAME을 스택 이름으로 교체하는 **--stack** ***STACK NAME*** 옵션으로 스택 이름을 설정합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. **ceph** 태그를 사용하여 Ceph Storage 외부 업그레이드 프로세스를 실행합니다.

```
$ openstack overcloud external-upgrade run --stack STACK NAME --tags ceph
```

3. Ceph Storage 업그레이드가 완료될 때까지 기다립니다.

24장. 파일 저장소에서 BLUESTORE로 OSD 마이그레이션

업그레이드 프로세스를 완료하고 확인한 후 파일 저장소 OSD를 BlueStore로 마이그레이션해야 합니다. 한 번에 하나의 노드로 마이그레이션을 완료해야 합니다. 다음 절차에서는 **ceph-anible** 을 사용하여 마이그레이션을 완료합니다. 이 절차는 director에서 Ceph 클러스터를 배포하는 경우에만 적용됩니다.

24.1. 클러스터가 파일 저장소를 실행하는지 확인하여 마이그레이션이 필요합니다.

절차

1. 컨트롤러 노드 또는 독립 실행형 Ceph MON 노드와 같이 Ceph MON 컨테이너가 있는 노드에 **heat-admin** 사용자로 로그인합니다. 예를 들어 표준 Overcloud 배포에서 **overcloud-controller-1** 은 Ceph MON 컨테이너를 사용합니다.

2. Ceph 클러스터를 쿼리하여 OSD에서 사용 중인 드라이버를 확인합니다.

```
[heat-admin@overcloud-controller-1 ~]$ sudo -i
[root@overcloud-controller-1 ~]# podman exec -it ceph-mon-overcloud-controller-1 sh -c
"ceph -f json osd metadata" | jq -c 'sort_by(.hostname) | .[] | ["host", .hostname, "osd_id", .id,
"objectstore", .osd_objectstore]'
[root@overcloud-controller-1 ~]#
```

3. **"objectstore": "filestore"**를 반환하는 행이 있으면 해당 노드에 OSD 마이그레이션이 필요합니다.

주의

마이그레이션 시간은 클러스터 크기에 따라 다를 수 있습니다. 매우 큰 클러스터가 있는 경우 마이그레이션 시간은 해당 클러스터의 OSD 수와 저장된 데이터 양에 비례합니다. 환경이 혼합 아키텍처 시나리오에 있지 않도록 최대한 빨리 마이그레이션을 완료해야 성능에 영향을 줄 수 있습니다.

주의

RHCS(Red Hat Ceph Storage) 4 버전의 **ceph-anible** 을 사용하여 파일 저장소 기반 OSD 관리는 지원되지 않으므로 스택 업데이트를 실행하기 전에 마이그레이션을 완료합니다.

24.2. 파일 저장소에서 BLUESTORE로 OSD 마이그레이션

파일 저장소에서 BlueStore로 마이그레이션하기 위해 director는 Ansible을 사용하여 노드의 모든 OSD를 삭제하고 다시 생성합니다. director는 마이그레이션 프로세스 전에 용량 검사를 수행합니다. 마지막으로, director가 BlueStore 백엔드를 사용하여 OSD를 재배포합니다.

사전 요구 사항

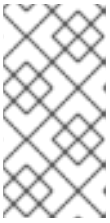
- 정상적이고 실행 중인 RHCS(Red Hat Ceph Storage) 4 클러스터. 컨트롤러 또는 독립 실행형 Ceph MON 노드의 ceph MON 컨테이너에 다음 명령을 입력하여 클러스터를 확인할 수 있습니다.

```
[root@overcloud-controller-1 ~]# podman exec -it ceph-mon-overcloud-controller-1 sh -c
"ceph  -s"
```

