Red Hat OpenStack Platform 13

Load Balancing-as-a-Service에 Octavia 사용

Octavia 관리 및 octavia를 사용하여 데이터 플레인 전체의 네트워크 트래픽의 부하를 분산하는 방법.

Last Updated: 2023-05-25

Red Hat OpenStack Platform 13 Load Balancing-as-a-Service에 Octavia 사용

Octavia 관리 및 octavia를 사용하여 데이터 플레인 전체의 네트워크 트래픽의 부하를 분산하는 방법.

OpenStack Team
rhos-docs@redhat.com

법적 공지

초록

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 설치, 구성, 작동, 문제 해결 및 업그레이드합니다.

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. 로드 밸런싱 서비스 소개

로드 밸런싱 서비스(octavia)는 RHOSP(Red Hat OpenStack Platform) 배포를 위한 LBaaS(Load Balancing-as-a-Service) API 버전 2 구현을 제공합니다. 로드 밸런싱 서비스는 여러 가상 머신, 컨테이너 또는 베어 메탈 서버를 관리합니다.집합적으로 amphorae-온디맨드로 시작하는 방법. 온디맨드 수평 확장을 제공하는 기능 덕분에 로드 밸런싱 서비스는 대규모 RHOSP 엔터프라이즈 배포에 적합한 완전한 기능을 갖춘 로드 밸런서 장치로 만듭니다.

참고

Red Hat은 Neutron-LBaaS에서 부하 분산 서비스로의 마이그레이션 경로를 지원하지 않습니다. 지원되지 않는 일부 오픈 소스 도구를 사용할 수 있습니다. 예를 들어 GitHub에서 nlbaas2octavia-lb-replicator를 검색합니다.

- 1.1절. "서비스 구성 요소 로드 밸런싱"
- 1.2절. "로드 밸런싱 서비스 오브젝트 모델"
- 1.3절. "Red Hat OpenStack Platform에서 로드 밸런싱 사용"

1.1. 서비스 구성 요소 로드 밸런싱

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)는 컴퓨팅 노드에 상주하는 *amphorae* 라는 VM 인스턴스 세트를 사용합니다. 로드 밸런싱 서비스 컨트롤러는 로드 밸런싱 관리 네트워크(**lb-mgmt-net**)를 통해 amphorae와 통신합니다.

octavia를 사용하는 경우 유동 IP(FIP)가 필요하지 않은 로드 밸런서 가상 IP(VIP)를 만들 수 있습니다. FIP를 사용하지 않는 경우 로드 밸런서를 통해 성능을 향상시키는 이점이 있습니다.

그림 1.1. 서비스 구성 요소 로드 밸런싱

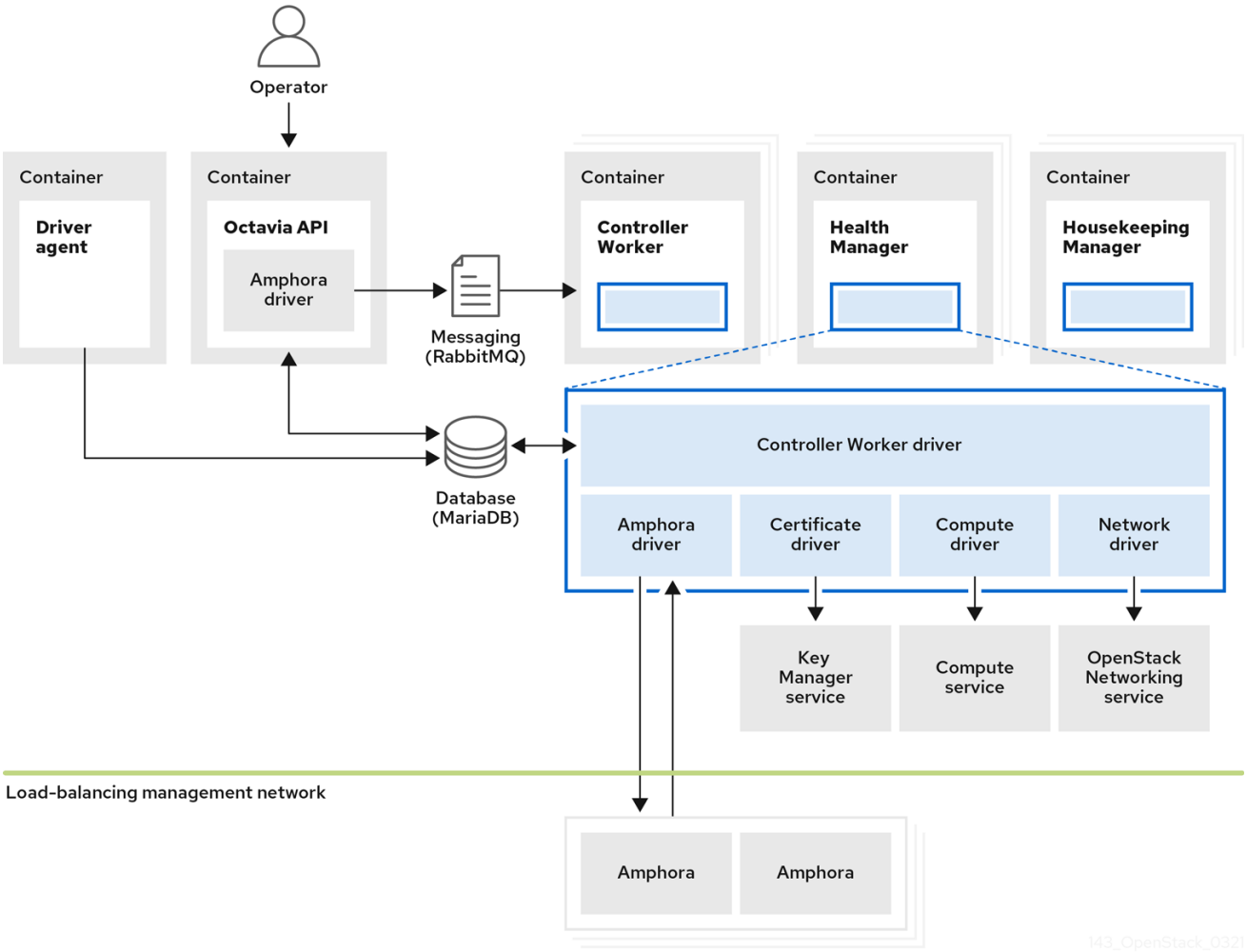


그림 1.1은 로드 밸런싱 서비스의 구성 요소가 기본적으로 컨트롤러 노드에 있는 Networking API 서버와 동일한 노드에서 호스팅됩니다. 로드 밸런싱 서비스는 다음 구성 요소로 구성됩니다.

Octavia API(octavia_api 컨테이너)

사용자가 octavia와 상호 작용할 수 있도록 REST API를 제공합니다.

컨트롤러 작업자 (octavia_worker 컨테이너)

부하 분산 관리 네트워크를 통해 구성 및 구성 업데이트를 Amphorae로 보냅니다.

Health Manager (octavia_health_manager 컨테이너)

개별 Amphorae의 상태를 모니터링하고 Amphora에 장애가 발생하면 장애 조치 이벤트를 처리합니다.

하우스키핑 관리자 (octavia_housekeeping container)

삭제된 데이터베이스 레코드를 정리하고 Amphora 인증서 교체를 관리합니다.

드라이버 에이전트(octavia_driver_agent 컨테이너)

OVN과 같은 기타 프로바이더 드라이버를 지원합니다.

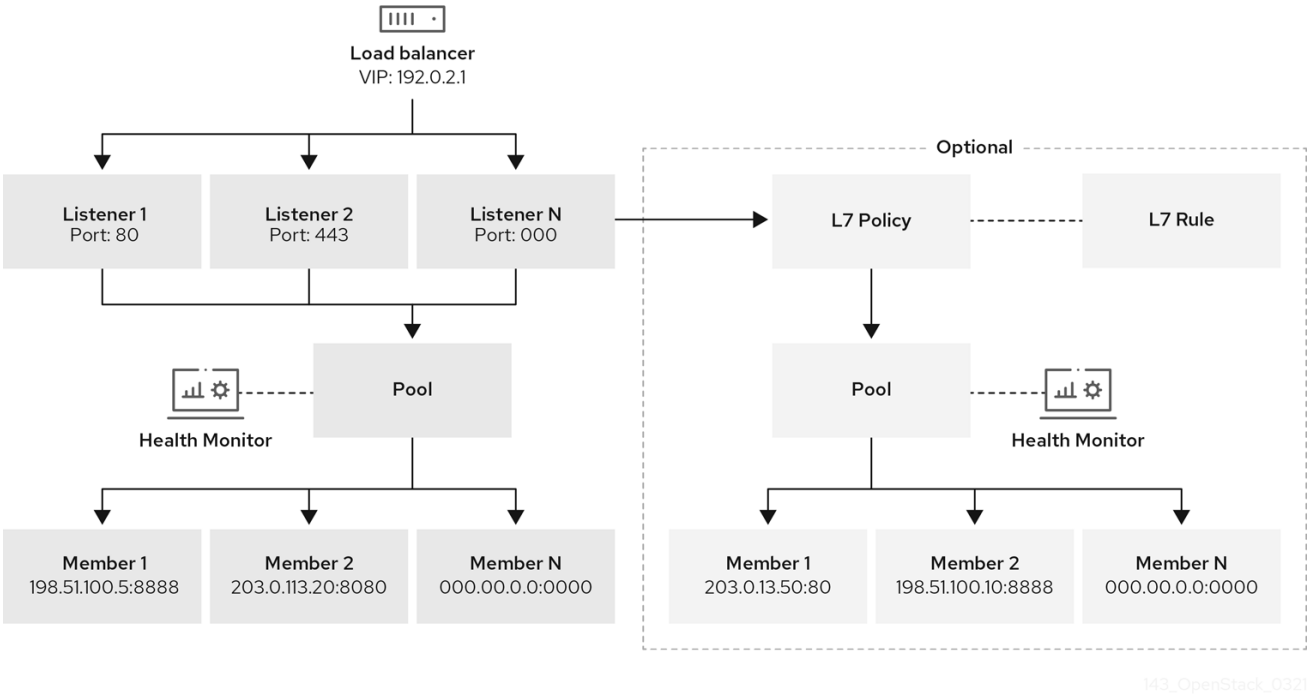
Amphora

부하 분산을 수행합니다. Amphorae는 일반적으로 리스너, 풀, 상태 모니터, L7 정책 및 멤버 구성에 따라 부하 분산 매개 변수를 사용하여 구성하는 컴퓨팅 노드에서 실행되는 인스턴스입니다. Amphorae는 정기적인 하트비트를 Health Manager로 전송합니다.

1.2. 로드 밸런싱 서비스 오브젝트 모델

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)는 일반적인 로드 밸런싱 오브젝트 모델을 사용합니다.

그림 1.2. 로드 밸런싱 서비스 개체 모델 다이어그램



로드 밸런서

로드 밸런싱 엔터티를 나타내는 최상위 API 오브젝트입니다. 로드 밸런서를 생성할 때 VIP 주소가 할당됩니다. Amphora 공급자를 사용하여 하나 이상의 컴퓨팅 노드에서 하나 이상의 Amphora 인스턴스가 시작되는 로드 밸런서 장치를 생성합니다.

리스너

로드 밸런서가 수신하는 포트(예: HTTP의 경우 TCP 포트 80).

상태 모니터

각 백엔드 구성원 서버에서 정기적인 상태 점검을 수행하여 실패한 서버를 미리 탐지하고 풀에서 일시적으로 제거하는 프로세스입니다.

풀

로드 밸런서의 클라이언트 요청을 처리하는 멤버 그룹입니다. API를 사용하여 두 개 이상의 리스너와 풀을 연결할 수 있습니다. 풀을 L7 정책과 공유할 수 있습니다.

멤버

백엔드 인스턴스 또는 서비스에 연결하는 방법을 설명합니다. 이 설명은 백엔드 멤버를 사용할 수 있는 IP 주소 및 네트워크 포트로 구성됩니다.

L7 규칙

L7 정책이 연결에 적용되는지 여부를 결정하는 계층 7(L7) 조건을 정의합니다.

L7 정책

리스너와 관련된 L7 규칙 컬렉션으로, 백엔드 풀에 대한 연관이 있을 수도 있습니다. 정책은 정책의 모든 규칙이 true인 경우 로드 밸런서에서 수행하는 작업을 설명합니다.

추가 리소스

- 1.1절. "서비스 구성 요소 로드 밸런싱"

1.3. RED HAT OPENSTACK PLATFORM에서 로드 밸런싱 사용

클라우드 배포를 위한 단순하거나 자동 제공 확장 및 가용성을 활성화하려면 로드 밸런싱이 필요합니다. 로드 밸런싱 서비스(octavia)는 다른 RHOSP(Red Hat OpenStack Platform) 서비스에 따라 다릅니다.

- **Compute 서비스(nova)** - 부하 분산 서비스 VM 인스턴스(amphora) 라이프사이클을 관리하고 필요에 따라 컴퓨팅 리소스를 생성합니다.
- **네트워킹 서비스(neutron)** - Amphorae, 테넌트 환경 및 외부 네트워크 간의 네트워크 연결.
- **Key Manager 서비스(barbican)** - 리스너에서 TLS 세션 종료가 구성된 경우 TLS 인증서 및 자격 증명을 관리합니다.
- **ID 서비스(keystone)** - octavia API에 대한 인증 요청 및 로드 밸런싱 서비스의 경우 다른 RHOSP 서비스로 인증합니다.
- **Image 서비스(glance)** - Amphora 가상 머신 이미지를 저장합니다.
- **공통 라이브러리(oslo)** - 부하 분산 서비스 컨트롤러 구성 요소 간의 통신으로, 로드 밸런싱 서비스가 표준 OpenStack 프레임워크 내에서 작동되고 시스템, 프로젝트 코드 구조 검토.
- **Taskflow** - 일반적인 라이브러리의 일부이며 부하 분산 서비스는 백엔드 서비스 구성 및 관리를 오케스트레이션할 때 이 작업 흐름 시스템을 사용합니다.

로드 밸런싱 서비스는 드라이버 인터페이스를 통해 다른 RHOSP 서비스와 상호 작용합니다. 외부 구성 요소를 기능적으로 동등한 서비스로 교체해야 하는 경우 드라이버 인터페이스에서 로드 밸런싱 서비스의 대규모 재구성을 방지합니다.

2장. 로드 밸런싱 서비스 구현을 위한 고려 사항

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 배포할 때 사용할 공급자 또는 고가용성 환경 구현 여부를 선택하는 등 몇 가지 결정을 내려야 합니다.

- 2.1절. "로드 밸런싱 서비스 공급자 드라이버"
- 2.2절. "로드 밸런싱 서비스(octavia) 기능 지원 매트릭스"
- 2.3절. "서비스 소프트웨어 요구사항 로드 밸런싱"
- 2.4절. "언더클라우드에 대한 서비스 사전 요구 사항 로드 밸런싱"
- 2.5절. "로드 밸런싱 서비스 인스턴스용 active-standby 토폴로지의 기본 사항"
- 2.6절. "로드 밸런싱 서비스의 배포 후 단계"

2.1. 로드 밸런싱 서비스 공급자 드라이버

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)는 Octavia v2 API를 사용하여 여러 공급자 드라이버 활성화를 지원합니다. 하나의 공급자 드라이버 또는 여러 공급자 드라이버를 동시에 사용하도록 선택할 수 있습니다.

RHOSP에서는 로드 밸런싱 공급자 Amphora 및 OVN(Open Virtual Network)을 제공합니다.

기본값인 Amphora는 컴퓨팅 환경과 함께 확장되는 기능 세트가 있는 고가용성 로드 밸런서 장치입니다. 이로 인해 Amphora는 대규모 배포에 적합합니다.

OVN 로드 밸런싱 공급자는 기본 기능 세트가 있는 경량 로드 밸런서입니다. OVN은 동-서 계층 네트워크 트래픽에 일반적으로 사용됩니다. OVN은 신속하게 프로비저닝하고 amphora와 같은 완전한 기능을 갖춘 부하 분산 공급자보다 적은 리소스를 사용합니다.

neutron 모듈 계층 2 플러그인을 OVN 메커니즘 드라이버(ML2/OVN)와 함께 사용하는 RHOSP 배포에서 RHOSP director는 추가 설치 또는 구성 없이도 로드 밸런싱 서비스에서 OVN 공급자 드라이버를 자동으로 활성화합니다.

중요

이 섹션의 정보는 별도로 명시하지 않는 한 Amphora 부하 분산 공급자에만 적용됩니다.

추가 리소스

- 2.2절. "로드 밸런싱 서비스(octavia) 기능 지원 매트릭스"

2.2. 로드 밸런싱 서비스(OCTAVIA) 기능 지원 매트릭스

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)는 두 개의 로드 밸런싱 공급자인 amphora 및 OVN(Open Virtual Network)을 제공합니다.

Amphora 는 완전한 기능을 갖춘 로드 밸런싱 공급자로서 별도의 haproxy VM과 추가 대기 시간 홉이 필요합니다.

OVN 은 모든 노드에서 실행되며 별도의 VM이나 추가 홉이 필요하지 않습니다. 그러나 OVN에는 amphora보다 로드 밸런싱 기능이 훨씬 적습니다.

다음 표에는 RHOSP(Red Hat OpenStack Platform) 13에서 지원하는 및 기능에 대한 유지 관리 릴리스 지원이 시작된 octavia의 기능이 나열되어 있습니다.

참고

기능이 나열되지 않으면 RHOSP 13에서 이 기능을 지원하지 않습니다.

표 2.1. 로드 밸런싱 서비스(octavia) 기능 지원 매트릭스

기능	RHOSP 13 릴리스의 지원 수준	
	Amphora 공급자	OVN 공급자
ML2/OVS L3 HA	완전 지원 (완전한 지원)13.0 및 모든 유지 관리 릴리스	해당 없음
ML2/OVS DVR	완전 지원 (완전한 지원)2018년 8월 29일 유지 관리 릴리스 및 이후	해당 없음
ML2/OVS L3 HA + 구성 가능한 네트워크 노드 [1]	완전 지원 (완전한 지원)2019년 3월 13일 유지 관리 릴리스 및 이후	해당 없음
ML2/OVS DVR + 구성 가능한 네트워크 노드 [1]	완전 지원 (완전한 지원)2019년 3월 13일 유지 관리 릴리스 및 이후	해당 없음
ML2/OVN L3 HA	완전 지원 (완전한 지원)2018년 8월 29일 유지 관리 릴리스 및 이후	완전 지원-28 2020년 10월 유지 관리 릴리스 이상
ML2/OVN DVR	완전 지원 (완전한 지원)2018년 11월 13일 유지 관리 릴리스 및 이후	완전 지원-28 2020년 10월 유지 관리 릴리스 이상
ML2/ODL	완전 지원 (완전한 지원)2019년 1월 16일 유지 관리 릴리스 및 이후	해당 없음
Amphora active-standby	완전 지원-28 2020년 10월 유지 관리 릴리스 및 이후	지원되지 않음
종료된 HTTPS 로드 밸런서	완전 지원 (완전한 지원)2020년 3월 10일 유지 관리 릴리스 및 이후	지원되지 않음
Amphora 예비 풀	기술 프리뷰 전용2019년 4월 30일 유지 관리 릴리스 및 이후	해당 없음
UDP	완전 지원-28 2020년 10월 유지 관리 릴리스 이상	완전 지원-28 2020년 10월 유지 관리 릴리스 이상
플레이버	기술 프리뷰 전용2020년 10월 28일 유지 관리 릴리스 및 이후	해당 없음

ACL을 사용하여 로드 밸런서 생성	기술 프리뷰 전용2020년 10월 28일 유지 관리 릴리스 및 이후	해당 없음

[1] OVS, 메타데이터, DHCP, L3 및 Octavia가 있는 네트워크 노드(작업자, 상태 모니터, 하우스키핑).

추가 리소스

- 2.1절. "로드 밸런싱 서비스 공급자 드라이버"

2.3. 서비스 소프트웨어 요구사항 로드 밸런싱

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)에는 다음과 같은 핵심 OpenStack 구성 요소를 구성해야 합니다.

- 컴퓨팅(nova)
- OpenStack Networking(neutron)
- 이미지 (glance)
- ID(keystone)
- RabbitMQ
- MySQL

2.4. 언더클라우드에 대한 서비스 사전 요구 사항 로드 밸런싱

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)에는 RHOSP 언더클라우드에 대한 다음과 같은 요구 사항이 있습니다.

- 성공적인 언더클라우드 설치
- 언더클라우드에 있는 로드 밸런싱 서비스.
- 컨테이너 기반 Overcloud 배포 계획.
- 컨트롤러 노드에 구성된 로드 밸런싱 서비스 구성 요소.

중요

기존 오버클라우드 배포에서 로드 밸런싱 서비스를 활성화하려면 언더클라우드를 준비해야 합니다. 이렇게 하지 않으면 부하 분산 서비스를 실행하지 않고 Overcloud 설치가 성공으로 보고됩니다. 언더클라우드를 준비하려면 *컨테이너화된 서비스로 전환 가이드를* 참조하십시오.

2.5. 로드 밸런싱 서비스 인스턴스용 ACTIVE-STANDBY 토폴로지의 기본 사항

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 배포할 때 기본적으로 사용자가 로드 밸런서를 생성할 때 로드 밸런서가 고가용성인지 여부를 결정할 수 있습니다. 사용자에게 선택권을

부여하려면 RHOSP 배포 후 독립 실행형 로드 밸런서를 생성하기 위한 고가용성 로드 밸런서 및 플레이버를 생성하는 로드 밸런싱 서비스 플레이버를 생성합니다.

기본적으로 Amphora 공급자 드라이버는 HA(고가용성)에 대한 지원이 제한된 단일 로드 밸런싱 서비스(amphora) 인스턴스 토폴로지에 대해 구성됩니다. 그러나 활성-대기 토폴로지를 구현할 때 로드 밸런싱 서비스 인스턴스를 고가용성으로 만들 수 있습니다.

이 토폴로지에서 로드 밸런싱 서비스는 각 로드 밸런서에 대해 활성 및 대기 중인 인스턴스를 부팅하고 각 로드 밸런서 간에 세션 지속성을 유지합니다. 활성 인스턴스가 비정상이 되면 인스턴스가 대기 인스턴스로 자동으로 실패하여 활성 상태가 됩니다. 로드 밸런싱 서비스 상태 관리자는 오류가 발생한 인스턴스를 자동으로 다시 빌드합니다.

추가 리소스

- 4.2절. "로드 밸런싱 서비스 인스턴스를 위한 active-standby 토폴로지 활성화"

2.6. 로드 밸런싱 서비스의 배포 후 단계

RHOSP(Red Hat OpenStack Platform)는 로드 밸런싱 서비스(octavia)의 배포 후 단계를 간소화하는 워크플로 작업을 제공합니다. 이 워크플로는 Ansible 플레이북 세트를 실행하여 오버클라우드 배포의 마지막 단계로 다음과 같은 배포 후 단계를 제공합니다.

