



# Red Hat OpenStack Platform 17.1

## 로드 밸런싱을 서비스로 구성

로드 밸런싱 서비스(octavia)를 사용하여 데이터 플레인에서 네트워크 트래픽 관리



# Red Hat OpenStack Platform 17.1 로드 밸런싱을 서비스로 구성

---

로드 밸런싱 서비스(octavia)를 사용하여 데이터 플레인에서 네트워크 트래픽 관리

OpenStack Team  
rhos-docs@redhat.com

## 법적 공지

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux<sup>®</sup> is the registered trademark of Linus Torvalds in the United States and other countries.

Java<sup>®</sup> is a registered trademark of Oracle and/or its affiliates.

XFS<sup>®</sup> is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL<sup>®</sup> is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js<sup>®</sup> is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack<sup>®</sup> Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

## 초록

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

## 차례

|                                                       |           |
|-------------------------------------------------------|-----------|
| 보다 포괄적 수용을 위한 오픈 소스 용어 교체 .....                       | 5         |
| RED HAT 문서에 관한 피드백 제공 .....                           | 6         |
| <b>1장. 로드 밸런싱 서비스 소개 .....</b>                        | <b>7</b>  |
| 1.1. 서비스 구성 요소 로드 밸런싱 .....                           | 7         |
| 1.2. 서비스 오브젝트 모델 로드 밸런싱 .....                         | 8         |
| 1.3. RED HAT OPENSTACK PLATFORM의 로드 밸런싱 사용 .....      | 10        |
| <b>2장. 로드 밸런싱 서비스 구현을 위한 고려 사항 .....</b>              | <b>11</b> |
| 2.1. 서비스 공급자 드라이버 로드 밸런싱 .....                        | 11        |
| 2.2. 로드 밸런싱 서비스(OCTAVIA) 기능 지원 매트릭스 .....             | 11        |
| 2.3. 서비스 소프트웨어 요구 사항 로드 밸런싱 .....                     | 13        |
| 2.4. 언더클라우드의 로드 밸런싱 서비스 사전 요구 사항 .....                | 13        |
| 2.5. 로드 밸런싱 서비스 인스턴스에 대한 활성-STANDBY 토폴로지의 기본 사항 ..... | 14        |
| 2.6. LOAD-BALANCING SERVICE에 대한 배포 후 단계 .....         | 14        |
| <b>3장. 로드 밸런싱 서비스 보안 .....</b>                        | <b>16</b> |
| 3.1. 로드 밸런싱 서비스의 양방향 TLS 인증 .....                     | 16        |
| 3.2. 로드 밸런싱 서비스의 인증서 라이프사이클 .....                     | 16        |
| 3.3. 서비스 인증서 및 키 로드 밸런싱 구성 .....                      | 16        |
| <b>4장. 로드 밸런싱 서비스 설치 및 구성 .....</b>                   | <b>20</b> |
| 4.1. 로드 밸런싱 서비스 배포 .....                              | 20        |
| 4.2. 로드 밸런싱 서비스 인스턴스에 대한 활성-STANDBY 토폴로지 활성화 .....    | 20        |
| 4.3. 로드 밸런싱 서비스 기본 설정 변경 .....                        | 22        |
| <b>5장. 로드 밸런싱 서비스 인스턴스 로그 관리 .....</b>                | <b>24</b> |
| 5.1. 로드 밸런싱 서비스 인스턴스 관리 로그 오프로드 활성화 .....             | 24        |
| 5.2. 로드 밸런싱 서비스 인스턴스에 대한 테넌트 흐름 로그 오프로드 활성화 .....     | 26        |
| 5.3. 로드 밸런싱 서비스 인스턴스 테넌트 흐름 로깅 비활성화 .....             | 28        |
| 5.4. 로드 밸런싱 서비스 인스턴스 로컬 로그 스토리지 비활성화 .....            | 29        |
| 5.5. 로드 밸런싱 서비스 인스턴스 로깅을 위한 HEAT 매개변수 .....           | 30        |
| 5.6. 서비스 인스턴스 테넌트 흐름 로그 형식 로드 밸런싱 .....               | 32        |
| <b>6장. 로드 밸런싱 서비스 플레이버 구성 .....</b>                   | <b>35</b> |
| 6.1. 로드 밸런싱 서비스 공급자 기능 나열 .....                       | 35        |
| 6.2. 플레이버 프로필 정의 .....                                | 36        |
| 6.3. 로드 밸런싱 서비스 플레이버 생성 .....                         | 37        |
| <b>7장. 로드 밸런싱 서비스 모니터링 .....</b>                      | <b>39</b> |
| 7.1. 로드 밸런싱 관리 네트워크 .....                             | 39        |
| 7.2. 서비스 인스턴스 로드 밸런싱 모니터링 .....                       | 40        |
| 7.3. 서비스 풀 멤버 모니터링 로드 밸런싱 .....                       | 40        |
| 7.4. 로드 밸런서 프로비저닝 상태 모니터링 .....                       | 40        |
| 7.5. 로드 밸런서 기능 모니터링 .....                             | 40        |
| 7.6. 로드 밸런싱 서비스 상태 모니터 정보 .....                       | 40        |
| 7.7. 로드 밸런싱 서비스 상태 모니터 생성 .....                       | 41        |
| 7.8. 로드 밸런싱 서비스 상태 모니터 수정 .....                       | 42        |
| 7.9. 로드 밸런싱 서비스 상태 모니터 삭제 .....                       | 43        |
| 7.10. 서비스 HTTP 상태 모니터 로드 밸런싱 모범 사례 .....              | 44        |
| <b>8장. 비보안 HTTP 로드 밸런서 생성 .....</b>                   | <b>45</b> |
| 8.1. 상태 모니터를 사용하여 HTTP 로드 밸런서 생성 .....                | 45        |

|                                                   |            |
|---------------------------------------------------|------------|
| 8.2. 부동 IP를 사용하는 HTTP 로드 밸런서 생성                   | 48         |
| 8.3. 세션 지속성을 사용하여 HTTP 로드 밸런서 생성                  | 53         |
| <b>9장. 보안 HTTP 로드 밸런서 생성</b>                      | <b>58</b>  |
| 9.1. 종료되지 않은 HTTPS 로드 밸런서 정보                      | 58         |
| 9.2. 종료되지 않은 HTTPS 로드 밸런서 생성                      | 58         |
| 9.3. TLS 종료 HTTPS 로드 밸런서 정보                       | 62         |
| 9.4. TLS 종료 HTTPS 로드 밸런서 생성                       | 63         |
| 9.5. SNI를 사용하여 TLS 종료 HTTPS 로드 밸런서 생성             | 68         |
| 9.6. HTTP/2 리스너를 사용하여 TLS 종료 로드 밸런서 생성            | 74         |
| 9.7. 동일한 IP 및 백엔드에서 HTTP 및 TLS 종료 HTTPS 로드 밸런싱 생성 | 80         |
| <b>10장. 다른 종류의 로드 밸런서 생성</b>                      | <b>86</b>  |
| 10.1. TCP 로드 밸런서 생성                               | 86         |
| 10.2. 상태 모니터를 사용하여 UDP 로드 밸런서 생성                  | 90         |
| 10.3. QOS 규칙 로드 밸런서 생성                            | 94         |
| 10.4. 액세스 제어 목록을 사용하여 로드 밸런서 생성                   | 98         |
| 10.5. OVN 로드 밸런서 생성                               | 102        |
| <b>11장. 계층 7 로드 밸런싱 구현</b>                        | <b>108</b> |
| 11.1. 계층 7 로드 밸런싱 정보                              | 109        |
| 11.2. 로드 밸런싱 서비스의 계층 7 로드 밸런싱                     | 110        |
| 11.3. 계층 7 로드 밸런싱 규칙                              | 110        |
| 11.4. 계층 7 로드 밸런싱 규칙 유형                           | 110        |
| 11.5. 계층 7 로드 밸런싱 규칙 비교 유형                        | 111        |
| 11.6. 계층 7 로드 밸런싱 규칙 결과 버전                        | 112        |
| 11.7. 계층 7 로드 밸런싱 정책                              | 112        |
| 11.8. 계층 7 로드 밸런싱 정책 논리                           | 112        |
| 11.9. 계층 7 로드 밸런싱 정책 작업                           | 113        |
| 11.10. 계층 7 로드 밸런싱 정책 위치                          | 113        |
| 11.11. 비보안 HTTP 요청을 보안 HTTP로 리디렉션                 | 114        |
| 11.12. 폴의 시작 경로를 기반으로 요청 리디렉션                     | 117        |
| 11.13. 하위 도메인 요청을 특정 폴로 전송                        | 120        |
| 11.14. 특정 폴로 끝나는 호스트 이름을 기반으로 요청 전송               | 123        |
| 11.15. 브라우저 쿠키의 부재에 따라 요청을 특정 폴에 전송               | 125        |
| 11.16. 브라우저 쿠키 없음 또는 잘못된 쿠키 값을 특정 폴로 통한 요청 전송     | 128        |
| 11.17. 이름이 호스트 이름 및 경로와 일치하는 폴에 요청 전송             | 132        |
| 11.18. 쿠키를 사용하여 기존 프로덕션 사이트에서 A-B 테스트 구성          | 135        |
| <b>12장. 태그를 사용하여 로드 밸런싱 서비스 오브젝트 그룹화</b>          | <b>141</b> |
| 12.1. 서비스 오브젝트 로드 밸런싱을 생성할 때 태그 추가                | 141        |
| 12.2. 기존 로드 밸런싱 서비스 오브젝트에 태그 추가 또는 제거             | 145        |
| 12.3. 태그를 사용하여 로드 밸런싱 서비스 오브젝트 필터링                | 149        |
| <b>13장. 에지에서 네트워크 트래픽 로드 밸런싱을 위한 가용성 영역 생성</b>    | <b>152</b> |
| 13.1. 로드 밸런싱 서비스를 위한 가용성 영역 생성                    | 152        |
| 13.2. 가용성 영역에서 로드 밸런서 생성                          | 158        |
| <b>14장. 로드 밸런싱 서비스 업데이트 및 업그레이드</b>               | <b>161</b> |
| 14.1. 로드 밸런싱 서비스 업데이트 및 업그레이드                     | 161        |
| 14.2. 실행 중인 로드 밸런싱 서비스 인스턴스 업데이트                  | 162        |
| <b>15장. 로드 밸런싱 서비스 문제 해결 및 유지 관리</b>              | <b>164</b> |
| 15.1. 로드 밸런서 확인                                   | 164        |
| 15.2. 서비스 인스턴스 관리 로그 로드 밸런싱                       | 170        |

|                                 |     |
|---------------------------------|-----|
| 15.3. 특정 로드 밸런싱 서비스 인스턴스 마이그레이션 | 171 |
| 15.4. SSH를 사용하여 인스턴스 로드 밸런싱에 연결 | 172 |
| 15.5. 리스너 통계 표시                 | 173 |
| 15.6. 리스너 요청 오류 해석              | 175 |



## 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 더 나은지 알려주십시오.

### Jira에서 문서 피드백 제공

**문제 생성** 양식을 사용하여 문서에 대한 피드백을 제공합니다. Jira 문제는 Red Hat OpenStack Platform Jira 프로젝트에서 생성되어 피드백의 진행 상황을 추적할 수 있습니다.

1. Jira에 로그인했는지 확인합니다. Jira 계정이 없는 경우 피드백을 제출할 계정을 생성합니다.
2. 다음 링크를 클릭하여 **문제 생성** 페이지를 엽니다.  
<https://issues.redhat.com/secure/CreateInfoDetails!init.jspx?pid=12336920&summary=Documentation%20feedback:%20%3CAdd%20summary%20here%3E&i<Include+the+documentation+URL,+the%20chapter+or+section+number,+and+a+detailed+descrip>
3. **요약** 및 **설명** 필드를 작성합니다. **설명** 필드에 문서 URL, 장 또는 섹션 번호, 문제에 대한 자세한 설명을 포함합니다. 양식의 다른 필드를 수정하지 마십시오.
4. **생성**을 클릭합니다.

## 1장. 로드 밸런싱 서비스 소개

로드 밸런싱 서비스(octavia)는 RHOSP(Red Hat OpenStack Platform) 배포를 위해 LBaaS(Load Balancing-as-a-Service) API 버전 2 구현을 제공합니다. 로드 밸런싱 서비스는 필요에 따라 시작되는 여러 가상 머신, 컨테이너 또는 베어 메탈 서버를 전체적으로 관리합니다. 온디맨드로 수평 스케일링을 제공하는 기능으로 인해 대규모 RHOSP 엔터프라이즈 배포에 적합한 로드 밸런싱 서비스가 완전히 제공되는 로드 밸런서가 됩니다.



### 참고

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

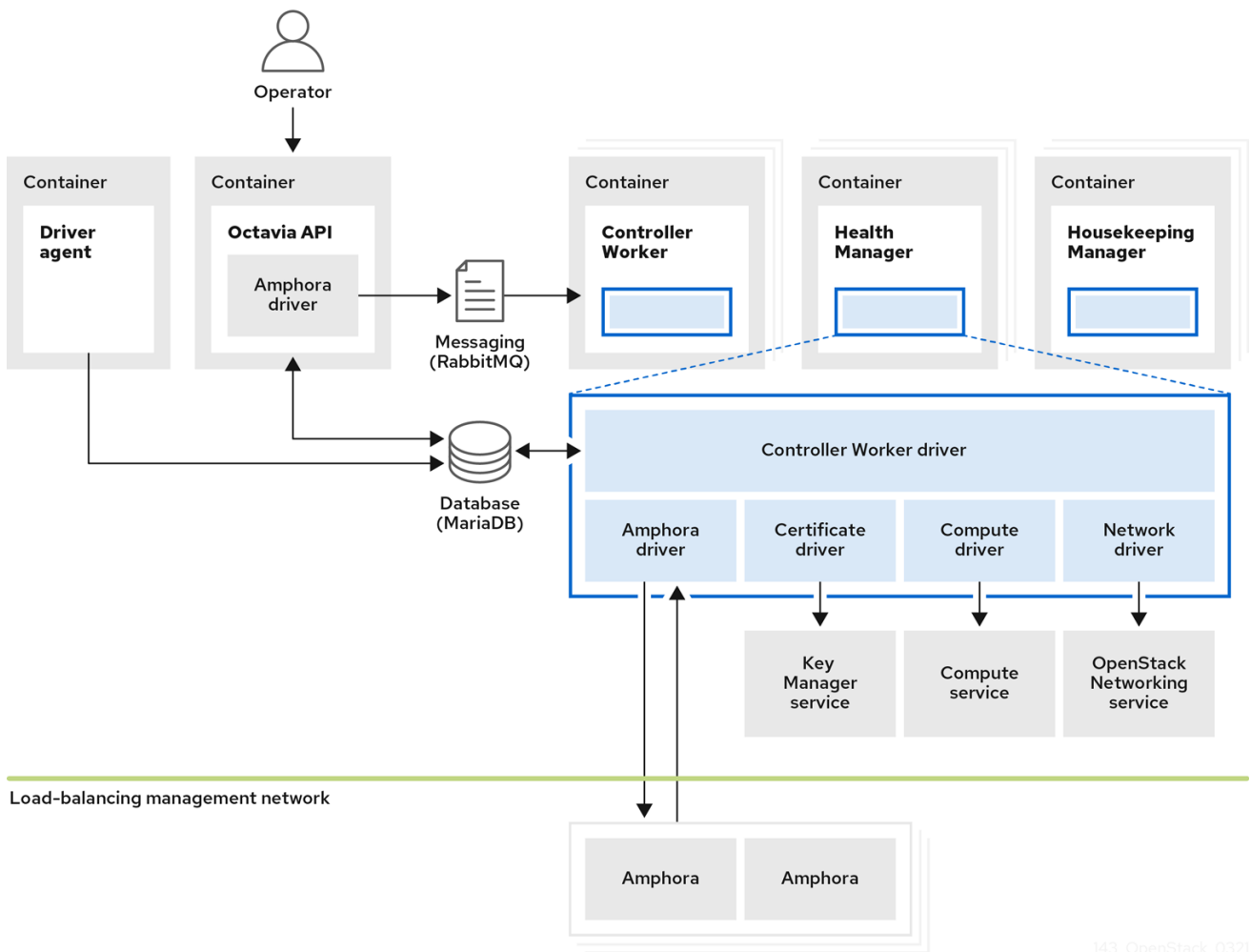
- 1.1절. "서비스 구성 요소 로드 밸런싱"
- 1.2절. "서비스 오브젝트 모델 로드 밸런싱"
- 1.3절. "Red Hat OpenStack Platform의 로드 밸런싱 사용"

### 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 *amphorae*와 통신합니다.

octavia를 사용하는 경우 부동 IP(FIP)가 필요하지 않은 로드 밸런서 가상 IP(VIP)를 생성할 수 있습니다. FIP를 사용하지 않는 경우 로드 밸런서를 통해 성능을 향상시킬 수 있다는 이점이 있습니다.

그림 1.1. 서비스 구성 요소 로드 밸런싱



143\_OpenStack\_0321

그림 1.1에서는 로드 밸런싱 서비스의 구성 요소가 기본적으로 컨트롤러 노드에 있는 Networking API 서버와 동일한 노드에서 호스팅되는 것을 보여줍니다. 로드 밸런싱 서비스는 다음 구성 요소로 구성됩니다.

#### Octavia API(octavia\_api 컨테이너)

사용자가 octavia와 상호 작용할 수 있는 REST API를 제공합니다.

#### Controller Worker(octavia\_worker 컨테이너)

로드 밸런싱 관리 네트워크를 통해 구성 및 구성 업데이트를 amphorae로 보냅니다.

#### 상태 관리자(octavia\_health\_manager 컨테이너)

개별 amphorae의 상태를 모니터링하고 amphora가 장애가 발생하면 페일오버 이벤트를 처리합니다.

#### Homekeeping Manager(octavia\_housekeeping 컨테이너)

삭제된 데이터베이스 레코드를 정리하고 amphora 인증서 교체를 관리합니다.

#### 드라이버 에이전트(octavia\_driver\_agent 컨테이너)

OVN과 같은 다른 공급자 드라이버를 지원합니다.

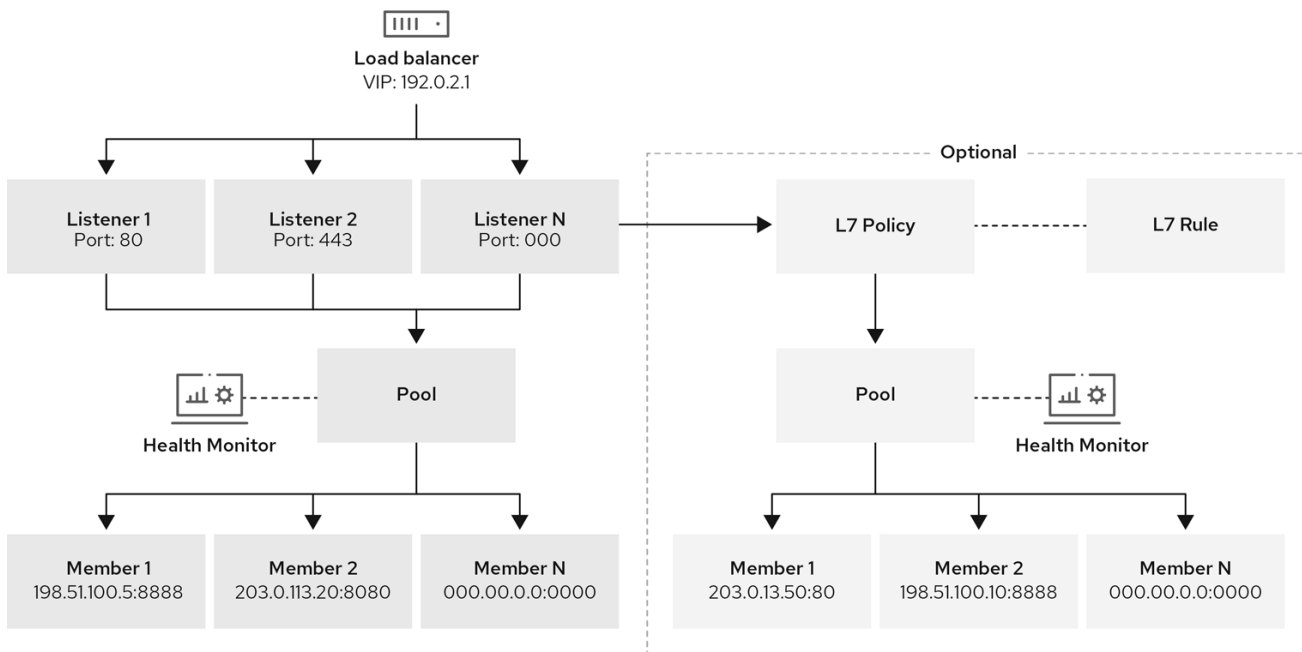
#### Amphora

부하 분산을 수행합니다. Amphorae는 일반적으로 리스너, 풀, 상태 모니터, L7 정책 및 멤버 구성에 따라 로드 밸런싱 매개변수로 구성된 컴퓨팅 노드에서 실행되는 인스턴스입니다. Amphorae는 주기적인 하트비트를 Health Manager로 보냅니다.

## 1.2. 서비스 오브젝트 모델 로드 밸런싱

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

그림 1.2. 서비스 오브젝트 모델 다이어그램 로드 밸런싱



143\_OpenStack\_0321

## 로드 밸런서

로드 밸런싱 엔터티를 나타내는 top API 오브젝트입니다. 로드 밸런서를 생성할 때 VIP 주소가 할당됩니다. amphora 공급자를 사용하여 로드 밸런서를 생성하는 경우 하나 이상의 컴퓨팅 노드에서 하나 이상의 amphora 인스턴스가 시작됩니다.

## 리스너

로드 밸런서가 수신 대기하는 포트(예: HTTP의 경우 TCP 포트 80)입니다.

## 상태 모니터

각 백엔드 멤버 서버에서 정기적으로 실패한 서버를 사전 감지하고 풀에서 임시로 제거하는 프로세스입니다.

## pool

로드 밸런서의 클라이언트 요청을 처리하는 멤버 그룹입니다. API를 사용하여 둘 이상의 리스너와 풀을 연결할 수 있습니다. L7 정책과 함께 풀을 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 연결된 L7 규칙 컬렉션이며 백엔드 풀에도 연결할 수 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 PLATFORM의 로드 밸런싱 사용

로드 밸런싱은 클라우드 배포를 위한 단순한 또는 자동 제공 확장 및 가용성을 가능하게 하는 데 중요합니다. 로드 밸런싱 서비스(octavia)는 다른 RHOSP(Red Hat OpenStack Platform) 서비스에 따라 다릅니다.

- **Compute 서비스(nova)** - 로드 밸런싱 서비스 VM 인스턴스(amphora) 라이프사이클을 관리하고 필요에 따라 컴퓨팅 리소스를 생성하기 위해 사용됩니다.
- **네트워킹 서비스(neutron)** - amphorae, 테넌트 환경 및 외부 네트워크 간의 네트워크 연결용입니다.
- **키 관리자 서비스(barbican)** - TLS 세션 종료가 리스너에 구성된 경우 TLS 인증서 및 인증 정보를 관리합니다.
- **ID 서비스(keystone)** - octavia API에 대한 인증 요청 및 로드 밸런싱 서비스에서 다른 RHOSP 서비스로 인증합니다.
- **Image 서비스(glance)** - amphora 가상 머신 이미지를 저장하는 데 사용됩니다.
- **공통 라이브러리(oslo)** - 로드 밸런싱 서비스 컨트롤러 구성 요소 간 통신의 경우 표준 OpenStack 프레임워크 내에서 로드 밸런싱 서비스가 작동하고 시스템을 검토하고 시스템 및 프로젝트 코드 구조를 검토합니다.
- **taskflow** - 공용 라이브러리의 일부입니다. 로드 밸런싱 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 인터페이스는 로드 밸런싱 서비스의 주요 재구성을 방지합니다.

## 2장. 로드 밸런싱 서비스 구현을 위한 고려 사항

사용할 공급자 선택 또는 고가용성 환경을 구현할지 여부와 같이 RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 배포할 때 여러 가지 결정을 수행해야 합니다.