절차

1. **CephAnsibleDisksConfig** 매개변수의 **osd_objectstore** 가 **filestore** 로 기본 설정되지 않는지 확인합니다. **osd_objectstore** 매개변수가 사용자 지정 heat 환경 파일에 있는 경우 **bluestore** 값을 명시적으로 정의하거나 제거해야 합니다.

```
parameter_defaults:
  CephAnsibleDisksConfig:
    devices:
      - /dev/sdb
      - /dev/sdc
      - /dev/sdd
    osd_scenario: lvm
   osd_objectstore: bluestore
```



참고

를 포함한 특정 FileStore 구성이 있는 경우(예: 저널) 구성을 적절하게 업데이트해야 합니다. 고급 구성에 대한 자세한 내용은 *컨테이너화된 Red Hat Ceph 가이드의 오버클라우드 배포 가이드의 Ceph Storage 노드 디스크 레이아웃 매핑* 을 참조하십시오.

2. **stack** 사용자로 언더클라우드에 로그인합니다.

3. **ceph_fstobs** 태그와 함께 **openstack overcloud external-upgrade run** 명령을 입력합니다. **<NODE_NAME>** 을 업그레이드할 Ceph OSD 노드의 이름으로 바꿉니다. **openstack server list** 명령을 사용하여 노드 이름을 찾을 수 있습니다.

```
[stack@undercloud ~] $ openstack overcloud external-upgrade run --tags ceph_fstobs -e
ceph_ansible_limit=<NODE_NAME> | tee oc-fstobs.log
```

4. Ceph MON 서비스가 있는 노드에 로그인하고 Ceph 클러스터를 쿼리하여 업그레이드한 노드의 OSD 상태를 확인합니다. 다음 OSD 노드의 마이그레이션을 시작하려면 업그레이드한 노드가 성공적으로 마이그레이션되었는지 확인해야 합니다.

```
[heat-admin@overcloud-controller-1 ~]$ sudo -i
[root@overcloud-controller-1 ~]# podman exec -it ceph-mon-overcloud-controller-1 sh -c
"ceph -f json osd metadata" | jq -c '.[] | select(.hostname == "<NODE_NAME>") | ["host",
.hostname, "osd_id", .id, "objectstore", .osd_objectstore]'
[root@overcloud-controller-1 ~]# exit
```

<NODE_NAME>을 마이그레이션된 노드의 이름으로 바꿉니다. OSD에서 BlueStore를 사용하는 것으로 표시되면 마이그레이션에 성공합니다.

5. 선택 사항: 특정 OSD에 대한 추가 세부 정보를 보려면 다음 명령을 입력합니다.

```
[root@overcloud-controller-1 ~]# podman exec -it ceph-mon-overcloud-controller-1 sh -c
"ceph osd metadata <ID>"
```

<ID>를 쿼리할 OSD의 ID로 바꿉니다.

6. 다음 노드에서 마이그레이션 프로세스를 시작하려면 클러스터가 동기화될 때까지 기다려야 합니다.

```
[root@overcloud-controller-1 ~]# podman exec -it ceph-mon-overcloud-controller-1 sh -c
"ceph  -s"
```

```
Review the command output and ensure that the health of the cluster is `HEALTH_OK` and
the PGs are in the `active+clean` state.
```

7. 다음 노드에서 마이그레이션 프로세스를 시작하려면 클러스터 리밸런싱 프로세스가 완료될 때까지 기다려야 합니다. 상태를 따르려면 다음 명령을 실행합니다.

```
[heat-admin@overcloud-controller-0 ~]$ sudo podman exec ceph-mon-<NODE_NAME>
ceph -w
```

<NODE_NAME>을 마이그레이션된 노드의 이름으로 바꿉니다.

8. 스토리지 클러스터의 각 노드에 대한 마이그레이션 프로세스를 반복합니다.

파일 저장소에서 BlueStore로의 마이그레이션에 대한 자세한 내용은 Red Hat Ceph Storage *관리 가이드의 BlueStore 를 참조하십시오*.