- 인증서 및 키 구성.
- Amphorae와 로드 밸런싱 서비스 컨트롤러 작업자 및 상태 관리자 간에 부하 분산 관리 네트워크를 구성합니다.

Amphora 이미지

사전 프로비저닝된 서버에서는 로드 밸런싱 서비스를 배포하기 전에 언더클라우드에 Amphora 이미지를 설치해야 합니다.

```
$ sudo dnf install octavia-amphora-image-x86_64.noarch
```

사전 프로비저닝되지 않은 서버에서 RHOSP director는 기본 amphora 이미지를 자동으로 다운로드하고 오버클라우드 이미지 서비스(glance)에 업로드한 다음 이 Amphora 이미지를 사용하도록 로드 밸런싱 서비스를 구성합니다. 스택 업데이트 또는 업그레이드 중에 director는 이 이미지를 최신 amphora 이미지로 업데이트합니다.

참고

사용자 지정 Amphora 이미지는 지원되지 않습니다.

추가 리소스

- 4.1절. "로드 밸런싱 서비스 배포"

3장. 로드 밸런싱 서비스 보안

Red Hat OpenStack 로드 밸런싱 서비스(octavia)의 다양한 구성 요소 간 통신을 보호하려면 TLS 암호화 프로토콜과 공개 키 암호화를 사용합니다.

- 3.1절. "로드 밸런싱 서비스의 양방향 TLS 인증"
- 3.2절. "로드 밸런싱 서비스의 인증서 라이프사이클"
- 3.3절. "로드 밸런싱 서비스 인증서 및 키 구성"

3.1. 로드 밸런싱 서비스의 양방향 TLS 인증

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)의 컨트롤러 프로세스는 TLS 연결을 통해 로드 밸런싱 서비스 인스턴스(amphorae)와 통신합니다. 부하 분산 서비스는 양방향 TLS 인증을 사용하여 양쪽이 모두 신뢰되는지 확인합니다.

참고

이는 전체 TLS 핸드셰이크 프로세스를 단순화한 것입니다. TLS 핸드셰이크 프로세스에 대한 자세한 내용은 TLS 1.3 RFC 330646을 참조하십시오.

양방향 TLS 인증에는 두 가지 단계가 포함됩니다. 로드 밸런싱 서비스 작업자 프로세스와 같은 컨트롤러 프로세스의 *1* 단계에서는 로드 밸런싱 서비스 인스턴스에 연결하고 인스턴스에서 해당 서버 인증서를 컨트롤러에 제공합니다. 그런 다음 컨트롤러는 컨트롤러에 저장된 서버 CA(인증 기관) 인증서에 대해 서버 인증서의 유효성을 검사합니다. 표시된 인증서의 유효성이 서버 CA 인증서에 대해 확인되면 연결은 2단계로 진행됩니다.

2단계에서 컨트롤러는 클라이언트 인증서를 로드 밸런싱 서비스 인스턴스에 제공합니다. 그런 다음 인스턴스는 인스턴스 내부에 저장된 클라이언트 CA 인증서에 대해 인증서의 유효성을 검사합니다. 이 인증서의 유효성을 검사하면 나머지 TLS 핸드셰이크는 컨트롤러와 로드 밸런싱 서비스 인스턴스 간에 보안 통신 채널을 계속 설정합니다.

추가 리소스

- 3.3절. "로드 밸런싱 서비스 인증서 및 키 구성"

3.2. 로드 밸런싱 서비스의 인증서 라이프사이클

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia) 컨트롤러는 서버 인증 기관 인증서 및 키를 사용하여 각 로드 밸런싱 서비스 인스턴스(amphora)에 대한 인증서를 고유하게 생성합니다.

로드 밸런싱 서비스 하우스키핑 컨트롤러 프로세스는 만료 날짜 가까이에 따라 이러한 서버 인증서를 자동으로 순환합니다.

로드 밸런싱 서비스 컨트롤러 프로세스는 클라이언트 인증서를 사용합니다. 이러한 TLS 인증서를 관리하는 사람 운영자는 일반적으로 클라우드 컨트롤 플레인에서 인증서를 사용하므로 만료 기간이 길어집니다.

추가 리소스

- 3.3절. "로드 밸런싱 서비스 인증서 및 키 구성"

3.3. 로드 밸런싱 서비스 인증서 및 키 구성

RHOSP(Red Hat OpenStack Platform) director를 구성하여 인증서 및 키를 생성하거나 자체 제공할 수 있습니다. 필요한 개인 인증 기관을 자동으로 생성하고 필요한 인증서를 발급하도록 director를 구성합니다. 이러한 인증서는 내부 로드 밸런싱 서비스(octavia) 통신 전용으로, 사용자에게 노출되지 않습니다.

중요

RHOSP director는 인증서와 키를 생성하고 만료되기 전에 자동으로 갱신합니다. 자체 인증서를 사용하는 경우 갱신해야 합니다.

참고

RHOSP director는 수동으로 생성된 인증서에서 자동으로 생성된 인증서로 전환할 수 없습니다. 그러나 **/var/lib/config-data/puppet-generated/octavia/etc/octavia/certs** 디렉터리에서 컨트롤러 노드에서 기존 인증서를 삭제하고 오버클라우드를 업데이트하여 인증서를 다시 만들 수 있습니다.

자체 인증서 및 키를 사용해야 하는 경우 다음 단계를 완료합니다.

사전 요구 사항

- 읽기 및 이해, "로드 밸런싱 서비스 기본 설정 변경" ("정보 리소스의 링크 참조")

절차

1. 언더클라우드 호스트에 **stack** 사용자로 로그인합니다.
2. 언더클라우드 인증 정보 파일을 소싱합니다.

```
$ source ~/stackrc
```

3. YAML 사용자 지정 환경 파일을 생성합니다.

예제

```
$ vi /home/stack/templates/my-octavia-environment.yaml
```

4. YAML 환경 파일에서 사이트에 적합한 값을 사용하여 다음 매개변수를 추가합니다.

- **OctaviaCaCert**:
  Octavia가 인증서를 생성하는 데 사용하는 CA의 인증서입니다.
- **OctaviaCaKey**:
  생성된 인증서에 서명하는 데 사용되는 개인 CA 키입니다.
- **OctaviaCaKeyPassphrase**:
  위의 개인 CA 키에 사용되는 암호입니다.
- **OctaviaClientCert**:
  컨트롤러에 대해 Octavia CA에서 발급한 클라이언트 인증서 및 암호화되지 않은 키입니다.
- **OctaviaGenerateCerts**:
  director에 자동 인증서 및 키 생성을 활성화하거나 비활성화(false)하도록 지시하는 부울입니다.

중요

OctaviaGenerateCerts 를 false로 설정해야 합니다.

예제

```
parameter_defaults:
  OctaviaCaCert: |
    -----BEGIN CERTIFICATE-----

MIIDgzCCAmugAwIBAgIJAKk46qw6ncJaMA0GCSqGSIb3DQEBCwUAMFgxCzAJBgNV
    [snip]
    sFW3S2roS4X0Af/kSSD8mlBBTFTCMBAj6rtLBKLaQbIxEpIzrgvp
    -----END CERTIFICATE-----

  OctaviaCaKey: |
    -----BEGIN RSA PRIVATE KEY-----
    Proc-Type: 4,ENCRYPTED
    [snip]
    -----END RSA PRIVATE KEY-----[

  OctaviaClientCert: |
    -----BEGIN CERTIFICATE-----

MIIDmjCCAoKgAwIBAgIBATANBgkqhkiG9w0BAQsFADBcMQswCQYDVQQGEwJVUzEP

    [snip]
    270l5ILSnfejLxDH+vI=
    -----END CERTIFICATE-----
    -----BEGIN PRIVATE KEY-----

MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQDU771O8MTQV8RY

    [snip]
    KfrjE3UqTF+ZaaIQaz3yayXW
    -----END PRIVATE KEY-----

  OctaviaCaKeyPassphrase:
    b28c519a-5880-4e5e-89bf-c042fc75225d

  OctaviaGenerateCerts: false
  [rest of file snipped]
```

5. **openstack overcloud deploy** 명령을 실행하고 코어 heat 템플릿, 환경 파일 및 이 새 사용자 지정 환경 파일을 포함합니다.

중요

후속 환경 파일에 정의된 매개 변수와 리소스가 우선하므로 환경 파일의 순서가 중요합니다.

예제

```
$ openstack overcloud deploy --templates \
```

```
-e <your_environment_files> \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/octavia.yaml \
-e /home/stack/templates/my-octavia-environment.yaml
```

추가 리소스

- 4.3절. "로드 밸런싱 서비스 기본 설정 변경"
- *Advanced Overcloud Customization* 가이드의 환경 파일
- *Advanced Overcloud Customization* 가이드 의 Overcloud 생성에 환경 파일 포함

4장. 로드 밸런싱 서비스 설치 및 구성

RHOSP director를 사용하여 RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 배포할 때 VM 인스턴스를 고가용성으로 설정할 수 있습니다. 로드 밸런싱 서비스를 구성하려는 경우에도 director를 사용합니다.

4.1. 로드 밸런싱 서비스 배포

RHOSP(Red Hat OpenStack Platform) director를 사용하여 로드 밸런싱 서비스(octavia)를 배포합니다. director는 사용자 환경에 대한 계획 집합인 Orchestration 서비스(heat) 템플릿을 사용합니다. 언더클라우드는 이러한 계획을 가져와서 해당 지침에 따라 로드 밸런싱 서비스와 RHOSP 환경을 생성합니다.

사전 요구 사항

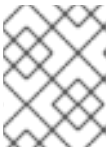
- 환경에서 octavia 이미지에 액세스할 수 있는지 확인합니다.

절차

- 배포 명령을 실행하고 코어 heat 템플릿, 환경 파일, the **octavia.yaml** heat 템플릿을 포함합니다.

  예제

  ```
  $ openstack overcloud deploy --templates \
  -e <your_environment_files> \
  -e /usr/share/openstack-tripleo-heat-templates/environments/services/octavia.yaml
  ```

  참고

  director는 스택을 업데이트하거나 업그레이드하는 동안 Amphorae를 최신 Amphora 이미지로 업데이트합니다.

추가 리소스

- *Director 설치 및 사용* 가이드의 배포 명령 옵션

4.2. 로드 밸런싱 서비스 인스턴스를 위한 ACTIVE-STANDBY 토폴로지 활성화

RHOSP(Red Hat OpenStack Platform) director를 사용하여 활성 대기 토폴로지를 구현할 때 로드 밸런싱 서비스 인스턴스(amphorae)를 고가용성으로 만들 수 있습니다. director는 사용자 환경에 대한 계획 집합인 Orchestration 서비스(heat) 템플릿을 사용합니다. 언더클라우드는 이러한 계획을 가져와서 해당 지침에 따라 로드 밸런싱 서비스와 RHOSP 환경을 생성합니다.

사전 요구 사항

- 계산 서비스에 대해 유사성 방지가 활성화되었는지 확인합니다. 이는 기본값입니다.

- 컴퓨팅 노드 호스트 3개 이상:
  - 서로 다른 호스트(Compute anti-affinity)에 Amphorae를 배치할 두 개의 컴퓨팅 노드 호스트입니다.
  - 문제가 발생할 때 활성 로드 밸런서를 통해 장애 조치를 수행하는 세 번째 호스트입니다.

절차

1. 언더클라우드 호스트에 **stack** 사용자로 로그인합니다.
2. 언더클라우드 인증 정보 파일을 소싱합니다.

```
$ source ~/stackrc
```

3. 사용자 지정 YAML 환경 파일을 생성합니다.

예제

```
$ vi /home/stack/templates/my-octavia-environment.yaml
```

4. 사용자 지정 환경 파일에서 다음 매개변수를 추가합니다.

```
parameter_defaults:
  OctaviaLoadBalancerTopology: "ACTIVE_STANDBY"
```

5. 배포 명령을 실행하고 핵심 heat 템플릿, 환경 파일 및 이 새 사용자 지정 환경 파일을 포함합니다.

중요

후속 환경 파일에 정의된 매개 변수와 리소스가 우선하므로 환경 파일의 순서가 중요합니다.

예제

```
$ openstack overcloud deploy --templates \
-e <your_environment_files> \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/octavia.yaml \
-e /home/stack/templates/my-octavia-environment.yaml
```

검증 단계

- 배포가 완료되고 로드 밸런서를 생성한 후 다음 명령을 실행합니다.

```
$ source overcloudrc
$ openstack loadbalancer amphora list
```

로드 밸런싱 서비스 인스턴스의 고가용성 구성에 성공하면 두 개의 인스턴스(amphorae)의 출력이 표시되고 **SINGLE** 과 동일한 **역할이** 발생하지 않습니다.

추가 리소스

- *Advanced Overcloud Customization* 가이드의 환경 파일

- *Advanced Overcloud Customization* 가이드 의 Overcloud 생성에 환경 파일 포함

4.3. 로드 밸런싱 서비스 기본 설정 변경

RHOSP(Red Hat OpenStack Platform) director를 사용하여 로드 밸런싱 서비스(octavia)의 구성을 변경합니다. director는 사용자 환경에 대한 계획 집합인 Orchestration 서비스(heat) 템플릿을 사용합니다. 언더클라우드는 이러한 계획을 가져와서 해당 지침에 따라 로드 밸런싱 서비스와 RHOSP 환경을 생성합니다.

사전 요구 사항

- 언더클라우드에서 다음 파일을 참조하여 director에서 로드 밸런싱 서비스를 배포하는 데 이미 사용하는 RHOSP Orchestration 서비스(heat) 매개변수를 확인합니다.

```
/usr/share/openstack-tripleo-heat-templates/deployment/octavia/octavia-deployment-config.j2.yaml
```

- 수정할 매개변수를 결정합니다.
  다음은 몇 가지 예입니다.
  - **OctaviaControlNetwork**
    로드 밸런서 관리 네트워크에 사용되는 neutron 네트워크의 이름입니다.
  - **OctaviaControlSubnetCidr**
    CIDR 형식으로 amphora 제어 서브넷의 서브넷입니다.
  - **OctaviaMgmtPortDevName**
    octavia worker/health-manager와 amphora 머신 간 통신에 사용되는 octavia 관리 네트워크 인터페이스의 이름입니다.

절차

1. 언더클라우드 호스트에 **stack** 사용자로 로그인합니다.
2. 언더클라우드 인증 정보 파일을 소싱합니다.

```
$ source ~/stackrc
```

3. 사용자 지정 YAML 환경 파일을 생성합니다.

   예제

```
$ vi /home/stack/templates/my-octavia-environment.yaml
```

4. 환경 파일에는 parameters **_defaults 키워드가** 포함되어야 합니다. parameter **_defaults 키워드 뒤에 매개** 변수 값 쌍을 넣습니다.

   예제

```
parameter_defaults:
  OctaviaMgmtPortDevName: "o-hm0"
  OctaviaControlNetwork: 'lb-mgmt-net'
  OctaviaControlSubnet: 'lb-mgmt-subnet'
  OctaviaControlSecurityGroup: 'lb-mgmt-sec-group'
```

```
OctaviaControlSubnetCidr: '172.24.0.0/16'
OctaviaControlSubnetGateway: '172.24.0.1'
OctaviaControlSubnetPoolStart: '172.24.0.2'
OctaviaControlSubnetPoolEnd: '172.24.255.254'
```

5. 배포 명령을 실행하고 핵심 heat 템플릿, 환경 파일 및 이 새 사용자 지정 환경 파일을 포함합니다.

중요

후속 환경 파일에 정의된 매개 변수와 리소스가 우선하므로 환경 파일의 순서가 중요합니다.

예제

```
$ openstack overcloud deploy --templates \
-e <your_environment_files> \
-e /usr/share/openstack-tripleo-heat-templates/environments/services/octavia.yaml \
-e /home/stack/templates/my-octavia-environment.yaml
```

추가 리소스

- *Advanced Overcloud Customization* 가이드의 환경 파일
- *Advanced Overcloud Customization* 가이드 의 Overcloud 생성에 환경 파일 포함

5장. 로드 밸런싱 서비스 플레이버 구성

로드 밸런싱 서비스(octavia) *플레이버*는 생성한 공급자 구성 옵션의 집합입니다. 사용자가 로드 밸런서를 요청할 때 정의된 플레이버 중 하나를 사용하여 로드 밸런서를 빌드하도록 지정할 수 있습니다. 각 공급업체의 고유한 기능을 표시하는 각 부하 분산 프로바이더 드라이버에 대한 플레이버를 정의합니다.

새 로드 밸런싱 서비스 플레이버를 생성하려면 다음을 수행합니다.

1. 플레이버에서 구성할 로드 밸런싱 프로바이더의 기능을 결정합니다.
2. 선택한 플레이버 기능을 사용하여 플레이버 프로필을 생성합니다.
3. 플레이버를 만듭니다.

중요

이 섹션에 포함된 플레이버에 대한 주제는 *Red Hat OpenStack Platform 13, 2020년 10월 28일 유지 관리 릴리스 등의 기술* 검토에 있습니다. 기술 프리뷰로 표시된 기능의 지원 범위에 대한 자세한 내용은 기술 프리뷰 기능 지원 범위 를 참조하십시오.

5.1. 로드 밸런싱 서비스 공급자 기능 나열

각 로드 밸런싱 서비스(octavia) 공급자 드라이버에서 노출하는 기능 목록을 검토할 수 있습니다.

중요

이 섹션에 설명된 플레이버 기능은 Red Hat OpenStack Platform 13, 2020년 10월 28일 유지 관리 릴리스 등의 *기술 프리뷰* 에 있습니다. 기술 프리뷰로 표시된 기능의 지원 범위에 대한 자세한 내용은 기술 프리뷰 기능 지원 범위 를 참조하십시오.

사전 요구 사항

- OpenStack 관리자 권한이 있어야 합니다.

절차

1. 언더클라우드 호스트에 **stack** 사용자로 로그인합니다.
2. 언더클라우드 인증 정보 파일을 소싱합니다.

```
$ source ~/stackrc
```

3. 각 드라이버의 기능을 나열합니다.

```
$ openstack loadbalancer provider capability list <provider>
```

<provider> 를 프로바이더의 이름 또는 UUID로 바꿉니다.

예제

```
$ openstack loadbalancer provider capability list amphora
```

명령 출력에는 프로바이더가 지원하는 모든 기능이 나열됩니다.

샘플 출력

```
+---------------------+-------------------------------------------------+
| name                | description                                     |
+---------------------+-------------------------------------------------+
| loadbalancer_topology | The load balancer topology. One of: SINGLE - One  |
|                     | amphora per load balancer. ACTIVE_STANDBY - Two   |
|                     | amphora per load balancer.                      |
| ...                 | ...                                             |
+---------------------+-------------------------------------------------+
```

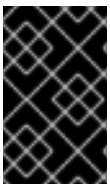
4. 생성 중인 플레이버에 포함하려는 기능의 이름을 확인합니다.

추가 리소스

- 5.2절. "플레이버 프로필 정의"
- *명령줄 인터페이스 참조*의 LoadBalancer 공급자 기능 목록

5.2. 플레이버 프로필 정의

로드 밸런싱 서비스(octavia) 플레이버 프로필에는 공급자 드라이버 이름과 기능 목록이 포함되어 있습니다. 플레이버 프로필을 사용하여 로드 밸런서를 만들기 위해 사용자가 지정하는 플레이버를 만듭니다.



중요

이 섹션에 설명된 플레이버 기능은 Red Hat OpenStack Platform 13, 2020년 10월 28일 유지 관리 릴리스 등의 *기술 프리뷰* 에 있습니다. 기술 프리뷰로 표시된 기능의 지원 범위에 대한 자세한 내용은 기술 프리뷰 기능 지원 범위 를 참조하십시오.

사전 요구 사항

- OpenStack 관리자 권한이 있어야 합니다.
- 어떤 로드 밸런싱 공급자와 플레이버 프로필에 포함하려는 기능을 알아야 합니다.

절차

1. 언더클라우드 호스트에 **stack** 사용자로 로그인합니다.
2. 언더클라우드 인증 정보 파일을 소싱합니다.

```
$ source ~/stackrc
```

3. 플레이버 프로필을 생성합니다.

```
$ openstack loadbalancer flavorprofile create --name <profile_name> --provider
<provider_name> --flavor-data '{"<capability>": "<value>"}'
```

예제

```
$ openstack loadbalancer flavorprofile create --name amphora-single-profile --provider
amphora --flavor-data '{"loadbalancer_topology": "SINGLE"}'
```

샘플 출력

```
+---------------+--------------------------------------+
| Field         | Value                                |
+---------------+--------------------------------------+
| id            | 72b53ac2-b191-48eb-8f73-ed012caca23a |
| name          | amphora-single-profile               |
| provider_name | amphora                              |
| flavor_data   | {"loadbalancer_topology": "SINGLE"}  |
+---------------+--------------------------------------+
```

이 예에서는 Amphora 공급자에 대한 플레이버 프로필이 생성됩니다. 플레이버에서 이 프로필을 지정하면 플레이버를 사용하여 사용자가 생성하는 로드 밸런서는 단일 amphora 로드 밸런서입니다.

검증 단계

- 플레이버 프로필을 생성할 때 로드 밸런싱 서비스는 프로바이더와 함께 플레이버 값을 검증하여 공급자가 지정한 기능을 지원할 수 있도록 합니다.

추가 리소스

- 5.3절. "로드 밸런싱 서비스 플레이버 생성"
- LoadBalancer flavorprofile create in the *Command Line Interface Reference*

5.3. 로드 밸런싱 서비스 플레이버 생성

플레이버 프로필을 사용하여 로드 밸런싱 서비스(octavia)에 사용할 사용자용 플레이버를 만듭니다. 플레이버에 할당하는 이름은 사용자가 로드 밸런서를 만들 때 지정하는 값입니다.

중요

이 섹션에 설명된 플레이버 기능은 Red Hat OpenStack Platform 13, 2020년 10월 28일 유지 관리 릴리스 등의 *기술 프리뷰* 에 있습니다. 기술 프리뷰로 표시된 기능의 지원 범위에 대한 자세한 내용은 기술 프리뷰 기능 지원 범위 를 참조하십시오.

사전 요구 사항

- OpenStack 관리자 권한이 있어야 합니다.
- 플레이버 프로필을 생성해야 합니다.

절차

1. 언더클라우드 호스트에 **stack** 사용자로 로그인합니다.
2. 언더클라우드 인증 정보 파일을 소싱합니다.

```
$ source ~/stackrc
```

3. 플레이버를 생성합니다.

```
$ openstack loadbalancer flavor create --name <flavor_name> \
--flavorprofile <flavor-profile> --description "<string>"
```

작은 정보

사용자가 제공하는 플레이버의 기능을 이해할 수 있도록 자세한 설명을 제공합니다.

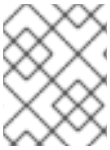
예제

```
$ openstack loadbalancer flavor create --name standalone-lb --flavorprofile amphora-single-profile --description "A non-high availability load balancer for testing."
```

샘플 출력

```
+-------------------+--------------------------------------+
| Field             | Value                                |
+-------------------+--------------------------------------+
| id                | 25cda2d8-f735-4744-b936-d30405c05359 |
| name              | standalone-lb                        |
| flavor_profile_id | 72b53ac2-b191-48eb-8f73-ed012caca23a |
| enabled           | True                                 |
| description       | A non-high availability load         |
|                   | balancer for testing.                |
+-------------------+--------------------------------------+
```

이 예에서는 플레이버가 정의되었습니다. 사용자가 이 플레이버를 지정하면 로드 밸런싱 서비스 인스턴스(amphora)를 사용하는 로드 밸런서를 만들고 고가용성이 아닙니다.