- 2.1절. "서비스 공급자 드라이버 로드 밸런싱"
- 2.2절. "로드 밸런싱 서비스(octavia) 기능 지원 매트릭스"
- 2.3절. "서비스 소프트웨어 요구 사항 로드 밸런싱"
- 2.4절. "언더클라우드의 로드 밸런싱 서비스 사전 요구 사항"
- 2.5절. "로드 밸런싱 서비스 인스턴스에 대한 활성-standby 토폴로지의 기본 사항"
- 2.6절. "Load-balancing service에 대한 배포 후 단계"

### 2.1. 서비스 공급자 드라이버 로드 밸런싱

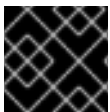
RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)는 Octavia v2 API를 사용하여 여러 공급자 드라이버 활성화를 지원합니다. 하나의 공급자 드라이버 또는 여러 공급자 드라이버를 동시에 사용하도록 선택할 수 있습니다.

RHOSP는 amphora 및 OVN(Open Virtual Network)이라는 두 가지 로드 밸런싱 공급자를 제공합니다.

기본 Amphora는 컴퓨팅 환경과 함께 확장되는 기능 세트가 있는 고가용성 로드 밸런서입니다. 이로 인해 amphora는 대규모 배포에 적합합니다.

OVN 로드 밸런싱 공급자는 기본 기능 세트가 있는 경량 로드 밸런서입니다. OVN은 east-west, 계층 4 네트워크 트래픽에 일반적으로 사용됩니다. OVN은 신속하게 프로비저닝하고 amphora와 같은 완전한 부하 분산 공급자보다 적은 리소스를 소비합니다.

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



#### 중요

이 섹션의 정보는 달리 표시되지 않는 한 amphora 로드 밸런싱 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 amphora 및 OVN(Open Virtual Network)이라는 두 가지 로드 밸런싱 공급자를 제공합니다.

*Amphora*는 별도의 haproxy VM과 추가 대기 시간 흡이 필요한 완전한 기능을 갖춘 로드 밸런싱 공급자입니다.

*OVN*은 모든 노드에서 실행되며 별도의 VM이나 추가 흡이 필요하지 않습니다. 그러나 OVN은 amphora보다 부하 분산 기능이 훨씬 적습니다.

다음 표에는 RHOSP(Red Hat OpenStack Platform) 17.1에서 지원하는 로드 밸런싱 서비스의 기능이 나열되어 있으며 해당 기능에 대한 유지 관리 릴리스 지원이 시작되었습니다.



## 참고

기능이 나열되지 않은 경우 RHOSP 17.1에서 해당 기능을 지원하지 않습니다.

표 2.1. 로드 밸런싱 서비스(octavia) 기능 지원 매트릭스

| 기능                                | RHOSP 17.1의 지원 수준 |         |
|-----------------------------------|-------------------|---------|
|                                   | Amphora 공급자       | OVN 공급자 |
| ML2/OVS L3 HA                     | 완전 지원             | 지원되지 않음 |
| ML2/OVS DVR                       | 완전 지원             | 지원되지 않음 |
| ML2/OVS L3 HA + 구성 가능 네트워크 노드 [1] | 완전 지원             | 지원되지 않음 |
| ML2/OVS DVR + 구성 가능 네트워크 노드 [1]   | 완전 지원             | 지원되지 않음 |
| ML2/OVN L3 HA                     | 완전 지원             | 완전 지원   |
| ML2/OVN DVR                       | 완전 지원             | 완전 지원   |
| DPDK                              | 지원되지 않음           | 지원되지 않음 |
| SR-IOV                            | 지원되지 않음           | 지원되지 않음 |
| 상태 모니터                            | 완전 지원             | 지원되지 않음 |
| Amphora active-standby            | 완전 지원             | 지원되지 않음 |
| HTTPS 로드 밸런서 종료(barbican)         | 완전 지원             | 지원되지 않음 |
| Amphora 스페어 풀                     | 기술 프리뷰만 지원        | 지원되지 않음 |
| UDP                               | 완전 지원             | 완전 지원   |
| 백업 멤버                             | 기술 프리뷰만 지원        | 지원되지 않음 |
| 공급자 프레임워크                         | 기술 프리뷰만 지원        | 지원되지 않음 |
| TLS 클라이언트 인증                      | 기술 프리뷰만 지원        | 지원되지 않음 |

|               |            |         |
|---------------|------------|---------|
| TLS 백엔드 암호화   | 기술 프리뷰만 지원 | 지원되지 않음 |
| Octavia 플레이버  | 완전 지원      | 지원되지 않음 |
| 오브젝트 태그       | 완전 지원      | 지원되지 않음 |
| 리스너 API 시간 초과 | 완전 지원      | 지원되지 않음 |
| 로그 오프로드       | 완전 지원      | 지원되지 않음 |
| VIP 액세스 제어 목록 | 완전 지원      | 지원되지 않음 |
| 가용성 영역        | 완전 지원      | 지원되지 않음 |
| 블롭 기반 amphora | 지원되지 않음    | 지원되지 않음 |

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

#### 추가 리소스

- [2.1절. "서비스 공급자 드라이버 로드 밸런싱"](#)

### 2.3. 서비스 소프트웨어 요구 사항 로드 밸런싱

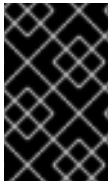
RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 사용하려면 다음과 같은 핵심 OpenStack 구성 요소를 구성해야 합니다.

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

### 2.4. 언더클라우드의 로드 밸런싱 서비스 사전 요구 사항

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

- 성공적인 언더클라우드 설치
- 언더클라우드의 로드 밸런싱 서비스입니다.
- 컨테이너 기반 오버클라우드 배포 계획.
- 컨트롤러 노드에 구성된 서비스 구성 요소를 로드 밸런싱합니다.



## 중요

기존 오버클라우드 배포에서 로드 밸런싱 서비스를 활성화하려면 언더클라우드를 준비해야 합니다. 이로 인해 로드 밸런싱 서비스가 실행되지 않고 오버클라우드 설치가 성공한 것으로 보고됩니다.

## 2.5. 로드 밸런싱 서비스 인스턴스에 대한 활성화-STANDBY 토폴로지의 기본 사항

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 배포할 때 사용자가 생성할 때 기본적으로 로드 밸런서를 사용할 수 있는지 여부를 결정할 수 있습니다. 사용자에게 선택권을 부여하려면 RHOSP 배포 후 고가용성 로드 밸런서와 독립 실행형 로드 밸런서를 생성하기 위한 플레이버를 생성하기 위한 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 active-standby 토폴로지를 구현할 때 로드 밸런싱 서비스 인스턴스를 고가용성으로 만들 수 있습니다.

이 토폴로지에서 로드 밸런싱 서비스는 각 로드 밸런서에 대한 활성화 및 대기 인스턴스를 부팅하고 각각에 대한 세션 지속성을 유지합니다. 활성화 인스턴스가 비정상 상태가 되면 인스턴스가 standby 인스턴스로 자동으로 장애 조치되어 활성화됩니다. 로드 밸런싱 서비스 상태 관리자는 실패한 인스턴스를 자동으로 다시 빌드합니다.

### 추가 리소스

- [4.2절. "로드 밸런싱 서비스 인스턴스에 대한 활성화-standby 토폴로지 활성화"](#)

## 2.6. LOAD-BALANCING SERVICE에 대한 배포 후 단계

RHOSP(Red Hat OpenStack Platform)는 로드 밸런싱 서비스(octavia)의 배포 후 단계를 단순화하는 워크플로우 작업을 제공합니다. 이 워크플로는 일련의 Ansible 플레이북을 실행하여 다음 배포 후 단계를 오버클라우드 배포의 마지막 단계로 제공합니다.

- 인증서 및 키를 구성합니다.
- amphorae와 Load-balancing service Controller worker 및 상태 관리자 간의 로드 밸런싱 관리 네트워크를 구성합니다.

### Amphora 이미지

사전 프로비저닝된 서버에서 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 다운로드하여 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 정의 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 간의 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. "서비스 인증서 및 키 로드 밸런싱 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. 로드 밸런싱 서비스는 양방향 TLS 인증을 사용하여 양면이 신뢰할 수 있는지 확인합니다.



#### 참고

이는 전체 TLS 핸드셰이크 프로세스를 단순화합니다. TLS 핸드셰이크 프로세스에 대한 자세한 내용은 [TLS 1.3 RFC 8446](#) 을 참조하십시오.

양방향 TLS 인증에는 두 단계가 포함됩니다. *1단계*에서는 로드 밸런싱 서비스 작업자 프로세스와 같은 컨트롤러 프로세스가 로드 밸런싱 서비스 인스턴스에 연결되고 인스턴스는 서버 인증서를 컨트롤러에 제공합니다. 그런 다음 컨트롤러는 컨트롤러에 저장된 서버 CA(인증 기관) 인증서에 대해 서버 인증서의 유효성을 검사합니다. 제공된 인증서가 서버 CA 인증서에 대해 검증되면 연결은 2단계로 진행됩니다.

*2단계*에서 컨트롤러는 클라이언트 인증서를 로드 밸런싱 서비스 인스턴스에 제공합니다. 그런 다음 인스턴스에서 인스턴스에 저장된 클라이언트 CA 인증서에 대해 인증서의 유효성을 검사합니다. 이 인증서의 유효성이 확인되면 나머지 TLS 핸드셰이크는 컨트롤러와 로드 밸런싱 서비스 인스턴스 간의 보안 통신 채널을 계속 설정합니다.

#### 추가 리소스

- 3.3절. "서비스 인증서 및 키 로드 밸런싱 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 만료일 내에 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 기간을 부여합니다.

#### 추가 리소스

- 3.3절. "서비스 인증서 및 키 로드 밸런싱 구성"

### 3.3. 서비스 인증서 및 키 로드 밸런싱 구성

인증서 및 키를 생성하도록 RHOSP(Red Hat OpenStack Platform) director를 구성하거나 자체적으로 제공할 수 있습니다. 필요한 개인 인증 기관을 자동으로 생성하고 필요한 인증서를 발행하도록 director를 구성합니다. 이러한 인증서는 내부 로드 밸런싱 서비스(octavia) 통신용이며 사용자에게 노출되지 않습니다.



### 중요

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



### 참고

수동으로 생성된 인증서에서 자동으로 생성된 인증서로 전환하는 것은 RHOSP director에서 지원되지 않습니다. 그러나 `/var/lib/config-data/puppet-generated/octavia/etc/octavia/certs` 디렉터리의 컨트롤러 노드에서 기존 인증서를 삭제하고 오버클라우드를 업데이트하여 인증서 다시 생성을 적용할 수 있습니다.

고유한 인증서 및 키를 사용해야 하는 경우 다음 단계를 완료합니다.

### 사전 요구 사항

- "로드 밸런싱 서비스 기본 설정 변경"을 읽고 이해하십시오('추가 리소스의 링크 참조).

### 절차

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

```
$ 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 키와 함께 사용되는 암호입니다.
- **OctaviaClientCert:**  
컨트롤러용으로 Octavia CA에서 발행한 클라이언트 인증서 및 암호화되지 않은 키입니다.
- **OctaviaGenerateCerts:**  
director에 자동 인증서 및 키 생성을 활성화(false)하도록 지시하는 부울입니다.

**중요**

**OctaviaGenerateCerts** 를 false로 설정해야 합니다.

**예제**

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

    MIIDgzCCAmugAwIBAgIJAKk46qw6ncJaMA0GCSqGSIb3DQEBCwUAMFgxGzAJBgNV
    [snip]
    sFW3S2roS4X0Af/kSSD8mIBBTFTCMBAj6rtLBKLaQblxEplzrgvp
    -----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-----

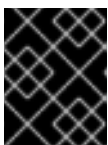
    MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQU771O8MTQV8RY

    [snip]
    KfrjE3UqTF+ZaalQaz3yayXW
    -----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 매개변수와 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"](#)
- *Red Hat OpenStack Platform* 배포 가이드의 [환경 파일](#) 사용자 지정
- *Red Hat OpenStack Platform* 배포 가이드 사용자 지정 [에](#) 오버클라우드 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 인스턴스(amphorae)를고가용성으로 설정할 수 있습니다. 로드 밸런싱 서비스를 설정하려면 director도 사용합니다.

- [4.1절. "로드 밸런싱 서비스 배포"](#)
- [4.2절. "로드 밸런싱 서비스 인스턴스에 대한 활성-standby 토폴로지 활성화"](#)
- [4.3절. "로드 밸런싱 서비스 기본 설정 변경"](#)

### 4.1. 로드 밸런싱 서비스 배포

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

#### 사전 요구 사항

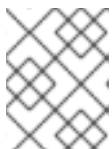
- 환경에 octavia 이미지에 액세스할 수 있는지 확인합니다.

#### 절차

- 배포 명령을 실행하고 코어 heat 템플릿, 환경 파일 및 **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 스택 업데이트 또는 업그레이드 중에 amphora를 최신 amphora 이미지로 업데이트합니다.

#### 추가 리소스

- *director 가이드를 사용하여 Red Hat OpenStack Platform 설치 및 관리의* [배포 명령 옵션](#)

### 4.2. 로드 밸런싱 서비스 인스턴스에 대한 활성-STANDBY 토폴로지 활성화

RHOSP(Red Hat OpenStack Platform) director를 사용하여 Active-standby 토폴로지를 구현할 때 로드 밸런싱 서비스 인스턴스(amphorae)를 사용할 수 있습니다. director는 사용자 환경의 계획 세트에 해당하는 오케스트레이션 서비스(heat) 템플릿을 사용합니다. 언더클라우드는 이러한 계획을 가져와서 해당 지침에 따라 로드 밸런싱 서비스와 RHOSP 환경을 생성합니다.

#### 사전 요구 사항

- Compute 서비스에 대해 anti-affinity가 활성화되어 있는지 확인합니다. 이는 기본값입니다.
- 컴퓨팅 노드 호스트 세 개 이상:

- amphorae를 다른 호스트에 배치하는 두 개의 컴퓨팅 노드 호스트 (Compute anti-affinity)
- 문제가 발생할 때 활성 로드 밸런서를 통해 성공적으로 실패하는 세 번째 호스트입니다.

## 절차

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

```
$ source ~/stackrc
```

3. 사용자 지정 YAML 환경 파일을 생성합니다.

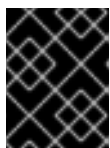
### 예제

```
$ 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 코어 heat 템플릿, 환경 파일 및 이 새 사용자 지정 환경 파일을 포함합니다.



### 중요

후속 환경 파일에 정의된 매개변수와 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

- 배포가 완료되고 로드 밸런서를 생성한 후 다음 명령을 실행합니다.

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

로드 밸런싱 서비스 인스턴스가 고가용성 구성이 성공하면 두 개의 인스턴스(amphorae)에 대한 출력이 표시되고 **SINGLE** 과 같은 역할이 발생하지 않습니다.

## 추가 리소스

- Red Hat OpenStack Platform 배포 가이드의 [환경 파일](#) 사용자 지정
- Red Hat OpenStack Platform 배포 가이드 사용자 지정 [예](#) 오버클라우드 생성에 환경 파일 포함

### 4.3. 로드 밸런싱 서비스 기본 설정 변경

RHOSP(Red Hat OpenStack Platform) director를 사용하여 로드 밸런싱 서비스(octavia)를 설정합니다. director는 사용자 환경의 계획 세트에 해당하는 오케스트레이션 서비스(heat) 템플릿을 사용합니다. 언더클라우드에서는 이러한 계획을 가져와서 해당 지침에 따라 로드 밸런싱 서비스와 RHOSP 환경을 생성합니다.

#### 사전 요구 사항

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

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

- 수정할 매개변수를 결정합니다.  
다음은 몇 가지 예입니다.

- **OctaviaControlNetwork**

로드 밸런싱 관리 네트워크에 사용되는 neutron 네트워크의 이름입니다.

- **OctaviaControlSubnetCidr**

CIDR 형식의 amphora 제어 서브넷의 서브넷입니다.

이 매개변수를 사용하여 amphora 제어 서브넷을 IPv6 CIDR로 지정할 수 있습니다.

**OctaviaControlSubnetCidr** 에서 IPv6 CIDR을 사용하려면

**OctaviaControlSubnetPoolStart** 및 **OctaviaControlSubnetPoolEnd** 에서 IPv6 주소도 설정해야 합니다.



#### 중요

로드 밸런싱 관리 네트워크에 IPv6 CIDR을 사용하는 이 기능은 이 릴리스에서 기술 프리뷰로 사용할 수 있으므로 Red Hat에서 완전히 지원하지 않습니다. 테스트 용도로만 사용해야 하며 프로덕션 환경에 배포해서는 안 됩니다.

기술 프리뷰 기능에 대한 자세한 내용은 [적용 범위 상세 정보](#)를 참조하십시오.



#### 중요

amphora 제어 서브넷에 대한 CIDR을 설정한 후에는 이러한 값을 수정할 수 없습니다. 즉, IPv4 서브넷에서 IPv6 서브넷으로 업그레이드할 수 없습니다.

- **OctaviaMgmtPortDevName**

amphora 시스템과 octavia worker/health-manager 간의 통신에 사용되는 octavia 관리 네트워크 인터페이스의 이름입니다.

#### 절차

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

```
$ source ~/stackrc
```

3. 사용자 지정 YAML 환경 파일을 생성합니다.

### 예제

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

- 환경 파일에는 키워드 **parameter\_defaults** 가 포함되어야 합니다. **parameter\_defaults** 키워드 뒤에 매개변수 값 쌍을 배치합니다.

### 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'
```

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



### 중요

후속 환경 파일에 정의된 매개변수와 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

- Red Hat OpenStack Platform 배포 가이드의 [환경 파일 사용자 지정](#)
- Red Hat OpenStack Platform 배포 가이드 사용자 지정 [예](#) 오버클라우드 생성에 환경 파일 포함

## 5장. 로드 밸런싱 서비스 인스턴스 로그 관리

테넌트 흐름 로깅을 활성화하거나 amphora 로컬 파일 시스템에 로깅을 억제할 수 있습니다. 컨테이너 세트 또는 선택한 끝점의 다른 syslog 수신자에 관리 또는 테넌트 흐름 로그를 전달할 수도 있습니다.

TCP syslog 프로토콜을 사용하도록 선택하는 경우 기본 엔드포인트가 실패하는 경우 관리 및 테넌트 로그 오프로드를 위해 하나 이상의 보조 끝점을 지정할 수 있습니다.

또한 syslog 기능 값 설정, 테넌트 흐름 로그 형식 변경, 커널 및 cron과 같은 소스의 로그를 포함하도록 관리 로깅 범위 확장 등 다양한 기타 로깅 기능을 제어할 수 있습니다.

- 5.1절. "로드 밸런싱 서비스 인스턴스 관리 로그 오프로드 활성화"
- 5.2절. "로드 밸런싱 서비스 인스턴스에 대한 테넌트 흐름 로그 오프로드 활성화"
- 5.3절. "로드 밸런싱 서비스 인스턴스 테넌트 흐름 로깅 비활성화"
- 5.4절. "로드 밸런싱 서비스 인스턴스 로컬 로그 스토리지 비활성화"
- 5.5절. "로드 밸런싱 서비스 인스턴스 로깅을 위한 heat 매개변수"
- 5.6절. "서비스 인스턴스 테넌트 흐름 로그 형식 로드 밸런싱"

### 5.1. 로드 밸런싱 서비스 인스턴스 관리 로그 오프로드 활성화

기본적으로 로드 밸런싱 서비스 인스턴스(amphorae)는 systemd 저널의 로컬 시스템에 로그를 저장합니다. 그러나 amphorae 오프로드가 syslog 수신기에 로그되어 관리 로그를 집계하도록 지정할 수 있습니다. 로그 오프로드를 사용하면 관리자가 로그의 한 위치로 이동하고 amphorae가 순환될 때 로그를 유지할 수 있습니다.

#### 절차

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

```
$ source ~/stackrc
```

3. 사용자 지정 YAML 환경 파일을 생성합니다.

#### 예제

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

4. **parameter\_defaults** 아래의 YAML 환경 파일에서 **OctaviaLogOffload**를 **true**로 설정합니다.

```
parameter_defaults:
  OctaviaLogOffload: true
...
```



## 참고

amphorae 오프로드 관리 로그는 **OctaviaAdminLogFacility** 매개변수로 다른 값을 지정하지 않는 한 기본적으로 **local1**의 syslog 기능 값을 사용합니다. 유효한 값은 0 Cryostat- Cryostat7입니다.

## 예제

```
parameter_defaults:
  OctaviaLogOffload: true
  OctaviaAdminLogFacility: 2
  ...
```

- amphorae는 haproxy 관리자 로그, keepalived 및 amphora 에이전트 로그와 같은 로드 밸런서 관련 관리 로그만 전달합니다. 커널, 시스템 및 보안 로그와 같은 amphorae에서 모든 관리 로그를 전송하도록 amphorae를 구성하려면 **OctaviaForwardAllLogs**를 **true**로 설정합니다.

## 예제

```
parameter_defaults:
  OctaviaLogOffload: true
  OctaviaForwardAllLogs: true
  ...
```

- 로그 프로토콜(기본값) 또는 TCP를 선택합니다.  
기본 끝점이 실패하는 경우 amphorae는 로그 프로토콜이 TCP인 경우에만 보조 끝점으로 로그를 보냅니다.

```
parameter_defaults:
  OctaviaLogOffload: true
  OctaviaConnectionLogging: false
  OctaviaLogOffloadProtocol: tcp
  ...
```

- amphorae는 로그 메시지를 수신 대기하는 syslog 수신자가 포함된 Orchestration 서비스(heat)에서 정의한 기본 컨테이너 세트를 사용합니다. 다른 끝점 세트를 사용하려면 **OctaviaAdminLogTargets** 매개변수로 지정할 수 있습니다.

테넌트 흐름 로그 오프로드에 구성된 엔드포인트는 관리 로그 오프로드에 사용되는 끝점과 같을 수 있습니다.

또한 로그 오프로드 프로토콜이 TCP인 경우 첫 번째 끝점에 연결할 수 없는 경우 연결에 성공할 때까지 amphorae는 추가 끝점을 나열한 순서로 시도합니다.

```
OctaviaAdminLogTargets: <ip_address>:<port>[, <ip_address>:<port>]
```

## 예제

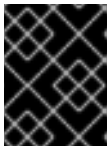
```
parameter_defaults:
  OctaviaLogOffload: true
  OctaviaLogOffloadProtocol: tcp
  OctaviaAdminLogTargets: 192.0.2.1:10514, 2001:db8:1::10:10514
  ...
```

8. 기본적으로 로그 오프로드를 활성화하면 테넌트 흐름 로그도 오프로드됩니다. 테넌트 흐름 로그 오프로드를 비활성화하려면 **OctaviaConnectionLogging** 을 **false** 로 설정합니다.

#### 예제

```
parameter_defaults:
  OctaviaLogOffload: true
  OctaviaConnectionLogging: false
  ...
```

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



#### 중요

후속 환경 파일에 정의된 매개변수와 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

- **OctaviaAdminLogTargets** 또는 **OctaviaTenantLogTargets** 를 사용하여 특정 끝점을 지정하지 않는 한 amphorae 오프로드는 다른 RHOSP 로그(/var/log/containers/octavia-amphorae/)와 동일한 위치에 RHOSP 컨트롤러에 로그됩니다.
- 다음 로그 파일이 있는지에 적합한 위치를 확인합니다.
  - **Octavia-amphora.log**-- 관리 로그의 로그 파일입니다.
  - (활성화된 경우) **octavia-tenant-traffic.log**-- 테넌트 트래픽 흐름 로그의 로그 파일입니다.

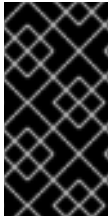
#### 추가 리소스

- [5.5절. "로드 밸런싱 서비스 인스턴스 로깅을 위한 heat 매개변수"](#)
- Red Hat OpenStack Platform 배포 가이드의 [환경 파일 사용자 지정](#)
- Red Hat OpenStack Platform 배포 가이드 사용자 지정 [예](#) 오버클라우드 생성에 환경 파일 포함

## 5.2. 로드 밸런싱 서비스 인스턴스에 대한 테넌트 흐름 로그 오프로드 활성화

기본적으로 로드 밸런싱 서비스 인스턴스(amphorae)는 systemd 저널의 로컬 시스템에 로그를 저장합니다. 그러나 대체 syslog 수신자 끝점을 대신 지정할 수 있습니다. 테넌트 흐름 로그는 테넌트 연결 수에 따라 크기가 증가하므로 대체 syslog 수신자에 충분한 디스크 공간이 포함되어 있는지 확인합니다.

관리 로그 오프로드가 활성화되면 기본적으로 테넌트 흐름 로그 오프로드가 자동으로 활성화됩니다. 테넌트 흐름 로그 오프로드를 끄려면 **OctaviaConnectionLogging** 매개변수를 **false** 로 설정합니다.



### 중요

테넌트 흐름 로깅은 로드 밸런서가 수신하는 연결 수에 따라 많은 syslog 메시지를 생성할 수 있습니다. 테넌트 흐름 로깅은 로드 밸런서에 대한 각 연결에 대해 하나의 로그 항목을 생성합니다. 로그 볼륨을 모니터링하고 로드 밸런서에서 관리하는 예상 연결 수에 따라 syslog 수신자를 적절하게 구성합니다.

### 절차

1. 언더클라우드 호스트에 **stack** 사용자로 로그인합니다.

2. 언더클라우드 인증 정보 파일을 가져옵니다.