24.3. 파일 저장소에서 BLUESTORE로 마이그레이션 확인

OSD의 상태를 확인하여 마이그레이션을 성공적으로 완료했는지 확인할 수 있습니다.

절차

1. **heat-admin** 사용자로 컨트롤러 노드 중 하나에서 호스팅되는 ceph-mon 컨테이너에 로그인합니다.

2. Ceph 클러스터를 쿼리합니다.

```
[heat-admin@overcloud-controller-1 ~]$ sudo -i
[root@overcloud-controller-1 ~]# podman exec -it ceph-mon-overcloud-controller-1 sh -c
"ceph -f json osd metadata" | jq -c 'sort_by(.hostname) | .[] | ["host", .hostname, "osd_id", .id,
"objectstore", .osd_objectstore]'
[root@overcloud-controller-1 ~]#
```

구성에서 클러스터의 모든 OSD가 BlueStore를 사용하는 것으로 표시되면 마이그레이션이 성공적으로 수행됩니다.

중요

권장되는 모범 사례는 멱등 스택 업데이트를 실행하여 구성 정의와 실제 구성이 일치하는지 확인하는 것입니다. 스택 업데이트 기간은 시스템 크기에 따라 다르므로 다운타임을 줄이기 위해 유지 관리 기간 동안 마이그레이션을 완료할 수 있습니다.

25장. 업그레이드 후 작업 수행

오버클라우드 업그레이드를 완료한 후 몇 가지 업그레이드 후 구성을 수행하여 환경이 완전히 지원되고 향후 작업을 수행할 준비가 되었는지 확인해야 합니다.

25.1. 언더클라우드에서 불필요한 패키지 및 디렉토리 제거

Leapp을 업그레이드한 후 언더클라우드에 남아 있는 불필요한 패키지 및 디렉터리를 제거합니다.

절차

1. 불필요한 패키지 제거

```
$ sudo dnf -y remove --exclude=python-pycadf-common python2*
```

2. Red Hat OpenStack 13에 사용된 이전 이미지를 포함하는 **/httpboot 및 /tftpboot** 디렉토리에서 콘텐츠를 제거합니다.

```
$ sudo rm -rf /httpboot /tftpboot
```

25.2. 업그레이드 후 기능 검증

post-upgrade 검증 그룹을 실행하여 post-upgrade 기능을 확인합니다.

절차

1. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

2. **--group post-upgrade** 옵션을 사용하여 **openstack tripleo validator run 명령을 실행합니다**.

```
$ openstack tripleo validator run --group post-upgrade
```

3. 검증 보고서 결과를 확인하십시오. 특정 검증의 자세한 출력을 보려면 보고서의 특정 검증 UUID에 대해 **openstack tripleo validator show run --full** 명령을 실행합니다.

```
$ openstack tripleo validator show run --full <UUID>
```

중요

검증 결과가 **FAILED**이더라도 Red Hat OpenStack Platform 배포나 실행을 방해할 수 없습니다. 그러나 **FAILED** 검증 결과는 프로덕션 환경에서 잠재적으로 문제가 발행할 수 있다는 것을 의미합니다.

25.3. 오버클라우드 이미지 업그레이드

현재 오버클라우드 이미지를 새 버전으로 교체해야 합니다. 새 이미지를 사용하면 최신 버전의 OpenStack Platform 소프트웨어를 사용하여 director에서 노드를 세부 검사하고 프로비저닝할 수 있습니다.