참고

비활성화된 플레이버는 여전히 사용자에게 표시되지만 비활성화된 플레이버를 사용하여 로드 밸런서를 생성할 수 없습니다.

추가 리소스

- 5.2절. “플레이버 프로필 정의”
- LoadBalancer 플레이버 create in the *Command Line Interface Reference*

6장. 로드 밸런싱 서비스 모니터링

로드 밸런싱 작동을 유지하기 위해 로드 밸런서 관리 네트워크를 사용하고 부하 분산 상태 모니터를 생성, 수정 및 삭제할 수 있습니다.

6.1. 로드 밸런싱 관리 네트워크

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)는 로드 밸런서 *관리 네트워크라고 하는 프로젝트 네트워크를 통해 로드 밸런서를 모니터링합니다.* 로드 밸런싱 서비스를 실행하는 호스트에는 로드 밸런서 관리 네트워크에 연결할 인터페이스가 있어야 합니다. 지원되는 인터페이스 구성은 Open Virtual Network 메커니즘 드라이버(ML2/OVN) 또는 Open vSwitch 메커니즘 드라이버(ML2/OVS)를 사용하는 neutron Modular Layer 2 플러그인과 함께 작동합니다. 다른 메커니즘 드라이버와 인터페이스 사용은 테스트되지 않았습니다.

배포에서 생성된 기본 인터페이스는 기본 통합 브리지 **br-int** 의 내부 OVS(Open vSwitch) 포트입니다. 이러한 인터페이스를 로드 밸런서 관리 네트워크에 할당된 실제 네트워킹 서비스(neutron) 포트와 연결해야 합니다.

기본 인터페이스는 기본적으로 **o-hm0** 이라고 합니다. 이러한 인터페이스는 로드 밸런싱 서비스 호스트의 표준 인터페이스 구성 파일을 통해 정의됩니다. RHOSP director는 배포 중에 각 로드 밸런싱 서비스 호스트에 대한 네트워킹 서비스 포트 및 인터페이스를 자동으로 구성합니다. 포트 정보와 템플릿은 다음을 포함하여 인터페이스 구성 파일을 생성하는 데 사용됩니다.

- IP 및 넷마스크를 포함한 IP 네트워크 주소 정보
- MTU 구성
- MAC 주소
- 네트워킹 서비스 포트 ID

기본 OVS 사례에서는 OVS 포트에 추가 데이터를 등록하는 데 Networking 서비스 포트 ID를 사용합니다. 네트워킹 서비스는 이 인터페이스를 포트에 속하는 것으로 인식하고, 로드 밸런서 관리 네트워크에서 통신할 수 있도록 OVS를 구성합니다.

기본적으로 RHOSP는 로드 밸런싱 서비스 컨트롤러가 TCP 포트 9443의 VM 인스턴스(앰포라)와 통신할

수 있도록 하는 보안 그룹 및 방화벽 규칙을 구성하고, amphorae의 하트비트 메시지가 UDP 포트 5555의 컨트롤러에 도착하도록 허용합니다. 다른 메커니즘 드라이버에는 로드 밸런싱 서비스와 로드 밸런서 간에 통신할 수 있도록 추가 또는 대체 요구 사항이 필요할 수 있습니다.

6.2. 로드 밸런싱 서비스 인스턴스 모니터링

로드 밸런싱 서비스(octavia)는 부하 분산 인스턴스(amphorae)를 모니터링하고 amphorae가 작동하지 않는 경우 페일오버 및 교체를 시작합니다. 장애 조치가 발생할 때마다 로드 밸런싱 서비스는 **/var/log/containers/octavia** 의 컨트롤러에 장애 조치(failover)를 기록합니다.

로그 분석을 사용하여 장애 조치(failover) 추세를 모니터링하여 문제를 조기에 해결합니다. 네트워킹 서비스(neutron) 연결 문제, 서비스 거부 공격, 계산 서비스(nova) 오작동과 같은 문제로 인해 로드 밸런서의 장애 조치 속도가 높아지는 경우가 많습니다.

6.3. 로드 밸런싱 서비스 풀 멤버 모니터링

로드 밸런싱 서비스(octavia)는 기본 로드 밸런싱 하위 시스템의 상태 정보를 사용하여 부하 분산 풀의 구성원의 상태를 결정합니다. 상태 정보는 로드 밸런싱 서비스 데이터베이스로 스트리밍되며 상태 트리 또는 다른 API 메서드에서 사용할 수 있습니다. 중요한 애플리케이션의 경우 정기적으로 상태 정보를 폴링해야 합니다.

6.4. 로드 밸런서 프로비저닝 상태 모니터링

로드 밸런서의 프로비저닝 상태를 모니터링하고 프로비저닝 상태가 **ERROR** 인지 경고를 보낼 수 있습니다. 애플리케이션이 풀을 정기적으로 변경하고 여러 **PENDING** 단계를 입력할 때 트리거하도록 경고를 구성하지 마십시오.

로드 밸런서 오브젝트의 프로비저닝 상태는 컨트롤 플레인에 연결하여 생성, 업데이트 및 삭제 요청을 성공적으로 프로비저닝할 수 있는 기능을 반영합니다. 로드 밸런서 오브젝트의 작동 상태는 로드 밸런서의 현재 기능에 대해 보고합니다.

예를 들어 로드 밸런서의 프로비저닝 상태가 **ERROR** 일 수 있지만 작동 상태는 **ONLINE** 입니다. 이는 마지막으로 로드 밸런서 구성에 대한 업데이트를 요청한 네트워킹(neutron) 실패로 인해 발생할 수 있습니다. 이 경우 로드 밸런서는 로드 밸런서를 통해 트래픽을 계속 처리하지만 아직 최신 구성 업데이트를 적용하지 않았을 수 있습니다.

6.5. 로드 밸런서 기능 모니터링

로드 밸런서 및 해당 하위 오브젝트의 작동 상태를 모니터링할 수 있습니다.

로드 밸런서 리스너에 연결하고 클라우드 외부에서 모니터링하는 외부 모니터링 서비스를 사용할 수도 있습니다. 외부 모니터링 서비스는 라우터 실패, 네트워크 연결 문제 등 로드 밸런서의 기능에 영향을 미칠 수 있는 로드 밸런싱 서비스(octavia) 외부에 오류가 있는지 여부를 나타냅니다.

6.6. 부하 분산 서비스 상태 모니터 정보

로드 밸런싱 서비스(octavia) 상태 모니터는 각 백엔드 구성원 서버에서 실패한 서버를 선점적으로 탐지하고 풀에서 임시로 가져오는 프로세스입니다.

상태 모니터가 실패한 서버를 감지하면 풀에서 서버를 제거하고 멤버를 **ERROR** 로 표시합니다. 서버를 수정하고 다시 작동하면 상태 모니터가 구성원의 상태를 자동으로 **ERROR** 에서 **ONLINE** 으로 변경하고 트래픽을 전달하는 것이 다시 시작됩니다.

프로덕션 로드 밸런서에서 항상 상태 모니터를 사용하십시오. 상태 모니터가 없으면 실패한 서버가 풀에서 제거되지 않습니다. 이로 인해 웹 클라이언트의 서비스 중단이 발생할 수 있습니다.

다음은 간단히 설명하는 몇 가지 유형의 상태 모니터가 있습니다.

HTTP

기본적으로 애플리케이션 서버의 / 경로를 검색합니다.

HTTPS

HTTP 상태 모니터와 동일하게 작동하지만 TLS 백엔드 서버를 사용합니다.
서버에서 클라이언트 인증서 검증을 수행하는 경우 HAProxy에 유효한 인증서가 없습니다. 이 경우 TLS-HELLO 상태 모니터링이 대안입니다.

TLS-HELLO

백엔드 서버가 SSLv3-client hello 메시지에 응답하는지 확인합니다.
TLS-HELLO 상태 모니터는 상태 코드 또는 본문 내용과 같은 다른 상태 지표를 확인하지 않습니다.

PING

백엔드 서버에 주기적인 ICMP ping 요청을 보냅니다.
이러한 상태 점검을 통과하도록 PING을 허용하도록 백엔드 서버를 구성해야 합니다.

중요

PING 상태 모니터는 멤버가 연결할 수 있는 경우에만 확인하고 ICMP 에코 요청에 응답합니다. PING 상태 모니터는 인스턴스에서 실행되는 애플리케이션이 정상인지 감지하지 않습니다. ICMP 에코 요청이 유효한 상태 점검인 경우에만 PING 상태 모니터를 사용합니다.

TCP

백엔드 서버 프로토콜 포트에 대한 TCP 연결을 엽니다.
TCP 애플리케이션은 TCP 연결을 열고 TCP 핸드셰이크 후 데이터를 전송하지 않고 연결을 종료합니다.

UDP-CONNECT

기본 UDP 포트 연결을 수행합니다.
Destination Unreachable(ICMP 유형 3)이 멤버 서버에서 활성화되지 않거나 보안 규칙에 의해 차단된 경우 UDP-CONNECT 상태 모니터가 제대로 작동하지 않을 수 있습니다. 이 경우 멤버 서버가 실제로 다운된 경우 **ONLINE** 의 운영 상태가 되는 것으로 표시될 수 있습니다.

6.7. 로드 밸런싱 서비스 상태 모니터 생성

사용자의 서비스 중단을 방지하려면 로드 밸런싱 서비스(octavia) 상태 모니터를 사용합니다. 상태 모니터는 각 백엔드 서버에서 주기적인 상태 점검을 실행하여 오류가 발생한 서버를 미리 감지하고 일시적으로 풀에서 서버를 끌어올 수 있습니다.

절차

1. 자격 증명 파일을 가져옵니다.

   예제

```
$ source ~/overcloudrc
```

2. 사이트에 적합한 인수 값을 사용하여 **openstack loadbalancer healthmonitor create** 명령을 실행합니다.

- 모든 상태 모니터 유형에는 다음과 같은 구성 가능한 인수가 필요합니다.

<pool>

모니터링할 백엔드 멤버 서버 풀의 이름 또는 ID입니다.

--type

상태 모니터의 유형입니다. **HTTP**,**HTTPS**,**PING**,**SCTP**,**TCP**,**TLS-HELLO** 또는 **UDP-CONNECT** 중 하나입니다.

--delay

상태 점검 사이에 대기할 시간(초)입니다.

--timeout

지정된 상태 점검이 완료될 때까지 대기하는 시간(초)입니다. **timeout** 은 항상 **지연** 보다 작아야 합니다.

--max-retries

백엔드 서버가 다운된 상태로 간주되기 전에 실패해야 하는 상태 점검 수입니다. 또한 실패한 백엔드 서버에서 전달해야 하는 상태 점검 수를 다시 고려해야 합니다.

- 또한 HTTP 상태 모니터 유형에는 기본적으로 설정된 다음과 같은 인수가 필요합니다.

--url-path

백엔드 서버에서 검색해야 하는 URL의 경로 부분입니다. 기본값은 / 입니다.

--http-method

url_path 를 검색하는 데 사용되는 HTTP 메서드입니다. 기본적으로 **GET** 입니다.

--expected-codes

OK 상태 점검을 나타내는 HTTP 상태 코드 목록입니다. 기본적으로 **200** 입니다.

예제

```
$ openstack loadbalancer healthmonitor create --name my-health-monitor --delay 10 --max-retries 4 --timeout 5 --type TCP lb-pool-1
```

검증

- **openstack loadbalancer healthmonitor list** 명령을 실행하고 상태 모니터가 실행 중인지 확인합니다.

추가 리소스

- *명령줄 인터페이스 참조*에서 LoadBalancer 상태 모니터 생성

6.8. 로드 밸런싱 서비스 상태 모니터 수정

프로브를 멤버로 보내는 간격, 연결 제한 간격, 요청의 HTTP 메서드를 변경할 때 로드 밸런싱 서비스(octavia) 상태 모니터에 대한 구성을 수정할 수 있습니다.

절차

1. 자격 증명 파일을 가져옵니다.

   예제

   ```
   $ source ~/overcloudrc
   ```

2. 상태 모니터 (**my-health-monitor**)를 수정합니다.
   이 예제에서 사용자는 상태 모니터가 프로브를 멤버로 보낼 때까지 대기하는 시간(초)을 변경하고 있습니다.

   예제

   ```
   $ openstack loadbalancer healthmonitor set my_health_monitor --delay 600
   ```

검증

- **openstack loadbalancer healthmonitor show** 명령을 실행하여 구성 변경 사항을 확인합니다.

  ```
  $ openstack loadbalancer healthmonitor show my_health_monitor
  ```

추가 리소스

- 명령줄 인터페이스 참조에 설정된 LoadBalancer 상태 모니터
- *명령줄 인터페이스 참조에 LoadBalancer healthmonitor 표시*

6.9. 로드 밸런싱 서비스 상태 모니터 삭제

로드 밸런싱 서비스(octavia) 상태 모니터를 제거할 수 있습니다.

작은 정보

*상태 모니터를 삭제하는 대안은 **openstack loadbalancer healthmonitor set --disable** 명령을 사용하여 비활성화하는 것입니다.*

절차

1. *자격 증명 파일을 가져옵니다.*

   예제

   ```
   $ source ~/overcloudrc
   ```

2. *상태 모니터(**my-health-monitor**)를 삭제합니다.*

   예제

   ```
   $ openstack loadbalancer healthmonitor delete my-health-monitor
   ```

검증

- ***openstack loadbalancer healthmonitor list*** *명령을 실행하여 삭제한 상태 모니터가 더 이상 존재하지 않는지 확인합니다.*

추가 리소스

- *명령줄 인터페이스 참조에서* LoadBalancer *healthmonitor delete*

6.10. 로드 밸런싱 서비스 HTTP 상태 모니터 모범 사례

웹 애플리케이션에서 상태 점검을 생성하는 코드를 작성할 때 다음 모범 사례를 사용하십시오.

- *상태 모니터* ***url-path*** *를 로드하기 위해 인증이 필요하지 않습니다.*
- *기본적으로 상태 모니터* ***url-path*** *는* ***HTTP 200 OK*** *상태 코드를 반환하여 다른* ***예상 코드를 지정하지 않는 한 정상 서버를 나타냅니다****.*
- *상태 점검은 애플리케이션이 정상이고 더 이상 발생하지 않도록 충분한 내부 검사를 수행합니다. 다음 조건이 애플리케이션에 대해 충족되었는지 확인합니다.*
  - *필요한 데이터베이스 또는 기타 외부 스토리지 연결이 실행 중입니다.*
  - *로드는 애플리케이션이 실행되는 서버에 대해 허용됩니다.*
  - *사이트는 유지 관리 모드가 아닙니다.*
  - *애플리케이션과 관련된 테스트가 작동 중입니다.*
- *상태 점검으로 생성된 페이지는 크기가 작아야 합니다.*
  - *1초 간격으로 반환됩니다.*
  - *애플리케이션 서버에 상당한 부하를 유발하지 않습니다.*
- *상태 점검으로 생성된 페이지는 캐시되지 않지만 상태 점검을 실행하는 코드는 캐시된 데이터를 참조할 수 있습니다.*
  예를 들어 cron을 사용하여 보다 광범위한 상태 점검을 실행하고 결과를 디스크에 저장하는 것이 유용할 수 있습니다. 상태 모니터 ***URL-path*** *에서 페이지를 생성하는 코드는 이 cron 작업의 결과를 수행하는 테스트에 통합합니다.*
- *로드 밸런싱 서비스는 반환된 HTTP 상태 코드만 처리하고 상태 점검이 너무 자주 실행되므로* ***HEAD*** *또는* ***OPTIONS*** *HTTP 메서드를 사용하여 전체 페이지 처리를 건너뛸 수 있습니다.*

7장. 비보안 HTTP 로드 밸런서 생성

비보안 HTTP 네트워크 트래픽에 대해 다음 로드 밸런서를 생성할 수 있습니다.

7.1. 상태 모니터를 사용하여 HTTP 로드 밸런서 생성

Red Hat OpenStack Platform Networking 서비스(neutron) 유동 IP와 호환되지 않는 네트워크의 경우 로드 밸런서를 생성하여 비보안 HTTP 애플리케이션의 네트워크 트래픽을 관리합니다. 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성합니다.

사전 요구 사항

- TCP 포트 80에서 비보안 HTTP 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷.
- 프라이빗 서브넷의 백엔드 서버는 URL 경로 /에 상태 점검으로 구성됩니다.
- 인터넷에서 도달할 수 있는 공유 외부(공용) 서브넷입니다.

절차

1. 자격 증명 파일을 가져옵니다.

   예제

   ```
   $ source ~/overcloudrc
   ```

2. 공용 서브넷(public_**subnet)에서 로드 밸런서(lb**1)를 만듭니다.

   참고

   괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

   예제

   ```
   $ openstack loadbalancer create --name lb1 --vip-subnet-id public_subnet
   ```

3. 로드 밸런서의 상태를 확인합니다.

   예제

   ```
   $ openstack loadbalancer show lb1
   ```

4. 다음 단계로 진행하기 전에 **provisioning_status** 가 EgressIP 인지 **확인합니다**.

5. 포트**(80)에 리스너**(리스너1)를 만듭니다.

예제

```
$ openstack loadbalancer listener create --name listener1 --protocol HTTP --protocol-port 80
lb1
```

6. 리스너의 상태를 확인합니다.

예제

```
$ openstack loadbalancer listener show listener1
```

다음 단계로 진행하기 전에 상태가 ACTIVE 인지 **확인합니다**.

7. 리스너 기본 풀(**pool1**)을 생성합니다.

예제

```
$ openstack loadbalancer pool create --name pool1 --lb-algorithm ROUND_ROBIN --listener
listener1 --protocol HTTP
```

8. 백엔드 서버에 연결하고 경로(/)를 테스트하는 풀(**pool1**)에 상태 모니터를 생성합니다.

예제

```
$ openstack loadbalancer healthmonitor create --delay 15 --max-retries 4 --timeout 10 --type
HTTP --url-path / pool1
```

9. 프라이빗 서브넷(**private_subnet**)에서 로드 밸런서 구성원 (**192.0.2.10 및 192.0.2.11**)을 기본 풀에 추가합니다.

예제

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.10 --
protocol-port 80 pool1
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.11 --
protocol-port 80 pool1
```

검증

1. 로드 밸런서(lb1) 설정을 보고 확인합니다.

예제

```
$ openstack loadbalancer show lb1
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
```

```
| description         |                                      |
| flavor              |                                      |
| id                  | 788fe121-3dec-4e1b-8360-4020642238b0 |
| listeners           | 09f28053-fde8-4c78-88b9-0f191d84120e |
| name                | lb1                                  |
| operating_status    | ONLINE                               |
| pools               | 627842b3-eed8-4f5f-9f4a-01a738e64d6a |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| provider            | amphora                              |
| provisioning_status | ACTIVE                               |
| updated_at          | 2022-01-15T11:12:13                  |
| vip_address         | 198.51.100.12                        |
| vip_network_id      | 9bca13be-f18d-49a5-a83d-9d487827fd16 |
| vip_port_id         | 69a85edd-5b1c-458f-96f2-b4552b15b8e6 |
| vip_qos_policy_id   | None                                 |
| vip_subnet_id       | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
+---------------------+--------------------------------------+
```

2. 상태 모니터가 올바르게 작동하고 있는 경우 각 멤버의 상태를 확인할 수 있습니다. 작동 중인 멤버(**b85c807e-4d7c-4cd-b725-5e8afddf80d2**)에는 **operating_status** 의 **ONLINE** 값이 있습니다.

예제

```
$ openstack loadbalancer member show pool1 b85c807e-4d7c-4cbd-b725-5e8afddf80d2
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| address             | 192.0.2.10                           |
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:16:23                  |
| id                  | b85c807e-4d7c-4cbd-b725-5e8afddf80d2 |
| name                |                                      |
| operating_status    | ONLINE                               |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| protocol_port       | 80                                   |
| provisioning_status | ACTIVE                               |
| subnet_id           | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
| updated_at          | 2022-01-15T11:20:45                  |
| weight              | 1                                    |
| monitor_port        | None                                 |
| monitor_address     | None                                 |
| backup              | False                                |
+---------------------+--------------------------------------+
```

추가 리소스

- 명령줄 인터페이스 참조 의 LoadBalancer

7.2. 유동 IP를 사용하는 HTTP 로드 밸런서 생성

비보안 HTTP 애플리케이션의 네트워크 트래픽을 관리하려면 유동 IP에 따라 VIP(가상 IP)를 사용하여 로드 밸런서를 만듭니다. 유동 IP를 사용하면 할당된 IP를 계속 제어할 수 있다는 장점이 있으며, 로드 밸런서를 이동, 제거 또는 재생성해야 하는 경우 필요합니다. 또한 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.

참고

유동 IP는 IPv6 네트워크에서 작동하지 않습니다.

사전 요구 사항

- TCP 포트 80에서 비보안 HTTP 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷.
- 백엔드 서버는 URL 경로 / 에 상태 점검으로 구성됩니다.
- 로드 밸런서 VIP와 함께 사용할 유동 IP입니다.
- 유동 IP에 사용하기 위해 인터넷에서 연결할 수 있는 Red Hat OpenStack Platform Networking 서비스(neutron) 공유 외부(공용) 서브넷입니다.

절차

1. 자격 증명 파일을 가져옵니다.

   예제

   ```
   $ source ~/overcloudrc
   ```

2. 프라이빗 서브넷(private_**subnet)에서 로드 밸런서(lb**1)를 만듭니다.

   참고

   괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

   예제

   ```
   $ openstack loadbalancer create --name lb1 --vip-subnet-id private_subnet
   ```

3. 이후 단계에서 제공해야 하므로 **load_balancer_vip_port_id** 의 값을 기록해 둡니다.

4. 로드 밸런서의 상태를 확인합니다.

   예제

   ```
   $ openstack loadbalancer show lb1
   ```

5. 다음 단계로 진행하기 전에 **provisioning_status** 가 EgressIP 인지 **확인합니다**.

6. 포트**(80)에 리스너**(리스너1)를 만듭니다.

   예제

```
$ openstack loadbalancer listener create --name listener1 --protocol HTTP --protocol-port 80
lb1
```

7. 리스너 기본 풀(**pool1**)을 생성합니다.

예제

```
$ openstack loadbalancer pool create --name pool1 --lb-algorithm ROUND_ROBIN --listener
listener1 --protocol HTTP
```

8. 백엔드 서버에 연결하고 경로(/)를 테스트하는 풀(**pool1**)에 상태 모니터를 생성합니다.

예제

```
$ openstack loadbalancer healthmonitor create --delay 15 --max-retries 4 --timeout 10 --type
HTTP --url-path / pool1
```

9. 프라이빗 서브넷에 로드 밸런서 구성원(19**2.0.2.10** 및 **192.0.2.11**)을 기본 풀에 추가합니다.

예제

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.10 --
protocol-port 80 pool1
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.11 --
protocol-port 80 pool1
```

10. 공유 외부 서브넷(**공용**)에 유동 IP 주소를 만듭니다.