```
$ source ~/stackrc
```

3. **OctaviaConnectionLogging** 매개변수가 설정된 환경 파일을 찾습니다.

```
$ grep -rl OctaviaConnectionLogging /home/stack/templates/
```

4. 파일을 찾을 수 없는 경우 환경 파일을 생성합니다.

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

5. **OctaviaLogOffload** 및 **OctaviaConnectionLogging** 매개변수를 환경 파일의 **parameter\_defaults** 섹션에 추가하고 값을 **true** 로 설정합니다.

```
parameter_defaults:
  OctaviaLogOffload: true
  OctaviaConnectionLogging: true
  ...
```



### 참고

amphorae는 다른 값을 지정하는 데 **OctaviaTenantLogFacility** 매개변수를 사용하지 않는 한, syslog 기능 기본값 **local0** 을 사용하여 테넌트 흐름 로그를 오프로드합니다. 유효한 값은 0 Cryostat- Cryostat7입니다.

6. 선택 사항: amphorae가 테넌트 및 관리 로그 오프로드 모두에 사용하는 기본 끝점을 변경하려면 각각 **OctaviaTenantLogTargets** 및 **OctaviaAdminLogTargets** 를 사용합니다. amphorae는 로그 메시지를 수신하는 syslog 수신자가 포함된 기본 컨테이너 세트를 사용합니다. 또한 로그 오프로드 프로토콜이 TCP인 경우 첫 번째 끝점에 연결할 수 없는 경우 연결에 성공할 때까지 amphorae는 추가 끝점을 나열한 순서로 시도합니다.

```
OctaviaAdminLogTargets: <ip-address>:<port>[, <ip-address>:<port>]
OctaviaTenantLogTargets: <ip-address>:<port>[, <ip-address>:<port>]
```

7. 배포 명령을 실행하고 수정된 코어 heat 템플릿, 환경 파일 및 사용자 지정 환경 파일을 포함합니다.



### 중요

후속 환경 파일에 정의된 매개변수와 리소스가 우선하므로 환경 파일의 순서가 중요합니다.

```
$ 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

- **OctaviaAdminLogTargets** 또는 **OctaviaTenantLogTargets** 를 사용하여 특정 끝점을 지정하지 않는 한 amphorae 오프로드는 다른 RHOSP 로그(/var/log/containers/octavia-amphorae/)와 동일한 위치에 RHOSP 컨트롤러에 로그됩니다.
- 다음 로그 파일이 있는지에 적합한 위치를 확인합니다.
  - **Octavia-amphora.log**-- 관리 로그의 로그 파일입니다.
  - **Octavia-tenant-traffic.log**-- 테넌트 트래픽 흐름 로그의 로그 파일입니다.

## 추가 리소스

- [5.5절. "로드 밸런싱 서비스 인스턴스 로깅을 위한 heat 매개변수"](#)
- Red Hat OpenStack Platform 배포 가이드의 [환경 파일 사용자 지정](#)
- Red Hat OpenStack Platform 배포 가이드 사용자 지정 [예](#) 오버클라우드 생성에 환경 파일 포함
- [5.6절. "서비스 인스턴스 테넌트 흐름 로그 형식 로드 밸런싱"](#)

## 5.3. 로드 밸런싱 서비스 인스턴스 테넌트 흐름 로깅 비활성화

관리 로그 오프로드를 활성화하면 로드 밸런싱 서비스 인스턴스(amphorae)에 대한 테넌트 흐름 로그 오프로드가 자동으로 활성화됩니다.

관리 로그 오프로드를 활성화하고 테넌트 흐름 로깅을 비활성화하려면 **OctaviaConnectionLogging** 매개변수를 **false** 로 설정해야 합니다.

**OctaviaConnectionLogging** 매개변수가 **false** 인 경우 amphorae는 amphorae 내의 디스크에 테넌트 흐름 로그를 작성하지 않으며 다른 곳에서 수신하는 syslog 수신자로 로그를 오프로드하지 않습니다.

## 절차

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

```
$ source ~/stackrc
```

3. amphora 로깅이 구성된 YAML 사용자 지정 환경 파일을 찾습니다.

### 예제

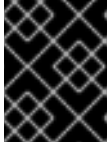
```
$ grep -rl OctaviaLogOffload /home/stack/templates/
```

4. 사용자 지정 환경 파일에서 **parameter\_defaults** 에서 **OctaviaConnectionLogging** 을 **false** 로 설정합니다.

## 예제

```
parameter_defaults:
  OctaviaLogOffload: true
  OctaviaConnectionLogging: false
  ...
```

5. 배포 명령을 실행하고 코어 heat 템플릿, 환경 파일 및 **OctaviaConnectionLogging** 을 **true** 로 설정한 사용자 지정 환경 파일을 포함합니다.



## 중요

후속 환경 파일에 정의된 매개변수와 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

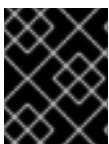
- **OctaviaAdminLogTargets** 또는 **OctaviaTenantLogTargets** 를 사용하여 특정 끝점을 지정하지 않는 한 amphorae 오프로드는 다른 RHOSP 로그(/var/log/containers/octavia-amphorae/)와 동일한 위치에 RHOSP 컨트롤러에 로그됩니다.
- **octavia-tenant-traffic.log** 가 없는 경우 적절한 위치를 확인합니다.

## 추가 리소스

- Red Hat OpenStack Platform 배포 가이드의 [환경 파일 사용자 지정](#)
- Red Hat OpenStack Platform 배포 가이드 사용자 지정 [에](#) 오버클라우드 생성에 환경 파일 포함

## 5.4. 로드 밸런싱 서비스 인스턴스 로컬 로그 스토리지 비활성화

관리 및 테넌트 흐름 로그를 오프로드하도록 로드 밸런싱 서비스 인스턴스(amphorae)를 구성하더라도 amphorae 내부의 디스크에 이러한 로그를 계속 작성합니다. 로드 밸런서의 성능을 개선하기 위해 로컬에서 로깅을 중지할 수 있습니다.



## 중요

로컬로 로깅을 비활성화하면 커널, 시스템, 보안 로깅을 포함하여 amphora의 모든 로그 스토리지도 비활성화합니다.



## 참고

로컬 로그 스토리지를 비활성화하고 **OctaviaLogOffload** 매개변수가 false로 설정된 경우 로드 밸런싱 성능을 개선하기 위해 **OctaviaConnectionLogging** 을 false로 설정해야 합니다.

## 절차

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

```
$ source ~/stackrc
```

3. 사용자 지정 YAML 환경 파일을 생성합니다.

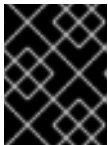
### 예제

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

4. **parameter\_defaults** 아래의 환경 파일에서 **OctaviaDisableLocalLogStorage** 를 **true** 로 설정합니다.

```
parameter_defaults:
  OctaviaDisableLocalLogStorage: true
...
```

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



### 중요

후속 환경 파일에 정의된 매개변수와 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

- amphora 인스턴스에서 로그 파일이 기록된 위치를 확인하고 새 로그 파일이 기록되지 않는지 확인합니다.

## 추가 리소스

- Red Hat OpenStack Platform 배포 가이드의 [환경 파일 사용자 지정](#)
- Red Hat OpenStack Platform 배포 가이드 사용자 지정 [에 오버클라우드 생성에 환경 파일 포함](#)

## 5.5. 로드 밸런싱 서비스 인스턴스 로깅을 위한 HEAT 매개변수

로드 밸런싱 서비스 인스턴스(amphora) 로깅을 구성하려면 로깅을 제어하고 **openstack overcloud deploy** 명령을 실행하는 하나 이상의 Orchestration 서비스(heat) 매개변수의 값을 설정합니다.

amphora 로깅에 대한 이러한 heat 매개변수를 사용하면 로그 오프로드를 켜거나, 사용자 지정 끝점을 통해 로그를 오프로드하도록 정의하며, 로그의 syslog 기능 값을 설정하는 등의 기능을 제어할 수 있습니다.

표 5.1. 모든 로그에 대한 heat 매개변수

| 매개변수                                 | Default      | 설명                                                                                                                                                                                        |
|--------------------------------------|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>OctaviaLogOffload</b>             | <b>false</b> | <b>true</b> 인 경우 인스턴스가 로그를 오프로드합니다. 엔드포인트를 지정하지 않으면 기본적으로 인스턴스가 다른 RHOSP 로그와 동일한 위치로 로그를 오프로드합니다 (/var/log/containers/octavia-amphorae/).                                                 |
| <b>OctaviaDisableLocalLogStorage</b> | <b>false</b> | <b>true</b> 인 경우 인스턴스는 인스턴스 호스트 파일 시스템에 로그를 저장하지 않습니다. 여기에는 모든 커널, 시스템 및 보안 로그가 포함됩니다.                                                                                                    |
| <b>OctaviaForwardAllLogs</b>         | <b>false</b> | <p><b>true</b> 인 경우 인스턴스는 cron 및 커널 로그와 같은 로드되지 않은 분산 관련 로그를 포함하여 모든 로그 메시지를 관리 로그 끝점에 전달합니다.</p> <p>인스턴스가 <b>OctaviaForwardAllLogs</b> 를 인식하려면 <b>OctaviaLogOffload</b> 도 활성화해야 합니다.</p> |

표 5.2. 관리자 로깅을 위한 heat 매개변수

| 매개변수                           | Default  | 설명                                                                                                                                                                                                                                                                                                                        |
|--------------------------------|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>OctaviaAdminLogTargets</b>  | 값이 없습니다. | <p>관리 로그 메시지를 수신하기 위한 쉘표로 구분된 syslog 끝점(&lt;host&gt;:&lt;port&gt;) 목록입니다.</p> <p>이러한 엔드포인트는 지정된 포트에서 로그 메시지를 수신 대기하는 프로세스를 실행하는 컨테이너, VM 또는 물리적 호스트일 수 있습니다.</p> <p><b>OctaviaAdminLogTargets</b> 가 없으면 인스턴스가 RHOSP director에서 정의한 컨테이너 세트에서 다른 RHOSP 로그(/var/log/containers/octavia-amphorae/)와 동일한 위치로 로그를 오프로드합니다.</p> |
| <b>OctaviaAdminLogFacility</b> | <b>1</b> | 관리 로그 메시지에 사용할 syslog "LOG_LOCAL" 기능인 0에서 7 사이의 번호입니다.                                                                                                                                                                                                                                                                    |

표 5.3. 테넌트 흐름 로깅을 위한 heat 매개변수

| 매개변수                            | Default     | 설명                                                                                                                                                                                                                                                           |
|---------------------------------|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>OctaviaConnectionLogging</b> | <b>true</b> | <p><b>true</b> 인 경우 테넌트 연결 흐름이 기록됩니다.</p> <p><b>OctaviaConnectionLogging</b> 이 false이면 amphorae는 <b>OctaviaLogOffload</b> 설정에 관계없이 로깅 테넌트 연결을 중지합니다. <b>OctaviaConnectionLogging</b> 은 로컬 테넌트 흐름 로그 스토리지를 비활성화하고, 로그 오프로드가 활성화된 경우 테넌트 흐름 로그를 전달하지 않습니다.</p> |

| 매개변수                            | Default                                                                                                                                                                 | 설명                                                                                                                                                                                                                                                                                                                                 |
|---------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>OctaviaTenantLogTargets</b>  | 값이 없습니다.                                                                                                                                                                | <p>테넌트 트래픽 흐름 로그 메시지를 수신하기 위한 쉘표로 구분된 syslog 끝점(&lt;host&gt;:&lt;port&gt;) 목록입니다.</p> <p>이러한 엔드포인트는 지정된 포트에서 로그 메시지를 수신 대기하는 프로세스를 실행하는 컨테이너, VM 또는 물리적 호스트일 수 있습니다.</p> <p><b>OctaviaTenantLogTargets</b> 가 없으면 인스턴스가 RHOSP director에서 정의한 컨테이너 세트에서 다른 RHOSP 로그(/var/log/containers/octavia-amphorae/)와 동일한 위치로 로그를 오프로드합니다.</p> |
| <b>OctaviaTenantLogFacility</b> | 0                                                                                                                                                                       | 테넌트 트래픽 흐름 로그 메시지에 사용할 syslog "LOG_LOCAL" 기능인 0에서 7 사이의 번호입니다.                                                                                                                                                                                                                                                                     |
| <b>OctaviaUserLogFormat</b>     | <pre>"{{ '{' }} project_id {{ '}}' {{ '}}' {{ '}}' {{ '}}' {{ '}}' {{ '}}' {{ '}}' %ci %{+Q}r %ST %B %U %B %B %U % [ssl_c_verify] %{+Q} [ssl_c_s_dn] %b %Tt %t %"</pre> | <p>테넌트 트래픽 흐름 로그의 형식입니다.</p> <p>영숫자는 특정 octavia 필드를 나타내며 중괄호({})는 대체 변수입니다.</p>                                                                                                                                                                                                                                                    |

## 추가 리소스

- Red Hat OpenStack Platform 배포 가이드 사용자 지정 [에](#) 오버클라우드 생성에 환경 파일 포함

## 5.6. 서비스 인스턴스 테넌트 흐름 로그 형식 로드 밸런싱

로드 밸런싱 서비스 인스턴스(amphorae)에 대한 테넌트 흐름 로그는 HAProxy 로그 형식입니다. 두 가지 예외는 amphora 공급자 드라이버에서 값을 제공하는 **project\_id** 및 **lb\_id** 변수입니다.

### 샘플

다음은 syslog 수신자로서 rsyslog가 포함된 샘플 로그 항목입니다.

```
Jun 12 00:44:13 amphora-3e0239c3-5496-4215-b76c-6abbe18de573 haproxy[1644]:
5408b89aa45b48c69a53dca1aaec58db fd8f23df-960b-4b12-ba62-2b1dff661ee7 261ecfc2-9e8e-
4bba-9ec2-3c903459a895 172.24.4.1 41152 12/Jun/2019:00:44:13.030 "GET / HTTP/1.1" 200 76 73
- "" e37e0e04-68a3-435b-876c-cffe4f2138a4 6f2720b3-27dc-4496-9039-1aaf2fee105 4 --
```

### 참고

- 하이픈(-)은 알 수 없거나 연결에 적용되지 않는 모든 필드를 나타냅니다.

- 이전 샘플 로그 항목의 접두사는 rsyslog 수신자에서 시작하며 amphora의 syslog 메시지의 일부가 아닙니다.

```
Jun 12 00:44:13 amphora-3e0239c3-5496-4215-b76c-6abbe18de573 haproxy[1644]:"
```

## Default

기본 amphora 테넌트 흐름 로그 형식은 다음과 같습니다.

```
`"{{ '{{' }} project_id {{ '{{' }} }} {{ '{{' }} lb_id {{ '{{' }} }} %f %ci %cp %t %{+Q}r %ST %B %U %[ssl_c_verify]
%{+Q}[ssl_c_s_dn] %b %s %Tt %tsc"
```

형식에 대한 설명은 다음 표를 참조하십시오.

표 5.4. 테넌트 흐름 로그 형식 변수 정의에 대한 데이터 변수입니다.

| Variable              | 유형     | 필드 이름                                        |
|-----------------------|--------|----------------------------------------------|
| <b>{{project_id}}</b> | UUID   | 프로젝트 ID( amphora 공급자 드라이버의 하위 변수)            |
| <b>{{lb_id}}</b>      | UUID   | 로드 밸런서 ID( amphora 공급자 드라이버의 하위 변수)          |
| <b>%f</b>             | string | <b>frontend_name</b>                         |
| <b>%ci</b>            | IP 주소  | <b>client_ip</b>                             |
| <b>%cp</b>            | 숫자     | <b>client_port</b>                           |
| <b>%t</b>             | date   | <b>date_time</b>                             |
| <b>%ST</b>            | 숫자     | <b>status_code</b>                           |
| <b>%B</b>             | 숫자     | <b>bytes_read</b>                            |
| <b>%U</b>             | 숫자     | <b>bytes_uploaded</b>                        |
| <b>%ssl_c_verify</b>  | 부울     | <b>client_certificate_verify</b> (0 또는 1)    |
| <b>%ssl_c_s_dn</b>    | string | <b>client_certificate_distinguished_name</b> |
| <b>%b</b>             | string | <b>pool_id</b>                               |
| <b>%s</b>             | string | <b>member_id</b>                             |
| <b>%Tt</b>            | 숫자     | <b>processing_time</b> (밀리초)                 |
| <b>%tsc</b>           | string | <b>termination_state</b> (쿠키 상태 포함)          |

추가 리소스

- *HAProxy* 문서의 [사용자 정의 로그 형식](#)

## 6장. 로드 밸런싱 서비스 플레이버 구성

로드 밸런싱 서비스(octavia) *플레이버*는 생성한 공급자 구성 옵션 세트입니다. 사용자가 로드 밸런서를 요청할 때 정의된 플레이버 중 하나를 사용하여 로드 밸런서를 빌드할 수 있습니다. 각 로드 밸런싱 공급자 드라이버에 대한 플레이버를 정의하여 각 공급자의 고유한 기능을 노출합니다.

새 로드 밸런싱 서비스 플레이버를 생성하려면 다음을 수행합니다.

1. 플레이버에서 구성할 로드 밸런싱 공급자의 기능을 결정합니다.
2. 선택한 플레이버 기능을 사용하여 플레이버 프로필을 만듭니다.
3. 플레이버를 만듭니다.
  - 6.1절. "로드 밸런싱 서비스 공급자 기능 나열"
  - 6.2절. "플레이버 프로필 정의"
  - 6.3절. "로드 밸런싱 서비스 플레이버 생성"

### 6.1. 로드 밸런싱 서비스 공급자 기능 나열

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

#### 절차

1. RHOSP 관리자 역할을 사용하여 오버클라우드에 액세스할 수 있는 인증 정보 파일을 가져옵니다.

#### 예제

```
$ source ~/my_overcloudrc
```

2. 각 드라이버의 기능을 나열합니다.

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

<provider>를 공급자의 이름 또는 UUID로 바꿉니다.

#### 예제

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

명령 출력에는 공급자가 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.                   |
| ...           | ...                                         |
+-----+-----+
```

3. 생성 중인 플레이버에 포함할 기능의 이름을 확인합니다.

## 추가 리소스

- [6.2절. "플레이버 프로파일 정의"](#)
- [명령줄 인터페이스 참조의 LoadBalancer 공급자 기능 목록](#)

## 6.2. 플레이버 프로파일 정의

로드 밸런싱 서비스(octavia) 플레이버 프로파일에는 공급자 드라이버 이름과 기능 목록이 포함되어 있습니다. 플레이버 프로파일을 사용하여 사용자가 로드 밸런서를 생성하도록 지정하는 플레이버를 만듭니다.

### 사전 요구 사항

- 어떤 로드 밸런싱 공급자와 플레이버 프로파일에 포함할 기능을 알아야 합니다.

### 절차

1. RHOSP 관리자 역할을 사용하여 오버클라우드에 액세스할 수 있는 인증 정보 파일을 가져옵니다.

#### 예제

```
$ source ~/my_overcloudrc
```

2. 플레이버 프로파일을 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 프로파일이 생성됩니다. 이 프로파일이 플레이버에 지정되면 플레이버를 사용하여 사용자가 생성하는 로드 밸런서는 단일 amphora 로드 밸런서입니다.

### 검증

- 플레이버 프로필을 생성할 때 Load-balancing 서비스는 플레이버 값을 공급자와 함께 검증하여 공급자가 지정한 기능을 지원할 수 있는지 확인합니다.

#### 추가 리소스

- 6.3절. "로드 밸런싱 서비스 플레이버 생성"
- 명령줄 인터페이스 참조에서 LoadBalancer [flavorprofile create](#)

### 6.3. 로드 밸런싱 서비스 플레이버 생성

플레이버 프로필을 사용하여 로드 밸런싱 서비스(octavia)에 대한 사용자용 플레이버를 생성합니다. 플레이버에 할당된 이름은 사용자가 로드 밸런서를 생성할 때 지정하는 값입니다.

#### 사전 요구 사항

- 플레이버 프로필을 생성해야 합니다.

#### 절차

1. RHOSP 관리자 역할을 사용하여 오버클라우드에 액세스할 수 있는 인증 정보 파일을 가져옵니다.

##### 예제