사전 요구 사항

- 언더클라우드를 최신 버전으로 업그레이드했습니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

3. 오버클라우드 QCOW2 아카이브가 포함된 패키지를 설치합니다.

```
$ sudo dnf install rhosp-director-images rhosp-director-images-ipa
```

4. **stack** 사용자 홈(**/home/stack/images**)의 **images** 디렉터리에서 기존 이미지를 제거합니다.

```
$ rm -rf ~/images/*
```

5. 아카이브를 추출합니다.

```
$ cd ~/images
$ for i in /usr/share/rhosp-director-images/overcloud-full-latest-16.1.tar /usr/share/rhosp-director-images/ironic-python-agent-latest-16.1.tar; do tar -xvf $i; done
$ cd ~
```

6. 최신 이미지를 director로 가져옵니다.

```
$ openstack overcloud image upload --update-existing --image-path /home/stack/images/
```

7. 새 이미지를 사용하도록 노드를 구성합니다.

```
$ openstack overcloud node configure $(openstack baremetal node list -c UUID -f value)
```

중요

오버클라우드 노드를 배포할 때 오버클라우드 이미지 버전이 해당 heat 템플릿 버전에 해당하는지 확인합니다. 예를 들어 OpenStack Platform 16.1 heat 템플릿에서만 OpenStack Platform 16.1 이미지를 사용합니다.

중요

새 **overcloud-full** 이미지가 이전 **overcloud-full** 이미지를 대체합니다. 이전 이미지를 변경한 경우 특히 나중에 새 노드를 배포하려는 경우 새 이미지의 변경 사항을 반복해야 합니다.

25.4. CPU 고정 매개변수 업데이트

Red Hat OpenStack Platform 16.1에서는 CPU 고정에 새로운 매개변수를 사용합니다.

NovaComputeCpuDedicatedSet

전용(고정) CPU를 설정합니다.

NovaComputeCpuSharedSet

공유(고정되지 않은) CPU를 설정합니다.

Red Hat OpenStack Platform 16.1로 업그레이드를 완료한 후 **NovaVcpuPinSet** 매개변수에서 **NovaComputeCpuDedicatedSet** 및 **NovaComputeCpuSharedSet** 매개변수로 CPU 고정 구성을 마이그레이션해야 합니다.

절차

1. **stack** 사용자로 언더클라우드에 로그인합니다.
2. 컴퓨팅 노드에서 동시 멀티스레딩(SMT)을 지원하지만 **hw:cpu_thread_policy=isolate** 정책으로 인스턴스를 생성한 경우 다음 옵션 중 하나를 수행해야 합니다.
   - **hw:cpu_thread_policy** 스레드 정책을 설정 해제하고 인스턴스 크기를 조정합니다.
     i. 오버클라우드 인증 파일을 소싱합니다.

     ```
     $ source ~/overcloudrc
     ```

     ii. 플레이버의 **hw:cpu_thread_policy** 속성을 설정 해제합니다.

     ```
     (overcloud) $ openstack flavor unset --property hw:cpu_thread_policy <flavor>
     ```

     참고

     - **hw:cpu_thread_policy** 특성을 설정 해제하면 정책을 기본 **prefer** 정책으로 설정합니다. 이 정책은 사용 가능한 경우 SMT 사용 가능한 컴퓨팅 노드를 사용하도록 설정합니다. SMT 사용 컴퓨팅 노드의 하드 요구 사항을 설정하는 데 **필요한 hw:cpu_thread_policy** 특성을 설정할 수도 있습니다.
     - 컴퓨팅 노드에 SMT 아키텍처가 없거나 사용 가능한 스레드 스레딩이 있는 CPU 코어가 충분한 경우 예약에 실패합니다. 이를 방지하려면 **require** 대신 **hw:cpu_thread_policy** 를 **prefer** 로 설정합니다. 기본 **prefer** 정책은 사용 가능한 경우 스레드 시블링을 사용하도록 합니다.
     - **hw:cpu_thread_policy=isolate** 을 사용하는 경우 SMT를 비활성화하거나 SMT를 지원하지 않는 플랫폼을 사용해야 합니다.

     iii. 새 스레드 정책을 사용하도록 인스턴스를 변환합니다.

     ```
     (overcloud) $ openstack server resize --flavor <flavor> <server>
     (overcloud) $ openstack server resize confirm <server>
     ```

     hw:cpu_thread_policy=isolated 정책을 사용하여 고정된 모든 인스턴스에 대해 이 단계를 반복합니다.