예제

```
$ openstack floating ip create public
```

11. 이후 단계에서 제공해야 하므로 **floating_ip_address** 의 값을 기록해 둡니다.

12. 이 유동 IP(**203.0.113.0**)를 로드 밸런서 **vip_port_id** (69**a85edd-5b1c-458f-96f2-b4552b15b8e6**)와 연결합니다.

예제

```
$ openstack floating ip set --port 69a85edd-5b1c-458f-96f2-b4552b15b8e6 203.0.113.0
```

검증

1. 유동 IP(**203.0.113.0**)를 사용하여 로드 밸런서 간에 **HTTP** 트래픽이 이동하는지 확인합니다.

예제

```
$ curl -v http://203.0.113.0 --insecure
```

샘플 출력

```
* About to connect() to 203.0.113.0 port 80 (#0)
```

```
*   Trying 203.0.113.0...
* Connected to 203.0.113.0 (203.0.113.0) port 80 (#0)
> GET / HTTP/1.1
> User-Agent: curl/7.29.0
> Host: 203.0.113.0
> Accept: */*
>
< HTTP/1.1 200 OK
< Content-Length: 30
<
* Connection #0 to host 203.0.113.0 left intact
```

2. 상태 모니터가 올바르게 작동하고 있는 경우 각 멤버의 상태를 확인할 수 있습니다. 작동 중인 멤버(**b85c807e-4d7c-4cd-b725-5e8afddf80d2**)에는 **operating_status** 의 **ONLINE** 값이 있습니다.

   예제

   ```
   $ openstack loadbalancer member show pool1 b85c807e-4d7c-4cbd-b725-5e8afddf80d2
   ```

   샘플 출력

   ```
   +---------------------+--------------------------------------+
   | Field               | Value                                |
   +---------------------+--------------------------------------+
   | address             | 192.0.02.10                          |
   | admin_state_up      | True                                 |
   | created_at          | 2022-01-15T11:11:23                  |
   | id                  | b85c807e-4d7c-4cbd-b725-5e8afddf80d2 |
   | name                |                                      |
   | operating_status    | ONLINE                               |
   | project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
   | protocol_port       | 80                                   |
   | provisioning_status | ACTIVE                               |
   | subnet_id           | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
   | updated_at          | 2022-01-15T11:28:42                  |
   | weight              | 1                                    |
   | monitor_port        | None                                 |
   | monitor_address     | None                                 |
   | backup              | False                                |
   +---------------------+--------------------------------------+
   ```

추가 리소스

- 명령줄 인터페이스 참조 의 LoadBalancer
- 명령줄 인터페이스 참조의 유동

7.3. 세션 지속성을 사용하여 HTTP 로드 밸런서 생성

비보안 HTTP 애플리케이션의 네트워크 트래픽을 관리하려면 세션 지속성을 추적하는 로드 밸런서를 생성할 수 있습니다. 이렇게 하면 로드 밸런서에서 요청이 입력될 때 동일한 클라이언트의 후속 요청을 동일한 백엔드 서버로 보냅니다. 세션 지속성은 시간과 메모리를 저장하여 로드 밸런싱을 최적화합니다.

사전 요구 사항

- *TCP 포트 80에서 비보안 HTTP 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷.*
- *백엔드 서버는 URL 경로 / 에 상태 점검으로 구성됩니다.*
- *인터넷에서 도달할 수 있는 공유 외부(공용) 서브넷입니다.*
- *부하 분산 중인 네트워크 트래픽이 있는 비보안 웹 애플리케이션에 쿠키가 활성화되어 있습니다.*

절차

1. *자격 증명 파일을 가져옵니다.*

   예제

   ```
   $ source ~/overcloudrc
   ```

2. *공용 서브넷(public_**subnet)에서 로드 밸런서(lb**1)를 만듭니다.*

   참고

   괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

   예제

   ```
   $ openstack loadbalancer create --name lb1 --vip-subnet-id public_subnet
   ```

3. *로드 밸런서의 상태를 확인합니다.*

   예제

   ```
   $ openstack loadbalancer show lb1
   ```

4. *다음 단계로 진행하기 전에 **provisioning_status** 가 EgressIP 인지 **확인합니다**.*

5. *포트**(80)에 리스너**(리스너1)를 만듭니다.*

   예제

   ```
   $ openstack loadbalancer listener create --name listener1 --protocol HTTP --protocol-port 80
   lb1
   ```

6. *쿠키(**PHPSESSIONID)에서 세션 지속성을 정의하는 리스너 기본 풀(pool**1)을 만듭니다.*

   예제

   ```
   $ openstack loadbalancer pool create --name pool1 --lb-algorithm ROUND_ROBIN --listener
   listener1 --protocol HTTP --session-persistence
   type=APP_COOKIE,cookie_name=PHPSESSIONID
   ```

7. 백엔드 서버에 연결하고 경로(/)를 테스트하는 풀(**pool1**)에 상태 모니터를 생성합니다.

예제

```
$ openstack loadbalancer healthmonitor create --delay 15 --max-retries 4 --timeout 10 --type
HTTP --url-path / pool1
```

8. 프라이빗 서브넷(**private_subnet**)에서 로드 밸런서 구성원 **(192.0.2.10 및 192.0.2.11)**을 기본 풀에 추가합니다.

예제

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.10 --
protocol-port 80 pool1
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.11 --
protocol-port 80 pool1
```

검증

1. 로드 밸런서(lb1) 설정을 보고 확인합니다.

예제

```
$ openstack loadbalancer show lb1
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:58                  |
| description         |                                      |
| flavor              |                                      |
| id                  | 788fe121-3dec-4e1b-8360-4020642238b0 |
| listeners           | 09f28053-fde8-4c78-88b9-0f191d84120e |
| name                | lb1                                  |
| operating_status    | ONLINE                               |
| pools               | 627842b3-eed8-4f5f-9f4a-01a738e64d6a |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| provider            | amphora                              |
| provisioning_status | ACTIVE                               |
| updated_at          | 2022-01-15T11:28:42                  |
| vip_address         | 198.51.100.22                        |
| vip_network_id      | 9bca13be-f18d-49a5-a83d-9d487827fd16 |
| vip_port_id         | 69a85edd-5b1c-458f-96f2-b4552b15b8e6 |
| vip_qos_policy_id   | None                                 |
| vip_subnet_id       | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
+---------------------+--------------------------------------+
```

2. 상태 모니터가 올바르게 작동하고 있는 경우 각 멤버의 상태를 확인할 수 있습니다.
작동 중인 멤버(**b85c807e-4d7c-4cd-b725-5e8afddf80d2**)에는 **operating_status** 의 **ONLINE** 값이 있습니다.

예제

```
$ openstack loadbalancer member show pool1 b85c807e-4d7c-4cbd-b725-5e8afddf80d2
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| address             | 192.0.02.10                          |
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:23                  |
| id                  | b85c807e-4d7c-4cbd-b725-5e8afddf80d2 |
| name                |                                      |
| operating_status    | ONLINE                               |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| protocol_port       | 80                                   |
| provisioning_status | ACTIVE                               |
| subnet_id           | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
| updated_at          | 2022-01-15T11:28:42                  |
| weight              | 1                                    |
| monitor_port        | None                                 |
| monitor_address     | None                                 |
| backup              | False                                |
+---------------------+--------------------------------------+
```

추가 리소스

- *명령줄 인터페이스 참조 의 LoadBalancer*

8장. 보안 HTTP 로드 밸런서 생성

다양한 유형의 로드 밸런서를 생성하여 HTTPS(Secure HTTP) 네트워크 트래픽을 관리할 수 있습니다.

8.1. 종료되지 않은 HTTPS 로드 밸런서 정보

종료되지 않은 HTTPS 로드 밸런서는 일반 TCP 로드 밸런서와 같이 효과적으로 작동합니다. 로드 밸런서는 웹 클라이언트에서 웹 클라이언트의 원시 TCP 트래픽을 웹 클라이언트로 전달하는 백엔드 서버로, 웹 클라이언트의 원시 TCP 트래픽을 웹 클라이언트로 전달합니다. 종료되지 않은 HTTPS 로드 밸런서는 계층 7 기능과 같은 고급 로드 밸런서 기능을 지원하지 않지만 인증서 및 키를 자체 관리하여 로드 밸런서 리소스 사용률을 줄입니다.

8.2. 종료되지 않은 HTTPS 로드 밸런서 생성

애플리케이션에 일반적으로 HTTPS 전달이라고 하는 백엔드 멤버 서버에서 HTTPS 트래픽이 종료되어야 하는 경우 로드 밸런서 리스너에 HTTPS 프로토콜을 사용할 수 있습니다.

사전 요구 사항

- TCP 포트 443에서 TLS 암호화 웹 애플리케이션으로 구성된 HTTPS 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷입니다.
- 백엔드 서버는 URL 경로 / 에 상태 점검으로 구성됩니다.
- 인터넷에서 도달할 수 있는 공유 외부(공용) 서브넷입니다.

절차

1. 자격 증명 파일을 가져옵니다.

   예제

   ```
   $ source ~/overcloudrc
   ```

2. 공용 서브넷(public_**subnet)에서 로드 밸런서(lb**1)를 만듭니다.

   참고

   괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

```
$ openstack loadbalancer create --name lb1 --vip-subnet-id public_subnet
```

3. 로드 밸런서의 상태를 모니터링합니다.

예제

```
$ openstack loadbalancer show lb1
```

4. 다음 단계로 진행하기 전에 **provisioning_status** 가 EgressIP 인지 **확인합니다**.

5. 포트(443)**에서 리스너**(리스너1)를 만듭니다**.**

예제

```
$ openstack loadbalancer listener create --name listener1 --protocol HTTPS --protocol-port
443 lb1
```

6. 리스너 기본 풀(**pool1)을 생성합니다.**

예제

```
$ openstack loadbalancer pool create --name pool1 --lb-algorithm ROUND_ROBIN --listener
listener1 --protocol HTTPS
```

7. 백엔드 서버에 연결하고 경로(/)를 테스트하는 풀(**pool1**)에 상태 모니터를 생성합니다.

예제

```
$ openstack loadbalancer healthmonitor create --delay 15 --max-retries 4 --timeout 10 --type
TLS-HELLO --url-path / pool1
```

8. 프라이빗 서브넷(**private_subnet**)에서 로드 밸런서 구성원 **(192.0.2.10 및 192.0.2.11**)을 기본 풀에 추가합니다.

예제

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.10 --
protocol-port 443 pool1
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.11 --
protocol-port 443 pool1
```

검증

1. 로드 밸런서(**lb1**) 설정을 보고 확인합니다.

예제

```
$ openstack loadbalancer show lb1
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
| description         |                                      |
| flavor              |                                      |
| id                  | 788fe121-3dec-4e1b-8360-4020642238b0 |
| listeners           | 09f28053-fde8-4c78-88b9-0f191d84120e |
| name                | lb1                                  |
| operating_status    | ONLINE                               |
| pools               | 627842b3-eed8-4f5f-9f4a-01a738e64d6a |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| provider            | amphora                              |
| provisioning_status | ACTIVE                               |
| updated_at          | 2022-01-15T11:12:42                  |
| vip_address         | 198.51.100.11                        |
| vip_network_id      | 9bca13be-f18d-49a5-a83d-9d487827fd16 |
| vip_port_id         | 69a85edd-5b1c-458f-96f2-b4552b15b8e6 |
| vip_qos_policy_id   | None                                 |
| vip_subnet_id       | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
+---------------------+--------------------------------------+
```

2. *상태 모니터가 올바르게 작동하고 있는 경우 각 멤버의 상태를 확인할 수 있습니다. 작동 중인 멤버(**b85c807e-4d7c-4cd-b725-5e8afddf80d2**)에는 **operating_status** 의 **ONLINE** 값이 있습니다.*

예제

```
$ openstack loadbalancer member show pool1 b85c807e-4d7c-4cbd-b725-5e8afddf80d2
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| address             | 192.0.2.10                           |
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
| id                  | b85c807e-4d7c-4cbd-b725-5e8afddf80d2 |
| name                |                                      |
| operating_status    | ONLINE                               |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| protocol_port       | 443                                  |
| provisioning_status | ACTIVE                               |
| subnet_id           | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
| updated_at          | 2022-01-15T11:12:42                  |
| weight              | 1                                    |
| monitor_port        | None                                 |
| monitor_address     | None                                 |
| backup              | False                                |
+---------------------+--------------------------------------+
```

추가 리소스

- OpenStack 키 관리자 가이드를 통해 시크릿 관리 .
- 명령줄 인터페이스 참조 의 LoadBalancer

8.3. TLS 종료 HTTPS 로드 밸런서 정보

TLS 터미널 HTTPS 로드 밸런서가 구현되면 웹 클라이언트는 TLS(Transport Layer Security) 프로토콜을 통해 로드 밸런서와 통신합니다. 로드 밸런서는 TLS 세션을 종료하고 암호 해독된 요청을 백엔드 서버에 전달합니다. 로드 밸런서에서 TLS 세션을 종료하면 CPU 집약적인 암호화 작업을 로드 밸런서에 오프로드하고 로드 밸런서에서 계층 7 검사와 같은 고급 기능을 사용할 수 있습니다.

8.4. TLS 종료 HTTPS 로드 밸런서 생성

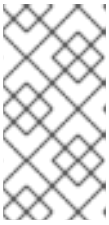
TLS 종료 HTTPS 로드 밸런서를 사용하는 경우 CPU 집약적인 암호화 작업을 로드 밸런서에 오프로드하고 로드 밸런서에서 계층 7 검사와 같은 고급 기능을 사용할 수 있습니다. 또한 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.

사전 요구 사항

- TCP 포트 80에서 비보안 HTTP 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷.
- 백엔드 서버는 URL 경로 / 에 상태 점검으로 구성됩니다.
- 인터넷에서 도달할 수 있는 공유 외부(공용) 서브넷입니다.
- TLS 공개 키 암호화는 다음과 같은 특성을 사용하여 구성됩니다.
  - TLS 인증서, 키 및 중간 인증서 체인은 로드 밸런서 VIP 주소에 할당된 DNS 이름의 외부 인증 기관(CA)에서 가져옵니다(예: **www.example.com**).
  - 인증서, 키 및 중간 인증서 체인은 현재 디렉터리의 별도의 파일에 있습니다.
  - 키와 인증서는 PEM으로 인코딩됩니다.
  - 키는 암호로 암호화되지 않습니다.
  - 중간 인증서 체인에는 PEM 인코딩 및 연결된 여러 인증서가 포함되어 있습니다.
- Key Manager 서비스(barbican)를 사용하도록 로드 밸런싱 서비스(octavia)를 구성해야 합니다. 자세한 내용은 OpenStack Key Manager를 사용하여 시크릿 관리 가이드 를 참조하십시오.

절차

1. 키(server**.key**), 인증서(server**.crt**) 및 중간 인증서 체인(**ca-chain.crt**)을 단일 PKCS12 파일(server.p12)으로 결합합니다.



참고

괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

```
$ openssl pkcs12 -export -inkey server.key -in server.crt -certfile ca-chain.crt -passout pass: -out server.p12
```

참고

*다음 절차는 암호가 **PKCS12** 파일을 보호하는 경우 작동하지 않습니다.*

2. *자격 증명 파일을 가져옵니다.*

예제

```
$ source ~/overcloudrc
```

3. ***Key Manager** 서비스를 사용하여 **PKCS12** 파일에 대한 시크릿 리소스(**tls_secret1**)를 생성합니다.*

예제

```
$ openstack secret store --name='tls_secret1' -t 'application/octet-stream' -e 'base64' --payload="$(base64 < server.p12)"
```

4. *공용 서브넷(**public_subnet**)에서 로드 밸런서(**lb1**)를 만듭니다.*

예제

```
$ openstack loadbalancer create --name lb1 --vip-subnet-id public_subnet
```

5. *로드 밸런서의 상태를 모니터링합니다.*

예제

```
$ openstack loadbalancer show lb1
```

6. *다음 단계로 진행하기 전에 **provisioning_status** 가 **EgressIP** 인지 확인합니다.*

7. ***TERMINATED_HTTPS** 리스너(**리스너1**)를 만들고 리스너의 기본 **TLS** 컨테이너로 시크릿 리소스를 참조합니다.*

예제

```
$ openstack loadbalancer listener create --protocol-port 443 --protocol
TERMINATED_HTTPS --name listener1 --default-tls-container=$(openstack secret list | awk
'/ tls_secret1 / {print $2}') lb1
```

8. *풀(**pool1**)을 만들고리스너의 기본 풀로 설정합니다.*

예제

```
$ openstack loadbalancer pool create --name pool1 --lb-algorithm ROUND_ROBIN --listener
listener1 --protocol HTTP
```

9. *백엔드 서버에 연결하고 경로(/)를 테스트하는 풀(**pool1**)에 상태 모니터를 생성합니다.*

예제

```
$ openstack loadbalancer healthmonitor create --delay 15 --max-retries 4 --timeout 10 --type
HTTP --url-path / pool1
```

10. *프라이빗 서브넷(**private_subnet**)의 비보안 **HTTP** 백엔드 서버(**192.0.2.10** 및 **192.0.2.11**)를 풀에 추가합니다.*

예제

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.10 --
protocol-port 80 pool1
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.11 --
protocol-port 80 pool1
```

검증

1. *로드 밸런서(**lb1**) 설정을 보고 확인합니다.*

예제

```
$ openstack loadbalancer show lb1
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
| description         |                                      |
| flavor              |                                      |
| id                  | 788fe121-3dec-4e1b-8360-4020642238b0 |
| listeners           | 09f28053-fde8-4c78-88b9-0f191d84120e |
| name                | lb1                                  |
| operating_status    | ONLINE                               |
| pools               | 627842b3-eed8-4f5f-9f4a-01a738e64d6a |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| provider            | amphora                              |
| provisioning_status | ACTIVE                               |
| updated_at          | 2022-01-15T11:12:42                  |
| vip_address         | 198.51.100.11                        |
| vip_network_id      | 9bca13be-f18d-49a5-a83d-9d487827fd16 |
| vip_port_id         | 69a85edd-5b1c-458f-96f2-b4552b15b8e6 |
| vip_qos_policy_id   | None                                 |
| vip_subnet_id       | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
+---------------------+--------------------------------------+
```

2. *상태 모니터가 올바르게 작동하고 있는 경우 각 멤버의 상태를 확인할 수 있습니다.*

*작동 중인 멤버(**b85c807e-4d7c-4cd-b725-5e8afddf80d2**)에는 **operating_status** 의 **ONLINE** 값이 있습니다.*

예제

```
$ openstack loadbalancer member show pool1 b85c807e-4d7c-4cbd-b725-5e8afddf80d2
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| address             | 192.0.2.10                           |
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
| id                  | b85c807e-4d7c-4cbd-b725-5e8afddf80d2 |
| name                |                                      |
| operating_status    | ONLINE                               |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| protocol_port       | 80                                   |
| provisioning_status | ACTIVE                               |
| subnet_id           | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
| updated_at          | 2022-01-15T11:12:42                  |
| weight              | 1                                    |
| monitor_port        | None                                 |
| monitor_address     | None                                 |
| backup              | False                                |
+---------------------+--------------------------------------+
```

추가 리소스

- OpenStack 키 관리자 가이드를 통해 시크릿 관리.
- 명령줄 인터페이스 참조 의 LoadBalancer

8.5. SNI를 사용하여 TLS 종료 HTTPS 로드 밸런서 생성

SNI(Server Name Indication) 기술을 사용하는 TLS 종료 HTTPS 로드 밸런서의 경우 단일 리스너에 여러 TLS 인증서를 포함하고 로드 밸런서에서 공유 IP를 사용할 때 표시할 인증서를 확인할 수 있습니다. 또한 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.

사전 요구 사항

- TCP 포트 80에서 비보안 HTTP 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷.
- 백엔드 서버는 URL 경로 / 에 상태 점검으로 구성됩니다.

- *인터넷에서 도달할 수 있는 공유 외부(공용) 서브넷입니다.*

- ***TLS** 공개 키 암호화는 다음과 같은 특성을 사용하여 구성됩니다.*

  - *로드 밸런서 **VIP** 주소에 할당된 **DNS** 이름(예: **www.example.com** 및 **www2.example.com**)의 외부 인증 기관(**CA**)에서 여러 **TLS** 인증서, 키 및 중간 인증서 체인을 확보할 수 있습니다.*

  - *키와 인증서는 **PEM**으로 인코딩됩니다.*

  - *키는 암호로 암호화되지 않습니다.*

- ***Key Manager** 서비스(**barbican**)를 사용하도록 로드 밸런싱 서비스(**octavia**)를 구성해야 합니다. 자세한 내용은 **OpenStack Key Manager**를 사용하여 시크릿 관리 가이드 를 참조하십시오.*

절차

1. ***SNI** 목록의 각 **TLS** 인증서에 대해 키(**server.key**), 인증서(**server.crt**) 및 중간 인증서 체인(**ca-chain.crt**)을 하나의 **PKCS12** 파일(**server.p12**)으로 결합합니다.*

   *이 예제에서는 인증서(**www.example.com** 및 **www2.example.com**)에 대해 두 개의 **PKCS12** 파일(**server.p12** 및 **server2.p12**) 을 생성합니다.*

   참고

   괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

   ```
   $ openssl pkcs12 -export -inkey server.key -in server.crt -certfile ca-chain.crt -passout pass: -out server.p12

   $ openssl pkcs12 -export -inkey server2.key -in server2.crt -certfile ca-chain2.crt -passout pass: -out server2.p12
   ```

2. *자격 증명 파일을 가져옵니다.*

예제

```
$ source ~/overcloudrc
```

3. *Key Manager 서비스를 사용하여 PKCS12 파일에 시크릿 리소스(tls_secret1 및 tls_secret2)를 생성합니다.*

```
$ openstack secret store --name='tls_secret1' -t 'application/octet-stream' -e 'base64' --
payload="$(base64 < server.p12)"
$ openstack secret store --name='tls_secret2' -t 'application/octet-stream' -e 'base64' --
payload="$(base64 < server2.p12)"
```

4. *공용 서브넷(public_subnet)에서 로드 밸런서(lb1)를 만듭니다.*

```
$ openstack loadbalancer create --name lb1 --vip-subnet-id public_subnet
```

5. *로드 밸런서의 상태를 모니터링합니다.*

예제

```
$ openstack loadbalancer show lb1
```

6. *다음 단계로 진행하기 전에 provisioning_status 가 EgressIP 인지 확인합니다.*

7. *TERMINATED_HTTPS 리스너(listener1)를 생성하고 SNI를 사용하여 보안 리소스를 모두 참조합니다.*

(리스너의 기본 TLS 컨테이너로 tls_secret1 참조)

```
$ openstack loadbalancer listener create --protocol-port 443 \
```

```
--protocol TERMINATED_HTTPS --name listener1 \
--default-tls-container=$(openstack secret list | awk '/ tls_secret1 / {print $2}') \
--sni-container-refs $(openstack secret list | awk '/ tls_secret1 / {print $2}') \
$(openstack secret list | awk '/ tls_secret2 / {print $2}') -- lb1
```

8. ***풀(pool1)을 만들고리스너의 기본 풀로 설정합니다.***

```
$ openstack loadbalancer pool create --name pool1 --lb-algorithm ROUND_ROBIN --listener
listener1 --protocol HTTP
```

9. ***백엔드 서버에 연결하고 경로(/)를 테스트하는 풀(pool1)에 상태 모니터를 생성합니다.***

예제

```
$ openstack loadbalancer healthmonitor create --delay 15 --max-retries 4 --timeout 10 --type
HTTP --url-path / pool1
```

10. ***프라이빗 서브넷(private_subnet)의 비보안 HTTP 백엔드 서버(192.0.2.10 및 192.0.2.11)를 풀에 추가합니다.***

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.10 --
protocol-port 80 pool1
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.11 --
protocol-port 80 pool1
```

검증

1. ***로드 밸런서(lb1) 설정을 보고 확인합니다.***

예제

```
$ openstack loadbalancer show lb1
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
| description         |                                      |
| flavor              |                                      |
| id                  | 788fe121-3dec-4e1b-8360-4020642238b0 |
| listeners           | 09f28053-fde8-4c78-88b9-0f191d84120e |
| name                | lb1                                  |
| operating_status    | ONLINE                               |
| pools               | 627842b3-eed8-4f5f-9f4a-01a738e64d6a |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| provider            | amphora                              |
| provisioning_status | ACTIVE                               |
| updated_at          | 2022-01-15T11:12:42                  |
| vip_address         | 198.51.100.11                        |
| vip_network_id      | 9bca13be-f18d-49a5-a83d-9d487827fd16 |
| vip_port_id         | 69a85edd-5b1c-458f-96f2-b4552b15b8e6 |
| vip_qos_policy_id   | None                                 |
| vip_subnet_id       | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
+---------------------+--------------------------------------+
```

2. *상태 모니터가 올바르게 작동하고 있는 경우 각 멤버의 상태를 확인할 수 있습니다.*

*작동 중인 멤버(**b85c807e-4d7c-4cd-b725-5e8afddf80d2**)에는 **operating_status** 의 **ONLINE** 값이 있습니다.*

예제

```
$ openstack loadbalancer member show pool1 b85c807e-4d7c-4cbd-b725-5e8afddf80d2
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| address             | 192.0.2.10                           |
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
| id                  | b85c807e-4d7c-4cbd-b725-5e8afddf80d2 |
| name                |                                      |
| operating_status    | ONLINE                               |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| protocol_port       | 80                                   |
| provisioning_status | ACTIVE                               |
| subnet_id           | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
| updated_at          | 2022-01-15T11:12:42                  |
| weight              | 1                                    |
| monitor_port        | None                                 |
| monitor_address     | None                                 |
| backup              | False                                |
+---------------------+--------------------------------------+
```

추가 리소스

- *OpenStack 키 관리자 가이드를 통해 시크릿 관리.*
- *명령줄 인터페이스 참조 의 LoadBalancer*

8.6. 동일한 IP 및 백엔드에서 HTTP 및 TLS 종료 HTTPS 로드 밸런싱 생성

클라이언트가 보안 또는 비보안 HTTP 프로토콜과 관계없이 동일한 로드 밸런서에서 비보안 리스너와 TLS 종료 HTTPS 리스너와 동일한 IP 주소를 구성할 수 있습니다. 또한 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.