```
$ source ~/my_overcloudrc
```

2. 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 정의되어 있습니다. 사용자가 이 플레이버를 지정하면 하나의 로드 밸런싱 서비스 인스턴스(amphora)를 사용하고 가용성이 높지 않는 로드 밸런서를 생성합니다.



### 참고

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

### 추가 리소스

- [6.2절. "플레이버 프로필 정의"](#)
- [명령줄 인터페이스 참조에서 LoadBalancer 플레이버 생성](#)

## 7장. 로드 밸런싱 서비스 모니터링

로드 밸런싱을 유지하려면 로드 밸런서 관리 네트워크를 사용하여 로드 밸런싱 상태 모니터를 생성, 수정 및 삭제할 수 있습니다.

- 7.1절. "로드 밸런싱 관리 네트워크"
- 7.2절. "서비스 인스턴스 로드 밸런싱 모니터링"
- 7.3절. "서비스 풀 멤버 모니터링 로드 밸런싱"
- 7.4절. "로드 밸런서 프로비저닝 상태 모니터링"
- 7.5절. "로드 밸런서 기능 모니터링"
- 7.6절. "로드 밸런싱 서비스 상태 모니터 정보"
- 7.7절. "로드 밸런싱 서비스 상태 모니터 생성"
- 7.8절. "로드 밸런싱 서비스 상태 모니터 수정"
- 7.9절. "로드 밸런싱 서비스 상태 모니터 삭제"
- 7.10절. "서비스 HTTP 상태 모니터 로드 밸런싱 모범 사례"

### 7.1. 로드 밸런싱 관리 네트워크

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)는 로드 밸런싱 관리 네트워크라는 프로젝트 네트워크를 통해 로드 밸런서를 모니터링합니다. 로드 밸런싱 서비스를 실행하는 호스트에는 로드 밸런싱 관리 네트워크에 연결하는 인터페이스가 있어야 합니다. 지원되는 인터페이스 구성은 Open Virtual Network 메커니즘 드라이버(ML2/OVN) 또는 ML2/OVS(Open vSwitch 메커니즘 드라이버)를 사용하여 neutron Modular Layer 2 플러그인에서 작동합니다. 다른 메커니즘 드라이버와의 인터페이스 사용은 테스트되지 않았습니다.

배포 시 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 인터페이스의 이름은 기본적으로 **o-hm0**입니다. 로드 밸런싱 서비스 호스트의 표준 인터페이스 구성 파일을 통해 정의됩니다. RHOSP director는 배포 중에 각 로드 밸런싱 서비스 호스트에 대한 네트워킹 서비스 포트와 인터페이스를 자동으로 설정합니다. 포트 정보 및 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 Networking 서비스 포트 ID를 사용하여 OVS 포트에 추가 데이터를 등록합니다. 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 9443에서 VM 인스턴스(amphorae)와

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

## 7.2. 서비스 인스턴스 로드 밸런싱 모니터링

로드 밸런싱 서비스(octavia)는 로드 밸런싱 인스턴스(amphorae)를 모니터링하고 amphorae 오작동이 발생하는 경우 페일오버 및 교체를 시작합니다. 장애 조치가 발생할 때마다 로드 밸런싱 서비스는 `/var/log/containers/octavia`의 컨트롤러에 해당 상태 관리자 로그에 장애 조치를 기록합니다.

로그 분석을 사용하여 장애 조치(failover) 추세를 모니터링하여 문제를 조기에 해결합니다. Networking 서비스(neutron) 연결 문제, 서비스 거부 공격 및 Compute 서비스(nova) 오작동과 같은 문제로 인해 로드 밸런서에 대한 페일오버 속도가 높아지는 경우가 많습니다.

## 7.3. 서비스 풀 멤버 모니터링 로드 밸런싱

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

## 7.4. 로드 밸런서 프로비저닝 상태 모니터링

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

로드 밸런서 오브젝트의 프로비저닝 상태는 생성, 업데이트 및 삭제 요청을 연결하고 성공적으로 프로비저닝하는 컨트롤 플레인의 기능을 반영합니다. 로드 밸런서 오브젝트의 작동 상태는 로드 밸런서의 현재 기능에 대해 보고합니다.

예를 들어 로드 밸런서의 프로비저닝 상태는 **ERROR**이지만 **ONLINE**의 운영 상태가 될 수 있습니다. 이는 로드 밸런서 구성에 대한 마지막으로 요청된 업데이트가 성공적으로 완료되지 못하도록 차단된 Networking(neutron) 실패로 인해 발생할 수 있습니다. 이 경우 로드 밸런서는 로드 밸런서를 통해 트래픽을 계속 처리하지만 아직 최신 구성 업데이트를 적용하지 못할 수 있습니다.

## 7.5. 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 문제와 같은 로드 밸런서의 기능에 영향을 미칠 수 있는 로드 밸런싱 서비스(octavia) 외부에 오류가 있는지 여부를 나타냅니다.

## 7.6. 로드 밸런싱 서비스 상태 모니터 정보

로드 밸런싱 서비스(octavia) 상태 모니터는 각 백엔드 구성원 서버에서 주기적으로 상태 점검을 수행하여 실패한 서버를 미리 탐지하고 일시적으로 풀에서 가져오는 프로세스입니다.

상태 모니터가 실패한 서버를 감지하면 풀에서 서버를 제거하고 프로세스를 **ERROR**로 표시합니다. 서버를 수정하고 다시 작동하면 상태 모니터에서 멤버의 상태를 **ERROR**에서 **ONLINE**으로 자동 변경하고 트래픽을 다시 전달합니다.

항상 프로덕션 로드 밸런서에서 상태 모니터를 사용합니다. 상태 모니터가 없으면 실패한 서버가 풀에서 제거되지 않습니다. 이로 인해 웹 클라이언트에 대한 서비스가 중단될 수 있습니다.

여기에 간단히 설명 된 바와 같이 몇 가지 유형의 상태 모니터가 있습니다.

## HTTP

기본적으로 애플리케이션 서버의 / 경로를 확인합니다.

## HTTPS

HTTP 상태 모니터와 동일하게 작동하지만 TLS 백엔드 서버에서 작동합니다.

서버에서 클라이언트 인증서 검증을 수행하는 경우 HAProxy에 유효한 인증서가 없습니다. 이러한 경우 TLS-HELLO 상태 모니터링이 대안입니다.

## TLS-HELLO

백엔드 서버가 SSLv3-client hello 메시지에 응답하는지 확인합니다.

TLS-HELLO 상태 모니터는 상태 코드 또는 본문 콘텐츠와 같은 기타 상태 메트릭을 확인하지 않습니다.

## PING

정기 ICMP ping 요청을 백엔드 서버로 보냅니다.

이러한 상태 점검이 통과되도록 백엔드 서버를 구성해야 합니다.



### 중요

PING 상태 모니터는 멤버에 연결할 수 있는 경우에만 확인하고 ICMP 에코 요청에 응답합니다. PING 상태 모니터는 인스턴스에서 실행되는 애플리케이션이 정상인지 여부를 탐지하지 않습니다. ICMP 에코 요청이 유효한 상태 점검인 경우에만 PING 상태 모니터를 사용하십시오.

## TCP

백엔드 서버 프로토콜 포트에 대한 TCP 연결을 엽니다.

TCP 애플리케이션은 TCP 연결을 열고 TCP 핸드셰이크 후 데이터를 보내지 않고 연결을 종료합니다.

## UDP-CONNECT

기본 UDP 포트 연결을 수행합니다.

대상 Unreachable(ICMP 유형 3)이 멤버 서버에서 활성화되지 않거나 보안 규칙에 의해 차단된 경우 UDP-CONNECT 상태 모니터가 제대로 작동하지 않을 수 있습니다. 이러한 경우 멤버 서버는 실제로 다운되었을 때 **ONLINE** 의 작동 상태를 표시 할 수 있습니다.

## 7.7. 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 실패한 서버를 미리 감지하고 서버를 일시적으로 폴링합니다.

### 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 대기하는 시간(초)입니다.

#### --timeout

지정된 상태 점검이 완료될 때까지 대기하는 시간(초)입니다. 시간 제한은 항상 **delay** 보다 작아야 합니다.

#### --max-retries

백엔드 서버가 다운된 것으로 간주되기 전에 상태 점검 수를 실패해야 합니다. 또한 실패한 백엔드 서버가 다시 고려되려면 상태 점검 수를 전달해야 합니다.

- 또한 HTTP 상태 모니터 유형에는 기본적으로 설정된 다음 인수도 필요합니다.

#### --url-path

백엔드 서버에서 검색해야 하는 URL의 경로 부분입니다. 기본적으로 이 값은 **/** 입니다.

#### --http-method

**url\_path** 를 검색하는 데 사용되는 HTTP 메서드입니다. 기본적으로 이 값은 **GET** 입니다.

#### --expected-codes

OK 상태 점검을 나타내는 HTTP 상태 코드 목록입니다. 기본적으로 이 값은 **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 참조에서 LoadBalancer [healthmonitor](#) 생성

## 7.8. 로드 밸런싱 서비스 상태 모니터 수정

프로브를 멤버로 보내는 간격을 변경하려는 경우 로드 밸런싱 서비스(octavia) 상태 모니터의 구성을 수정할 수 있습니다. 연결 시간 초과 간격, 요청에 대한 HTTP 메서드 등을 변경할 수 있습니다.

#### 절차

1. 자격 증명 파일을 가져옵니다.

#### 예제

```
$ source ~/overcloudrc
```

2. 상태 모니터를 수정합니다(**my-health-monitor**).

이 예에서 사용자는 상태 모니터가 프로브를 멤버로 보내는 사이에 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 LoadBalancer healthmonitor 설정](#)
- [명령줄 인터페이스 참조에 LoadBalancer healthmonitor 표시](#)

## 7.9. 로드 밸런싱 서비스 상태 모니터 삭제

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

#### 작은 정보

상태 모니터를 삭제하는 대신 **openstack loadbalancer healthmonitor set --disable** 명령을 사용하여 비활성화합니다.

#### 절차

1. 자격 증명 파일을 가져옵니다.

#### 예제

```
$ source ~/overcloudrc
```

2. 상태 모니터를 삭제합니다(**my-health-monitor**).

#### 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 참조의 LoadBalancer healthmonitor 삭제](#)

## 7.10. 서비스 HTTP 상태 모니터 로드 밸런싱 모범 사례

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

- 상태 모니터 **url-path** 는 로드하기 위해 인증이 필요하지 않습니다.
- 기본적으로 상태 모니터 **url-path** 는 대체 **expected** 코드를 지정하지 않는 한 정상 서버를 나타내는 **HTTP 200 OK** 상태 코드를 반환합니다.
- 상태 점검은 애플리케이션이 정상이고 더 이상 필요하지 않은지 확인하기에 충분한 내부 검사를 수행합니다. 애플리케이션에 대해 다음 조건이 충족되었는지 확인합니다.
  - 필요한 데이터베이스 또는 기타 외부 스토리지 연결이 실행 중입니다.
  - 애플리케이션이 실행되는 서버에 대해 로드가 허용됩니다.
  - 귀하의 사이트는 유지 관리 모드에 있지 않습니다.
  - 애플리케이션과 관련된 테스트가 작동합니다.
- 상태 점검으로 생성된 페이지는 크기가 작아야 합니다.
  - 하위 초 간격으로 반환됩니다.
  - 애플리케이션 서버에 상당한 로드를 유발하지 않습니다.
- 상태 점검을 실행하는 코드에서 캐시된 데이터를 참조할 수 있지만 상태 점검에서 생성한 페이지는 캐시되지 않습니다.  
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 통합됩니다.
- 로드 밸런싱 서비스는 반환된 HTTP 상태 코드만 처리하고 상태 점검이 자주 실행되므로 **HEAD** 또는 **OPTIONS** HTTP 메서드를 사용하여 전체 페이지 처리를 건너뛸 수 있습니다.

## 8장. 비보안 HTTP 로드 밸런서 생성

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

- 8.1절. "상태 모니터를 사용하여 HTTP 로드 밸런서 생성"
- 8.2절. "부동 IP를 사용하는 HTTP 로드 밸런서 생성"
- 8.3절. "세션 지속성을 사용하여 HTTP 로드 밸런서 생성"

### 8.1. 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

- 인터넷에서 연결할 수 있는 공유 외부(공용) 서브넷입니다.

#### 절차

1. 자격 증명 파일을 가져옵니다.

##### 예제

```
$ source ~/overcloudrc
```

2. 공용 서브넷(**public\_subnet**)에 로드 밸런서(**lb1**)를 만듭니다.



##### 참고

controlPlane 내부의 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 --wait
```

3. 포트(**80**)에서 리스너(**listener1**)를 생성합니다.

##### 예제

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

4. 리스너 상태를 확인합니다.

##### 예제

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

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

5. 리스너 기본 풀(**pool1**)을 만듭니다.

#### 예제

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

6. 백엔드 서버에 연결하는 풀(**pool1**)에 상태 모니터(**healthmon1**)를 생성하고 경로(/)를 테스트합니다.

상태 점검이 권장되지만 필수는 아닙니다. 상태 모니터가 정의되지 않은 경우 멤버 서버는 **ONLINE** 으로 간주됩니다.

#### 예제

```
$ openstack loadbalancer healthmonitor create --name healthmon1 \
--delay 15 --max-retries 4 --timeout 10 --type HTTP --url-path / pool1
```

7. 프라이빗 서브넷(**private\_subnet**)에서 로드 밸런서 멤버(**192.0.2.10** 및 **192.0.2.11**)를 기본 풀에 추가합니다.

#### 예제

이 예에서 백엔드 서버 **192.0.2.10** 및 **192.0.2.11** 은 각각 **member1** 및 **member2** 로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 --subnet-id \
private_subnet --address 192.0.2.10 --protocol-port 80 pool1
```

```
$ openstack loadbalancer member create --name member2 --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 존재하고 제대로 작동하는 경우 각 멤버의 상태를 확인할 수 있습니다.

작업 멤버(member1)에는 **operating\_status** 에 대한 **ONLINE** 값이 있습니다.

#### 예제

```
$ openstack loadbalancer member show pool1 member1
```

## 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        | member1                             |
| 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 참조의 LoadBalancer [https://access.redhat.com/documentation/en-us/red\\_hat\\_openshift\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openshift_platform/17.1/html/command_line_interface_reference/loadbalancer)

## 8.2. 부동 IP를 사용하는 HTTP 로드 밸런서 생성

비보안 HTTP 애플리케이션에 대한 네트워크 트래픽을 관리하려면 유동 IP에 따라 VIP(가상 IP)를 사용하여 로드 밸런서를 생성합니다. 유동 IP를 사용하면 할당된 IP를 제어할 수 있으므로 로드 밸런서를 이동, 제거 또는 다시 생성해야 하는 경우 필요합니다. 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.



## 참고

유동 IP는 IPv6 네트워크에서 작동하지 않습니다.

## 사전 요구 사항

- 로드 밸런서 **VIP**와 함께 사용할 부동 **IP**입니다.
- 부동 **IP**에 사용할 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**)에 로드 밸런서(**lb1**)를 만듭니다.



### 참고

**controlPlane** 내부의 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 --wait
```

3. 2단계의 출력에서는 이후 단계에서 제공해야 하므로 **load\_balancer\_vip\_port\_id** 값을 기록합니다.

4. 포트(80)에서 리스너(listener1)를 생성합니다.

예제

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

5. 리스너 기본 풀(pool1)을 만듭니다.

예제

이 예제의 명령은 **TCP** 포트 **80**에서 비보안 **HTTP** 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷을 사용하는 **HTTP** 풀을 생성합니다.

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

6. 백엔드 서버에 연결하는 풀(pool1)에 상태 모니터(healthmon1)를 생성하고 경로(/)를 테스트합니다.

상태 점검이 권장되지만 필수는 아닙니다. 상태 모니터가 정의되지 않은 경우 멤버 서버는 **ONLINE** 으로 간주됩니다.

예제

```
$ openstack loadbalancer healthmonitor create --name healthmon1 \
--delay 15 --max-retries 4 --timeout 10 --type HTTP --url-path / pool1
```

7. 프라이빗 서브넷의 로드 밸런서 멤버(192.0.2.10 및 192.0.2.11)를 기본 풀에 추가합니다.

예제

이 예에서 백엔드 서버 **192.0.2.10** 및 **192.0.2.11** 은 각각 **member1** 및 **member2** 로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 --subnet-id \
private_subnet --address 192.0.2.10 --protocol-port 80 pool1
```

```
$ openstack loadbalancer member create --name member2 --subnet-id \
private_subnet --address 192.0.2.11 --protocol-port 80 pool1
```

8. 공유 외부 서브넷(공용)에 유동 IP 주소를 만듭니다.

예제

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

9. 8단계의 출력에서 **floating\_ip\_address** 값을 기록합니다. 이후 단계에서 제공해야 하기 때문입니다.

10. 이 유동 IP(**203.0.113.0**)를 로드 밸런서 **vip\_port\_id** (**69a85edd-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 밸런서에서 **HTTP** 트래픽 흐름을 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 존재하고 제대로 작동하는 경우 각 멤버의 상태를 확인할 수 있습니다.

작업 멤버(**member1**)에는 **operating\_status** 에 대한 **ONLINE** 값이 있습니다.

#### 예제

```
$ openstack loadbalancer member show pool1 member1
```

#### 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         | member1              |
| 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 참조의 LoadBalancer** [https://access.redhat.com/documentation/en-us/red\\_hat\\_openshift\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openshift_platform/17.1/html/command_line_interface_reference/loadbalancer)
- **명령줄 인터페이스 참조에서 floating**

### 8.3. 세션 지속성을 사용하여 HTTP 로드 밸런서 생성

비보안 HTTP 애플리케이션의 네트워크 트래픽을 관리하려면 세션 지속성을 추적하는 로드 밸런서를 생성할 수 있습니다. 이렇게 하면 요청이 들어오면 로드 밸런서에서 동일한 클라이언트의 후속 요청을 동일한 백엔드 서버로 보냅니다. 세션 지속성은 시간과 메모리를 절약하여 부하 분산을 최적화합니다.

#### 사전 요구 사항

- 인터넷에서 연결할 수 있는 공유 외부(공용) 서브넷입니다.
- 로드 밸런싱을 수행하는 네트워크 트래픽의 비보안 웹 애플리케이션에는 쿠키가 활성화되어 있습니다.

#### 절차

1.

자격 증명 파일을 가져옵니다.

예제

```
$ source ~/overcloudrc
```

2.

공용 서브넷(**public\_subnet**)에 로드 밸런서(**lb1**)를 만듭니다.



참고

**controlPlane** 내부의 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 --wait
```

3.

포트(**80**)에서 리스너(**listener1**)를 생성합니다.

예제

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

4.

쿠키에서 세션 지속성을 정의하는 리스너 기본 풀(pool1)을 생성합니다(PHPSESSIONID).

#### 예제

이 예제의 명령은 TCP 포트 80에서 비보안 HTTP 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷을 사용하는 HTTP 풀을 생성합니다.

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

5.

백엔드 서버에 연결하는 풀(pool1)에 상태 모니터(healthmon1)를 생성하고 경로(/)를 테스트합니다.

상태 점검이 권장되지만 필수는 아닙니다. 상태 모니터가 정의되지 않은 경우 멤버 서버는 **ONLINE** 으로 간주됩니다.

#### 예제

```
$ openstack loadbalancer healthmonitor create --name healthmon1 \
--delay 15 --max-retries 4 --timeout 10 --type HTTP --url-path / pool1
```

6.

프라이빗 서브넷(private\_subnet)에서 로드 밸런서 멤버(192.0.2.10 및 192.0.2.11)를 기본 풀에 추가합니다.

#### 예제

이 예에서 백엔드 서버 192.0.2.10 및 192.0.2.11 은 각각 member1 및 member2 로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 --subnet-id \
private_subnet --address 192.0.2.10 --protocol-port 80 pool1

$ openstack loadbalancer member create --name member2 --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 존재하고 제대로 작동하는 경우 각 멤버의 상태를 확인할 수 있습니다.

작업 멤버(member1)에는 **operating\_status** 에 대한 **ONLINE** 값이 있습니다.

예제

```
$ openstack loadbalancer member show pool1 member1
```

### 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        | member1                             |
| 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 참조의 LoadBalancer [https://access.redhat.com/documentation/en-us/red\\_hat\\_openshift\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openshift_platform/17.1/html/command_line_interface_reference/loadbalancer)

## 9장. 보안 HTTP 로드 밸런서 생성

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

- [9.1절. “종료되지 않은 HTTPS 로드 밸런서 정보”](#)
- [9.2절. “종료되지 않은 HTTPS 로드 밸런서 생성”](#)
- [9.3절. “TLS 종료 HTTPS 로드 밸런서 정보”](#)
- [9.4절. “TLS 종료 HTTPS 로드 밸런서 생성”](#)
- [9.5절. “SNI를 사용하여 TLS 종료 HTTPS 로드 밸런서 생성”](#)
- [9.6절. “HTTP/2 리스너를 사용하여 TLS 종료 로드 밸런서 생성”](#)
- [9.7절. “동일한 IP 및 백엔드에서 HTTP 및 TLS 종료 HTTPS 로드 밸런싱 생성”](#)

### 9.1. 종료되지 않은 HTTPS 로드 밸런서 정보

종료되지 않은 HTTPS 로드 밸런서는 일반 TCP 로드 밸런서와 같이 작동합니다. 로드 밸런서는 웹 클라이언트에서 HTTPS 연결이 웹 클라이언트로 종료되는 백엔드 서버로 원시 TCP 트래픽을 전달합니다. 종료되지 않은 HTTPS 로드 밸런서는 계층 7 기능과 같은 고급 로드 밸런서 기능을 지원하지 않지만 인증서와 키를 자체적으로 관리하여 로드 밸런서 리소스 사용률을 줄입니다.

### 9.2. 종료되지 않은 HTTPS 로드 밸런서 생성

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

사전 요구 사항

- 인터넷에서 연결할 수 있는 공유 외부(공용) 서브넷입니다.

## 절차

1. 자격 증명 파일을 가져옵니다.

### 예제

```
$ source ~/overcloudrc
```

2. 공용 서브넷(**public\_subnet**)에 로드 밸런서(**lb1**)를 만듭니다.



### 참고

**controlPlane** 내부의 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 --wait
```

3. 포트(**443**)에서 리스너(**listener1**)를 생성합니다.

### 예제

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

4.

리스너 기본 풀(pool1)을 만듭니다.

#### 예제

이 예제의 명령은 **TCP** 포트 **443**에서 **TLS** 암호화 웹 애플리케이션으로 구성된 **HTTPS** 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷을 사용하는 **HTTPS** 풀을 생성합니다.

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

5.

백엔드 서버에 연결하고 경로(/)를 테스트하는 풀(pool1) 유형(TLS-HELLO)에 상태 모니터(healthmon1)를 만듭니다.

상태 점검이 권장되지만 필수는 아닙니다. 상태 모니터가 정의되지 않은 경우 멤버 서버는 **ONLINE** 으로 간주됩니다.

#### 예제

```
$ openstack loadbalancer healthmonitor create --name healthmon1 \
--delay 15 --max-retries 4 --timeout 10 --type TLS-HELLO \
--url-path / pool1
```

6.

프라이빗 서브넷(private\_subnet)에서 로드 밸런서 멤버(192.0.2.10 및 192.0.2.11)를 기본 풀에 추가합니다.

#### 예제

이 예에서 백엔드 서버 **192.0.2.10** 및 **192.0.2.11** 은 각각 **member1** 및 **member2** 로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 --subnet-id \
private_subnet --address 192.0.2.10 --protocol-port 443 pool1
```

```
$ openstack loadbalancer member create --name member2 --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 존재하고 제대로 작동하는 경우 각 멤버의 상태를 확인할 수 있습니다.

## 예제

작업 멤버(member1)에는 **operating\_status** 에 대한 **ONLINE** 값이 있습니다.

```
$ openstack loadbalancer member show pool1 member1
```

## 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        | member1                             |
| 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

- [키 관리자 서비스 가이드를 사용하여 보안 관리](#)
- 명령줄 인터페이스 참조의 **LoadBalancer** [https://access.redhat.com/documentation/en-us/red\\_hat\\_openshift\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openshift_platform/17.1/html/command_line_interface_reference/loadbalancer)

## 9.3. TLS 종료 HTTPS 로드 밸런서 정보

TLS 종료 HTTPS 로드 밸런서가 구현되면 웹 클라이언트는 TLS(Transport Layer Security) 프로토콜을 통해 로드 밸런서와 통신합니다. 로드 밸런서는 TLS 세션을 종료하고 해독된 요청을 백엔드 서버로 전

달합니다. 로드 밸런서에서 **TLS** 세션을 종료하면 **CPU** 집약적인 암호화 작업을 로드 밸런서에 오프로드하고 로드 밸런서에서 계층 7 검사와 같은 고급 기능을 사용하도록 허용합니다.

#### 9.4. TLS 종료 HTTPS 로드 밸런서 생성

**TLS** 종료 **HTTPS** 로드 밸런서를 사용하면 **CPU** 집약적인 암호화 작업을 로드 밸런서에 오프로드하고 로드 밸런서에서 계층 7 검사와 같은 고급 기능을 사용하도록 허용합니다. 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.

##### 사전 요구 사항

- 인터넷에서 연결할 수 있는 공유 외부(공용) 서브넷입니다.
- **TLS** 공개 키 암호화는 다음과 같은 특성으로 구성됩니다.
  - **TLS** 인증서, 키 및 중간 인증서 체인은 로드 밸런서 **VIP** 주소에 할당된 **DNS** 이름의 외부 **CA**(인증 기관)에서 가져옵니다(예: **www.example.com**).
  - 인증서, 키 및 중간 인증서 체인은 현재 디렉터리에 별도의 파일에 있습니다.
  - 키와 인증서는 **PEM**으로 인코딩됩니다.
  - 중간 인증서 체인에는 **PEM** 인코딩 및 연결된 여러 인증서가 포함되어 있습니다.
- 키 관리자 서비스(**barbican**)를 사용하도록 로드 밸런싱 서비스(**octavia**)를 구성해야 합니다. 자세한 내용은 [키 관리자 서비스 가이드를 사용하여 시크릿](#) 관리를 참조하십시오.

##### 프로세스

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



## 참고

**controlPlane** 내부의 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

암호가 **PKCS12** 파일을 보호하면 다음 절차가 작동하지 않습니다.

2.

자격 증명 파일을 가져옵니다.

## 예제

```
$ source ~/overcloudrc
```

3.

**Key Manager** 서비스를 사용하여 **PKCS12** 파일의 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**)에 로드 밸런서(**lb1**)를 만듭니다.

예제

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

5.

**TERMINATED\_HTTPS** 리스너(**listener1**)를 생성하고 시크릿 리소스를 리스너의 기본 **TLS** 컨테이너로 참조합니다.

예제

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

6.

풀(**pool1**)을 만들고 리스너의 기본 풀로 설정합니다.

예제

이 예제의 명령은 **TCP** 포트 **80**에서 비보안 **HTTP** 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷을 사용하는 **HTTP** 풀을 생성합니다.

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

7.

백엔드 서버에 연결하는 풀(**pool1**)에 상태 모니터(**healthmon1**)를 생성하고 경로를 테스트합니다.

상태 점검이 권장되지만 필수는 아닙니다. 상태 모니터가 정의되지 않은 경우 멤버 서버는 **ONLINE** 으로 간주됩니다.

#### 예제

```
$ openstack loadbalancer healthmonitor create --name healthmon1 \
--delay 15 --max-retries 4 --timeout 10 --type HTTP --url-path / pool1
```

8.

프라이빗 서브넷(**private\_subnet**)에서 비보안 **HTTP** 백엔드 서버( **192.0.2.10** 및 **192.0.2.11**)를 풀에 추가합니다.

#### 예제

이 예에서 백엔드 서버 **192.0.2.10** 및 **192.0.2.11** 은 각각 **member1** 및 **member2** 로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 --subnet-id \
private_subnet --address 192.0.2.10 --protocol-port 443 pool1
```

```
$ openstack loadbalancer member create --name member2 --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 존재하고 제대로 작동하는 경우 각 멤버의 상태를 확인할 수 있습니다.

예제

```
$ openstack loadbalancer member show pool1 member1
```

작업 멤버(member1)에는 **operating\_status** 에 대한 **ONLINE** 값이 있습니다.