   - 컴퓨팅 노드에서 인스턴스를 마이그레이션하고 컴퓨팅 노드에서 SMT를 비활성화합니다.
     i. 오버클라우드 인증 파일을 소싱합니다.

     ```
     $ source ~/overcloudrc
     ```

ii. 컴퓨팅 노드가 새 가상 머신을 수락하지 않도록 비활성화합니다.

```
(overcloud) $ openstack compute service list
(overcloud) $ openstack compute service set <hostname> nova-compute --disable
```

iii. 컴퓨팅 노드에서 모든 인스턴스를 마이그레이션합니다. 인스턴스 마이그레이션에 대한 자세한 내용은 컴퓨팅 노드 간 가상 머신 인스턴스 마이그레이션을 참조하십시오.

iv. Compute 노드를 재부팅하고 Compute 노드 BIOS에서 SMT를 비활성화합니다.

v. 컴퓨팅 노드를 부팅합니다.

vi. 컴퓨팅 노드를 다시 활성화합니다.

```
(overcloud) $ openstack compute service set <hostname> nova-compute --enable
```

3. **stackrc** 파일을 소싱합니다.

```
$ source ~/stackrc
```

4. **NovaVcpuPinSet** 매개 변수가 포함된 환경 파일을 편집합니다.

5. **NovaVcpuPinSet 매개변수에서 NovaComputeCpuDedicatedSet 및 NovaComputeCpuSharedSet** 로 CPU 고정 구성을 마이그레이션합니다.

- 고정 인스턴스에 사용된 호스트의 **NovaVcpuPinSet 값을 NovaComputeCpuDedicatedSet** 로 마이그레이션합니다.
- 이전에 고정되지 않은 인스턴스에 사용된 호스트의 경우 **NovaVcpuPinSet 의 값을 NovaComputeCpuSharedSet** 로 마이그레이션합니다.
- NovaVcpuPinSet에 대한 값이 설정되지 않은 경우 모든 컴퓨팅 노드 코어를 노드에서 호스트하려는 인스턴스 유형에 따라 **NovaComputeCpuDedicatedSet 또는 NovaComputeCpuSharedSet** 에 할당해야 합니다.

예를 들어 이전 환경 파일에 다음과 같은 고정 구성이 포함될 수 있습니다.

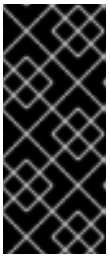
```
parameter_defaults:
  ...
  NovaVcpuPinSet: 1,2,3,5,6,7
  ...
```

구성을 고정 구성으로 마이그레이션하려면 **NovaComputeCpuDedicatedSet** 매개변수를 설정하고 **NovaVcpuPinSet 매개변수를 설정** 해제합니다.

```
parameter_defaults:
  ...
  NovaComputeCpuDedicatedSet: 1,2,3,5,6,7
  NovaVcpuPinSet: ""
  ...
```

구성을 고정 해제 구성으로 마이그레이션하려면 **NovaComputeCpuSharedSet** 매개변수를 설정하고 **NovaVcpuPinSet 매개변수를 설정** 취소합니다.

```
parameter_defaults:
  ...
  NovaComputeCpuSharedSet: 1,2,3,5,6,7
  NovaVcpuPinSet: ""
  ...
```



중요

NovaComputeCpuDedicatedSet 또는 **NovaComputeCpuSharedSet** 의 구성이 **NovaVcpuPinSet** 에 정의된 구성과 일치하는지 확인합니다. 이러한 구성을 변경하거나 **NovaComputeCpuDedicatedSet** 또는 **NovaComputeCpuSharedSet** 을 둘 다 구성하려면 고정 구성이 있는 컴퓨팅 노드가 구성을 업데이트하기 전에 인스턴스를 실행하지 않는지 확인합니다.