사전 요구 사항

- *TCP 포트 80에서 비보안 HTTP 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷.*
- *백엔드 서버는 URL 경로 / 에 상태 점검으로 구성됩니다.*

- *인터넷에서 도달할 수 있는 공유 외부(공용) 서브넷입니다.*

- ***TLS** 공개 키 암호화는 다음과 같은 특성을 사용하여 구성됩니다.*

  - *로드 밸런서 **VIP** 주소(예: **www.example.com**)에 할당된 **DNS** 이름의 외부 인증 기관 **(CA)**에서 **TLS** 인증서, 키 및 선택적 중간 인증서 체인을 가져왔습니다.*

  - *인증서, 키 및 중간 인증서 체인은 현재 디렉터리의 별도의 파일에 있습니다.*

  - *키와 인증서는 **PEM**으로 인코딩됩니다.*

  - *키는 암호로 암호화되지 않습니다.*

  - *중간 인증서 체인에는 **PEM** 인코딩 및 연결된 여러 인증서가 포함되어 있습니다.*

- *키 관리자 서비스(**barbican**)를 사용하도록 로드 밸런싱 서비스(**octavia**)를 구성했습니다. 자세한 내용은 **OpenStack Key Manager**를 사용하여 시크릿 관리 가이드 를 참조하십시오.*

- *비보안 **HTTP** 리스너는 **HTTPS TLS** 종료 로드 밸런서와 동일한 풀로 구성됩니다.*

절차

1. *키(**server.key**), 인증서(**server.crt**) 및 중간 인증서 체인(**ca-chain.crt**)을 단일 **PKCS12** 파일 **(server.p12)**으로 결합합니다.*

참고

괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

```
$ openssl pkcs12 -export -inkey server.key -in server.crt -certfile ca-chain.crt -passout pass:
-out server.p12
```

2. *자격 증명 파일을 가져옵니다.*

예제

```
$ source ~/overcloudrc
```

3. ***Key Manager** 서비스를 사용하여 **PKCS12** 파일에 대한 시크릿 리소스(**tls_secret1**)를 생성합니다.*

```
$ openstack secret store --name='tls_secret1' -t 'application/octet-stream' -e 'base64' --
payload="$(base64 < server.p12)"
```

4. *공용 서브넷(**public_subnet**)에서 로드 밸런서(**lb1**)를 만듭니다.*

```
$ openstack loadbalancer create --name lb1 --vip-subnet-id public_subnet
```

5. *로드 밸런서의 상태를 모니터링합니다.*

예제

```
$ openstack loadbalancer show lb1
```

6. *다음 단계로 진행하기 전에 **provisioning_status** 가 **EgressIP** 인지 확인합니다.*

7. ***TERMINATED_HTTPS**리스너(리스너1)를 만들고 리스너의 기본 **TLS** 컨테이너로 시크릿 리소스를 참조합니다.*

```
$ openstack loadbalancer listener create --protocol-port 443 --protocol
TERMINATED_HTTPS --name listener1 --default-tls-container=$(openstack secret list | awk
'/ tls_secret1 / {print $2}') lb1
```

8. *풀(**pool1**)을 만들고리스너의 기본 풀로 설정합니다.*

```
$ openstack loadbalancer pool create --name pool1 --lb-algorithm ROUND_ROBIN --listener
listener1 --protocol HTTP
```

9. *백엔드 서버에 연결하고 경로(/)를 테스트하는 풀(**pool1**)에 상태 모니터를 생성합니다.*

예제

```
$ openstack loadbalancer healthmonitor create --delay 15 --max-retries 4 --timeout 10 --type
HTTP --url-path / pool1
```

10. *프라이빗 서브넷(**private_subnet**)의 비보안 **HTTP** 백엔드 서버(**192.0.2.10** 및 **192.0.2.11**)를 풀에 추가합니다.*

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.10 --
protocol-port 80 pool1
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.11 --
protocol-port 80 pool1
```

11. *비보안 **HTTP**리스너(리스너**2**)를 만들고기본 풀을 보안 리스너와 동일하게 설정합니다.*

```
$ openstack loadbalancer listener create --protocol-port 80 --protocol HTTP --name listener2
--default-pool pool1 lb1
```

검증

1. *로드 밸런서(**lb1**) 설정을 보고 확인합니다.*

예제

```
$ openstack loadbalancer show lb1
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
| description         |                                      |
| flavor              |                                      |
| id                  | 788fe121-3dec-4e1b-8360-4020642238b0 |
| listeners           | 09f28053-fde8-4c78-88b9-0f191d84120e |
| name                | lb1                                  |
| operating_status    | ONLINE                               |
| pools               | 627842b3-eed8-4f5f-9f4a-01a738e64d6a |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| provider            | amphora                              |
| provisioning_status | ACTIVE                               |
| updated_at          | 2022-01-15T11:12:42                  |
| vip_address         | 198.51.100.11                        |
| vip_network_id      | 9bca13be-f18d-49a5-a83d-9d487827fd16 |
| vip_port_id         | 69a85edd-5b1c-458f-96f2-b4552b15b8e6 |
| vip_qos_policy_id   | None                                 |
| vip_subnet_id       | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
+---------------------+--------------------------------------+
```

2. *상태 모니터가 올바르게 작동하고 있는 경우 각 멤버의 상태를 확인할 수 있습니다.*

*작동 중인 멤버(**b85c807e-4d7c-4cd-b725-5e8afddf80d2**)에는 **operating_status** 의 **ONLINE** 값이 있습니다.*

예제

```
$ openstack loadbalancer member show pool1 b85c807e-4d7c-4cbd-b725-5e8afddf80d2
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| address             | 192.0.2.10                           |
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
| id                  | b85c807e-4d7c-4cbd-b725-5e8afddf80d2 |
| name                |                                      |
| operating_status    | ONLINE                               |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| protocol_port       | 80                                   |
| provisioning_status | ACTIVE                               |
| subnet_id           | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
| updated_at          | 2022-01-15T11:12:42                  |
| weight              | 1                                    |
| monitor_port        | None                                 |
| monitor_address     | None                                 |
| backup              | False                                |
+---------------------+--------------------------------------+
```

추가 리소스

- ***OpenStack** 키 관리자 가이드를 통해 시크릿 관리.*
- *명령줄 인터페이스 참조 의 **LoadBalancer***

9장. 다른 종류의 로드 밸런서 생성

로드 밸런싱 서비스(octavia)를 사용하여 관리할 HTTP가 아닌 네트워크 트래픽 유형과 일치하는 로드 밸런서 유형을 생성합니다.

9.1. TCP 로드 밸런서 생성

비HTTP, TCP 기반 서비스 및 애플리케이션의 네트워크 트래픽을 관리해야 하는 경우 로드 밸런서를 생성할 수 있습니다. 또한 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.

사전 요구 사항

- 특정 TCP 포트에서 사용자 지정 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷입니다.
- 백엔드 서버는 URL 경로 / 에 상태 점검으로 구성됩니다.
- 인터넷에서 도달할 수 있는 공유 외부(공용) 서브넷입니다.

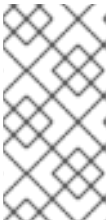
절차

1. 자격 증명 파일을 가져옵니다.

예제

```
$ source ~/overcloudrc
```

2. *공용 서브넷(**public_subnet**)에서 로드 밸런서(**lb1**)를 만듭니다.*



참고

괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

```
$ openstack loadbalancer create --name lb1 --vip-subnet-id public_subnet
```

3. *로드 밸런서의 상태를 확인합니다.*

예제

```
$ openstack loadbalancer show lb1
```

4. *다음 단계로 진행하기 전에 **provisioning_status** 가 **EgressIP** 인지 확인합니다.*

5. *사용자 지정 애플리케이션이 구성된 지정된 포트(**23456**)에 **TCP** 리스너(**리스너1**)를만듭니다.*

예제

```
$ openstack loadbalancer listener create --name listener1 --protocol TCP --protocol-port
23456 lb1
```

6. *풀(**pool1**)을 만들고리스너의 기본 풀로 설정합니다.*

예제

```
$ openstack loadbalancer pool create --name pool1 --lb-algorithm ROUND_ROBIN --listener
listener1 --protocol TCP
```

7. *백엔드 서버에 연결하고 **TCP** 서비스 포트를 프로브하는 풀(**pool1**)에 상태 모니터를 생성합니다.*

예제

```
$ openstack loadbalancer healthmonitor create --delay 15 --max-retries 4 --timeout 10 --type
TCP pool1
```

8. *사설 서브넷(**private_subnet**)에 백엔드 서버(**192.0.2.10** 및 **192.0.2.11**)를 풀에 추가합니다.*

예제

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.10 --
protocol-port 80 pool1
```

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.11 --
protocol-port 80 pool1
```

검증

1. *로드 밸런서(**lb1**) 설정을 보고 확인합니다.*

예제

```
$ openstack loadbalancer show lb1
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
| description         |                                      |
| flavor              |                                      |
| id                  | 788fe121-3dec-4e1b-8360-4020642238b0 |
| listeners           | 09f28053-fde8-4c78-88b9-0f191d84120e |
| name                | lb1                                  |
| operating_status    | ONLINE                               |
| pools               | 627842b3-eed8-4f5f-9f4a-01a738e64d6a |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| provider            | amphora                              |
| provisioning_status | ACTIVE                               |
| updated_at          | 2022-01-15T11:12:42                  |
| vip_address         | 198.51.100.11                        |
| vip_network_id      | 9bca13be-f18d-49a5-a83d-9d487827fd16 |
| vip_port_id         | 69a85edd-5b1c-458f-96f2-b4552b15b8e6 |
| vip_qos_policy_id   | None                                 |
| vip_subnet_id       | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
+---------------------+--------------------------------------+
```

2.

*상태 모니터가 올바르게 작동하고 있는 경우 각 멤버의 상태를 확인할 수 있습니다. 다음 명령을 사용하여 멤버 **ID**를 가져옵니다.*

예제

```
$ openstack loadbalancer member list pool1
```

*작동 중인 멤버(**b85c807e-4d7c-4cd-b725-5e8afddf80d2**)에는 **operating_status** 의 **ONLINE** 값이 있습니다.*

예제

```
$ openstack loadbalancer member show pool1 b85c807e-4d7c-4cbd-b725-5e8afddf80d2
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| address             | 192.0.2.10                           |
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
| id                  | b85c807e-4d7c-4cbd-b725-5e8afddf80d2 |
| name                |                                      |
| operating_status    | ONLINE                               |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| protocol_port       | 80                                   |
| provisioning_status | ACTIVE                               |
| subnet_id           | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
| updated_at          | 2022-01-15T11:12:42                  |
| weight              | 1                                    |
| monitor_port        | None                                 |
| monitor_address     | None                                 |
| backup              | False                                |
+---------------------+--------------------------------------+
```

추가 리소스

- 명령줄 인터페이스 참조 의 **LoadBalancer**

9.2. 상태 모니터를 사용하여 UDP 로드 밸런서 생성

UDP 포트에서 네트워크 트래픽을 관리해야 하는 경우 로드 밸런서를 생성할 수 있습니다. 또한 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.

사전 요구 사항

- **UDP** 포트를 사용하도록 구성된 하나 이상의 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷입니다.
- 인터넷에서 도달할 수 있는 공유 외부(공용) 서브넷입니다.
- 백엔드 서버는 **UDP** 상태 점검으로 구성됩니다.
- **ICMP** 대상 **Unreachable** 메시지를 차단하는 보안 규칙이 없습니다(**ICMP** 유형 **3**).

절차

1. 자격 증명 파일을 가져옵니다.

예제

```
$ source ~/overcloudrc
```

2. *프라이빗 서브넷(**private_subnet**)에서 로드 밸런서(**lb1**)를 만듭니다.*

참고

괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

```
$ openstack loadbalancer create --name lb1 --vip-subnet-id private_subnet
```

3. *로드 밸런서의 상태를 확인합니다.*

예제

```
$ openstack loadbalancer show lb1
```

4. *다음 단계로 진행하기 전에 **provisioning_status** 가 **EgressIP** 인지 확인합니다.*

5. *포트(**1234**)에 리스너(**리스너1**)를 만듭니다.*

예제

```
$ openstack loadbalancer listener create --name listener1 --protocol UDP --protocol-port
1234 lb1
```

6. *리스너 기본 풀(**pool1**)을 생성합니다.*

예제

```
$ openstack loadbalancer pool create --name pool1 --lb-algorithm ROUND_ROBIN --listener
listener1 --protocol UDP
```

7. ***UDP(UDP-CONNECT)**를 사용하여 백엔드 서버에 연결하는 풀(**pool1**)에 상태 모니터를 만듭니다.*

예제

```
$ openstack loadbalancer healthmonitor create --delay 5 --max-retries 2 --timeout 3 --type
UDP-CONNECT pool1
```

8. *프라이빗 서브넷(**private_subnet**)에서 로드 밸런서 구성원 (**192.0.2.10** 및 **192.0.2.11**)을 기본 풀에 추가합니다.*

예제

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.10 --
protocol-port 1234 pool1
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.11 --
protocol-port 1234 pool1
```

검증

1. *로드 밸런서(**lb1**) 설정을 보고 확인합니다.*

예제

```
$ openstack loadbalancer show lb1
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
| description         |                                      |
| flavor              |                                      |
| id                  | 788fe121-3dec-4e1b-8360-4020642238b0 |
| listeners           | 09f28053-fde8-4c78-88b9-0f191d84120e |
| name                | lb1                                  |
| operating_status    | ONLINE                               |
| pools               | 627842b3-eed8-4f5f-9f4a-01a738e64d6a |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| provider            | amphora                              |
| provisioning_status | ACTIVE                               |
| updated_at          | 2022-01-15T11:12:42                  |
| vip_address         | 198.51.100.11                        |
| vip_network_id      | 9bca13be-f18d-49a5-a83d-9d487827fd16 |
| vip_port_id         | 69a85edd-5b1c-458f-96f2-b4552b15b8e6 |
| vip_qos_policy_id   | None                                 |
| vip_subnet_id       | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
+---------------------+--------------------------------------+
```

2. *상태 모니터가 올바르게 작동하고 있는 경우 각 멤버의 상태를 확인할 수 있습니다. 다음 명령을 사용하여 멤버 **ID**를 가져옵니다.*

예제

```
$ openstack loadbalancer member list pool1
```

작동 중인 멤버(**b85c807e-4d7c-4cd-b725-5e8afddf80d2**)에는 **operating_status** 의 **ONLINE** 값이 있습니다.

예제

```
$ openstack loadbalancer member show pool1 b85c807e-4d7c-4cbd-b725-5e8afddf80d2
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| address             | 192.0.2.10                           |
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
| id                  | b85c807e-4d7c-4cbd-b725-5e8afddf80d2 |
| name                |                                      |
| operating_status    | ONLINE                               |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| protocol_port       | 1234                                 |
| provisioning_status | ACTIVE                               |
| subnet_id           | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
| updated_at          | 2022-01-15T11:12:42                  |
| weight              | 1                                    |
| monitor_port        | None                                 |
| monitor_address     | None                                 |
| backup              | False                                |
+---------------------+--------------------------------------+
```

추가 리소스

- 명령줄 인터페이스 참조 의 **LoadBalancer**

9.3. QOS-RULED 로드 밸런서 생성

RHOSP(Red Hat OpenStack Platform) 네트워킹 서비스(**neutron**) **QoS(Quality of Service)** 정책을 로드 밸런서를 사용하는 **VIP(**가상 **IP** 주소**)**에 적용할 수 있습니다. 이러한 방식으로 **QoS** 정책을 사용하여 로드 밸런서에서 관리할 수 있는 수신 또는 발신 네트워크 트래픽을 제한할 수 있습니다. 또한 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.

사전 요구 사항

- **TCP** 포트 **80**에서 **HTTP** 애플리케이션으로 구성된 백엔드 서버가 포함된 프라이빗 서브넷입니다.
- 백엔드 서버는 **URL** 경로 **/** 에 상태 점검으로 구성됩니다.
- 인터넷에서 도달할 수 있는 공유 외부(공용) 서브넷입니다.
- **RHOSP Networking** 서비스에 대해 생성된 대역폭 제한 규칙이 포함된 **QoS** 정책입니다.

절차

1. 자격 증명 파일을 가져옵니다.

   예제

   ```
   $ source ~/overcloudrc
   ```

2. 최대 **1024 kbps** 및 최대 **Burst** 속도 **1024 kbps**를 사용하여 네트워크 대역폭 **QoS** 정책 **(qos_policy_bandwidth)**을 생성합니다.