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         | member1             |
| 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

- [키 관리자 서비스 가이드를 사용하여 보안 관리](#)
- 명령줄 인터페이스 참조의 **LoadBalancer** [https://access.redhat.com/documentation/en-us/red\\_hat\\_openshift\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openshift_platform/17.1/html/command_line_interface_reference/loadbalancer)

### 9.5. SNI를 사용하여 TLS 종료 HTTPS 로드 밸런서 생성

**SNI(Server Name Indication)** 기술을 사용하는 **TLS 종료 HTTPS** 로드 밸런서의 경우 단일 리스너는 여러 **TLS** 인증서를 포함하고 로드 밸런서에서 공유 **IP**를 사용할 때 제공할 인증서를 확인할 수 있습니다. 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.

#### 사전 요구 사항

- 인터넷에서 연결할 수 있는 공유 외부(공용) 서브넷입니다.
- **TLS** 공개 키 암호화는 다음과 같은 특성으로 구성됩니다.
  - 로드 밸런서 **VIP** 주소에 할당된 **DNS** 이름의 외부 인증 기관(**CA**)에서 여러 **TLS** 인증서, 키 및 중간 인증서 체인을 가져왔습니다(예: **www.example.com** 및 **www2.example.com**).

- 키와 인증서는 **PEM**으로 인코딩됩니다.

- 키 관리자 서비스(**barbican**)를 사용하도록 로드 밸런싱 서비스(**octavia**)를 구성해야 합니다. 자세한 내용은 [키 관리자 서비스 가이드를 사용하여 시크릿](#) 관리를 참조하십시오.

## 프로세스

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

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



### 참고

**controlPlane** 내부의 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

$ 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.

키 관리자 서비스를 사용하여 **PKCS12** 파일의 시크릿 리소스(**tls\_secret1** 및 **tls\_secret2**)를 생성합니다.

예제

```
$ 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**)에 로드 밸런서(**lb1**)를 만듭니다.

예제

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

5.

**TERMINATED\_HTTPS** 리스너(**listener1**)를 생성하고 **SNI**를 사용하여 보안 리소스를 모두 참조합니다.

(참고로 **tls\_secret1** 을 리스너의 기본 **TLS** 컨테이너로 참조합니다.)

예제

```
$ openstack loadbalancer listener create --name listener1 \
```

```
--protocol-port 443 --protocol TERMINATED_HTTPS \
--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
```

6.

풀(pool1)을 만들고 리스너의 기본 풀로 설정합니다.

예제

이 예제의 명령은 **TCP** 포트 **80**에서 비보안 **HTTP** 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷을 사용하는 **HTTP** 풀을 생성합니다.

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

7.

백엔드 서버에 연결하는 풀(pool1)에 상태 모니터(healthmon1)를 생성하고 경로(/)를 테스트합니다.

상태 점검이 권장되지만 필수는 아닙니다. 상태 모니터가 정의되지 않은 경우 멤버 서버는 **ONLINE** 으로 간주됩니다.

예제

```
$ openstack loadbalancer healthmonitor create --name healthmon1 \
--delay 15 --max-retries 4 --timeout 10 --type HTTP --url-path / pool1
```

8.

프라이빗 서브넷(private\_subnet)에서 비보안 **HTTP** 백엔드 서버( 192.0.2.10 및 192.0.2.11)를 풀에 추가합니다.

예제

이 예에서 백엔드 서버 **192.0.2.10** 및 **192.0.2.11** 은 각각 **member1** 및 **member2** 로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 --subnet-id \
private_subnet --address 192.0.2.10 --protocol-port 443 pool1
```

```
$ openstack loadbalancer member create --name member2 --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 존재하고 제대로 작동하는 경우 각 멤버의 상태를 확인할 수 있습니다.

예제

```
$ openstack loadbalancer member show pool1 member1
```

샘플 출력

작업 멤버(member1)에는 **operating\_status** 에 대한 **ONLINE** 값이 있습니다.

```
+-----+-----+
| 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        | member1                             |
| 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

- [키 관리자 서비스 가이드를 사용하여 보안 관리](#)
- 명령줄 인터페이스 참조의 **LoadBalancer** [https://access.redhat.com/documentation/en-us/red\\_hat\\_openshift\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openshift_platform/17.1/html/command_line_interface_reference/loadbalancer)

## 9.6. HTTP/2 리스너를 사용하여 TLS 종료 로드 밸런서 생성

**TLS 종료 HTTPS** 로드 밸런서를 사용하면 **CPU** 집약적인 암호화 작업을 로드 밸런서에 오프로드하고 로드 밸런서에서 계층 7 검사와 같은 고급 기능을 사용하도록 허용합니다. **HTTP/2** 리스너를 추가하면 **HTTP/2** 프로토콜을 활용하여 페이지를 더 빠르게 로드하여 성능을 향상시킬 수 있습니다. 로드 밸런서는 **ALPN(Application-Layer Protocol Negotiation)** **TLS** 확장을 사용하여 **HTTP/2**를 클라이언트와 협상합니다.

로드 밸런싱 서비스(**octavia**)는 엔드 투 엔드 **HTTP/2** 트래픽을 지원하므로 요청이 로드 밸런서에서 응답이 반환될 때까지 요청이 리스너에 도달하는 지점에서 **HAProxy**에서 **HTTP2** 트래픽을 번역하지 않습니다. 엔드 투 엔드 **HTTP/2** 트래픽을 달성하려면 백엔드 재암호화가 있는 **HTTP** 풀: **HTTPS** 트래픽을 위해 구성된 보안 포트 및 웹 애플리케이션에서 수신 대기하는 풀 멤버가 있어야 합니다.

백엔드 재암호화 없이 **HTTP/2** 트래픽을 **HTTP** 풀로 보낼 수 있습니다. 이 경우 **HAProxy**는 풀에 도달하기 전에 트래픽을 변환하고, 응답이 로드 밸런서에서 반환되기 전에 **HTTP/2**로 다시 변환됩니다.

백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.



### 참고

현재 로드 밸런싱 서비스는 **HTTP/2** 리스너를 사용하는 **TLS** 종료 로드 밸런서에 대한 상태 모니터링을 지원하지 않습니다.

### 사전 요구 사항

- **TLS** 공개 키 암호화는 다음과 같은 특성으로 구성됩니다.
  - **TLS** 인증서, 키 및 중간 인증서 체인은 로드 밸런서 **VIP** 주소에 할당된 **DNS** 이름의 외부 **CA**(인증 기관)에서 가져옵니다(예: **www.example.com**).
  - 인증서, 키 및 중간 인증서 체인은 현재 디렉터리에 별도의 파일에 있습니다.
  - 키와 인증서는 **PEM**으로 인코딩됩니다.
  - 중간 인증서 체인에는 **PEM** 인코딩 및 연결된 여러 인증서가 포함되어 있습니다.

•

키 관리자 서비스(**barbican**)를 사용하도록 로드 밸런싱 서비스(**octavia**)를 구성해야 합니다. 자세한 내용은 [키 관리자 서비스 가이드를 사용하여 시크릿](#) 관리를 참조하십시오.

## 절차

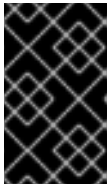
1.

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



### 참고

**controlPlane** 내부의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.



### 중요

**PKCS12** 파일을 생성할 때 암호를 사용하지 마십시오.

## 예제

이 예에서는 **PKCS12** 파일이 암호 없이 생성됩니다.

```
$ 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** 파일의 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**)에 로드 밸런서(**lb1**)를 만듭니다.

## 예제

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

5.

**TERMINATED\_HTTPS** 리스너(**listener1**)를 생성하고 다음을 수행합니다.

- 시크릿 리소스(**tls\_secret1**)를 리스너의 기본 **TLS** 컨테이너로 참조합니다.
- **ALPN** 프로토콜(**h2**)을 설정합니다.
- 클라이언트가 **HTTP/2(http/1.1)**를 지원하지 않는 경우 대체 프로토콜을 설정합니다.

## 예제

```
$ openstack loadbalancer listener create --name listener1 \
--protocol-port 443 --protocol TERMINATED_HTTPS --alpn-protocol h2 \
--alpn-protocol http/1.1 --default-tls-container=\
$(openstack secret list | awk ' / tls_secret1 / {print $2}') lb1
```

6. 풀(pool1)을 만들고 리스너의 기본 풀로 설정합니다.

#### 예제

이 예제의 명령은 **TCP** 포트 **80**에서 웹 애플리케이션으로 구성된 **HTTP** 애플리케이션을 호스팅하는 백엔드 서버를 포함하는 **HTTP** 풀을 생성합니다.

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

7. 백엔드 서버에 연결하는 풀(pool1)에 상태 모니터(healthmon1) 유형을 만듭니다.

상태 점검이 권장되지만 필수는 아닙니다. 상태 모니터가 정의되지 않은 경우 멤버 서버는 **ONLINE** 으로 간주됩니다.

#### 예제

```
$ openstack loadbalancer healthmonitor create --name healthmon1 \
--delay 15 --max-retries 4 --timeout 10 --type TCP pool1
```

8. 프라이빗 서브넷(private\_subnet)의 HTTP 백엔드 서버( 192.0.2.10 및 192.0.2.11)를 풀에 추가합니다.

#### 예제

이 예에서 백엔드 서버 **192.0.2.10** 및 **192.0.2.11** 은 각각 **member1** 및 **member2** 로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 --subnet-id \
private_subnet --address 192.0.2.10 --protocol-port 80 pool1
```

```
$ openstack loadbalancer member create --name member2 --subnet-id \
private_subnet --address 192.0.2.11 --protocol-port 80 pool1
```

## 검증

1. 로드 밸런서(**lb1**) 설정을 보고 확인합니다.

## 예제

```
$ openstack loadbalancer status show lb1
```

## 샘플 출력

```
{
  "loadbalancer": {
    "id": "936dad29-4c3f-4f24-84a8-c0e6f10ed810",
    "name": "lb1",
    "operating_status": "ONLINE",
    "provisioning_status": "ACTIVE",
    "listeners": [
      {
        "id": "708b82c6-8a6b-4ec1-ae53-e619769821d4",
        "name": "listener1",
        "operating_status": "ONLINE",
        "provisioning_status": "ACTIVE",
        "pools": [
          {
            "id": "5ad7c678-23af-4422-8edb-ac3880bd888b",
            "name": "pool1",
            "provisioning_status": "ACTIVE",
            "operating_status": "ONLINE",
            "health_monitor": {
              "id": "4ad786ef-6661-4e31-a325-eca07b2b3dd1",
              "name": "healthmon1",
              "type": "TCP",
              "provisioning_status": "ACTIVE",
              "operating_status": "ONLINE"
            },
            "members": [
              {
                "id": "facca0d3-61a7-4b46-85e8-da6994883647",
                "name": "member1",
                "operating_status": "ONLINE",
                "provisioning_status": "ACTIVE",
                "address": "192.0.2.10",
                "protocol_port": 80
              }
            ]
          }
        ]
      }
    ]
  }
}
```

```

{
  "id": "2b0d9e0b-8e0c-48b8-aa57-90b2fde2eae2",
  "name": "member2",
  "operating_status": "ONLINE",
  "provisioning_status": "ACTIVE",
  "address": "192.0.2.11",
  "protocol_port": 80
}
...

```

2.

상태 모니터가 존재하고 제대로 작동하는 경우 각 멤버의 상태를 확인할 수 있습니다.

예제

```
$ openstack loadbalancer member show pool1 member1
```

샘플 출력

작업 멤버(member1)에는 **operating\_status** 에 대한 **ONLINE** 값이 있습니다.

```

+-----+-----+
| Field      | Value                                |
+-----+-----+
| address     | 192.0.2.10                          |
| admin_state_up | True                                |
| created_at  | 2023-11-16T20:08:01                 |
| id          | facca0d3-61a7-4b46-85e8-da6994883647 |
| name        | member1                             |
| operating_status | ONLINE                             |
| project_id  | 9b29c91f67314bd09eda9018616851cf   |
| protocol_port | 80                                  |
| provisioning_status | ACTIVE                             |
| subnet_id   | 3b459c95-64d2-4cfa-b348-01aacc4b3fa9 |
| updated_at  | 2023-11-16T20:08:42                 |
| weight      | 1                                    |
| monitor_port | None                                |
| monitor_address | None                                |
| backup      | False                               |
| tags        |                                     |
+-----+-----+

```

## 추가 리소스

- [키 관리자 서비스 가이드를 사용하여 보안 관리](#)
- 명령줄 인터페이스 참조의 LoadBalancer [https://access.redhat.com/documentation/en-us/red\\_hat\\_openshift\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openshift_platform/17.1/html/command_line_interface_reference/loadbalancer)

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

클라이언트가 보안 또는 비보안 HTTP 프로토콜에 연결되어 있는지 여부에 관계없이 동일한 로드 밸런서와 동일한 콘텐츠로 웹 클라이언트에 응답하려는 경우 비보안 리스너 및 TLS 종료 HTTPS 리스너를 구성할 수 있습니다. 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.

### 사전 요구 사항

- 인터넷에서 연결할 수 있는 공유 외부(공용) 서브넷입니다.
- TLS 공개 키 암호화는 다음과 같은 특성으로 구성됩니다.
  - TLS 인증서, 키 및 선택적 중간 인증서 체인은 로드 밸런서 VIP 주소에 할당된 DNS 이름의 외부 CA(인증 기관)에서 가져왔습니다(예: [www.example.com](http://www.example.com)).
  - 인증서, 키 및 중간 인증서 체인은 현재 디렉터리에 별도의 파일에 있습니다.
  - 키와 인증서는 PEM으로 인코딩됩니다.
  - 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 [키 관리자 서비스 가이드를 사용하여 시크릿](#) 관리를 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**)로 결합합니다.



## 참고

**controlPlane** 내부의 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
```

2. 자격 증명 파일을 가져옵니다.

## 예제

```
$ source ~/overcloudrc
```

3. **Key Manager** 서비스를 사용하여 **PKCS12** 파일의 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**)에 로드 밸런서(**lb1**)를 만듭니다.

예제

```
$ openstack loadbalancer create --name lb1 \
--vip-subnet-id external_subnet --wait
```

5. **TERMINATED\_HTTPS** 리스너(**listener1**)를 생성하고 시크릿 리소스를 리스너의 기본 **TLS** 컨테이너로 참조합니다.

예제

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

6. 풀(**pool1**)을 만들고 리스너의 기본 풀로 설정합니다.

예제

이 예제의 명령은 **TCP** 포트 **80**에서 비보안 **HTTP** 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷을 사용하는 **HTTP** 풀을 생성합니다.

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

7. 백엔드 서버에 연결하는 풀(**pool1**)에 상태 모니터(**healthmon1**)를 생성하고 경로(/)를 테스트합니다.

상태 점검이 권장되지만 필수는 아닙니다. 상태 모니터가 정의되지 않은 경우 멤버 서버는 **ONLINE** 으로 간주됩니다.

예제

```
$ openstack loadbalancer healthmonitor create --name healthmon1 \
--delay 15 --max-retries 4 --timeout 10 --type HTTP --url-path / pool1
```

8.

프라이빗 서브넷(**private\_subnet**)에서 비보안 **HTTP** 백엔드 서버( **192.0.2.10** 및 **192.0.2.11**)를 풀에 추가합니다.

예제

이 예에서 백엔드 서버 **192.0.2.10** 및 **192.0.2.11** 은 각각 **member1** 및 **member2** 로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 --subnet-id \
private_subnet --address 192.0.2.10 --protocol-port 443 pool1
```

```
$ openstack loadbalancer member create --name member2 --subnet-id \
private_subnet --address 192.0.2.11 --protocol-port 443 pool1
```

9.

비보안 **HTTP** 리스너(**listener2**)를 만들고 보안 리스너와 동일하게 기본 풀을 만듭니다.

예제

```
$ openstack loadbalancer listener create --name listener2 \
--protocol-port 80 --protocol HTTP --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 존재하고 제대로 작동하는 경우 각 멤버의 상태를 확인할 수 있습니다.

예제

```
$ openstack loadbalancer member show pool1 member1
```

### 샘플 출력

작업 멤버(member1)에는 **operating\_status** 에 대한 **ONLINE** 값이 있습니다.

```
+-----+
| 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        | member1                             |
| 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

- [키 관리자 서비스 가이드를 사용하여 보안 관리](#)
- 명령줄 인터페이스 참조의 **LoadBalancer** [https://access.redhat.com/documentation/en-us/red\\_hat\\_openstack\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/17.1/html/command_line_interface_reference/loadbalancer)

## 10장. 다른 종류의 로드 밸런서 생성

로드 밸런싱 서비스(**octavia**)를 사용하여 관리하려는 **HTTP** 이외의 네트워크 트래픽 유형과 일치하는 로드 밸런서 유형을 생성합니다.

- [10.1절. “TCP 로드 밸런서 생성”](#)
- [10.2절. “상태 모니터를 사용하여 UDP 로드 밸런서 생성”](#)
- [10.3절. “QoS 규칙 로드 밸런서 생성”](#)
- [10.4절. “액세스 제어 목록을 사용하여 로드 밸런서 생성”](#)
- [10.5절. “OVN 로드 밸런서 생성”](#)

### 10.1. TCP 로드 밸런서 생성

**HTTP** 이외의 **TCP** 기반 서비스 및 애플리케이션에 대한 네트워크 트래픽을 관리해야 하는 경우 로드 밸런서를 생성할 수 있습니다. 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.

#### 사전 요구 사항

- 인터넷에서 연결할 수 있는 공유 외부(공용) 서브넷입니다.

#### 절차

1. 자격 증명 파일을 가져옵니다.

#### 예제

```
$ source ~/overcloudrc
```

2.

공용 서브넷(**public\_subnet**)에 로드 밸런서(**lb1**)를 만듭니다.



참고

**controlPlane** 내부의 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 --wait
```

3.

사용자 지정 애플리케이션이 구성된 지정된 포트(**23456**)에 **TCP** 리스너(**listener1**)를 만듭니다.

예제

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

4.

풀(**pool1**)을 만들고 리스너의 기본 풀로 설정합니다.

예제

이 예에서는 특정 **TCP** 포트에서 사용자 지정 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷을 사용하는 풀이 생성됩니다.

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

5.

백엔드 서버에 연결하고 **TCP** 서비스 포트를 프로브하는 풀(pool1)에 상태 모니터(**healthmon1**)를 만듭니다.

예제

상태 점검이 권장되지만 필수는 아닙니다. 상태 모니터가 정의되지 않은 경우 멤버 서버는 **ONLINE** 으로 간주됩니다.

```
$ openstack loadbalancer healthmonitor create --name healthmon1 \
--delay 15 --max-retries 4 --timeout 10 --type TCP pool1
```

6.

프라이빗 서브넷(**private\_subnet**)의 백엔드 서버(**192.0.2.10** 및 **192.0.2.11**)를 풀에 추가합니다.

예제

이 예에서 백엔드 서버 **192.0.2.10** 및 **192.0.2.11** 은 각각 **member1** 및 **member2** 로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 --subnet-id \
private_subnet --address 192.0.2.10 --protocol-port 443 pool1

$ openstack loadbalancer member create --name member2 --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 존재하고 제대로 작동하는 경우 각 멤버의 상태를 확인할 수 있습니다. 다음 명령을 사용하여 멤버 ID를 가져옵니다.

예제

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

작업 멤버(member1)에는 **operating\_status** 에 대한 **ONLINE** 값이 있습니다.

예제

```
$ openstack loadbalancer member show pool1 member1
```

## 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        | member1                            |
| 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 참조의 **LoadBalancer** [https://access.redhat.com/documentation/en-us/red\\_hat\\_openstack\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/17.1/html/command_line_interface_reference/loadbalancer)

## 10.2. 상태 모니터를 사용하여 UDP 로드 밸런서 생성

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

## 사전 요구 사항

- 인터넷에서 연결할 수 있는 공유 외부(공용) 서브넷입니다.

- **ICMP Destination Unreachable** 메시지를 차단하는 보안 규칙이 없습니다(**ICMP 유형 3**).

## 절차

1. 자격 증명 파일을 가져옵니다.

### 예제

```
$ source ~/overcloudrc
```

2. 프라이빗 서브넷(**private\_subnet**)에 로드 밸런서(**lb1**)를 만듭니다.



### 참고

**controlPlane** 내부의 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 --wait
```

3. 포트1234에서 리스너(**listener1**)를 만듭니다.

### 예제

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

4.

리스너 기본 풀(**pool1**)을 만듭니다.

#### 예제

이 예제의 명령은 **UDP** 포트를 사용하도록 구성된 하나 이상의 애플리케이션을 호스팅하는 백엔드 서버가 포함된 개인 서브넷을 사용하는 풀을 생성합니다.

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

5.

**UDP(UDP-CONNECT)**를 사용하여 백엔드 서버에 연결하는 풀(**pool1**)에 상태 모니터 (**healthmon1**)를 만듭니다.

상태 점검이 권장되지만 필수는 아닙니다. 상태 모니터가 정의되지 않은 경우 멤버 서버는 **ONLINE** 으로 간주됩니다.

#### 예제

```
$ openstack loadbalancer healthmonitor create --name healthmon1 \
--delay 5 --max-retries 2 --timeout 3 --type UDP-CONNECT pool1
```

6.

프라이빗 서브넷(**private\_subnet**)의 백엔드 서버( **192.0.2.10** 및 **192.0.2.11**)를 기본 풀에 추가합니다.

#### 예제

이 예에서 백엔드 서버 **192.0.2.10** 및 **192.0.2.11** 은 각각 **member1** 및 **member2** 로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 --subnet-id \
private_subnet --address 192.0.2.10 --protocol-port 1234 pool1

$ openstack loadbalancer member create --name member2 --subnet-id \
private_subnet --address 192.0.2.11 --protocol-port 1234 pool1
```

## 검증

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

- 상태 모니터가 존재하고 제대로 작동하는 경우 각 멤버의 상태를 확인할 수 있습니다.

## 예제

```
$ openstack loadbalancer member show pool1 member1
```

작업 멤버(member1)에는 **operating\_status** 에 대한 **ONLINE** 값이 있습니다.

#### 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        | member1                             |
| 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 참조의 LoadBalancer [https://access.redhat.com/documentation/en-us/red\\_hat\\_openstack\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/17.1/html/command_line_interface_reference/loadbalancer)

### 10.3. QOS 규칙 로드 밸런서 생성

RHOSP(Red Hat OpenStack Platform) Networking 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. 백엔드 멤버를 계속 사용할 수 있도록 상태 모니터를 생성하는 것이 좋습니다.

## 사전 요구 사항

- 인터넷에서 연결할 수 있는 공유 외부(공용) 서브넷입니다.
- RHOSP 네트워킹 서비스에 대해 생성된 대역폭 제한 규칙이 포함된 **QoS** 정책입니다.

## 절차

1. 자격 증명 파일을 가져옵니다.

### 예제

```
$ source ~/overcloudrc
```

2. 최대 **1024 kbps** 및 최대 **1024kb** 비율로 네트워크 대역폭 **QoS** 정책 (**qos\_policy\_bandwidth**)을 생성합니다.



### 참고

**controlPlane** 내부의 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

를 생성합니다.

예제

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

4. 포트(80)에서 리스너(listener1)를 생성합니다.

예제

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

5. 리스너 기본 풀(pool1)을 만듭니다.

예제

이 예제의 명령은 **TCP** 포트 **80**에서 **HTTP** 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷을 사용하는 **HTTP** 풀을 생성합니다.

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

6. 백엔드 서버에 연결하고 경로(/)를 테스트하는 풀에 상태 모니터(healthmon1)를 만듭니다.

상태 점검이 권장되지만 필수는 아닙니다. 상태 모니터가 정의되지 않은 경우 멤버 서버는 **ONLINE** 으로 간주됩니다.

예제

```
$ openstack loadbalancer healthmonitor create --name healthmon1 \
--delay 15 --max-retries 4 --timeout 10 --type HTTP --url-path /\
pool1
```

7.

프라이빗 서브넷(**private\_subnet**)에서 로드 밸런서 멤버(**192.0.2.10** 및 **192.0.2.11**)를 기본 풀에 추가합니다.

예제

이 예에서 백엔드 서버 **192.0.2.10** 및 **192.0.2.11** 은 각각 **member1** 및 **member2** 로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 --subnet-id \
private_subnet --address 192.0.2.10 --protocol-port 443 pool1
```

```
$ openstack loadbalancer member create --name member2 --subnet-id \
private_subnet --address 192.0.2.11 --protocol-port 443 pool1
```

검증

- 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 **vip\_qos\_policy\_id** 매개 변수에는 정책 ID가 포함되어 있습니다.

#### 추가 리소스

- 

명령줄 인터페이스 참조의 **LoadBalancer** [https://access.redhat.com/documentation/en-us/red\\_hat\\_opensstack\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_opensstack_platform/17.1/html/command_line_interface_reference/loadbalancer)

## 10.4. 액세스 제어 목록을 사용하여 로드 밸런서 생성

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

#### 사전 요구 사항

- 

인터넷에서 연결할 수 있는 공유 외부(공용) 서브넷입니다.