6. 파일을 저장합니다.

7. 배포 명령을 실행하여 새 CPU 고정 매개 변수로 Overcloud를 업데이트합니다.

```
(undercloud) $ openstack overcloud deploy \
  --stack _STACK NAME_ \
  --templates \
  ...
  -e /home/stack/templates/<compute_environment_file>.yaml
  ...
```

추가 리소스

- 컴퓨팅 노드에서 CPU 고정 구성

25.5. 멤버 역할로 사용자 마이그레이션

Red Hat OpenStack Platform 13에서 기본 멤버 역할을 **_member_** 라고 합니다.
Red Hat OpenStack Platform 16.1에서 기본 멤버 역할은 **member** 라고 합니다.

Red Hat OpenStack Platform 13에서 Red Hat OpenStack Platform 16.1로 업그레이드를 완료하면 **_member_** 역할에 할당된 사용자에게 여전히 해당 역할이 있습니다. 다음 단계를 사용하여 모든 사용자를 **member** 역할로 마이그레이션할 수 있습니다.

사전 요구 사항

- 오버클라우드를 최신 버전으로 업그레이드했습니다.

절차

1. 클라우드의 모든 사용자를 나열하십시오. **_member_** 역할이 있습니다.

```
openstack role assignment list --names --role _member_ --sort-column project
```

2. 각 사용자에 대해 **_member_** 역할을 제거하고 **member** 역할을 적용합니다.

```
openstack role remove --user <user> --project <project>  _member_
openstack role add --user <user> --project <project>  member
```

26장. 업그레이드 문제 해결

업그레이드 프로세스 중에 에 문제가 발생하는 경우 이 섹션의 지침을 참조하십시오.

26.1. 환경 파일 수정

사용자 지정 환경 파일의 매개변수가 실수한 경우 환경 파일을 수정하고 업그레이드 중에 언제든지 **openstack overcloud upgrade prepare** 명령을 실행할 수 있습니다. 이 명령은 새 버전의 오버클라우드 계획을 director에 업로드하여 새 **config-download** 플레이북 세트를 생성합니다.

이 예에서는 upgrade **-environment.yaml 파일에서 리포지터리 이름 실수가 있습니다.**

```
parameter_defaults:
  UpgradeLeappEnabled: true
  UpgradeLeappCommandOptions: "--enablerepo rhel-7-for-x86_64-baseos-eus-rpms --enablerepo
rhel-8-for-x86_64-appstream-eus-rpms --enablerepo fast-datapath-for-rhel-8-x86_64-rpms"
  CephAnsibleRepo: rhceph-4-tools-for-rhel-8-x86_64-rpms
```

이러한 실수로 인해 컨트롤러 노드의 Leapp 업그레이드 중에 문제가 발생합니다. 이 문제를 해결하려면 오류를 수정하고 **openstack overcloud upgrade prepare** 명령을 실행합니다.

절차

1. 파일에서 실수를 수정합니다.

```
parameter_defaults:
  UpgradeLeappEnabled: true
  UpgradeLeappCommandOptions: "--enablerepo rhel-8-for-x86_64-baseos-eus-rpms --
enablerepo rhel-8-for-x86_64-appstream-eus-rpms --enablerepo fast-datapath-for-rhel-8-
x86_64-rpms"
  CephAnsibleRepo: rhceph-4-tools-for-rhel-8-x86_64-rpms
```

2. 수정된 파일을 사용하여 업그레이드 준비 명령을 실행합니다.

```
$ openstack overcloud upgrade prepare \
   --stack STACK NAME \
   --templates \
   -e ENVIRONMENT FILE
   …
   -e /home/stack/templates/upgrades-environment.yaml \
   …
```

   오버클라우드 스택 업데이트가 완료될 때까지 기다립니다.

3. 실패한 업그레이드 작업 단계를 계속 진행합니다.