참고

괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

```
$ openstack network qos policy create qos_policy_bandwidth
$ openstack network qos rule create --type bandwidth-limit --max-kbps 1024 --max-burst-kbits 1024 qos-policy-bandwidth
```

3. ***QoS** 정책(**qos-policy-bandwidth**)을 사용하여 공용 서브넷(**public_subnet**)에 로드 밸런서(**lb1**)를 만듭니다.*

예제

```
$ openstack loadbalancer create --name lb1 --vip-subnet-id public_subnet --vip-qos-policy-id qos-policy-bandwidth
```

4. *로드 밸런서의 상태를 확인합니다.*

예제

```
$ openstack loadbalancer show lb1
```

5. *다음 단계로 진행하기 전에 **provisioning_status** 가 **EgressIP** 인지 확인합니다.*

6. *포트(**80**)에 리스너(리스너**1**)를 만듭니다.*

예제

```
$ openstack loadbalancer listener create --name listener1 --protocol HTTP --protocol-port 80
lb1
```

7. *리스너 기본 풀(**pool1**)을 생성합니다.*

예제

```
$ openstack loadbalancer pool create --name pool1 --lb-algorithm ROUND_ROBIN --listener
listener1 --protocol HTTP
```

8. *백엔드 서버에 연결하고 경로(/)를 테스트하는 풀에 상태 모니터를 생성합니다.*

예제

```
$ openstack loadbalancer healthmonitor create --delay 15 --max-retries 4 --timeout 10 --type
HTTP --url-path / pool1
```

9. *프라이빗 서브넷(**private_subnet**)에서 로드 밸런서 구성원 (**192.0.2.10** 및 **192.0.2.11**)을 기본 풀에 추가합니다.*

예제

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.10 --
protocol-port 80 pool1

$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.11 --
protocol-port 80 pool1
```

검증

1. *리스너(**listener1**) 설정을 보고 확인합니다.*

예제

```
$ openstack loadbalancer list
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
| description         |                                      |
| flavor              |                                      |
| id                  | 788fe121-3dec-4e1b-8360-4020642238b0 |
| listeners           | 09f28053-fde8-4c78-88b9-0f191d84120e |
| name                | lb1                                  |
| operating_status    | ONLINE                               |
| pools               | 627842b3-eed8-4f5f-9f4a-01a738e64d6a |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| provider            | amphora                              |
| provisioning_status | ACTIVE                               |
| updated_at          | 2022-01-15T11:12:42                  |
| vip_address         | 198.51.100.11                        |
| vip_network_id      | 9bca13be-f18d-49a5-a83d-9d487827fd16 |
| vip_port_id         | 69a85edd-5b1c-458f-96f2-b4552b15b8e6 |
```

```
| vip_qos_policy_id   | cdfc3398-997b-46eb-9db1-ebbd88f7de05 |
| vip_subnet_id       | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
+---------------------+--------------------------------------+
```

이 예제에서 매개 변수 **vip_qos_policy_id** 에는 정책 **ID**가 포함됩니다.

추가 리소스

- 명령줄 인터페이스 참조 의 **LoadBalancer**

9.4. 액세스 제어 목록을 사용하여 로드 밸런서 생성

ACL(액세스 제어 목록)을 생성하여 리스너로 들어오는 트래픽을 허용된 소스 **IP** 주소 세트로 제한할 수 있습니다. 기타 들어오는 트래픽은 거부됩니다. 또한 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.

사전 요구 사항

- **TCP** 포트 **80**에서 사용자 지정 애플리케이션으로 구성된 백엔드 서버가 포함된 프라이빗 서브넷입니다.
- 백엔드 서버는 **URL** 경로 **/** 에 상태 점검으로 구성됩니다.
- 인터넷에서 도달할 수 있는 공유 외부(공용) 서브넷입니다.

절차

1. 자격 증명 파일을 가져옵니다.

예제

```
$ source ~/overcloudrc
```

2. *공용 서브넷(**public_subnet**)에서 로드 밸런서(**lb1**)를 만듭니다.*

참고

괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

```
$ openstack loadbalancer create --name lb1 --vip-subnet-id public_subnet
```

3. *로드 밸런서의 상태를 확인합니다.*

예제

```
$ openstack loadbalancer show lb1
```

4. *다음 단계로 진행하기 전에 **provisioning_status** 가 **EgressIP** 인지 확인합니다.*

5. *허용된 **CIDR(192.0.2.0/24** 및 **198.51.100.0/24**)을 사용하여 리스너(**리스너1**)를 만듭니다.*

예제

```
$ openstack loadbalancer listener create --name listener1 --protocol TCP --protocol-port 80 --allowed-cidr 192.0.2.0/24 --allowed-cidr 198.51.100.0/24 lb1
```

-

6. *리스너 기본 풀(**pool1**)을 생성합니다.*

예제

```
$ openstack loadbalancer pool create --name pool1 --lb-algorithm ROUND_ROBIN --listener
listener1 --protocol TCP
```

7. *백엔드 서버에 연결하고 경로(/)를 테스트하는 풀에 상태 모니터를 생성합니다.*

예제

```
$ openstack loadbalancer healthmonitor create --delay 15 --max-retries 4 --timeout 10 --type
HTTP --url-path / pool1
```

8. *프라이빗 서브넷(**private_subnet**)에서 로드 밸런서 구성원 (**192.0.2.10** 및 **192.0.2.11**)을 기본 풀에 추가합니다.*

예제

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.10 --
protocol-port 80 pool1

$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.11 --
protocol-port 80 pool1
```

검증

1. *리스너(**listener1**) 설정을 보고 확인합니다.*

예제

```
$ openstack loadbalancer listener show listener1
```

샘플 출력

```
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| connection_limit          | -1                                   |
| created_at                | 2022-01-15T11:11:09                  |
| default_pool_id           | None                                 |
| default_tls_container_ref | None                                 |
| description               |                                      |
| id                        | d26ba156-03c3-4051-86e8-f8997a202d8e |
| insert_headers            | None                                 |
| l7policies                |                                      |
| loadbalancers             | 2281487a-54b9-4c2a-8d95-37262ec679d6 |
| name                      | listener1                            |
| operating_status          | ONLINE                               |
| project_id                | 308ca9f600064f2a8b3be2d57227ef8f     |
| protocol                  | TCP                                  |
| protocol_port             | 80                                   |
| provisioning_status       | ACTIVE                               |
| sni_container_refs        | []                                   |
| timeout_client_data       | 50000                                |
| timeout_member_connect    | 5000                                 |
| timeout_member_data       | 50000                                |
| timeout_tcp_inspect       | 0                                    |
| updated_at                | 2022-01-15T11:12:42                  |
| client_ca_tls_container_ref | None                               |
| client_authentication     | NONE                                 |
| client_crl_container_ref  | None                                 |
| allowed_cidrs             | 192.0.2.0/24                         |
|                           | 198.51.100.0/24                      |
+---------------------------+--------------------------------------+
```

*이 예에서 **allowed_cidrs** 매개 변수는 **192.0.2.0/24** 및 **198.51.100.0/24**의 트래픽만 허용하도록 설정됩니다.*

2. *로드 밸런서가 안전한지 확인하려면 **CIDR**이 **allowed_cidrs** 목록에 없는 클라이언트의 리스너에 대한 요청을 확인하십시오. 요청이 성공하지 못합니다.*

샘플 출력

```
curl: (7) Failed to connect to 203.0.113.226 port 80: Connection timed out
curl: (7) Failed to connect to 203.0.113.226 port 80: Connection timed out
curl: (7) Failed to connect to 203.0.113.226 port 80: Connection timed out
curl: (7) Failed to connect to 203.0.113.226 port 80: Connection timed out
```

추가 리소스

- *명령줄 인터페이스 참조 의 **LoadBalancer***

9.5. OVN 로드 밸런서 생성

***RHOSP(Red Hat OpenStack Platform)** 클라이언트를 사용하여 **RHOSP** 배포에서 네트워크 트래픽을 관리하는 로드 밸런서를 생성할 수 있습니다. **RHOSP Load-Balancing** 서비스는 **ML2/OVN(Open Virtual Network** 메커니즘 드라이버)을 사용하여 **neutron Modular Layer 2** 플러그인을 지원합니다.*

사전 요구 사항

- ***ML2/OVN** 프로바이더 드라이버를 배포해야 합니다.*

중요

***OVN** 공급자는 계층 **4 TCP** 및 **UDP** 네트워크 트래픽 및 **SOURCE_IP_PORT** 로드 밸런서 알고리즘만 지원합니다. **OVN** 공급자는 상태 모니터링을 지원하지 않습니다.*

- *특정 **TCP** 포트에서 사용자 지정 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷입니다.*

- *인터넷에서 도달할 수 있는 공유 외부(공용) 서브넷입니다.*

절차

1. *자격 증명 파일을 가져옵니다.*

   예제

   ```
   $ source ~/overcloudrc
   ```

2. ***--provider ovn** 인수를 사용하여 프라이빗 서브넷(**private_subnet**)에 로드 밸런서 장치(**lb1**)를 만듭니다.*

   참고

   괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

   예제

   ```
   $ openstack loadbalancer create --name lb1 --provider ovn --vip-subnet-id private_subnet
   ```

3. *로드 밸런서의 상태를 확인합니다.*

   ```
   $ openstack loadbalancer show lb1
   ```

4.

*다음 단계로 진행하기 전에 **provisioning_status** 가 **EgressIP** 인지 확인합니다.*

5.

*사용자 지정 애플리케이션이 구성된 지정된 포트(**80**)에서 프로토콜(**tcp**)을 사용하는 리스너(리스너**1**)를 만듭니다.*

참고

***OVN** 공급자는 계층 **4 TCP** 및 **UDP** 네트워크 트래픽만 지원합니다.*

예제

```
$ openstack loadbalancer listener create --name listener1 --protocol tcp --protocol-port 80
lb1
```

6.

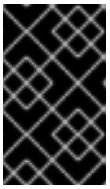
*리스너 기본 풀(**pool1**)을 생성합니다.*

참고

***OVN**에 대해 지원되는 유일한 부하 분산 알고리즘은 **SOURCE_IP_PORT** 입니다.*

예제

```
$ openstack loadbalancer pool create --name pool1 --lb-algorithm SOURCE_IP_PORT --
listener listener1 --protocol tcp
```



중요

***OVN**은 부하 분산을 위해 상태 모니터 기능을 지원하지 않습니다.*

7. *사설 서브넷(**private_subnet**)에 백엔드 서버(**192.0.2.10** 및 **192.0.2.11**)를 풀에 추가합니다.*

예제

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.10 --
protocol-port 80 pool1
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.11 --
protocol-port 80 pool1
```

검증

1. *로드 밸런서(**lb1**) 설정을 보고 확인합니다.*

예제

```
$ openstack loadbalancer show lb1
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:11:09                  |
| description         |                                      |
| flavor              |                                      |
| id                  | 788fe121-3dec-4e1b-8360-4020642238b0 |
| listeners           | 09f28053-fde8-4c78-88b9-0f191d84120e |
```

```
| name                | lb1                                  |
| operating_status    | ONLINE                               |
| pools               | 627842b3-eed8-4f5f-9f4a-01a738e64d6a |
| project_id          | dda678ca5b1241e7ad7bf7eb211a2fd7     |
| provider            | ovn                                  |
| provisioning_status | ACTIVE                               |
| updated_at          | 2022-01-15T11:12:42                  |
| vip_address         | 198.51.100.11                        |
| vip_network_id      | 9bca13be-f18d-49a5-a83d-9d487827fd16 |
| vip_port_id         | 69a85edd-5b1c-458f-96f2-b4552b15b8e6 |
| vip_qos_policy_id   | None                                 |
| vip_subnet_id       | 5bd7334b-49b3-4849-b3a2-b0b83852dba1 |
+---------------------+--------------------------------------+
```

2. ***openstack loadbalancer*** *리스너* ***show*** *명령을 실행하여 리스너 세부 정보를 확인합니다.*

예제

```
$ openstack loadbalancer listener show listener1
```

샘플 출력

```
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| connection_limit          | -1                                   |
| created_at                | 2022-01-15T11:13:52                  |
| default_pool_id           | a5034e7a-7ddf-416f-9c42-866863def1f2 |
| default_tls_container_ref | None                                 |
| description               |                                      |
| id                        | a101caba-5573-4153-ade9-4ea63153b164 |
| insert_headers            | None                                 |
| l7policies                |                                      |
| loadbalancers             | 653b8d79-e8a4-4ddc-81b4-e3e6b42a2fe3 |
| name                      | listener1                            |
| operating_status          | ONLINE                               |
| project_id                | 7982a874623944d2a1b54fac9fe46f0b     |
| protocol                  | TCP                                  |
| protocol_port             | 64015                                |
```

```
| provisioning_status        | ACTIVE                               |
| sni_container_refs         | []                                   |
| timeout_client_data        | 50000                                |
| timeout_member_connect     | 5000                                 |
| timeout_member_data        | 50000                                |
| timeout_tcp_inspect        | 0                                    |
| updated_at                 | 2022-01-15T11:15:17                  |
| client_ca_tls_container_ref | None                                |
| client_authentication      | NONE                                 |
| client_crl_container_ref   | None                                 |
| allowed_cidrs              | None                                 |
+----------------------------+--------------------------------------+
```

3. ***openstack loadbalancer pool show*** *명령을 실행하여 풀(**pool1**) 및로드 밸런서 구성원을 확인합니다.*

예제

```
$ openstack loadbalancer pool show pool1
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| admin_state_up      | True                                 |
| created_at          | 2022-01-15T11:17:34                  |
| description         |                                      |
| healthmonitor_id    |                                      |
| id                  | a5034e7a-7ddf-416f-9c42-866863def1f2 |
| lb_algorithm        | SOURCE_IP_PORT                       |
| listeners           | a101caba-5573-4153-ade9-4ea63153b164 |
| loadbalancers       | 653b8d79-e8a4-4ddc-81b4-e3e6b42a2fe3 |
| members             | 90d69170-2f73-4bfd-ad31-896191088f59 |
| name                | pool1                                |
| operating_status    | ONLINE                               |
| project_id          | 7982a874623944d2a1b54fac9fe46f0b     |
| protocol            | TCP                                  |
| provisioning_status | ACTIVE                               |
| session_persistence | None                                 |
```

```
| updated_at          | 2022-01-15T11:18:59                  |
| tls_container_ref   | None                                 |
| ca_tls_container_ref | None                                |
| crl_container_ref   | None                                 |
| tls_enabled         | False                                |
+---------------------+--------------------------------------+
```

추가 리소스

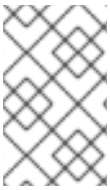
- *명령줄 인터페이스 참조 의* ***LoadBalancer***

10장. 계층 7 로드 밸런싱 구현

계층 7 정책과 함께 **Red Hat OpenStack Platform** 로드 밸런싱 서비스(**octavia**)를 사용하여 비즈니스 요구 사항을 충족하기 위해 여러 기준을 사용하여 **HTTP** 요청을 특정 애플리케이션 서버 풀로 리디렉션할 수 있습니다.

10.1. 계층 7 로드 밸런싱 정보

계층 7(L7) 로드 밸런싱은 OSI(Open Systems Interconnection) 모델에서 이름을 사용하여 로드 밸런서에서 계층 7(애플리케이션) 데이터를 기반으로 백엔드 애플리케이션 서버 풀에 요청을 배포합니다. 다음은 L7 부하 분산을 의미하는 다양한 용어입니다. 요청 스위칭,애플리케이션 부하 분산, 컨텐츠 기반 라우팅,스위칭 또는 밸런싱 입니다. Red Hat OpenStack Platform 로드 밸런싱 서비스(octavia)는 L7 로드 밸런싱에 대한 강력한 지원을 제공합니다.



참고

UDP 로드 밸런서를 사용하여 L7 정책 및 규칙을 생성할 수 없습니다.

L7 로드 밸런서는 여러 백엔드 풀을 대신하여 요청을 수락하고 애플리케이션 데이터를 사용하여 지정된 요청을 처리하는 정책을 기반으로 이러한 요청을 배포하는 리스너로 구성됩니다. 이를 통해 애플리케이션 인프라를 구체적으로 조정하고 특정 유형의 콘텐츠를 제공하도록 최적화할 수 있습니다. 예를 들어 백엔드 서버 한 그룹(풀)을 튜닝하여 이미지만 제공할 수 있습니다. 다른 그룹은 PHP 및 ASP와 같은 서버 측 스크립팅 언어 실행 및 HTML, CSS 및 JavaScript와 같은 정적 콘텐츠를 위해 다른 그룹입니다.

낮은 수준의 부하 분산과 달리 L7 로드 밸런싱에서는 부하 분산 서비스의 모든 풀에 동일한 콘텐츠를 포함할 필요가 없습니다. L7 로드 밸런서는 애플리케이션 메시지의 URI, 호스트, HTTP 헤더 및 기타 데이터를 기반으로 요청을 보낼 수 있습니다.

10.2. 로드 밸런싱 서비스의 7 계층 로드 밸런싱

잘 정의된 L7 애플리케이션 인터페이스에 대해 7 계층(L7) 로드 밸런싱을 구현할 수 있지만 Red Hat OpenStack Platform 로드 밸런싱 서비스(octavia)의 L7 기능은 HTTP 및 TERMINATED_HTTPS 프로토콜 및 의미만 나타냅니다.

Neutron LBaaS 및 로드 밸런싱 서비스는 L7 로드 밸런싱 논리에 L7 규칙과 정책을 사용합니다. L7 규칙은 true 또는 false로 평가되는 간단한 단일 논리 테스트입니다. L7 정책은 정책과 관련된 모든 규칙이 일치하는 경우 L7 규칙 및 정의된 작업을 수행하는 작업입니다.

10.3. 계층 7 로드 밸런싱 규칙

Red Hat OpenStack Platform 로드 밸런싱 서비스(octavia)의 경우 계층 7(L7) 로드 밸런싱 규칙은 true 또는 false를 반환하는 간단한 단일 논리 테스트입니다. 규칙 유형, 비교 유형, 값 및 규칙 유형에 따라 사용되는 선택적 키로 구성됩니다. L7 규칙은 항상 L7 정책과 연결되어야 합니다.

참고

UDP 로드 밸런서를 사용하여 L7 정책 및 규칙을 생성할 수 없습니다.

추가 리소스

- 10.4절. “계층 7 로드 밸런싱 규칙 유형”

10.4. 계층 7 로드 밸런싱 규칙 유형

Red Hat OpenStack Platform 로드 밸런싱 서비스(octavia)에는 다음과 같은 7 계층 로드 밸런싱 규칙이 있습니다.

- HOST_NAME: 규칙은 요청의 HTTP/1.1 호스트 이름을 규칙의 value 매개변수와 비교합니다.
- PATH: 규칙은 HTTP URI의 경로 부분을 규칙의 value 매개 변수와 비교합니다.
- FILE_TYPE: 규칙은 URI의 마지막 부분을 규칙의 value 매개변수(예: txt, mtls 등)와 비교합니다.

- *헤더: 규칙은 **key** 매개 변수에 정의된 헤더를 찾고 규칙의 **value** 매개변수와 비교합니다.*

- ***COOKIE:** 규칙은 **key** 매개 변수로 이름이 지정된 쿠키를 찾고 규칙의 **value** 매개변수와 비교합니다.*

- ***SSL_CONN_HAS_CERT:** 클라이언트에 **TLS** 클라이언트 인증에 대한 인증서가 제공된 경우 규칙과 일치합니다. 이는 인증서가 유효함을 의미하는 것은 아닙니다.*

- ***SSL_VERIFY_RESULT:** 이 규칙은 **TLS** 클라이언트 인증 인증서 검증 결과와 일치합니다. 값 **0(0)**은 인증서가 성공적으로 검증되었음을나타냅니다. **0**보다 큰 값은 인증서가 검증에 실패했음을 의미합니다. 이 값은 **openssl-ver** 결과 코드를 따릅니다.*

- ***SSL_DN_FIELD:** 규칙은 **key** 매개 변수에 정의된 고유 이름 필드를 찾고 규칙의 **value** 매개변수와 비교합니다.*

추가 리소스

- *10.3절. “계층 7 로드 밸런싱 규칙”*

10.5. 계층 7 로드 밸런싱 규칙 비교 유형

***Red Hat OpenStack Platform** 로드 밸런싱 서비스(**octavia**)의 경우 지정된 유형의 계층 **7** 로드 밸런싱 규칙이 항상 비교를 수행합니다. 로드 밸런싱 서비스는 다음 유형의 비교를 지원합니다. 모든 규칙 유형이 모든 비교 유형을 지원하는 것은 아닙니다.*

- ***REGEX: Perl** 유형 정규식 일치*

- ***STARTS_WITH:** 문자열 시작*

- ***ENDS_WITH:** 문자열이 다음과 같이 끝납니다.*

- *포함: 문자열 포함*

- *EQUAL_TO: 문자열과 같음*

추가 리소스

- *10.4절. "계층 7 로드 밸런싱 규칙 유형"*

10.6. 계층 7 로드 밸런싱 규칙 결과 인버전

일부 정책에 필요한 논리와 Red Hat OpenStack Platform 로드 밸런싱 서비스(octavia)가 사용하는 논리를 완전히 표현하기 위해 계층 7 로드 밸런싱 규칙이 반전될 수 있습니다. 지정된 규칙의 invert 매개 변수가 true이면 비교 결과가 반전됩니다.

예를 들어, 반전 된 규칙과 효과적으로 같으면 규칙과 같지 않습니다. 반전된 regex 규칙은 지정된 정규 표현식이 일치하지 않는 경우에만 true 를 반환합니다.

추가 리소스

- *10.7절. "계층 7 로드 밸런싱 정책"*

10.7. 계층 7 로드 밸런싱 정책

Red Hat OpenStack Platform 로드 밸런싱 서비스(octavia)의 경우 7 계층(L7) 로드 밸런싱 정책은 리스너와 연결된 L7 규칙 컬렉션이며 백엔드 풀에 연결할 수도 있습니다. 정책은 정책의 모든 규칙이 true인 경우 로드 밸런서에서 수행하는 작업입니다.

참고

UDP 로드 밸런서를 사용하여 L7 정책 및 규칙을 생성할 수 없습니다.

추가 리소스

- *10.3절. "계층 7 로드 밸런싱 규칙"*

10.8. 계층 7 로드 밸런싱 정책 논리

Red Hat OpenStack Platform 로드 밸런싱 서비스(octavia), 계층 7 로드 밸런싱 정책에서는 다음 논

리를 사용합니다. 즉, 지정된 정책과 관련된 모든 규칙은 논리적으로 **AND-ed**됩니다. 요청은 정책과 일치하려면 모든 정책 규칙과 일치해야 합니다.

규칙 간에 논리 **OR** 작업을 표현해야 하는 경우 동일한 작업으로 여러 정책을 만들거나 더 정교한 정규 표현식을 만듭니다.

추가 리소스

- 10.3절. "계층 7 로드 밸런싱 규칙"

10.9. 계층 7 로드 밸런싱 정책 작업

계층 **7** 로드 밸런싱 정책이 지정된 요청과 일치하면 해당 정책 작업이 실행됩니다. 다음은 **L7** 정책이 수행할 수 있는 작업입니다.

- 거부: 적절한 응답 코드를 사용하여 요청이 거부되고 백엔드 풀로 전달되지 않습니다.
- **REDIRECT_TO_URL:** 요청은 **redirect_url** 매개 변수에 정의된 **URL**로 **HTTP** 리디렉션을 전송합니다.
- **REDIRECT_PREFIX:** 이 정책과 일치하는 요청은 이 접두사 **URL**로 리디렉션됩니다.
- **REDIRECT_TO_POOL:** 요청은 **L7** 정책과 연결된 백엔드 풀로 전달됩니다.

추가 리소스

- 10.7절. "계층 7 로드 밸런싱 정책"

10.10. 계층 7 로드 밸런싱 정책 위치

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스**(octavia):** 여러 계층 **7(L7)** 로드 밸런싱 정책이 리스너와 연결된 경우, 정책 위치 매개 변수의 값이 중요합니다. **Location** 매개 변수는 **L7** 정책이 평가되는 순서를 결정할 때 사용됩니다. 정책 위치는 다음과 같은 방식으로 리스너 동작에 영향을 미칩니다.

- 로드 밸런싱 서비스**(haproxy amphorae)**의 참조 구현에서 **HAProxy**는 정책 작업과 관련하여

다음과 같은 순서를 적용합니다.

- **REJECT** 정책은 다른 모든 정책보다 우선합니다.
- **REDIRECT_TO_URL** 정책이 **REDIRECT_TO_POOL** 정책보다 우선합니다.
- **REDIRECT_TO_POOL** 정책은 위의 모든 항목 이후에만 평가되며 정책의 위치가 지정하는 순서로만 평가됩니다.

- **L7** 정책은 **position** 속성에 정의된 대로 특정 순서로 평가되며, 지정된 요청과 일치하는 첫 번째 정책은 작업이 따르는 것입니다.
- 정책이 지정된 요청과 일치하지 않으면 요청이 있는 경우 리스너의 기본 풀로 라우팅됩니다. 리스너에 기본 풀이 없으면 **503** 오류가 반환됩니다.
- 정책 위치 번호 지정은**1**부터 시작합니다.
- 기존 정책과 일치하는 위치로 새 정책이 생성되면 새 정책이 지정된 위치에 삽입됩니다.
- 위치를 지정하지 않고 새 정책을 만들거나 목록에 이미 있는 정책 수보다 큰 위치를 지정하면 새 정책이 목록에 추가됩니다.
- 정책을 삽입, 삭제 또는 목록에 추가하면 정책 위치 값이 번호를 건너뛰지 않고 하나(**1**에서 다시 정렬됩니다. 예를 들어 정책 **A, B** 및 **C**에 각각 **1,2** 및 **3** 의 위치 값이 있는 경우 목록에서 정책 **B**를 삭제하면 정책 **C**의 위치가 **2** 가 됩니다.

추가 리소스

- 10.7절. “계층 7 로드 밸런싱 정책”

10.11. 보안되지 않은 HTTP 요청 리디렉션

7 계층(L7) 정책과 함께 RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 사용하여 비보안 TCP 포트에서 수신한 HTTP 요청을 보안 TCP 포트로 리디렉션할 수 있습니다.

이 예에서 비보안 TCP 포트 80에 도착하는 HTTP 요청은 보안 TCP 포트 443으로 리디렉션됩니다.

사전 요구 사항

- *리스너(리스너1)및풀(pool1)이 있는TLS 종료 HTTPS 로드 밸런서(lb1). 자세한 내용은 TLS 종료 HTTPS 로드 밸런서 생성 을 참조하십시오.*

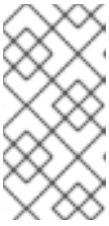
절차

1. *자격 증명 파일을 가져옵니다.*

   예제

   ```
   $ source ~/overcloudrc
   ```

2. *로드 밸런서(lb1) 포트(80)에서 HTTP 리스너(http_listener)를 만듭니다.*

   참고

   괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

   예제

   ```
   $ openstack loadbalancer listener create --name http_listener --protocol HTTP --protocol-port
   80 lb1
   ```

3. *리스너(http_listener)에 L7 정책(policy1)을 만듭니다. 정책에는 작업*

(REDIRECT_TO_URL)이 포함되어야 하며 URL(https://www.example.com/)을 가리켜야 합니다.

예제