#### 절차

- 1.

자격 증명 파일을 가져옵니다.

예제

```
$ source ~/overcloudrc
```

2.

공용 서브넷(**public\_subnet**)에 로드 밸런서(**lb1**)를 만듭니다.



참고

**controlPlane** 내부의 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 --wait
```

3.

**CIDR(192.0.2.0/24 및 198.51.100.0/24)**을 사용하여 리스너(**listener1**)를 생성합니다.

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
```

4.

리스너 기본 풀(**pool1**)을 만듭니다.

예제

이 예에서는 **TCP** 포트 **80**에서 사용자 지정 애플리케이션으로 구성된 백엔드 서버가 포함된 프라이빗 서브넷을 사용하는 풀이 생성됩니다.

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

5.

백엔드 서버에 연결하는 풀에 상태 모니터를 생성하고 경로(/)를 테스트합니다.

상태 점검이 권장되지만 필수는 아닙니다. 상태 모니터가 정의되지 않은 경우 멤버 서버는 **ONLINE** 으로 간주됩니다.

예제

```
$ openstack loadbalancer healthmonitor create --name healthmon1 \
--delay 15 --max-retries 4 --timeout 10 --type HTTP --url-path / pool1
```

6.

프라이빗 서브넷(**private\_subnet**)에서 로드 밸런서 멤버(**192.0.2.10** 및 **192.0.2.11**)를 기본 풀에 추가합니다.

예제

이 예에서 백엔드 서버 **192.0.2.10** 및 **192.0.2.11** 은 각각 **member1** 및 **member2** 로 이름이 지정됩니다.

```
$ 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** 매개변수는 **192.0.2.0/24** 및 **198.51.100.0/24**의 트래픽만 허용하도록 설정됩니다.

로드 밸런서가 안전한지 확인하려면 **CIDR**이 **allowed\_cidrs** 목록에 없는 클라이언트의 리스너에 대한 요청이 성공하지 않았는지 확인합니다.

#### 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 참조의 **LoadBalancer** [https://access.redhat.com/documentation/en-us/red\\_hat\\_openshift\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openshift_platform/17.1/html/command_line_interface_reference/loadbalancer)

### 10.5. 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 플러그인**을 지원합니다.

#### 사전 요구 사항

- ML2/OVN** 공급자 드라이버를 배포해야 합니다.



#### 중요

**OVN** 공급자는 계층 4 **TCP** 및 **UDP** 네트워크 트래픽과 **SOURCE\_IP\_PORT** 로드 밸런서 알고리즘만 지원합니다. **OVN** 공급자는 상태 모니터링을 지원하지 않습니다.

- 인터넷에서 연결할 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 밸런서를 생성합니다.



참고

**controlPlane** 내부의 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 --wait
```

3. 사용자 지정 애플리케이션이 구성된 지정된 포트(**80**)에서 프로토콜(**tcp**)을 사용하는 리스너(**listener1**)를 생성합니다.



참고

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

예제

■

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

4.

리스너 기본 풀(**pool1**)을 만듭니다.



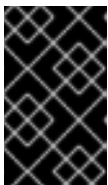
참고

OVN에 지원되는 유일한 로드 밸런싱 알고리즘은 **SOURCE\_IP\_PORT**입니다.

예제

이 예제의 명령은 특정 **TCP** 포트에서 사용자 지정 애플리케이션을 호스팅하는 백엔드 서버가 포함된 프라이빗 서브넷을 사용하는 **HTTP** 풀을 생성합니다.

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



중요

OVN은 로드 밸런싱을 위해 상태 모니터 기능을 지원하지 않습니다.

5.

프라이빗 서브넷(**private\_subnet**)의 백엔드 서버(**192.0.2.10** 및 **192.0.2.11**)를 풀에 추가합니다.

예제

이 예에서 백엔드 서버 **192.0.2.10** 및 **192.0.2.11**은 각각 **member1** 및 **member2**로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 --subnet-id \
private_subnet --address 192.0.2.10 --protocol-port 80 pool1
```

```
$ openstack loadbalancer member create --name member2 --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 listener 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) 및 로드 밸런서 멤버를 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                               |
| crt_container_ref | None                               |
| tls_enabled     | False                              |
+-----+-----+
```

### 추가 리소스

- 

명령줄 인터페이스 참조의 **LoadBalancer** [https://access.redhat.com/documentation/en-us/red\\_hat\\_openshift\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openshift_platform/17.1/html/command_line_interface_reference/loadbalancer)

## 11장. 계층 7 로드 밸런싱 구현

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

- - 11.1절. “계층 7 로드 밸런싱 정보”
- - 11.2절. “로드 밸런싱 서비스의 계층 7 로드 밸런싱”
- - 11.3절. “계층 7 로드 밸런싱 규칙”
- - 11.4절. “계층 7 로드 밸런싱 규칙 유형”
- - 11.5절. “계층 7 로드 밸런싱 규칙 비교 유형”
- - 11.6절. “계층 7 로드 밸런싱 규칙 결과 버전”
- - 11.7절. “계층 7 로드 밸런싱 정책”
- - 11.8절. “계층 7 로드 밸런싱 정책 논리”
- - 11.9절. “계층 7 로드 밸런싱 정책 작업”
- - 11.10절. “계층 7 로드 밸런싱 정책 위치”
- - 11.11절. “비보안 HTTP 요청을 보안 HTTP로 리디렉션”
- - 11.12절. “풀의 시작 경로를 기반으로 요청 리디렉션”

- 11.13절. “하위 도메인 요청을 특정 풀로 전송”
- 11.14절. “특정 풀로 끝나는 호스트 이름을 기반으로 요청 전송”
- 11.15절. “브라우저 쿠키의 부재에 따라 요청을 특정 풀에 전송”
- 11.16절. “브라우저 쿠키 없음 또는 잘못된 쿠키 값을 특정 풀로 통한 요청 전송”
- 11.17절. “이름이 호스트 이름 및 경로와 일치하는 풀에 요청 전송”
- 11.18절. “쿠키를 사용하여 기존 프로덕션 사이트에서 A-B 테스트 구성”

## 11.1. 계층 7 로드 밸런싱 정보

계층 7(L7) 로드 밸런싱은 OSI(Open Systems Interconnection) 모델에서 이름을 사용하여 로드 밸런서가 계층 7(애플리케이션) 데이터를 기반으로 백엔드 애플리케이션 서버 풀에 요청을 배포함을 나타냅니다. 다음은 모두 L7 로드 밸런싱을 의미하는 다른 용어입니다. *요청 전환, 애플리케이션 부하 분산 및 콘텐츠 기반 라우팅, 전환 또는 밸런싱*입니다. Red Hat OpenStack Platform 로드 밸런싱 서비스(octavia)는 L7 로드 밸런싱을 위한 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 사용하는 정책을 기반으로 해당 요청을 수락하여 지정된 요청의 풀 서비스를 결정하는 리스너로 구성됩니다. 이를 통해 특정 유형의 콘텐츠를 제공하도록 애플리케이션 인프라를 구체적으로 조정하고 최적화할 수 있습니다. 예를 들어 하나의 백엔드 서버 그룹(풀)을 조정하여 이미지만 제공하도록 조정할 수 있습니다. 또 다른 하나는 PHP 및 httpd와 같은 서버 측 스크립팅 언어 실행을 위한 것이며, 다른 하나는 HTML, CSS 및 JavaScript와 같은 정적 콘텐츠를 위한 것입니다.

낮은 수준의 로드 밸런싱과 달리 L7 로드 밸런싱에서는 부하 분산 서비스 뒤의 모든 풀에 동일한 콘텐츠가 있어야 하는 것은 아닙니다. L7 로드 밸런서는 애플리케이션 메시지의 URI, 호스트, HTTP 헤더 및 기타 데이터를 기반으로 요청을 보낼 수 있습니다.

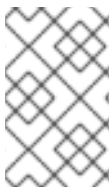
## 11.2. 로드 밸런싱 서비스의 계층 7 로드 밸런싱

잘 정의된 모든 L7 애플리케이션 인터페이스에 대해 계층 7(L7) 로드 밸런싱을 구현할 수 있지만 Red Hat OpenStack Platform 로드 밸런싱 서비스(octavia)의 L7 기능은 HTTP 및 TERMINATED\_HTTPS 프로토콜 및 의미 체계를 나타냅니다.

Neutron LBaaS 및 로드 밸런싱 서비스는 L7 로드 밸런싱 논리에 대해 L7 규칙과 정책을 사용합니다. L7 규칙은 **true** 또는 **false**로 평가되는 간단한 단일 논리 테스트입니다. L7 정책은 정책과 연결된 모든 규칙이 일치하는 경우 수행할 L7 규칙 컬렉션이며 정의된 작업입니다.

## 11.3. 계층 7 로드 밸런싱 규칙

Red Hat OpenStack Platform 로드 밸런싱 서비스(octavia)의 경우 계층 7(L7) 로드 밸런싱 규칙은 **true** 또는 **false**를 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

- [11.4절. “계층 7 로드 밸런싱 규칙 유형”](#)

## 11.4. 계층 7 로드 밸런싱 규칙 유형

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

- **HOST\_NAME:** 규칙은 요청의 HTTP/1.1 호스트 이름을 규칙의 **value** 매개변수와 비교합니다.
- **PATH:** 규칙은 HTTP URI의 **path** 부분을 규칙의 **value** 매개변수와 비교합니다.
- **FILE\_TYPE:** 이 규칙은 URI의 마지막 부분을 규칙의 **value** 매개 변수(예: **txt**, **kafka** 등)와 비교합니다.

- **HEADER:** 규칙은 **key** 매개변수에 정의된 헤더를 찾고 규칙의 **value** 매개변수와 비교합니다.
- **COOKIE:** 규칙은 **key** 매개변수로 이름이 지정된 쿠키를 찾아 규칙의 **value** 매개변수와 비교합니다.
- **SSL\_CONN\_HAS\_CERT:** 클라이언트가 **TLS** 클라이언트 인증을 위한 인증서를 제공한 경우 규칙과 일치합니다. 인증서가 유효한 것을 의미하지는 않습니다.
- **SSL\_VERIFY\_RESULT:** 이 규칙은 **TLS** 클라이언트 인증 인증서 검증 결과와 일치합니다. 값이 0(0)이면 인증서의 유효성이 성공적으로 검증되었습니다. 값이 0이면 인증서가 검증에 실패했습니다. 이 값은 **openssl-verify** 결과 코드를 따릅니다.
- **SSL\_DN\_FIELD:** 규칙은 **key** 매개변수에 정의된 **Distinguished Name** 필드를 찾아 규칙의 **value** 매개변수와 비교합니다.

#### 추가 리소스

- [11.3절. “계층 7 로드 밸런싱 규칙”](#)

### 11.5. 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 다음과 같은 비교 유형을 지원합니다. 모든 규칙 유형이 모든 비교 유형을 지원하는 것은 아닙니다.

- **REGEX:** Perl 유형 정규식 일치
- **STARTS\_WITH:** 문자열로 시작
- **ENDS\_WITH:** 문자열로 끝납니다.
- **CONTAINS:** 문자열이 포함되어 있습니다.

- **EQUAL\_TO:** 문자열은 동일합니다.

추가 리소스

- [11.4절. “계층 7 로드 밸런싱 규칙 유형”](#)

## 11.6. 계층 7 로드 밸런싱 규칙 결과 버전

일부 정책에 필요한 논리와 **Red Hat OpenStack Platform** 로드 밸런싱 서비스(**octavia**)가 사용하는 논리를 보다 완전히 표현하려면 계층 7 로드 밸런싱 규칙에 따라 결과가 반전될 수 있습니다. 지정된 규칙의 반전 매개 변수가 **true**이면 비교 결과가 반전됩니다.

예를 들어, 규칙 반전은 효과적으로 규칙 과 일치하지 않습니다. 반전된 **regex** 규칙은 지정된 정규식이 일치하지 않는 경우에만 **true** 를 반환합니다.

추가 리소스

- [11.7절. “계층 7 로드 밸런싱 정책”](#)

## 11.7. 계층 7 로드 밸런싱 정책

**Red Hat OpenStack Platform** 로드 밸런싱 서비스(**octavia**)의 경우 계층 7(**L7**) 로드 밸런싱 정책은 리스너와 연결된 **L7** 규칙 컬렉션이며 백엔드 풀에도 연관될 수 있습니다. 정책은 정책의 모든 규칙이 **true**인 경우 로드 밸런서에서 수행하는 작업입니다.



참고

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

추가 리소스

- [11.3절. “계층 7 로드 밸런싱 규칙”](#)

## 11.8. 계층 7 로드 밸런싱 정책 논리

**Red Hat OpenStack Platform** 로드 밸런싱 서비스(**octavia**) 계층 7 로드 밸런싱 정책은 다음 논리를

사용합니다. 지정된 정책과 연결된 모든 규칙은 논리적으로 **AND**로 설정됩니다. 정책과 일치하도록 요청이 모든 정책 규칙과 일치해야 합니다.

규칙 간에 논리 **OR** 작업을 표현해야 하는 경우 동일한 작업을 사용하여 여러 정책을 생성하거나 보다 정교한 정규식을 만듭니다.

추가 리소스

- [11.3절. “계층 7 로드 밸런싱 규칙”](#)

## 11.9. 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.

- **REJECT:** 요청이 적절한 응답 코드로 거부되고 백엔드 풀로 전달되지 않습니다.
- **REDIRECT\_TO\_URL:** 요청이 `redirect_url` 매개변수에 정의된 URL로 HTTP 리디렉션을 보냅니다.
- **REDIRECT\_PREFIX:** 이 정책과 일치하는 요청은 이 접두사 URL로 리디렉션됩니다.
- **REDIRECT\_TO\_POOL:** 요청은 L7 정책과 연결된 백엔드 풀로 전달됩니다.

추가 리소스

- [11.7절. “계층 7 로드 밸런싱 정책”](#)

## 11.10. 계층 7 로드 밸런싱 정책 위치

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)의 경우 여러 계층 7(L7) 로드 밸런싱 정책이 리스너와 연결된 경우 정책 위치 매개변수의 값이 중요합니다. **position** 매개변수는 L7 정책이 평가되는 순서를 결정할 때 사용됩니다. 정책 위치는 다음과 같은 방법으로 리스너 동작에 영향을 미칩니다.

-

로드 밸런싱 서비스(haproxy amphorae)의 참조 구현에서 HAProxy는 정책 작업과 관련하여 다음 순서를 적용합니다.

- **REJECT** 정책이 다른 모든 정책보다 우선합니다.
- **REDIRECT\_TO\_URL** 정책은 **REDIRECT\_TO\_POOL** 정책보다 우선합니다.
- **REDIRECT\_TO\_POOL** 정책은 위의 모든 후와 정책의 위치가 지정하는 순서대로 평가됩니다.
- **L7** 정책은 위치 특성에 정의된 대로 특정 순서로 평가되며 지정된 요청과 일치하는 첫 번째 정책은 작업이 수행되는 작업입니다.
- 지정된 요청과 일치하는 정책이 없는 경우 요청은 리스너의 기본 풀(있는 경우)으로 라우팅됩니다. 리스너에 기본 풀이 없으면 **503** 오류가 반환됩니다.
- 정책 위치 번호 지정은 **1개(1)**로 시작합니다.
- 기존 정책의 해당 위치와 일치하는 위치로 새 정책이 생성되면 지정된 위치에 새 정책이 삽입됩니다.
- 위치를 지정하지 않고 새 정책을 생성하거나 목록에 이미 있는 정책 수보다 큰 위치를 지정하면 새 정책이 목록에 추가됩니다.
- 정책이 목록에 삽입, 삭제 또는 추가되면 번호 건너뛰지 않고 정책 위치 값이 1에서 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

- [11.7절. “계층 7 로드 밸런싱 정책”](#)

## 11.11. 비보안 HTTP 요청을 보안 HTTP로 리디렉션

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

이 예에서는 비보안 TCP 포트 80에 도달하는 모든 HTTP 요청이 보안 TCP 포트 443으로 리디렉션됩니다.

#### 사전 요구 사항

- 리스너(listener1)와 풀(pool1)이 있는 TLS 종료 HTTPS 로드 밸런서(lb1)입니다. 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 생성합니다.



#### 참고

controlPlane 내부의 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**)에서 **L7** 정책(**policy1**)을 생성합니다. 정책에는 작업 (**REDIRECT\_TO\_URL**)이 포함되어야 하며 **URL**(**https://www.example.com/**)을 가리켜야 합니다.

예제

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

4.

모든 요청과 일치하는 **L7** 규칙을 정책에 추가합니다(**policy1**).

예제

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

검증

1.

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

2.

**openstack loadbalancer l7rule list <l7policy >** 명령을 실행하고 **compare\_type of STARTS\_WITH** 이 있는지 확인합니다.

예제

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

## 추가 리소스

- [명령줄 인터페이스 참조에서 LoadBalancer 리스너 생성](#)
- [명령줄 인터페이스 참조에서 LoadBalancer l7policy 생성](#)
- [명령줄 인터페이스 참조에서 LoadBalancer l7rule create](#)

### 11.12. 풀의 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 **URL**에서 하나 이상의 시작 경로와 일치하도록 계층 **7(L7)** 정책을 정의할 수 있습니다.

이 예에서는 **/js** 또는 **/images** 로 시작하는 **URL**이 포함된 모든 요청은 정적 콘텐츠 서버의 대체 풀로 리디렉션됩니다.

## 사전 요구 사항

- 리스너(**listener1**)와 풀(**pool1**)이 있는 **HTTP** 로드 밸런서(**lb1**)입니다. 자세한 내용은 [상태 모니터링을 사용하여 HTTP 로드 밸런서 생성](#)을 참조하십시오.

## 절차

1. 자격 증명 파일을 가져옵니다.

### 예제

```
$ source ~/overcloudrc
```

2. 로드 밸런서(**lb1**)에 두 번째 풀(**static\_pool**)을 만듭니다.



## 참고

**controlPlane** 내부의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

## 예제

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

3.

프라이빗 서브넷(**private\_subnet**)의 로드 밸런서 멤버(**192.0.2.10** 및 **192.0.2.11**)를 풀(**static\_pool**)에 추가합니다.

## 예제

이 예에서 백엔드 서버 **192.0.2.10** 및 **192.0.2.11** 은 각각 **member1** 및 **member2** 로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 --subnet-id \
private_subnet --address 192.0.2.10 --protocol-port 80 static_pool
```

```
$ openstack loadbalancer member create --name member2 --subnet-id \
private_subnet --address 192.0.2.11 --protocol-port 80 static_pool
```

4.

리스너(**listener1**)에 **L7** 정책(**policy1**)을 생성합니다. 정책에는 작업(**REDIRECT\_TO\_POOL**)이 포함되어야 하며 풀(**static\_pool**)을 가리켜야 합니다.

## 예제

```
$ openstack loadbalancer l7policy create --name policy1 \
--action REDIRECT_TO_POOL --redirect-pool static_pool 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(**policy2**)을 생성하고 풀을 가리키는 리스너(**listener1**)를 추가합니다.

예제

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

7.

정책의 요청 경로 시작 부분에 **/images** 를 찾는 **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 **policy2** 가 있는지 확인합니다.

2.

**openstack loadbalancer l7rule list <l7policy >** 명령을 실행하고 각 정책에 대해 **compare\_type of STARTS\_WITH** 이 있는 규칙이 있는지 확인합니다.

예제

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

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

추가 리소스

- [명령줄 인터페이스 참조에서 LoadBalancer 풀 생성](#)
- [명령줄 인터페이스 참조에서 LoadBalancer 멤버 생성](#)
- [명령줄 인터페이스 참조에서 LoadBalancer l7policy 생성](#)
- [명령줄 인터페이스 참조에서 LoadBalancer l7rule create](#)

### 11.13. 하위 도메인 요청을 특정 풀로 전송

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

이 예에서는 **HTTP/1.1** 호스트 이름 **www2.example.com** 이 포함된 모든 요청이 대체 풀 애플리케이션 서버인 **pool2** 로 리디렉션됩니다.

사전 요구 사항

- 리스너(**listener1**)와 풀(**pool1**)이 있는 **HTTP** 로드 밸런서(**lb1**)입니다.

자세한 내용은 [상태 모니터를 사용하여 HTTP 로드 밸런서 생성](#)을 참조하십시오.

## 절차

1. 자격 증명 파일을 가져옵니다.

### 예제

```
$ source ~/overcloudrc
```

2. 로드 밸런서에 두 번째 풀(pool2)을 만듭니다(lb1).



### 참고

**controlPlane** 내부의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

### 예제

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

3. 리스너(listener1)에 L7 정책(policy1)을 생성합니다. 정책에는 작업(**REDIRECT\_TO\_POOL**)이 포함되어야 하며 풀(pool2)을 가리켜야 합니다.

### 예제

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

4.

**HTTP/1.1** 호스트 이름 **www2.example.com**을 사용하여 요청을 두 번째 풀(pool2)으로 전송하는 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 **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

•

명령줄 인터페이스 참조에서 **LoadBalancer** 풀 생성

•

명령줄 인터페이스 참조에서 **LoadBalancer l7policy** 생성

•

명령줄 인터페이스 참조에서 [LoadBalancer l7rule create](#)

#### 11.14. 특정 풀로 끝나는 호스트 이름을 기반으로 요청 전송

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

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

#### 사전 요구 사항

•

리스너(**listener1**)와 풀(**pool1**)이 있는 **HTTP** 로드 밸런서(**lb1**)입니다.

자세한 내용은 [상태 모니터를 사용하여 HTTP 로드 밸런서 생성](#)을 참조하십시오.

#### 절차

1.

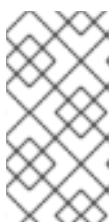
자격 증명 파일을 가져옵니다.

#### 예제

```
$ source ~/overcloudrc
```

2.

로드 밸런서에 두 번째 풀(**pool2**)을 만듭니다(**lb1**).



#### 참고

**controlPlane** 내부의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

## 예제

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

3.

리스너(**listener1**)에 **L7 정책(policy1)**을 생성합니다. 정책에는 작업(**REDIRECT\_TO\_POOL**)이 포함되어야 하며 풀(**pool2**)을 가리켜야 합니다.

## 예제

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

4.

**HTTP/1.1** 호스트 이름(**www2.example.com**)을 사용하는 요청을 두 번째 풀(**pool2**)으로 전송하는 모든 요청을 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 **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
```

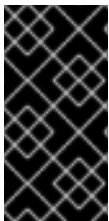
## 추가 리소스

- [명령줄 인터페이스 참조에서 LoadBalancer 풀 생성](#)
- [명령줄 인터페이스 참조에서 LoadBalancer l7policy 생성](#)
- [명령줄 인터페이스 참조에서 LoadBalancer l7rule create](#)

## 11.15. 브라우저 쿠키의 부재에 따라 요청을 특정 풀에 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 쿠키를 누락하는지 여부를 결정합니다.

이 예에서 브라우저 쿠키 `auth_token` 이 없는 웹 클라이언트는 인증 서버가 포함된 대체 풀로 리디렉션됩니다.



## 중요

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

## 사전 요구 사항

- 리스너(`listener1`)와 풀(`pool1`)이 있는 TLS 종료 HTTPS 로드 밸런서(`lb1`)입니다.

자세한 내용은 [TLS 종료 HTTPS 로드 밸런서 생성](#) 을 참조하십시오.

-

보안 인증 서버가 웹 사용자를 인증하는 두 번째 **RHOSP Networking 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

**controlPlane** 내부의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

### 예제

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

3. 인증 서브넷(**secure\_subnet**)의 **member**, **secure authentication server(192.0.2.10)**를 두 번째 풀에 추가합니다.

### 예제

이 예에서 백엔드 서버 **192.0.2.10**의 이름은 **member1**입니다.

```
$ openstack loadbalancer member create --name member1 \
--address 192.0.2.10 --protocol-port 80 --subnet-id secure_subnet \
login_pool
```

4.

리스너(**listener1**)에 **L7 정책(policy1)**을 생성합니다. 정책에는 작업(**REDIRECT\_TO\_POOL**)이 포함되어야 하며 두 번째 풀(**login\_pool**)을 가리켜야 합니다.

예제

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

5.

브라우저 쿠키(**auth\_token**)를 값으로 검색하고 쿠키가 없는 경우 일치하는 **L7 규칙(policy1)**을 추가합니다.

예제

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

검증

1.

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

2.

**openstack loadbalancer l7rule list <l7policy >** 명령을 실행하고 **REGEX** 의 **compare\_type** 이 있는 규칙이 있는지 확인합니다.

예제

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

## 추가 리소스

- [명령줄 인터페이스 참조에서 LoadBalancer 풀 생성](#)
- [명령줄 인터페이스 참조에서 LoadBalancer 멤버 생성](#)
- [명령줄 인터페이스 참조에서 LoadBalancer l7policy 생성](#)
- [명령줄 인터페이스 참조에서 LoadBalancer l7rule create](#)

## 11.16. 브라우저 쿠키 없음 또는 잘못된 쿠키 값을 특정 풀로 통한 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 쿠키를 누락하는지 또는 특정 값이 있는 인증 쿠키를 포함하는지 여부를 결정합니다.

이 예에서 브라우저 쿠키 `auth_token`이 없거나 `INVALID` 값이 있는 `auth_token` 이 있는 웹 클라이언트는 인증 서버를 포함하는 대체 풀로 리디렉션됩니다.



### 중요

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

## 사전 요구 사항

- 리스너(`listener1`)와 풀(`pool1`)이 있는 TLS 종료 HTTPS 로드 밸런서(`lb1`)입니다.

자세한 내용은 [TLS 종료 HTTPS 로드 밸런서 생성](#) 을 참조하십시오.

- 보안 인증 서버가 웹 사용자를 인증하는 두 번째 RHOSP Networking 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

**controlPlane** 내부의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

## 예제

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

3. 인증 서브넷(**secure\_subnet**)의 **member**, **secure authentication server**(192.0.2.10)를 두 번째 풀에 추가합니다.

## 예제

이 예에서 백엔드 서버 **192.0.2.10**의 이름은 **member1**입니다.

```
$ openstack loadbalancer member create --name member1 \
--address 192.0.2.10 --protocol-port 80 --subnet-id secure_subnet \
login_pool
```

4.

리스너(**listener1**)에 **L7 정책(policy1)**을 생성합니다. 정책에는 작업 (**REDIRECT\_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 값으로 검색하고 쿠키가 없는 경우 일치하는 **L7 규칙(policy1)**을 추가합니다.

예제

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

6.

리스너(**listener1**)에 두 번째 **L7정책( policy2)**을 생성합니다. 정책에는 작업 (**REDIRECT\_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 같은 경우 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 **policy2** 가 있는지 확인합니다.
2. **openstack loadbalancer l7rule list <l7policy>** 명령을 실행하고 **policy1** 에 대해 **REGEX** 의 **compare\_type** 이 있는 규칙과 **policy2** 용으로 **compare\_type of EQUAL\_TO** 규칙이 있는지 확인합니다.

예제

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

추가 리소스

- 명령줄 인터페이스 참조에서 **LoadBalancer** 풀 생성
- 명령줄 인터페이스 참조에서 **LoadBalancer** 멤버 생성
- 명령줄 인터페이스 참조에서 **LoadBalancer l7policy** 생성

•

명령줄 인터페이스 참조에서 [LoadBalancer l7rule create](#)

### 11.17. 이름이 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 애플리케이션 서버 대체 풀로 리디렉션할 수 있습니다. 비즈니스 논리 기준은 사전 정의된 호스트 이름 및 요청 경로와 일치하도록 시도하는 계층 **7(L7)** 정책을 통해 수행됩니다.

이 예에서 호스트 이름 **api.example.com** 과 일치하는 웹 클라이언트 요청 및 요청 경로 시작 시 **/api** 가 있는 모든 웹 클라이언트는 대체 풀 **api\_pool** 로 리디렉션됩니다.

#### 사전 요구 사항

•

리스너(**listener1**)와 풀(**pool1**)이 있는 **HTTP** 로드 밸런서(**lb1**)입니다. 자세한 내용은 [상태 모니터를 사용하여 HTTP 로드 밸런서 생성](#)을 참조하십시오.

#### 절차

1.

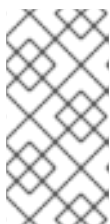
자격 증명 파일을 가져옵니다.

#### 예제

```
$ source ~/overcloudrc
```

2.

로드 밸런서(**lb1**)에 두 번째 풀(**api\_pool**)을 만듭니다.



#### 참고

**controlPlane** 내부의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

#### 예제

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

3.

프라이빗 서브넷(**private\_subnet**)의 로드 밸런서 멤버(**192.0.2.10** 및 **192.0.2.11**)를 풀(**static\_pool**)에 추가합니다.

예제

이 예에서 백엔드 서버 **192.0.2.10** 및 **192.0.2.11** 은 각각 **member1** 및 **member2** 로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 --subnet-id \
private_subnet --address 192.0.2.10 --protocol-port 80 static_pool
```

```
$ openstack loadbalancer member create --name member2 --subnet-id \
private_subnet --address 192.0.2.11 --protocol-port 80 static_pool
```

4.

리스너(**listener1**)에 **L7** 정책(**policy1**)을 생성합니다. 정책에는 작업(**REDIRECT\_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.

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

이 규칙은 첫 번째 규칙에 따라 논리적으로 **AND**됩니다.

예제

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

검증

1.

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

2.

**openstack loadbalancer l7rule list <l7policy >** 명령을 실행하고 **policy1** 용으로 각각 **EQUAL\_TO** 및 **STARTS\_WITH** 의 **compare\_type** 이 있는 규칙이 있는지 확인합니다.

예제

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

추가 리소스

•

명령줄 인터페이스 참조에서 **LoadBalancer** 플 생성

- 명령줄 인터페이스 참조에서 **LoadBalancer** **멤버** 생성
- 명령줄 인터페이스 참조에서 **LoadBalancer** **l7policy** 생성
- 명령줄 인터페이스 참조에서 **LoadBalancer** **l7rule create**

#### 11.18. 쿠키를 사용하여 기존 프로덕션 사이트에서 A-B 테스트 구성

**RHOSP(Red Hat OpenStack Platform)** 로드 밸런싱 서비스(**octavia**)를 계층 7(**L7**) 정책과 함께 사용하여 **A-B** 테스트를 구성하거나 프로덕션 웹 사이트에 대한 테스트를 분할할 수 있습니다.

이 예에서 웹 사이트의 "**B**" 버전으로 라우팅되는 웹 클라이언트는 풀의 멤버 서버에 의해 쿠키 **site\_version** 을 **B** 로 설정합니다(**pool1**).

#### 사전 요구 사항

- 두 개의 프로덕션 웹사이트(사이트 **A** 및 사이트 **B**).
- "풀의 시작 경로에 따라 요청 리디렉션"에 대한 지침에 따라 **HTTP** 로드 밸런서를 구성했습니다. 필요한 구성에 대한 요약은 다음과 같습니다.
  - 로드 밸런서의 리스너(**listener1**)입니다.
  - **/js** 또는 **/images** 로 시작하는 **URL**이 있는 **HTTP** 요청은 풀(**static\_pool**)으로 전송됩니다.
  - 다른 모든 요청은 리스너 기본 풀(**pool1**)으로 전송됩니다.
  - 구성에 대한 자세한 내용은 **11.12절. “풀의 시작 경로를 기반으로 요청 리디렉션”** 을 참조하십시오.

#### 절차

1.

자격 증명 파일을 가져옵니다.

예제

```
$ source ~/overcloudrc
```

2.

로드 밸런서(lb1)에 세 번째 풀(pool\_B)을 만듭니다.



참고

**controlPlane** 내부의 값은 이 절차의 예제 명령에 사용되는 샘플 값입니다. 이러한 샘플 값을 사이트에 적합한 값으로 바꿉니다.

예제

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

3.

프라이빗 서브넷(private\_subnet)의 로드 밸런서 멤버(192.0.2.50 및 192.0.2.51)를 풀(pool\_B)에 추가합니다.

예제

이 예에서 백엔드 서버 192.0.2.50 및 192.0.2.51 은 각각 member1 및 member2 로 이름이 지정됩니다.

```
$ openstack loadbalancer member create --name member1 \
--address 192.0.2.50 --protocol-port 80 \
--subnet-id private_subnet pool_B
```

```
$ openstack loadbalancer member create --name member2 \
--address 192.0.2.51 --protocol-port 80 \
--subnet-id private_subnet pool_B
```

4.

로드 밸런서(**lb1**)에 네 번째 풀(**static\_pool\_B**)을 만듭니다.

예제

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

5.

프라이빗 서브넷(**private\_subnet**)의 로드 밸런서 멤버(**192.0.2.100** 및 **192.0.2.101**)를 풀(**static\_pool\_B**)에 추가합니다.

예제

```
$ openstack loadbalancer member create --name member3 \
--address 192.0.2.100 --protocol-port 80 \
--subnet-id private_subnet static_pool_B
```

```
$ openstack loadbalancer member create --name member4 \
--address 192.0.2.101 --protocol-port 80 \
--subnet-id private_subnet static_pool_B
```

6.

리스너(**listener1**)에 **L7 정책(policy2)**을 생성합니다. 정책에는 작업(**REDIRECT\_TO\_POOL**)이 포함되어야 하며 풀(**static\_pool\_B**)을 가리켜야 합니다. **1** 위치에 정책을 삽입합니다.

예제

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

7.

요청 경로 시작 시 **/js** 또는 **/images** 와 일치하도록 정규식을 사용하는 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 정책(**policy2**)에 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(**listener1**)에 **L7** 정책(**policy3**)을 생성합니다. 정책에는 작업(**REDIRECT\_TO\_POOL**)이 포함되어야 하며 풀(**pool\_B**)을 가리켜야 합니다. **2** 위치에 정책을 삽입합니다.

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

가장 구체적인 규칙과 함께 **L7** 정책을 더 낮은 위치에 할당하는 것이 중요합니다. 모든 규칙이 **True**로 평가되는 첫 번째 정책은 작업이 따르는 정책이기 때문입니다. 이 절차에서는 요청을 잘못된 풀로 보내지 않도록 **policy3** 전에 **policy2**를 평가해야 합니다.

검증

1.

**openstack loadbalancer l7policy list** 명령을 실행하고 정책, **policy2** 및 **policy3**이 있는지 확인합니다.

2.

**openstack loadbalancer l7rule list <l7policy>** 명령을 실행하고 각 정책에 대해 **compare\_type of STARTS\_WITH**이 있는 규칙이 있는지 확인합니다.

예제

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

추가 리소스

•

명령줄 인터페이스 참조에서 **LoadBalancer** 풀 생성

- 명령줄 인터페이스 참조에서 **LoadBalancer** [멤버](#) 생성
- 명령줄 인터페이스 참조에서 **LoadBalancer** [l7policy](#) 생성
- 명령줄 인터페이스 참조에서 **LoadBalancer** [l7rule create](#)

## 12장. 태그를 사용하여 로드 밸런싱 서비스 오브젝트 그룹화

태그는 그룹으로 분류하기 위해 **Red Hat OpenStack Platform** 로드 밸런싱 서비스(**octavia**) 오브젝트에 추가할 수 있는 임의의 문자열입니다. 태그는 로드 밸런서, 리스너, 풀, 구성원, 상태 모니터, 규칙, 정책 등 부하 분산 오브젝트의 기능에는 영향을 미치지 않습니다. 오브젝트를 생성할 때 태그를 추가하거나 오브젝트를 생성한 후 태그를 추가하거나 제거할 수 있습니다.

특정 태그를 로드 밸런싱 오브젝트와 연결하면 **list** 명령을 실행하여 하나 이상의 그룹에 속하는 오브젝트를 필터링할 수 있습니다. 오브젝트를 하나 이상의 그룹으로 필터링할 수 있다는 것은 로드 밸런싱 서비스 리소스의 사용, 할당 및 유지 관리를 관리하는 시작점일 수 있습니다. 오브젝트 태그 기능은 자동화된 구성 관리 툴에서도 활용할 수 있습니다.

이 섹션에 포함된 항목은 다음과 같습니다.

- [12.1절. “서비스 오브젝트 로드 밸런싱을 생성할 때 태그 추가”](#)
- [12.2절. “기존 로드 밸런싱 서비스 오브젝트에 태그 추가 또는 제거”](#)
- [12.3절. “태그를 사용하여 로드 밸런싱 서비스 오브젝트 필터링”](#)

### 12.1. 서비스 오브젝트 로드 밸런싱을 생성할 때 태그 추가

**Red Hat OpenStack** 로드 밸런싱 서비스(**octavia**) 오브젝트를 생성할 때 선택한 태그를 추가할 수 있습니다. 태그가 적용되면 해당 로드 밸런서 목록 명령을 사용하여 로드 밸런서, 리스너, 풀, 멤버, 상태 모니터, 규칙 및 정책을 필터링할 수 있습니다.

#### 프로세스

1. 자격 증명 파일을 가져옵니다.

#### 예제

```
$ source ~/overcloudrc
```

2.

오브젝트에 적절한 **create** 명령과 함께 **--tag <tag>** 옵션을 사용하여 로드 밸런싱 오브젝트에 태그를 추가합니다.

- **OpenStack loadbalancer create --tag <tag> ...**
- **OpenStack Loadbalancer 리스너 create --tag <tag> ...**
- **OpenStack loadbalancer pool create --tag <tag> ...**
- **OpenStack Loadbalancer 멤버 create --tag <tag> ...**
- **OpenStack loadbalancer healthmonitor create --tag <tag> ...**
- **OpenStack loadbalancer l7policy create --tag <tag> ...**
- **openstack loadbalancer l7rule create --tag <tag> ...**



참고

태그는 최대 **255**자인 유효한 유니코드 문자열일 수 있습니다.

예 - 로드 밸런서 생성 및 태그

이 예제에서는 로드 밸런서( **lb1** )는 **Finance** 및 **Sales**: 두 개의 태그로 생성됩니다.

```
$ openstack loadbalancer create --name lb1 \
--vip-subnet-id public_subnet --tag Finance --tag Sales
```



## 참고

서비스 오브젝트에는 하나 이상의 태그가 있을 수 있습니다. 추가할 각 추가 태그에 대해 **--tag <tag>** 옵션을 반복합니다.

## 예 - 리스너 생성 및 태그

이 예에서는 **listener1** 이 태그를 사용하여 생성됩니다. **Sales:**

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

## 예 - 풀 생성 및 태그 지정

이 예에서는 **pool1** 이 태그인 **Sales:**을 사용하여 생성됩니다.

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

## 예 - 풀에서 멤버 생성 및 태그 지정

이 예에서는 **members**인 **192.0.2.10** 이 **tag**인 **Sales:**을 사용하여 **pool1** 에 생성됩니다.

```
$ openstack loadbalancer member create --name member1 \
--subnet-id private_subnet --address 192.0.2.10 --protocol-port 80 \
--tag Sales pool1
```

## 예 - 상태 모니터 생성 및 태그 지정

이 예에서는 **health monitor**인 **healthmon1** 이 태그로 생성됩니다. **Sales:**

```
$ openstack loadbalancer healthmonitor create --name healthmon1 \
--delay 15 --max-retries 4 --timeout 10 --type HTTP --url-path /\
--tag Sales pool1
```

## 예 - L7 정책 생성 및 태그

이 예에서는 **L7 정책** **policy1** 이 태그를 사용하여 생성됩니다. **Sales:**

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

■

**예 - L7 규칙 생성 및 태그 지정**

이 예제에서는 L7 규칙인 **rule1** 이 태그를 사용하여 생성됩니다. **Sales:**

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

**검증**

●

생성한 오브젝트가 있는지 확인하고 오브젝트에 적절한 **show** 명령을 사용하여 추가한 태그가 포함되어 있는지 확인합니다.

**예제**

이 예제에서 **show** 명령은 **loadbalancer, lb1** 에서 실행됩니다.

```
$ openstack loadbalancer show lb1
```

**샘플 출력**

```
+-----+-----+
| admin_state_up | True          |
| availability_zone | None         |
| created_at     | 2022-08-26T19:34:15 |
| description    |               |
| flavor_id     | None         |
| id            | 7975374b-3367-4436-ab19-2d79d8c1f29b |
| listeners     |              |
| name          | lb1          |
| operating_status | ONLINE       |
| pools        |              |
| project_id    | 2eee3b86ca404cdd977281dac385fd4e |
| provider      | amphora      |
| provisioning_status | ACTIVE      |
| updated_at    | 2022-08-30T13:30:17 |
| vip_address    | 172.24.3.76   |
| vip_network_id | 4c241fc4-95eb-491a-affe-26c53a8805cd |
| vip_port_id    | 9978a598-cc34-47f7-ba28-49431d570fd1 |
| vip_qos_policy_id | None         |
| vip_subnet_id | e999d323-bd0f-4469-974f-7f66d427e507 |
| tags          | Finance      |
|              | Sales        |
+-----+-----+
```

- 명령줄 인터페이스 참조에서 **LoadBalancer** [생성](#)
- 명령줄 인터페이스 참조에 **LoadBalancer** [표시](#)

## 12.2. 기존 로드 밸런싱 서비스 오브젝트에 태그 추가 또는 제거

**Red Hat OpenStack Load-balancing service(octavia)** 오브젝트에서 선택한 태그를 추가하고 제거할 수 있습니다. 태그가 적용되면 해당 로드 밸런서 목록 명령을 사용하여 로드 밸런서, 리스너, 풀, 멤버, 상태 모니터, 규칙 및 정책을 필터링할 수 있습니다.

### 프로세스

1. 자격 증명 파일을 가져옵니다.

### 예제

```
$ source ~/overcloudrc
```

2. 다음 중 하나를 수행합니다.

- 오브젝트에 적절한 **set** 명령과 함께 **--tag <tag>** 옵션을 사용하여 기존 로드 밸런싱 오브젝트에 태그를 추가합니다.
  - **openstack loadbalancer set --tag <tag> <load\_balancer\_name\_or\_ID>**
  - **openstack loadbalancer listener set --tag <tag> <listener\_name\_or\_ID>**

- `openstack loadbalancer pool set --tag <tag> <pool_name_or_ID>`
- `openstack loadbalancer member set --tag <tag> <pool_name_or_ID> <member_name_or_ID>`
- `openstack loadbalancer healthmonitor set --tag <tag> <healthmon_name_or_ID>`
- `openstack loadbalancer l7policy set --tag <tag> <l7policy_name_or_ID>`
- `openstack loadbalancer l7rule set --tag <tag> <l7policy_name_or_ID> <l7rule_ID>`



## 참고

태그는 최대 **255**자인 유효한 유니코드 문자열일 수 있습니다.

## 예제

이 예제에서 태그인 **Finance** 및 **Sales** 가 로드 밸런서인 **lb1** 에 추가됩니다.

```
$ openstack loadbalancer set --tag Finance --tag Sales lb1
```



## 참고

서비스 오브젝트에는 하나 이상의 태그가 있을 수 있습니다. 추가할 각 추가 태그에 대해 **--tag <tag>** 옵션을 반복합니다.

- 오브젝트에 대한 적절한 설정되지 않은 명령과 **--tag <tag>** 옵션을 사용하여 기존 로드 밸런싱 오브젝트에서 태그를 제거합니다.
  - `openstack loadbalancer unset --tag <tag> <load_balancer_name_or_ID>`

- `openstack loadbalancer listener unset --tag <tag> <listener_name_or_ID>`
- `openstack loadbalancer pool unset --tag <tag> <pool_name_or_ID>`
- `openstack loadbalancer member unset --tag <tag> <pool_name_or_ID> <member_name_or_ID>`
- `openstack loadbalancer healthmonitor unset --tag <tag> <healthmon_name_or_ID>`
- `openstack loadbalancer l7policy unset --tag <tag> <policy_name_or_ID>`
- `openstack loadbalancer l7rule unset --tag <tag> <policy_name_or_ID> <l7rule_ID>`

#### 예제

이 예에서 **Sales** 태그는 로드 밸런서인 **lb1** 에서 제거됩니다.

```
$ openstack loadbalancer unset --tag Sales lb1
```

- 오브젝트에 적절한 **set** 명령과 함께 **--no-tag** 옵션을 사용하여 기존 로드 밸런싱 오브젝트에서 모든 태그를 제거합니다.
  - `openstack loadbalancer set --no-tag <load_balancer_name_or_ID>`
  - `openstack loadbalancer listener set --no-tag <listener_name_or_ID>`
  - `openstack loadbalancer pool set --no-tag <pool_name_or_ID>`
  - `openstack loadbalancer member set --no-tag <pool_name_or_ID> <member_name_or_ID>`

- ```
openstack loadbalancer healthmonitor set --no-tag
<healthmon_name_or_ID>
```
- ```
openstack loadbalancer l7policy set --no-tag <l7policy_name_or_ID>
```
- ```
openstack loadbalancer l7rule set --no-tag <l7policy_name_or_ID>
<l7rule_ID>
```

#### 예제

이 예제에서는 로드 밸런서에서 모든 태그인 **lb1** 이 제거됩니다.

```
$ openstack loadbalancer set --no-tag lb1
```

#### 검증

- 오브젝트에 적절한 **show** 명령을 사용하여 로드 밸런싱 오브젝트에 하나 이상의 태그를 추가하거나 제거했는지 확인합니다.

#### 예제

이 예제에서 **show** 명령은 **loadbalancer, lb1** 에서 실행됩니다.

```
$ openstack loadbalancer show lb1
```

#### 샘플 출력

```
+-----+-----+
| admin_state_up | True |
| availability_zone | None |
| created_at      | 2022-08-26T19:34:15 |
| description     |      |
| flavor_id       | None |
| id              | 7975374b-3367-4436-ab19-2d79d8c1f29b |
| listeners       |      |
| name            | lb1 |
| operating_status | ONLINE |
| pools           |      |
| project_id      | 2eee3b86ca404cdd977281dac385fd4e |
| provider        | amphora |
| provisioning_status | ACTIVE |
```

```
| updated_at      | 2022-08-30T13:30:17 |
| vip_address     | 172.24.3.76         |
| vip_network_id  | 4c241fc4-95eb-491a-affe-26c53a8805cd |
| vip_port_id     | 9978a598-cc34-47f7-ba28-49431d570fd1 |
| vip_qos_policy_id | None                |
| vip_subnet_id   | e999d323-bd0f-4469-974f-7f66d427e507 |
| tags            | Finance              |
|                 | Sales                 |
+-----+-----+
```

- 명령줄 인터페이스 참조예LoadBalancer 설정
- 명령줄 인터페이스 참조예LoadBalancer 표시

### 12.3. 태그를 사용하여 로드 밸런싱 서비스 오브젝트 필터링

**Red Hat OpenStack** 로드 밸런싱 서비스(octavia)를 사용하여 오브젝트 목록을 생성할 수 있습니다. 태그된 오브젝트의 경우 필터링된 목록(오브젝트에 지정된 태그 중 하나 이상에 포함되어 있는지 여부에 따라 오브젝트를 포함하거나 제외하는 목록)을 생성할 수 있습니다. 태그를 사용하여 로드 밸런서, 리스너, 풀, 멤버, 상태 모니터, 규칙 및 정책을 필터링할 수 있다는 것은 로드 밸런싱 서비스 리소스의 사용, 할당 및 유지 관리를 관리하는 시작점이 될 수 있습니다.

#### 프로세스

1. 자격 증명 파일을 가져옵니다.

#### 예제

```
$ source ~/overcloudrc
```

2. 태그 옵션 중 하나를 사용하여 오브젝트에 대한 적절한 로드 밸런서 목록 명령을 실행하여 나열 하려는 오브젝트를 필터링합니다.

표 12.1. 오브젝트 필터링을 위한 태그 옵션

| 내 목록에서는...                       | 예                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 지정된 모든 태그와 일치하는 오브젝트를 포함합니다.     | <b>\$ OpenStack 로드 밸런서 목록 --tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer listener list --tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer l7pool list --tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer member list --tags Sales,Finance pool1</b><br><b>\$ OpenStack loadbalancer healthmonitor list --tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer l7policy list --tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer l7rule list --tags Sales,Finance policy1</b>                                     |
| 하나 이상의 지정된 태그와 일치하는 오브젝트를 포함합니다. | <b>\$ OpenStack loadbalancer list --any-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer listener list --any-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer l7pool list --any-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer member list --any-tags Sales,Finance pool1</b><br><b>\$ OpenStack loadbalancer healthmonitor list --any-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer l7policy list --any-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer l7rule list --any-tags Sales,Finance policy1</b> |

| 내 목록에서는...                       | 예                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 지정된 모든 태그와 일치하는 오브젝트를 제외합니다.     | <b>\$ OpenStack 로드 밸런서 목록 --not-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer listener list --not-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer l7pool list --not-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer member list --not-tags Sales,Finance pool1</b><br><b>\$ OpenStack loadbalancer healthmonitor list --not-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer l7policy list --not-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer l7rule list --not-tags Sales,Finance policy1</b>                                                 |
| 하나 이상의 지정된 태그와 일치하는 오브젝트를 제외합니다. | <b>\$ OpenStack 로드 밸런서 목록 --not-any-any-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer listener list --not-any-any-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer l7pool list --not-any-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer member list --not-any-any-tags Sales,Finance pool1</b><br><b>\$ OpenStack loadbalancer healthmonitor list --not-any-any-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer l7policy list --not-any-tags Sales,Finance</b><br><b>\$ OpenStack loadbalancer l7rule list --not-any-any-tags Sales,Finance policy1</b> |



## 참고

두 개 이상의 태그를 지정하는 경우 쉼표를 사용하여 태그를 구분합니다.

## 추가 리소스



명령줄 인터페이스 참조의 **LoadBalancer** 목록

## 13장. 에지에서 네트워크 트래픽 로드 밸런싱을 위한 가용성 영역 생성

가용성 영역에서 로드 밸런서를 생성하여 트래픽 처리량을 늘리고 RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 사용하여 대기 시간을 줄일 수 있습니다.

이 섹션에 포함된 항목은 다음과 같습니다.

- [로드 밸런싱 서비스 가용성 영역 생성](#)
- [가용성 영역에서 로드 밸런서 생성](#)

### 13.1. 로드 밸런싱 서비스를 위한 가용성 영역 생성

RHOSP(Red Hat OpenStack Platform) 로드 밸런싱 서비스(octavia)를 사용하면 RHOSP 관리자가 프로젝트 사용자가 분산 컴퓨팅 노드(DCN)에서 로드 밸런서를 생성하여 트래픽 처리량을 늘리고 대기 시간을 줄일 수 있는 AZ(가용성 영역)를 생성할 수 있습니다.

로드 밸런싱 서비스 AZ를 생성하려면 두 단계가 필요합니다. RHOSP 관리자는 먼저 AZ 프로필을 생성한 다음 프로필을 사용하여 사용자에게 표시되는 로드 밸런싱 서비스 AZ를 생성해야 합니다.

AZ 프로필에는 다음이 있어야 합니다.

- Compute 서비스(nova) AZ의 이름입니다.
- 사용할 관리 네트워크입니다.

각 AZ에 대해 하나의 고유 네트워크인 여러 관리 네트워크가 있습니다. 중앙 AZ는 기존 부하 분산 관리 네트워크인 lb-mgmt-net 을 사용하고 추가 AZ는 해당 네트워크인 lb-mgmt-<AZ\_name>-net 을 사용합니다(예: lb-mgmt-az-dcn1-net, lb-#159-az-dcn2-net 등).

사전 요구 사항

- octavia-dcn-deployment.yaml Ansible 플레이북을 실행하여 필요한 네트워킹 리소스가 생성된 DCN 환경이 있어야 합니다.

자세한 내용은 **DCN(Distributed Compute Node) 아키텍처 배포**의 **로드 밸런싱 서비스 가용성 영역에 대한 네트워크 리소스 생성**을 참조하십시오.

- 로드 밸런싱 서비스 공급자 드라이버는 **amphora**여야 합니다. **OVN** 공급자 드라이버는 **AZ**를 지원하지 않습니다.
- **admin** 역할이 있는 **RHOSP** 사용자여야 합니다.

## 프로세스

1. 자격 증명 파일을 가져옵니다.

## 예제

```
$ source ~/centralrc
```

2. 로드 밸런싱 서비스 **AZ**의 이름을 수집하는 데 사용할 **Compute** 서비스 **AZ**의 이름을 수집합니다.



## 중요

생성하는 로드 밸런싱 서비스 **AZ**의 이름은 **Compute** 서비스 **AZ**의 이름과 일치해야 합니다.