```
$ openstack loadbalancer l7policy create --action REDIRECT_PREFIX --redirect-prefix
https://www.example.com/ --name policy1 http_listener
```

4. *모든 요청과 일치하는 L7 규칙을 정책(policy1)에 추가합니다.*

예제

```
$ openstack loadbalancer l7rule create --compare-type STARTS_WITH --type PATH --value
/ policy1
```

검증

1. *openstack loadbalancer l7policy list 명령을 실행하고 정책, policy1 이 있는지 확인합니다.*

2. *openstack loadbalancer l7rule list <l7policy> 명령을 실행하고 STARTS_WITH 의 compare_type 이 있는 규칙이 있는지 확인합니다.*

예제

```
$ openstack loadbalancer l7rule list policy1
```

추가 리소스

- *LoadBalancer 리스너가 명령줄 인터페이스 참조에서 생성*
- *명령줄 인터페이스 참조에 LoadBalancer l7policy 생성*
- *명령줄 인터페이스 참조에 LoadBalancer l7rule 생성*

10.12. 풀에 대한 시작 경로를 기반으로 요청 리디렉션

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 사용하여 HTTP 요청을 대체 서버 풀로 리디렉션할 수 있습니다. 요청의 URL에서 하나 이상의 시작 경로와 일치하도록 계층 7(L7) 정책을 정의할 수 있습니다.

이 예에서는 /js 또는 / images 로 시작하는 URL이 포함된 모든 요청은 대체 정적 콘텐츠 서버 풀로 리디렉션됩니다.

사전 요구 사항

- *리스너(리스너1)및풀(pool1)이있는 HTTP 로드 밸런서(lb1). 자세한 내용은 상태 모니터를 사용하여 HTTP 로드 밸런서 생성을 참조하십시오.*

절차

1. *자격 증명 파일을 가져옵니다.*

   예제

   ```
   $ source ~/overcloudrc
   ```

2. *로드 밸런서(lb1)에 두 번째 풀(static_pool)을 만듭니다.*

참고

괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

```
$ openstack loadbalancer pool create --lb-algorithm ROUND_ROBIN --loadbalancer lb1 --
name static_pool --protocol HTTP
```

3. *프라이빗 서브넷(**private_subnet**)에 로드 밸런서 구성원 (**192.0.2.10** 및 **192.0.2.11**)을 풀 (**static_pool**)에 추가합니다.*

예제

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.10 --
protocol-port 80 static_pool
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.11 --
protocol-port 80 static_pool
```

4. *리스너(리스너**1**)에 **L7** 정책(**policy1**)을 만듭니다. 정책에는 작업(**REIRECT_TO_POOL**)이 포함되어야 하며 풀(**static_pool**)을 가리켜야 합니다.*

예제

```
$ openstack loadbalancer l7policy create --action REDIRECT_TO_POOL --redirect-pool
static_pool --name policy1 listener1
```

5. *정책의 요청 경로 시작 시 /**js** 를 찾는 **L7** 규칙을 추가합니다.*

예제

```
$ openstack loadbalancer l7rule create --compare-type STARTS_WITH --type PATH --value
/js policy1
```

6. *작업(**REDIRECT_TO_POOL**)을 사용하여 **L7** 정책(**정책2**)을 만들고 풀에서 가리키는리스너(**리스너**)를 추가합니다.*

예제

```
$ openstack loadbalancer l7policy create --action REDIRECT_TO_POOL --redirect-pool
static_pool --name policy2 listener1
```

7. *정책에 대한 요청 경로 시작 시 /**images** 를 검색하는 **L7** 규칙을 추가합니다.*

예제

```
$ openstack loadbalancer l7rule create --compare-type STARTS_WITH --type PATH --value
/images policy2
```

검증

1. ***openstack loadbalancer l7policy list** 명령을 실행하고 정책, **policy1** 및 **policy 2** 가 있는지 확인합니다.*

2. *openstack loadbalancer l7rule list <l7policy> 명령을 실행하고 각 정책에 대해 비교 유형 STARTS_WITH 가 있는 규칙이 있는지 확인합니다.*

예제

```
$ openstack loadbalancer l7rule list policy1
$ openstack loadbalancer l7rule list policy2
```

추가 리소스

- *LoadBalancer 풀은 명령줄 인터페이스 참조에 생성됩니다.*
- *LoadBalancer member create in the Command Line Interface Reference*
- *명령줄 인터페이스 참조에 LoadBalancer l7policy 생성*
- *명령줄 인터페이스 참조에 LoadBalancer l7rule 생성*

10.13. 특정 풀에 하위 도메인 요청 전송

7 계층(L7) 정책과 함께 RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 사용하여 특정 HTTP/1.1 호스트 이름을 포함하는 요청을 다른 애플리케이션 서버 풀로 리디렉션할 수 있습니다.

이 예제에서는 HTTP/1.1 호스트 이름이 포함된 모든 요청 대체 풀 애플리케이션 서버인 pool2 로 리디렉션됩니다.

사전 요구 사항

- *리스너(리스너1)및풀(pool1)이있는 HTTP 로드 밸런서(lb1). 자세한 내용은 상태 모니터를 사용하여 HTTP 로드 밸런서 생성을 참조하십시오.*

설자

1. *자격 증명 파일을 가져옵니다.*

   예제

   ```
   $ source ~/overcloudrc
   ```

2. *로드 밸런서(**lb1**)에 두 번째 풀(**pool2**)을 만듭니다.*

   참고

   괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

   예제

   ```
   $ openstack loadbalancer pool create --lb-algorithm ROUND_ROBIN --loadbalancer lb1 --
   name pool2 --protocol HTTP
   ```

3. *리스너(리스너**1**)에 **L7** 정책(**policy1**)을 만듭니다. 정책에는 작업(**REIRECT_TO_POOL**)이 포함되어야 하며 풀(**pool2**)을 가리켜야 합니다.*

   예제

   ```
   $ openstack loadbalancer l7policy create --action REDIRECT_TO_POOL --redirect-pool
   pool2 --name policy1 listener1
   ```

4. **HTTP/1.1** 호스트 이름 사용하여 요청을 두 번째 풀**(pool2)**에 보내는 정책에 **L7** 규칙을 추가합니다.

예제

```
$ openstack loadbalancer l7rule create --compare-type EQUAL_TO --type HOST_NAME --value www2.example.com policy1
```

검증

1. **openstack loadbalancer l7policy list** 명령을 실행하고 정책**, policy1** 이 있는지 확인합니다.

2. **openstack loadbalancer l7rule list <l7policy>** 명령을 실행하고 정책에 대해 **EQUAL _TO** 의 **compare_ type** 이 있는 규칙이 있는지 확인합니다.

예제

```
$ openstack loadbalancer l7rule list policy1
```

추가 리소스

- **LoadBalancer** 풀은 명령줄 인터페이스 참조에 생성됩니다.
- 명령줄 인터페이스 참조에 **LoadBalancer l7policy** 생성
- 명령줄 인터페이스 참조에 **LoadBalancer l7rule** 생성

10.14. 특정 풀로 끝나는 호스트 이름에 따라 요청 전송

7 계층(L7) 정책과 함께 RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 사용하여 특정 문자열에 끝나는 HTTP/1.1 호스트 이름이 포함된 요청을 다른 애플리케이션 서버 풀로 리디렉션할 수 있습니다.

이 예에서 .example.com 으로 끝나는 HTTP/1.1 호스트 이름이 포함된 모든 요청은 대체 풀 애플리케이션 서버인 pool2 로 리디렉션됩니다.

사전 요구 사항

- *리스너(리스너1)및풀(pool1)이있는 HTTP 로드 밸런서(lb1). 자세한 내용은 상태 모니터를 사용하여 HTTP 로드 밸런서 생성을참조하십시오.*

절차

1. *자격 증명 파일을 가져옵니다.*

예제

```
$ source ~/overcloudrc
```

2. *로드 밸런서(lb1)에 두 번째 풀(pool2)을 만듭니다.*

참고

괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

```
$ openstack loadbalancer pool create --lb-algorithm ROUND_ROBIN --loadbalancer lb1 --
name pool2 --protocol HTTP
```

3. *리스너(리스너**1**)에 **L7** 정책(**policy1**)을 만듭니다. 정책에는 작업(**REIRECT_TO_POOL**)이 포함되어야 하며 풀(**pool2**)을 가리켜야 합니다.*

예제

```
$ openstack loadbalancer l7policy create --action REDIRECT_TO_POOL --redirect-pool
pool2 --name policy1 listener1
```

4. ***HTTP/1.1** 호스트 이름(**www2.example.com**)을 사용하는 모든 요청을 두 번째 풀(**pool2**)으로 전송하는 정책에 **L7** 규칙을 추가합니다.*

예제

```
$ openstack loadbalancer l7rule create --compare-type ENDS_WITH --type HOST_NAME --
value .example.com policy1
```

검증

1. ***openstack loadbalancer l7policy list** 명령을 실행하고 정책, **policy1** 이 있는지 확인합니다.*

2. ***openstack loadbalancer l7rule list <l7policy>** 명령을 실행하고 정책에 대해 **EQUAL _TO** 의 **compare_ type** 이 있는 규칙이 있는지 확인합니다.*

예제

```
$ openstack loadbalancer l7rule list policy1
```

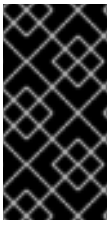
추가 리소스

- LoadBalancer 풀은 명령줄 인터페이스 참조에 생성됩니다.
- 명령줄 인터페이스 참조에 LoadBalancer l7policy 생성
- 명령줄 인터페이스 참조에 LoadBalancer l7rule 생성

10.15. 브라우저 쿠키가 없는 경우 특정 풀에 브라우저 쿠키가 없는 요청 전송

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 사용하여 인증되지 않은 웹 클라이언트 요청을 하나 이상의 인증 서버가 포함된 다른 풀로 리디렉션할 수 있습니다. 계층 7(L7) 정책은 들어오는 요청이 인증 쿠키가 누락되었는지 여부를 결정합니다.

이 예에서 브라우저 쿠키가 없는 모든 웹 클라이언트 요청 auth_token 은 인증 서버가 포함된 대체 풀로 리디렉션됩니다.



중요

다음 절차에서는 브라우저 쿠키를 사용하여 L7 애플리케이션 라우팅을 수행하는 방법에 대한 예를 제공하며 보안 문제를 다루지 않습니다.

사전 요구 사항

- 리스너(리스너1)및풀(pool1)이 있는TLS 종료 HTTPS 로드 밸런서(lb1). 자세한 내용은 TLS 종료 HTTPS 로드 밸런서 생성 을 참조하십시오.
- 보안 인증 서버에서 웹 사용자를 인증하는 두 번째 RHOSP 네트워킹 서비스(neutron) 서브넷입니다.

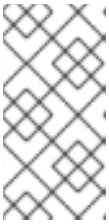
절차

1. 자격 증명 파일을 가져옵니다.

예제

```
$ source ~/overcloudrc
```

2. *로드 밸런서(**lb1**)에 두 번째 풀(**login_pool**)을 만듭니다.*



참고

괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

```
$ openstack loadbalancer pool create --lb-algorithm ROUND_ROBIN --loadbalancer lb1 --
name login_pool --protocol HTTP
```

3. *인증 서브넷(**secure_subnet**)에 멤버, 보안 인증 서버(**192.0.2.10**)를 두 번째 풀에 추가합니다.*

예제

```
$ openstack loadbalancer member create --address 192.0.2.10 --protocol-port 80 --subnet-id
secure_subnet login_pool
```

4. *리스너(리스너**1**)에 **L7** 정책(**policy1**)을 만듭니다. 정책에는 작업(**REIRECT_TO_POOL**)이 포함되어야 하며 두 번째 풀(**login_pool**)을 가리켜야 합니다.*

예제

```
$ openstack loadbalancer l7policy create --action REDIRECT_TO_POOL --redirect-pool
login_pool --name policy1 listener1
```

5. *브라우저 쿠키(**auth_token**)를 임의의 값으로 검색하는 정책(**policy1**)에 **L7** 규칙을 추가하고, 쿠키가 없는 경우 일치시킵니다.*

예제

```
$ openstack loadbalancer l7rule create --compare-type REGEX --key auth_token --type
COOKIE --value '.*' --invert policy1
```

검증

1. ***openstack loadbalancer l7policy list** 명령을 실행하고 정책, **policy1** 이 있는지 확인합니다.*

2. ***openstack loadbalancer l7rule list <l7policy>** 명령을 실행하고 **REGEX** 의 비교 유형과 규칙이 있는지 확인합니다.*

예제

```
$ openstack loadbalancer l7rule list policy1
```

추가 리소스

- ***LoadBalancer** 풀은 명령줄 인터페이스 참조에 생성됩니다.*

- *LoadBalancer member create in the Command Line Interface Reference*

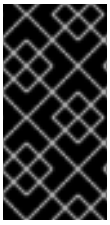
- *명령줄 인터페이스 참조에 LoadBalancer l7policy 생성*

- *명령줄 인터페이스 참조에 LoadBalancer l7rule 생성*

10.16. 브라우저 쿠키가 없거나 잘못된 쿠키 값을 특정 풀로 전송

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 사용하여 인증되지 않은 웹 클라이언트 요청을 하나 이상의 인증 서버가 포함된 다른 풀로 리디렉션할 수 있습니다. 계층 7(L7) 정책은 들어오는 요청이 인증 쿠키가 누락되었는지 또는 특정 값이 있는 인증 쿠키를 포함하는지 여부를 결정합니다.

이 예에서 웹 클라이언트는 브라우저 쿠키, auth_token 또는 값이 INVALID 인 auth_token 이 없는 경우 인증 서버가 포함된 대체 풀로 리디렉션됩니다.



중요

이 절차에서는 브라우저 쿠키를 사용하여 L7 애플리케이션 라우팅을 수행하는 방법에 대한 예를 제공하며 보안 문제를 해결하지 않습니다.

사전 요구 사항

- *리스너(리스너1)및풀(pool1)이 있는TLS 종료 HTTPS 로드 밸런서(lb1). 자세한 내용은 TLS 종료 HTTPS 로드 밸런서 생성 을 참조하십시오.*

- *보안 인증 서버에서 웹 사용자를 인증하는 두 번째 RHOSP 네트워킹 서비스(neutron) 서브넷입니다.*

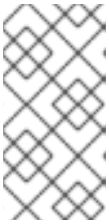
절차

1. *자격 증명 파일을 가져옵니다.*

예제

```
$ source ~/overcloudrc
```

2. *로드 밸런서(**lb1**)에 두 번째 풀(**login_pool**)을 만듭니다.*



참고

괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

```
$ openstack loadbalancer pool create --lb-algorithm ROUND_ROBIN --loadbalancer lb1 --
name login_pool --protocol HTTP
```

3. *인증 서브넷(**secure_subnet**)에 멤버, 보안 인증 서버(**192.0.2.10**)를 두 번째 풀에 추가합니다.*

예제

```
$ openstack loadbalancer member create --address 192.0.2.10 --protocol-port 80 --subnet-id
secure_subnet login_pool
```

4. *리스너(리스너**1**)에 **L7** 정책(**policy1**)을 만듭니다. 정책에는 작업(**REIRECT_TO_POOL**)이 포함되어야 하며 두 번째 풀(**login_pool**)을 가리켜야 합니다.*

예제

```
$ openstack loadbalancer l7policy create --action REDIRECT_TO_POOL --redirect-pool
login_pool --name policy1 listener1
```

5. *브라우저 쿠키(**auth_token**)를 임의의 값으로 검색하는 정책(**policy1**)에 **L7** 규칙을 추가하고, 쿠키가 없는 경우 일치시킵니다.*

예제

```
$ openstack loadbalancer l7rule create --compare-type REGEX --key auth_token --type
COOKIE --value '.*' --invert policy1
```

6. *리스너(리스너**1**)에 두 번째 **L7** 정책(**policy2**)을 만듭니다. 정책에는 작업 (**REIRECT_TO_POOL**)이 포함되어야 하며 두 번째 풀(**login_pool**)을 가리켜야 합니다.*

예제

```
$ openstack loadbalancer l7policy create --action REDIRECT_TO_POOL --redirect-pool
login_pool --name policy2 listener1
```

7. *브라우저 쿠키(**auth_token**)를 검색하고 쿠키 값이 **INVALID** 문자열과 일치하는지 일치하는 두 번째 정책(**policy2**)에 **L7** 규칙을 추가합니다.*

예제

```
$ openstack loadbalancer l7rule create --compare-type EQUAL_TO --key auth_token --type
COOKIE --value INVALID policy2
```

검증

1. **openstack loadbalancer l7policy list** 명령을 실행하고 정책, **policy1** 및 **policy 2** 가 있는지 확인합니다.

2. **openstack loadbalancer l7rule list <l7policy>** 명령을 실행하고 **policy1**에 비교_유형 이 있는 규칙과 **policy 2** 에 대해 **EQUAL _TO** 의 비교 유형이 있는지 확인합니다.

예제

```
$ openstack loadbalancer l7rule list policy1
$ openstack loadbalancer l7rule list policy2
```

추가 리소스

- **LoadBalancer** 풀은 명령줄 인터페이스 참조에 생성됩니다.
- **LoadBalancer member create in the Command Line Interface Reference**
- 명령줄 인터페이스 참조에 **LoadBalancer l7policy** 생성
- 명령줄 인터페이스 참조에 **LoadBalancer l7rule** 생성

10.17. 호스트 이름 및 경로와 일치하는 풀에 요청 전송

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스**(octavia)**를 사용하여 특정 기준과 일치하는 웹 클라이언트 요청을 대체 애플리케이션 서버 풀로 리디렉션할 수 있습니다. 비즈니스 로직 기준은 사전 정의된 호스트 이름 및 요청 경로와 일치하려는 계층 **7(L7)** 정책을 통해 수행됩니다.

이 예에서 호스트 이름 **api.example.com** 과 일치하는 웹 클라이언트 요청에는 요청 경로 시작 시 **/api**

*가 대체 풀인 **api_pool** 로 리디렉션됩니다.*

사전 요구 사항

- *리스너(**리스너1**)및풀(**pool1**)이있는 **HTTP** 로드 밸런서(**lb1**). 자세한 내용은 상태 모니터를 사용하여 **HTTP** 로드 밸런서 생성을 참조하십시오.*

절차

1. *자격 증명 파일을 가져옵니다.*

   예제

   ```
   $ source ~/overcloudrc
   ```

2. *로드 밸런서(**lb1**)에 두 번째 풀(**api_pool**)을 만듭니다.*

   참고

   괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

   예제

   ```
   $ openstack loadbalancer pool create --lb-algorithm ROUND_ROBIN --loadbalancer lb1 --
   name api_pool --protocol HTTP
   ```

3. *프라이빗 서브넷(**private_subnet**)에 로드 밸런서 구성원 (**192.0.2.10** 및 **192.0.2.11**)을 풀 (**static_pool**)에 추가합니다.*

예제

```
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.10 --
protocol-port 80 static_pool
$ openstack loadbalancer member create --subnet-id private_subnet --address 192.0.2.11 --
protocol-port 80 static_pool
```

4. *리스너(리스너1)에 **L7** 정책(**policy1**)을 만듭니다. 정책에는 작업(**REIRECT_TO_POOL**)이 포함되어야 하며 풀(**api_pool**)을 가리켜야 합니다.*

예제

```
$ openstack loadbalancer l7policy create --action REDIRECT_TO_POOL --redirect-pool
api_pool --name policy1 listener1
```

5. *호스트 이름 **api.example.com** 과 일치하는 정책에 **L7** 규칙을 추가합니다.*

예제

```
$ openstack loadbalancer l7rule create --compare-type EQUAL_TO --type HOST_NAME --
value api.example.com policy1
```

6. *두 번째 **L7** 규칙을 요청 경로 시작 시 **/api** 와 일치하는 정책에 추가합니다.*

이 규칙은 첫 번째 규칙에 논리적으로 적용됩니다.

예제

```
$ openstack loadbalancer l7rule create --compare-type STARTS_WITH --type PATH --value
/api policy1
```

검증

1. ***openstack loadbalancer l7policy list** 명령을 실행하고 정책, **policy1** 이 있는지 확인합니다.*

2. ***openstack loadbalancer l7rule list <l7policy>** 명령을 실행하고 각각 **EQUAL_TO** 및 **STARTS _WITH** 의 비교_ 유형 과 규칙이 있는지 확인합니다. 둘 다 **policy1** 에 있습니다.*

   예제

```
$ openstack loadbalancer l7rule list policy1
$ openstack loadbalancer l7rule list policy2
```

추가 리소스

- ***LoadBalancer** 풀은 명령줄 인터페이스 참조에 생성됩니다.*
- ***LoadBalancer member create** in the Command Line Interface Reference*
- *명령줄 인터페이스 참조에 **LoadBalancer** **l7policy** 생성*
- *명령줄 인터페이스 참조에 **LoadBalancer** **l7rule** 생성*

10.18. 쿠키를 사용하여 기존 프로덕션 사이트에서 A-B 테스트 구성

7 계층(L7) 정책이 포함된 RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 사

*용하여 프로덕션 웹 사이트에 **A-B** 테스트를 구성하거나 분할 테스트를 수행할 수 있습니다.*

*이 예에서 **"B"** 버전으로 라우팅되는 웹 클라이언트는 사이트 사이트_**version** 을 풀의 멤버 서버에 의해 **(pool1)**에 의해 **B** 로 설정합니다.*

사전 요구 사항

- *두 개의 생산 웹사이트(사이트 **A** 및 사이트 **B**).*
- *"풀의 시작 경로에 따라 요청 재지정"에 대한 지침에 따라 **HTTP** 로드 밸런서를 구성했습니다. 필수 구성의 요약은 다음과 같습니다.*
  - *로드 밸런서**(lb1)**의 리스너(리스너**1**).*
  - */**js** 또는 / **images** 로 시작하는 **URL**이 있는 **HTTP** 요청은 풀**(static_pool)**으로 전송됩니다.*
  - *기타 모든 요청은 리스너 기본 풀**(pool1)**으로 전송됩니다.*
  - *구성에 대한 자세한 내용은 **10.12**절. “풀에 대한 시작 경로를 기반으로 요청 리디렉션”을 참조하십시오.*

절차

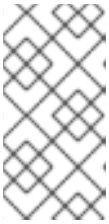
1. *자격 증명 파일을 가져옵니다.*

   예제

   ```
   $ source ~/overcloudrc
   ```

2.