```
$ openstack availability zone list --compute
```

## 샘플 출력

```
+-----+-----+
| Zone Name | Zone Status |
+-----+-----+
| az-central | available |
| az-dcn1 | available |
```

```
| az-dcn2 | available |
| internal | available |
+-----+-----+
```

3.

로드 밸런싱 서비스 **AZ**를 생성하는 데 사용할 관리 네트워크의 **ID**를 수집합니다.

```
$ openstack network list -c Name -c ID
```

샘플 출력

```
+-----+-----+
| ID                  | Name                  |
+-----+-----+
| 10458d6b-e7c9-436f-92d9-711677c9d9fd | lb-mgmt-az-dcn2-net   |
| 662a94f5-51eb-4a4c-86c4-52dcbf471ef9 | lb-mgmt-net           |
| 6b97ef58-2a25-4ea5-931f-b7c07cd09474 | lb-mgmt-backbone-net  |
| 99f4215b-fad8-432d-8444-1f894154dc30 | heat_tempestconf_network |
| a2884aaf-846c-4936-9982-3083f6a71d9b | lb-mgmt-az-dcn1-net   |
| d7f7de6c-0e84-49e2-9042-697fa85d2532 | public                |
| e887a9f9-15f7-4854-a797-033cedbfe5f3 | public2               |
+-----+-----+
```

4.

**AZ** 프로필을 만듭니다. 이 단계를 반복하여 생성할 각 로드 밸런싱 서비스 **AZ**에 대한 **AZ** 프로필을 생성합니다.

```
$ openstack loadbalancer availabilityzoneprofile create \
--name <AZ_profile_name> --provider amphora --availability-zone-data '{"compute_zone": "
<compute_AZ_name>","management_network": "<lb_mgmt_AZ_net_UUID>"}
```

예 - **az-central**의 프로필 생성

이 예에서는 **Compute AZ(az-central)**에서 실행되는 컴퓨팅 노드의 관리 네트워크(**lb-mgmt-net**)를 사용하는 **AZ** 프로필(**az\_profile\_central**)이 생성됩니다.

```
$ openstack loadbalancer availabilityzoneprofile create \
--name az_profile_central --provider amphora --availability-zone-data \
'{"compute_zone": "az-central","management_network": \
"662a94f5-51eb-4a4c-86c4-52dcbf471ef9"}'
```

5.

4단계를 반복하여 생성하려는 각 로드 밸런싱 서비스 **AZ**에 대한 **AZ** 프로필을 생성합니다.

예 - **az-dcn1**의 프로필 생성

이 예에서는 **Compute AZ(az-dcn1)**에서 실행되는 컴퓨팅 노드의 관리 네트워크(**lb-mgmt-az-dcn1-net**)를 사용하는 **AZ** 프로필(**az-profile-dcn1**)이 생성됩니다.

```
$ openstack loadbalancer availabilityzoneprofile create \
--name az-profile-dcn1 --provider amphora --availability-zone-data \
'{"compute-zone": "az-dcn1","management-network": \
"a2884aaf-846c-4936-9982-3083f6a71d9b"}'
```

예 - **az-dcn2**의 프로필 생성

이 예에서는 **Compute AZ(az-dcn2)**에서 실행되는 컴퓨팅 노드의 관리 네트워크(**lb-mgmt-az-dcn2-net**)를 사용하는 **AZ** 프로필(**az-profile-dcn2**)이 생성됩니다.

```
$ openstack loadbalancer availabilityzoneprofile create \
--name az-profile-dcn2 --provider amphora --availability-zone-data \
'{"compute-zone": "az-dcn2","management-network": \
"10458d6b-e7c9-436f-92d9-711677c9d9fd"}'
```

6.

**AZ** 프로필을 사용하여 로드 밸런싱 서비스 **AZ**를 생성합니다. 각 **AZ**에 적절한 프로필을 사용하여 추가 **AZ**에 대해 이 단계를 반복합니다.

예 - **AZ** 생성: **az-central**

이 예에서는 **AZ** 프로필(**az-profile-central**)을 사용하여 로드 밸런싱 서비스 **AZ(az-central)**가 생성됩니다.

```
$ openstack loadbalancer availabilityzone create --name az-central \
--availabilityzoneprofile az-profile-central \
--description "AZ for Headquarters" --enable
```

예 - **AZ** 생성: **az-dcn1**

이 예에서는 **AZ** 프로필(**az-profile-az-dcn1**)을 사용하여 로드 밸런싱 서비스 **AZ(az-dcn1)**가 생성됩니다.

```
$ openstack loadbalancer availabilityzone create --name az-dcn1 \
--availabilityzoneprofile az-profile-az-dcn1 \
--description "AZ for South Region" --enable
```

**예 - AZ 생성: az-dcn2**

이 예에서는 **AZ** 프로파일(**az-profile-az-dcn2**)을 사용하여 로드 밸런싱 서비스 **AZ**(**az-dcn2**)가 생성됩니다.

```
$ openstack loadbalancer availabilityzone create --name az-dcn2 \
--availabilityzoneprofile az-profile-az-dcn2 \
--description "AZ for North Region" --enable
```

**검증**

- **AZ**(**az-central**)가 생성되었는지 확인합니다. 각 **AZ**에 적절한 이름을 사용하여 추가 **AZ**에 대해 이 단계를 반복합니다.

**예 - az-central 확인**

```
$ openstack loadbalancer availabilityzone show az-central
```

**샘플 출력**

```
+-----+-----+
| Field          | Value                                |
+-----+-----+
| name           | az-central                          |
| availability_zone_profile_id | 5ed25d22-52a5-48ad-85ec-255910791623 |
| enabled        | True                                |
| description     | AZ for Headquarters                 |
+-----+-----+
```

**예 - az-dcn1 확인**

```
$ openstack loadbalancer availabilityzone show az-dcn1
```

## 샘플 출력

```
+-----+
| Field          | Value                               |
+-----+
| name           | az-dcn1                             |
| availability_zone_profile_id | e0995a82-8e67-4cea-b32c-256cd61f9cf3 |
| enabled        | True                                |
| description     | AZ for South Region                 |
+-----+
```

## 예 - az-dcn2 확인

```
$ openstack loadbalancer availabilityzone show az-dcn2
```

## 샘플 출력

```
+-----+
| Field          | Value                               |
+-----+
| name           | az-dcn2                             |
| availability_zone_profile_id | 306a4725-7dac-4046-8f16-f2e668ee5a8d |
| enabled        | True                                |
| description     | AZ for North Region                 |
+-----+
```

## 다음 단계

- [가용성 영역에서 로드 밸런서 생성](#)

## 추가 리소스

- **DCN(Distributed Compute Node) 아키텍처** 가이드에서 **로드 밸런싱 서비스 가용성 영역의 네트워크 리소스 생성**.
- 명령줄 인터페이스 참조에서 **LoadBalancer availabilityzoneprofile create**
- 명령줄 인터페이스 참조에서 **LoadBalancer availabilityzone 생성**

## 13.2. 가용성 영역에서 로드 밸런서 생성

**RHOSP(Red Hat OpenStack Platform)** 로드 밸런싱 서비스(**octavia**)를 사용하면 분산 컴퓨팅 노드(**DCN**) 환경에서 가용성 영역(**AZ**)에 로드 밸런서를 생성하여 트래픽 처리량을 늘리고 대기 시간을 줄일 수 있습니다.

### 사전 요구 사항

- **RHOSP** 관리자가 제공하는 로드 밸런싱 서비스 **AZ**가 있어야 합니다.
- 로드 밸런서와 연결된 **VIP** 네트워크는 로드 밸런서가 멤버인 **AZ**에서 사용할 수 있어야 합니다.

### 프로세스

1. 자격 증명 파일을 가져옵니다.

#### 예제

```
$ source ~/centralrc
```

2. **DCN** 환경의 로드 밸런서를 생성하려면 **--availability-zone** 옵션과 함께 **loadbalancer create** 명령을 사용하고 적절한 **AZ**를 지정합니다.

## 예제

예를 들어 가용성 영역(**az-central**)의 공용 서브넷(**public\_subnet**)에 종료되지 않은 **HTTPS** 로드 밸런서(**lb1**)를 생성하려면 다음 명령을 입력합니다.

```
$ openstack loadbalancer create --name lb1 --vip-subnet-id \
public_subnet --availability-zone az-central
```

3.

리스너, 풀, 상태 모니터 및 로드 밸런서 멤버를 추가하여 로드 밸런서를 계속 생성합니다.

## 검증

•

로드 밸런서(**lb1**)가 가용성 영역(**az-central**)의 멤버인지 확인합니다.

## 예제

```
$ openstack loadbalancer show lb1
```

## 샘플 출력

```
+-----+-----+
| Field      | Value                                |
+-----+-----+
| admin_state_up | True                                |
| availability_zone | az-central                          |
| created_at    | 2023-07-12T16:35:05                |
| description   |                                     |
| flavor_id     | None                                |
| id            | 85c7e567-a0a7-4fcb-af89-a0bbc9abe3aa |
| listeners    |                                     |
| name          | lb1                                 |
| operating_status | ONLINE                              |
| pools        |                                     |
| project_id    | d303d3bda9b34d73926dc46f4d0cb4bc   |
| provider      | amphora                             |
| provisioning_status | ACTIVE                              |
| updated_at    | 2023-07-12T16:36:45                |
| vip_address   | 10.101.10.229                      |
| vip_network_id | d7f7de6c-0e84-49e2-9042-697fa85d2532 |
| vip_port_id   | 7f916764-d171-4317-9c86-a1750a54b16e |
```

```
| vip_qos_policy_id | None |
| vip_subnet_id    | a421cbcf-c5db-4323-b7ab-1df20ee6acab |
| tags             | |
+-----+-----+
```

## 추가 리소스

- [로드 밸런싱 서비스를 위한 가용성 영역 생성](#)
- [명령줄 인터페이스 참조의 LoadBalancer](#) [https://access.redhat.com/documentation/en-us/red\\_hat\\_openstack\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/17.1/html/command_line_interface_reference/loadbalancer)

## 14장. 로드 밸런싱 서비스 업데이트 및 업그레이드

최신 **Red Hat OpenStack Platform** 로드 밸런싱 서비스(**octavia**) 기능 및 버그 수정을 받을 수 있도록 정기적인 업데이트 및 업그레이드를 수행합니다.

- [14.1절. “로드 밸런싱 서비스 업데이트 및 업그레이드”](#)
- [14.2절. “실행 중인 로드 밸런싱 서비스 인스턴스 업데이트”](#)

### 14.1. 로드 밸런싱 서비스 업데이트 및 업그레이드

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

#### 사전 요구 사항

- 업그레이드 중에 로드 밸런싱 서비스 컨트롤 플레인이 완전히 작동하지 않기 때문에 업그레이드를 수행하도록 유지보수 기간을 예약합니다.

#### 프로세스

1. **Red Hat OpenStack Platform** 가이드의 마이너 업데이트 수행에 설명된 대로 **RHOSP** 업데이트를 수행합니다.
2. 유지 관리 릴리스가 적용된 후 새 기능을 사용해야 하는 경우 실행 중인 **amphorae**를 최신 **amphora** 이미지로 업데이트합니다.

#### 추가 리소스

- [Red Hat OpenStack Platform 가이드의 마이너 업데이트 수행](#)
- [14.2절. “실행 중인 로드 밸런싱 서비스 인스턴스 업데이트”](#)
- [2.2절. “로드 밸런싱 서비스\(\*\*octavia\*\*\) 기능 지원 매트릭스”](#)

## 14.2. 실행 중인 로드 밸런싱 서비스 인스턴스 업데이트

주기적으로 최신 이미지로 실행 중인 로드 밸런싱 서비스 인스턴스(**amphora**)를 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 플레이버로 변경합니다.

**RHOSP** 업데이트 또는 업그레이드 중에 **director**는 기본 **amphora** 이미지를 자동으로 다운로드하여 오버클라우드 이미지 서비스(**glance**)에 업로드한 다음 새 이미지를 사용하도록 로드 밸런싱 서비스(**octavia**)를 구성합니다. 로드 밸런서를 통해 실패하면 로드 밸런싱 서비스가 새 **amphora** 이미지를 사용하는 인스턴스(**amphora**)를 시작하도록 강제 적용합니다.

### 사전 요구 사항

- **amphora**를 위한 새로운 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 밸런서에 대해 실패합니다.

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



#### 참고

로드 밸런서를 통해 장애 조치(**failover**)를 시작하고 시스템 사용률을 모니터링하고 필요에 따라 페일오버를 수행하는 속도를 조정합니다. 로드 밸런서 장애 조치로 인해 새 가상 머신 및 포트가 생성되어 **OpenStack Networking**의 로드가 일시적으로 증가할 수 있습니다.

4. 로드 밸런서에서 실패한 상태를 모니터링합니다.

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

로드 밸런서 상태가 **ACTIVE** 이면 업데이트가 완료됩니다.

#### 추가 리소스

- 명령줄 인터페이스 참조의 **LoadBalancer** [https://access.redhat.com/documentation/en-us/red\\_hat\\_openstack\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/17.1/html/command_line_interface_reference/loadbalancer)

## 15장. 로드 밸런싱 서비스 문제 해결 및 유지 관리

로드 밸런싱 서비스(**octavia**)의 기본 문제 해결 및 유지 관리는 상태를 표시하고 인스턴스를 마이그레이션하는 **OpenStack** 클라이언트 명령을 잘 알고 있으며 로그에 액세스하는 방법을 알고 있습니다. 보다 깊이 문제를 해결해야 하는 경우 하나 이상의 로드 밸런싱 서비스 인스턴스(**amphorae**)로 **SSH**를 수행할 수 있습니다.

- 15.1절. “로드 밸런서 확인”
- 15.2절. “서비스 인스턴스 관리 로그 로드 밸런싱”
- 15.3절. “특정 로드 밸런싱 서비스 인스턴스 마이그레이션”
- 15.4절. “SSH를 사용하여 인스턴스 로드 밸런싱에 연결”
- 15.5절. “리스너 통계 표시”
- 15.6절. “리스너 요청 오류 해석”

### 15.1. 로드 밸런서 확인

로드 밸런서 **show** 및 **list** 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

**controlPlane** 내부의 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 단계에서 로드 밸런서 ID(**265d0b71-c073-40f4-9718-8a182c6d53ca**)를 사용하여 로드 밸런서와 연결된 **amphora**의 ID를 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 단계에서 **amphora ID (1afabefd-ba09-49e1-8c39-41770aa25070)**를 사용하여 **amphora** 정보를 봅니다.

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 확인합니다.

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                              |
| crt_container_ref | None                                |
| tls_enabled      | False                              |
+-----+-----+

```

7.

로드 밸런서의 VIP 주소(192.0.2.177)에 연결하여 리스너가 **HTTPS** 또는 **TERMINATED\_HTTPS** 프로토콜에 대해 구성된 로드 밸런서에서 **HTTPS** 트래픽 흐름을 확인합니다.

작은 정보

**openstack loadbalancer show <load\_balancer\_name>** 명령을 사용하여 로드 밸런서 VIP 주소를 가져옵니다.



참고

로드 밸런서 VIP에 대해 구현된 보안 그룹은 필요한 프로토콜 및 포트에 대한 데이터 트래픽만 허용합니다. 따라서 **ICMP** 트래픽이 차단되므로 로드 밸런서 VIP를 **ping**할 수 없습니다.

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 참조의 **LoadBalancer** [https://access.redhat.com/documentation/en-us/red\\_hat\\_openstack\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/17.1/html/command_line_interface_reference/loadbalancer)

## 15.2. 서비스 인스턴스 관리 로그 로드 밸런싱

로드 밸런싱 서비스 인스턴스(**amphora**)의 관리 로그 오프로드 기능은 테넌트 흐름 로그를 제외하고 **amphora** 내부의 모든 시스템 로깅을 다룹니다. 관리 로그가 전송되는 동일한 **syslog** 수신자로 테넌트 흐름 로그를 보낼 수 있습니다. 관리 로그를 처리하는 것과 동일한 **syslog** 수신자로 테넌트 흐름 로그를 보낼 수 있지만 테넌트 흐름 로그를 별도로 구성해야 합니다.

**amphora**는 메시지를 전송하는 애플리케이션에 네이티브 로그 형식을 사용하여 모든 관리 로그 메시지를 보냅니다. 다른 **RHOSP** 로그(**/var/log/containers/octavia-amphorae/**)와 동일한 위치에 있는 **RHOSP**(Red Hat OpenStack Platform) 컨트롤러 노드에 대한 **amphorae** 로그입니다.

## 추가 리소스



**5장. 로드 밸런싱 서비스 인스턴스 로그 관리**

### 15.3. 특정 로드 밸런싱 서비스 인스턴스 마이그레이션

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

#### 절차

1. 자격 증명 파일을 가져옵니다.

#### 예제

```
$ source ~/overcloudrc
```

2. 마이그레이션하려는 **amphora**의 ID를 찾습니다. 이후 단계에서 ID를 제공해야 합니다.

```
$ openstack loadbalancer amphora list
```

3. **Compute** 스케줄러 서비스에서 비우는 컴퓨팅 노드에 새 **amphorae**를 예약하지 못하도록 컴퓨팅 노드 (**compute-host-1**)를 비활성화합니다.



#### 참고

**controlPlane** 내부의 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.

얻은 **amphora ID (ea17210a-1076-48ff-8a1f-ced49ccb5e53)**를 사용하여 **amphora**를 페일 오버하십시오.

예제

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

추가 리소스

- [명령줄 인터페이스 참조에 설정된 Compute 서비스](#)
- [명령줄 인터페이스 참조의 LoadBalancer \[https://access.redhat.com/documentation/en-us/red\\\_hat\\\_openstack\\\_platform/17.1/html/command\\\_line\\\_interface\\\_reference/loadbalancer\]\(https://access.redhat.com/documentation/en-us/red\_hat\_openstack\_platform/17.1/html/command\_line\_interface\_reference/loadbalancer\)](#)

#### 15.4. SSH를 사용하여 인스턴스 로드 밸런싱에 연결

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

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

사전 요구 사항

- 로드 밸런싱 서비스(**octavia**) **SSH** 개인 키가 있어야 합니다.

절차

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

```
$ eval $(ssh-agent -s)
$ sudo -E ssh-add /etc/octavia/ssh/octavia_id_rsa
```

2.

자격 증명 파일을 가져옵니다.

예제

```
$ source ~/overcloudrc
```

3.

연결하려는 **amphora**의 로드 밸런싱 관리 네트워크 (**lb\_network\_ip**)에서 IP 주소를 확인합니다.

```
$ openstack loadbalancer amphora list
```

4.

**SSH**를 사용하여 **amphora**에 연결합니다.

```
$ ssh -A -t tripleo-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 참조의 LoadBalancer [https://access.redhat.com/documentation/en-us/red\\_hat\\_openshift\\_platform/17.1/html/command\\_line\\_interface\\_reference/loadbalancer](https://access.redhat.com/documentation/en-us/red_hat_openshift_platform/17.1/html/command_line_interface_reference/loadbalancer)

## 15.5. 리스너 통계 표시

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

- 

현재 활성 연결(**active\_connections**).

- 수신된 총 바이트(**바이트\_in**)입니다.
- 전송된 총 바이트(**바이트\_out**).
- 수행할 수 없는 총 요청(**request\_errors**).
- 처리된 총 연결(**total\_connections**).

## 절차

1. 자격 증명 파일을 가져옵니다.

### 예제

```
$ source ~/overcloudrc
```

2. 리스너(**listener1**)의 통계를 확인합니다.



### 참고

**controlPlane** 내부의 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 **loadbalancer listener** 목록을 입력합니다.

## 샘플 출력

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

## 추가 리소스

- [명령줄 인터페이스 참조](#)에 **LoadBalancer 리스너 통계**가 표시됨
- [15.6절. “리스너 요청 오류 해석”](#)

## 15.6. 리스너 요청 오류 해석

특정 **RHOSP(Red Hat OpenStack Platform)** 로드 밸런서의 리스너에 대한 통계를 얻을 수 있습니다. 자세한 내용은 [15.5절. “리스너 통계 표시”](#)의 내용을 참조하십시오.

**RHOSP** 로드 밸런서에서 추적하는 통계 중 하나인 **request\_errors**는 로드 밸런서에 연결하는 최종 사용자의 요청에 발생한 오류만 계산합니다. **request\_errors** 변수는 멤버 서버에서 보고한 오류를 측정하지 않습니다.

예를 들어 테넌트가 **RHOSP** 로드 밸런싱 서비스(**octavia**)를 통해 **HTTP** 상태 코드 **400(Bad Request)**을 반환하는 웹 서버에 연결하는 경우 이 오류는 로드 밸런싱 서비스에 의해 수집되지 않습니다. **LoadBalancer**s는 데이터 트래픽의 콘텐츠를 검사하지 않습니다. 이 예에서 로드 밸런서는 사용자와 웹 서버 간에 정보를 올바르게 전송했기 때문에 이 흐름을 성공으로 해석합니다.

다음 조건으로 인해 **request\_errors** 변수가 증가할 수 있습니다.

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

추가 리소스

- [명령줄 인터페이스 참조에 LoadBalancer 리스너 통계](#)가 표시됨
- [15.5절. “리스너 통계 표시”](#)