*로드 밸런서(**lb1**)에서 세 번째 풀(**pool_B**)을 만듭니다.*



참고

괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

```
$ openstack loadbalancer pool create --lb-algorithm ROUND_ROBIN --loadbalancer lb1 --
name pool_B --protocol HTTP
```

3.

*프라이빗 서브넷(**private_subnet**)에 로드 밸런서 구성원(**192.0. 2.50**및 **192.0.2.51**)을 풀(**pool_B**)에 추가합니다.*

예제

```
$ openstack loadbalancer member create --address 192.0.2.50 --protocol-port 80 --subnet-id
private_subnet pool_B
$ openstack loadbalancer member create --address 192.0.2.51 --protocol-port 80 --subnet-id
private_subnet pool_B
```

4.

*로드 밸런서(**lb1**)에서 네 번째 풀(**static_pool_B**)을 만듭니다.*

예제

```
$ openstack loadbalancer pool create --lb-algorithm ROUND_ROBIN --loadbalancer lb1 --
name static_pool_B --protocol HTTP
```

5. *프라이빗 서브넷(**private_subnet**)에 로드 밸런서 구성원 (**192.0.2.100** 및 **192.0.2.101**)을 풀 (**static_pool_B**)에 추가합니다.*

예제

```
$ openstack loadbalancer member create --address 192.0.2.100 --protocol-port 80 --subnet-id private_subnet static_pool_B
$ openstack loadbalancer member create --address 192.0.2.101 --protocol-port 80 --subnet-id private_subnet static_pool_B
```

6. *리스너(리스너**1**)에 **L7** 정책(**policy2**)을 만듭니다. 정책에는 작업(**REIRECT_TO_POOL**)이 포함되어야 하며 풀(**static_pool_B**)을 가리켜야 합니다. 위치에 정책을 삽입합니다 . **1.***

예제

```
$ openstack loadbalancer l7policy create --action REDIRECT_TO_POOL --redirect-pool static_pool_B --name policy2 --position 1 listener1
```

7. *정규 표현식을 사용하여 요청 경로 시작시 /**js** 또는 / **images** 와 일치하는 정책(**policy2**)에 **L7** 규칙을 추가합니다.*

예제

```
$ openstack loadbalancer l7rule create --compare-type REGEX --type PATH --value '^/(js|images)' policy2
```

8.

*쿠키(**site_version**)와 일치하는 두 번째 **L7** 규칙을 정확한 문자열(**B**)에 추가합니다.*

예제

```
$ openstack loadbalancer l7rule create --compare-type EQUAL_TO --key site_version --type
COOKIE --value B policy2
```

9.

*리스너(리스너**1**)에 **L7** 정책(**policy3**)을 만듭니다. 정책에는 작업(**REIRECT_TO_POOL**)이 포함되어야 하며 풀(**pool_B**)을 가리켜야 합니다. 위치에 정책을 삽입합니다 **. 2.***

예제

```
$ openstack loadbalancer l7policy create --action REDIRECT_TO_POOL --redirect-pool
pool_B --name policy3 --position 2 listener1
```

10.

*쿠키(**site_version**)와 일치하는 정책(**policy3**)에 **L7** 규칙을 정확한 문자열(**B**)에 추가합니다.*

예제

```
$ openstack loadbalancer l7rule create --compare-type EQUAL_TO --key site_version --type
COOKIE --value B policy3
```

참고

*규칙이 **True**로 평가되는 첫 번째 정책은 작업이 따르는 정책이므로 **L7** 정책을 가장 낮은 위치에 할당하는 것이 중요합니다. 이 절차에서는 요청이 잘못된 풀로 전송되지 않도록 **policy3** 전에 **policy 2** 를 평가해야 합니다.*

검증

1. ***openstack loadbalancer l7policy list** 명령을 실행하고 정책, **policy2** 및 **policy 3** 이 있는지 확인합니다.*

2. ***openstack loadbalancer l7rule list <l7policy>** 명령을 실행하고 각 정책에 대해 비교 유형 **STARTS_WITH** 가 있는 규칙이 있는지 확인합니다.*

   예제

```
$ openstack loadbalancer l7rule list policy2
$ openstack loadbalancer l7rule list policy3
```

추가 리소스

- ***LoadBalancer** 풀은 명령줄 인터페이스 참조에 생성됩니다.*
- ***LoadBalancer member create** in the Command Line Interface Reference*
- *명령줄 인터페이스 참조에 **LoadBalancer** **l7policy** 생성*
- *명령줄 인터페이스 참조에 **LoadBalancer** **l7rule** 생성*

11장. 로드 밸런싱 서비스 업데이트 및 업그레이드

최신 **Red Hat OpenStack Platform** 로드 밸런싱 서비스 기능을 사용할 수 있도록 정기적인 업데이트 및 업그레이드를 수행하고 자주 업데이트 및 업그레이드로 인해 발생할 수 있는 길고 문제가 발생하는 문제를 방지합니다.

- 11.1절. “로드 밸런싱 서비스 업데이트 및 업그레이드”
- 11.2절. “실행 중인 로드 밸런싱 서비스 인스턴스 업데이트”

11.1. 로드 밸런싱 서비스 업데이트 및 업그레이드

로드 밸런싱 서비스(**octavia**)는 **RHOSP(Red Hat OpenStack Platform)** 업데이트 또는 업그레이드의 일부입니다.

사전 요구 사항

- 로드 밸런싱 서비스 컨트롤 플레인을 업그레이드하는 동안 완전히 작동하지 않기 때문에 업그레이드를 수행하도록 유지보수 기간을 예약합니다.

절차

1. **Red Hat OpenStack Platform** 업데이트 가이드에 설명된 대로 **RHOSP** 업데이트를 수행합니다.
2. 유지 관리 릴리스가 적용된 후 새 기능을 사용해야 하는 경우 **amphorae** 실행을 교체하여 최신 **amphora** 이미지로 업데이트합니다.

추가 리소스

- **Red Hat OpenStack Platform** 업데이트 가이드 유지
- 11.2절. “실행 중인 로드 밸런싱 서비스 인스턴스 업데이트”
- 2.2절. “로드 밸런싱 서비스(octavia) 기능 지원 매트릭스”

11.2. 실행 중인 로드 밸런싱 서비스 인스턴스 업데이트

주기적으로 실행 중인 로드 밸런싱 서비스 인스턴스(amphora)를 최신 이미지로 업데이트할 수 있습니다. 예를 들어 다음 이벤트 중에 amphora 인스턴스를 업데이트할 수 있습니다.

- RHOSP(Red Hat OpenStack Platform) 업데이트 또는 업그레이드.
- 시스템에 대한 보안 업데이트.
- 기본 가상 시스템의 다른 플레이버로의 변경.

RHOSP 업데이트 또는 업그레이드 중에 director는 기본 amphora 이미지를 자동으로 다운로드하여 overcloud Image 서비스(glance)에 업로드한 다음 새 이미지를 사용하도록 로드 밸런싱 서비스(octavia)를 구성합니다. 로드 밸런서가 실패하면 로드 밸런싱 서비스에서 새 amphora 이미지를 사용하는 인스턴스(amphora)를 시작하도록 합니다.

사전 요구 사항

- Amphora의 새 이미지. RHOSP 업데이트 또는 업그레이드 중에 사용할 수 있습니다.

절차

1. 자격 증명 파일을 가져옵니다.

   예제

   ```
   $ source ~/overcloudrc
   ```

2. 업데이트할 모든 로드 밸런서의 ID를 나열합니다.

   ```
   $ openstack loadbalancer list -c id -f value
   ```

3.

각 로드 밸런서에서 실패합니다.

```
$ openstack loadbalancer failover <loadbalancer_id>
```

참고

*로드 밸런서에 장애가 발생하면 시스템 사용률을 모니터링하고 필요에 따라 장애 조치를 수행하는 비율을 조정합니다. 로드 밸런서 장애 조치를 통해 새 가상 시스템 및 포트를 생성하여 **OpenStack Networking**의 로드를 일시적으로 늘릴 수 있습니다.*

4.

로드 밸런서를 통해 실패한 상태를 모니터링합니다.

```
$ openstack loadbalancer show <loadbalancer_id>
```

*로드 밸런서 상태가 **ACTIVE** 이면 업데이트가 완료됩니다.*

추가 리소스

- *명령줄 인터페이스 참조 의 **LoadBalancer***

12장. 로드 밸런싱 서비스 문제 해결 및 유지 관리

로드 밸런싱 서비스(octavia)의 기본 문제 해결 및 유지 관리는 상태 및 인스턴스 마이그레이션 및 로그 액세스 방법을 확인하기 위한 OpenStack 클라이언트 명령에 익숙하기 시작합니다. 더 자세한 문제를 해결해야 하는 경우 하나 이상의 로드 밸런싱 서비스 인스턴스(amphorae)에 SSH를 적용할 수 있습니다.

12.1. 로드 밸런서 확인

로드 밸런서 표시 및 나열 명령의 출력을 확인하여 로드 밸런싱 서비스(octavia) 및 다양한 구성 요소의 문제를 해결할 수 있습니다.

절차

1. 자격 증명 파일을 가져옵니다.

   예제

   ```
   $ source ~/overcloudrc
   ```

2. 로드 밸런서(lb1) 설정을 확인합니다.

참고

괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

```
$ openstack loadbalancer show lb1
```

샘플 출력

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| admin_state_up      | True                                 |
| created_at          | 2022-02-17T15:59:18                  |
| description         |                                      |
| flavor_id           | None                                 |
| id                  | 265d0b71-c073-40f4-9718-8a182c6d53ca |
| listeners           | 5aaa67da-350d-4125-9022-238e0f7b7f6f |
| name                | lb1                                  |
| operating_status    | ONLINE                               |
| pools               | 48f6664c-b192-4763-846a-da568354da4a |
| project_id          | 52376c9c5c2e434283266ae7cacd3a9c     |
| provider            | amphora                              |
| provisioning_status | ACTIVE                               |
| updated_at          | 2022-02-17T16:01:21                  |
| vip_address         | 192.0.2.177                          |
| vip_network_id      | afeaf55e-7128-4dff-80e2-98f8d1f2f44c |
| vip_port_id         | 94a12275-1505-4cdc-80c9-4432767a980f |
| vip_qos_policy_id   | None                                 |
| vip_subnet_id       | 06ffa90e-2b86-4fe3-9731-c7839b0be6de |
+---------------------+--------------------------------------+
```

3. *이전 단계의 로드 밸런서 **ID(265d0b71-c073-40f4-9718-8a182d53ca)**를 사용하여 로드 밸런서와 연결된 **amphora ID**를 가져옵니다.*

예제

```
$ openstack loadbalancer amphora list | grep 265d0b71-c073-40f4-9718-8a182c6d53ca
```

샘플 출력

```
| 1afabefd-ba09-49e1-8c39-41770aa25070 | 265d0b71-c073-40f4-9718-8a182c6d53ca |
ALLOCATED | STANDALONE | 198.51.100.7  | 192.0.2.177   |
```

4. *이전 단계의* ***amphora ID (1afabefd-ba09-49e1-8c39-41770aa25070)****를 사용하여* ***amphora*** *정보를 볼 수 있습니다.*

예제

```
$ openstack loadbalancer amphora show 1afabefd-ba09-49e1-8c39-41770aa25070
```

샘플 출력

```
+-----------------+--------------------------------------+
| Field           | Value                                |
+-----------------+--------------------------------------+
| id              | 1afabefd-ba09-49e1-8c39-41770aa25070 |
| loadbalancer_id | 265d0b71-c073-40f4-9718-8a182c6d53ca |
| compute_id      | ba9fc1c4-8aee-47ad-b47f-98f12ea7b200 |
| lb_network_ip   | 198.51.100.7                         |
| vrrp_ip         | 192.0.2.36                           |
| ha_ip           | 192.0.2.177                          |
| vrrp_port_id    | 07dcd894-487a-48dc-b0ec-7324fe5d2082 |
| ha_port_id      | 94a12275-1505-4cdc-80c9-4432767a980f |
| cert_expiration | 2022-03-19T15:59:23                  |
| cert_busy       | False                                |
| role            | STANDALONE                           |
```

```
| status          | ALLOCATED                            |
| vrrp_interface  | None                                 |
| vrrp_id         | 1                                    |
| vrrp_priority   | None                                 |
| cached_zone     | nova                                 |
| created_at      | 2022-02-17T15:59:22                  |
| updated_at      | 2022-02-17T16:00:50                  |
| image_id        | 53001253-5005-4891-bb61-8784ae85e962 |
| compute_flavor  | 65                                   |
+-----------------+--------------------------------------+
```

5. *리스너(**listener1**) 세부 정보를 봅니다.*

예제

```
$ openstack loadbalancer listener show listener1
```

샘플 출력

```
+-----------------------------+--------------------------------------+
| Field                       | Value                                |
+-----------------------------+--------------------------------------+
| admin_state_up              | True                                 |
| connection_limit            | -1                                   |
| created_at                  | 2022-02-17T16:00:59                  |
| default_pool_id             | 48f6664c-b192-4763-846a-da568354da4a |
| default_tls_container_ref   | None                                 |
| description                 |                                      |
| id                          | 5aaa67da-350d-4125-9022-238e0f7b7f6f |
| insert_headers              | None                                 |
| l7policies                  |                                      |
| loadbalancers               | 265d0b71-c073-40f4-9718-8a182c6d53ca |
| name                        | listener1                            |
| operating_status            | ONLINE                               |
| project_id                  | 52376c9c5c2e434283266ae7cacd3a9c     |
| protocol                    | HTTP                                 |
| protocol_port               | 80                                   |
| provisioning_status         | ACTIVE                               |
| sni_container_refs          | []                                   |
| timeout_client_data         | 50000                                |
```

```
| timeout_member_connect      | 5000                                 |
| timeout_member_data         | 50000                                |
| timeout_tcp_inspect         | 0                                    |
| updated_at                  | 2022-02-17T16:01:21                  |
| client_ca_tls_container_ref | None                                 |
| client_authentication       | NONE                                 |
| client_crl_container_ref    | None                                 |
| allowed_cidrs               | None                                 |
+-----------------------------+--------------------------------------+
```

6. ***풀(pool1) 및 로드 밸런서 멤버를 확인합니다.***

예제

```
$ openstack loadbalancer pool show pool1
```

샘플 출력

```
+----------------------+--------------------------------------+
| Field                | Value                                |
+----------------------+--------------------------------------+
| admin_state_up       | True                                 |
| created_at           | 2022-02-17T16:01:08                  |
| description          |                                      |
| healthmonitor_id     | 4b24180f-74c7-47d2-b0a2-4783ada9a4f0 |
| id                   | 48f6664c-b192-4763-846a-da568354da4a |
| lb_algorithm         | ROUND_ROBIN                          |
| listeners            | 5aaa67da-350d-4125-9022-238e0f7b7f6f |
| loadbalancers        | 265d0b71-c073-40f4-9718-8a182c6d53ca |
| members              | b92694bd-3407-461a-92f2-90fb2c4aedd1 |
|                      | 4ccdd1cf-736d-4b31-b67c-81d5f49e528d |
| name                 | pool1                                |
| operating_status     | ONLINE                               |
| project_id           | 52376c9c5c2e434283266ae7cacd3a9c     |
| protocol             | HTTP                                 |
| provisioning_status  | ACTIVE                               |
| session_persistence  | None                                 |
| updated_at           | 2022-02-17T16:01:21                  |
| tls_container_ref    | None                                 |
| ca_tls_container_ref | None                                 |
```

```
| crl_container_ref    | None                                 |
| tls_enabled          | False                                |
+----------------------+--------------------------------------+
```

7. *로드 밸런서의 **VIP** 주소(**192.0.2.177**)에 연결하여 **HTTPS** 트래픽이 **HTTPS** 또는 **TERMINATED_HTTPS** 프로토콜용으로 구성된 로드 밸런서 전체에서 **HTTPS** 트래픽이 이동하는지 확인합니다.*

작은 정보

*명령을 사용하여 로드 밸런서 **VIP** 주소를 가져옵니다. **openstack loadbalancer show <load_balancer_name>.***

예제

```
$ curl -v https://192.0.2.177 --insecure
```

샘플 출력

```
* About to connect() to 192.0.2.177 port 443 (#0)
*   Trying 192.0.2.177...
* Connected to 192.0.2.177 (192.0.2.177) port 443 (#0)
* Initializing NSS with certpath: sql:/etc/pki/nssdb
* skipping SSL peer certificate verification
* SSL connection using TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
* Server certificate:
*  subject: CN=www.example.com,O=Dis,L=Springfield,ST=Denial,C=US
*  start date: Jan 15 09:21:45 2021 GMT
*  expire date: Jan 15 09:21:45 2021 GMT
*  common name: www.example.com
*  issuer: CN=www.example.com,O=Dis,L=Springfield,ST=Denial,C=US
> GET / HTTP/1.1
> User-Agent: curl/7.29.0
> Host: 192.0.2.177
> Accept: */*
>
< HTTP/1.1 200 OK
```

```
< Content-Length: 30
<
* Connection #0 to host 192.0.2.177 left intact
```

추가 리소스

- 명령줄 인터페이스 참조 의 **LoadBalancer**

12.2. 특정 로드 밸런싱 서비스 인스턴스 마이그레이션

경우에 따라 로드 밸런싱 서비스 인스턴스(**amphora**)를 마이그레이션해야 합니다. 예를 들어, 유지 관리를 위해 호스트가 종료되는 경우

절차

1. 자격 증명 파일을 가져옵니다.

   예제

   ```
   $ source ~/overcloudrc
   ```

2. 마이그레이션할 **amphora**의 **ID**를 찾습니다. 이후 단계에서 **ID**를 제공해야 합니다.

   ```
   $ openstack loadbalancer amphora list
   ```

3. **Compute** 스케줄러 서비스에서 새 **Amphorae**를 비우고 **Compute** 노드에 예약하지 않도록 **Compute** 노드(**compute-host-1**)를 비활성화합니다.

참고

괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

```
$ openstack compute service set compute-host-1 nova-compute --disable
```

4. 가져온 **amphora ID (ea17210a-1076-48ff-8a1f-ced49ccb5e53)**를 사용하여 **amphora**를 통해 실패합니다.

예제

```
$ openstack loadbalancer amphora failover ea17210a-1076-48ff-8a1f-ced49ccb5e53
```

추가 리소스

- 명령줄 인터페이스 참조에 설정된 계산 서비스
- 명령줄 인터페이스 참조 의 **LoadBalancer**

12.3. SSH를 사용하여 인스턴스 로드 밸런싱에 연결

서비스 문제를 해결할 때 **SSH**를 사용하여 로드 밸런싱 서비스 인스턴스**(amphorae)**에 로그인합니다.

SSH(Secure Shell)를 사용하여 서비스 문제를 해결할 때 로드 밸런싱 서비스 인스턴스**(amphorae)** 실행에 로그인하는 것이 유용할 수 있습니다.

사전 요구 사항

- *로드 밸런싱 서비스(**octavia**) **SSH** 개인 키가 있어야 합니다.*
- *로드 밸런서를 생성하기 전에 로드 밸런싱 서비스 구성에서 **SSH**를 활성화해야 합니다.*

절차

1. ***director** 노드에서 **ssh-agent** 를 시작하고 사용자 **ID** 키를 에이전트에 추가합니다.*

```
$ eval $(ssh-agent -s)
$ ssh-add
```

2. *자격 증명 파일을 가져옵니다.*

   예제

```
$ source ~/overcloudrc
```

3. *연결할 **amphora**의 로드 밸런싱 관리 네트워크(**lb_network_ip**)에서 **IP** 주소를 결정합니다.*

```
$ openstack loadbalancer amphora list
```

4. ***SSH**를 사용하여 **amphora**에 연결합니다.*

```
$ ssh -A -t heat-admin@<controller_node_IP_address> ssh cloud-user@<lb_network_ip>
```

5. *완료되면 **amphora**에 대한 연결을 닫고 **SSH** 에이전트를 중지합니다.*

```
$ exit
```

추가 리소스

- *명령줄 인터페이스 참조 의 **LoadBalancer***

12.4. 리스너 통계 표시

***OpenStack Client**를 사용하면 특정 **RHOSP(Red Hat OpenStack Platform)** 로드 밸런서의 리스너에 대한 통계를 가져올 수 있습니다.*

- *현재 활성 연결**(active_connections).***
- *수신한 총 바이트**(bytes_in).***
- *총 바이트(바이트_**out**).*
- *이행할 수 없는 총 요청**(request_errors).***
- *처리된 총 연결**(total_connections).***

절차

1. *자격 증명 파일을 가져옵니다.*

   예제

   ```
   $ source ~/overcloudrc
   ```

2. *리스너 통계**(listener1)**를 확인합니다.*

참고

괄호 안의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

```
$ openstack loadbalancer listener stats show listener1
```

작은 정보

리스너 이름을 모르는 경우 **command loadbalancer** 리스너 목록을 입력합니다.

샘플 출력

```
+--------------------+-------+
| Field              | Value |
+--------------------+-------+
| active_connections | 0     |
| bytes_in           | 0     |
| bytes_out          | 0     |
| request_errors     | 0     |
| total_connections  | 0     |
+--------------------+-------+
```

추가 리소스

- **LoadBalancer** 리스너 통계가 명령줄 인터페이스 참조에 표시됩니다.
- 12.5절. “리스너 요청 오류 해석”

12.5. 리스너 요청 오류 해석

특정 RHOSP(Red Hat OpenStack Platform) 로드 밸런서의 리스너에 대한 통계를 가져올 수 있습니다. 자세한 내용은 12.4절. “리스너 통계 표시”의 내용을 참조하십시오.

RHOSP 로드 밸런서에서 추적한 통계 중 request_errors 는 최종 사용자가 로드 밸런서에 연결하는 요청에 발생한 오류만 계산하는 것입니다. request_errors 변수는 멤버 서버에서 보고한 오류를 측정하지 않습니다.

예를 들어 테넌트가 RHOSP 로드 밸런싱 서비스(octavia)를 통해 400(Bad Request) 의 HTTP 상태 코드를 반환하는 웹 서버에 연결하는 경우 로드 밸런싱 서비스에 의해 이 오류가 수집되지 않습니다. 로드 밸런서는 데이터 트래픽의 콘텐츠를 검사하지 않습니다. 이 예에서 로드 밸런서는 사용자와 웹 서버 간에 정보를 올바르게 전송했기 때문에 이 흐름을 성공한 것으로 해석합니다.

다음 조건에 따라 request_errors 변수가 증가할 수 있습니다.

- 요청이 전송되기 전에 클라이언트에서 조기 종료.
- 클라이언트에서 오류를 읽습니다.
- 클라이언트 시간 제한.
- 클라이언트가 연결을 종료합니다.
- 클라이언트의 다양한 잘못된 요청.

추가 리소스

- LoadBalancer 리스너 통계가 명령줄 인터페이스 참조에 표시됩니다.
- 12.4절. “리스너 통계 표시”