Red Hat Enterprise Linux 8

Red Hat OpenStack Platform에서 Red Hat High Availability 클러스터 구성

RHOSP 인스턴스에 HA 클러스터 및 클러스터 리소스 설치 및 구성

Last Updated: 2023-09-26

Red Hat Enterprise Linux 8 Red Hat OpenStack Platform에서 Red Hat High Availability 클러스터 구성

RHOSP 인스턴스에 HA 클러스터 및 클러스터 리소스 설치 및 구성

법적 공지

Copyright © 2023 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 ® is the registered trademark of Linus Torvalds in the United States and other countries.

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

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

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

Node.js ® 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 ® 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

Red Hat High Availability Add-On을 사용하여 RHOSP(Red Hat OpenStack Platform) 인스턴스에서 HA(고가용성) 클러스터를 구성할 수 있습니다. 이 제목에서는 필요한 패키지 및 에이전트를 설치하는 방법과 기본 클러스터, 펜싱 리소스 및 HA 클러스터 리소스를 구성하는 예제를 설명합니다.

차례

보다 포괄적 수용을 위한 오픈 소스 용어 교체

Red Hat은 코드, 문서 및 웹 속성에서 문제가 있는 언어를 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 의견에 감사드립니다. 어떻게 개선할 수 있는지 알려주십시오.

특정 문구에 대한 의견 제출

1. **Multi-page HTML** 형식으로 설명서를 보고 페이지가 완전히 로드된 후 오른쪽 상단 모서리에 **피드백** 버튼이 표시되는지 확인합니다.
2. 커서를 사용하여 주석 처리할 텍스트 부분을 강조 표시합니다.
3. 강조 표시된 텍스트 옆에 표시되는 **피드백 추가** 버튼을 클릭합니다.
4. 의견을 추가하고 **제출**을 클릭합니다.

Jira를 통해 피드백 제출 (등록 필요)

1. Jira 웹 사이트에 로그인합니다.
2. 상단 탐색 모음에서 **생성** 을 클릭합니다.
3. **요약** 필드에 설명 제목을 입력합니다.
4. **설명** 필드에 개선을 위한 제안을 입력합니다. 문서의 관련 부분에 대한 링크를 포함합니다.
5. 대화 상자 하단에서 **생성** 을 클릭합니다.

1장. 머리말

Red Hat High Availability Add-On을 사용하여 RHOSP(Red Hat OpenStack Platform) 인스턴스에서 HA(Red Hat High Availability) 클러스터를 구성할 수 있습니다. 이를 위해서는 필수 패키지 및 에이전트를 설치하고, 기본 클러스터를 구성하고, 펜싱 리소스를 구성하고, HA 클러스터 리소스를 구성해야 합니다.

RHOSP 문서는 Red Hat Openstack Platform 제품 설명서를 참조하십시오 .

RHEL High Availability 클러스터에서 RHOSP 인스턴스 사용에 적용되는 Red Hat의 정책, 요구 사항 및 제한 사항은 RHEL High Availability Clusters - OpenStack Virtual Machines as Cluster Members - Red Hat Customer Portal 을 참조하십시오.

2장. HA 인스턴스의 RHOSP 서버 그룹 구성

RHOSP HA 클러스터 노드 인스턴스를 생성하기 전에 인스턴스 서버 그룹을 생성합니다. 선호도 정책에 따라 인스턴스를 그룹화합니다. 여러 클러스터를 구성하는 경우 클러스터당 하나의 서버 그룹만 있어야 합니다.

서버 그룹에 설정한 선호도 정책은 하이퍼바이저가 실패하는 경우 클러스터가 계속 작동하는지 여부를 결정할 수 있습니다.

기본 선호도 정책은 **유사성** 입니다. 이 유사성 정책을 사용하면 모든 클러스터 노드를 동일한 RHOSP 하이퍼바이저에서 생성할 수 있었습니다. 이 경우 하이퍼바이저가 실패하면 전체 클러스터가 실패합니다. 따라서 anti-affinity 또는 **soft- anti-affinity 의 서버 그룹에 대한 선호도** 정책을 설정합니다.

- 유사성 **방지 정책의 유사성** 정책을 사용하면 서버 그룹에서 컴퓨팅 노드당 클러스터 노드를 하나만 허용합니다. 컴퓨팅 노드보다 더 많은 클러스터 노드를 생성하려고 하면 오류가 발생합니다. 이 구성은 RHOSP 하이퍼바이저 장애에 대해 최고 수준의 보호 기능을 제공하지만 사용 가능한 것보다 대규모 클러스터를 배포하기 위해 더 많은 리소스가 필요할 수 있습니다.
- **soft-anti-affinity** 의 유사성 정책을 사용하면 서버 그룹이 모든 컴퓨팅 노드에서 가능한 한 균등하게 클러스터 노드를 배포합니다. 이 **경우 유사성 방지 정책보다 하이퍼바이저 장애에 대한 보호가 줄어들지만 선호도** 정책보다 높은 고가용성을 제공합니다.

배포의 서버 그룹 유사성 정책을 확인하려면 다음 클러스터 구성 요소를 고려하여 클러스터 요구 사항의 균형을 조정해야 합니다.

- **클러스터의 노드 수**
- **사용 가능한 RHOSP 컴퓨팅 노드 수**
- **클러스터 쿼럼이 클러스터 작업을 유지하는 데 필요한 노드 수**

선호도 및 인스턴스 서버 그룹 생성에 대한 자세한 내용은 Compute 스케줄러 필터 및 명령줄 인터페이스 참조.

3장. 고가용성 및 RHOSP 패키지 및 에이전트 설치

RHOSP(Red Hat OpenStack Platform)에서 Red Hat High Availability 클러스터를 구성하는 데 필요한 패키지를 설치합니다. 클러스터 구성원으로 사용할 각 노드에 패키지를 설치해야 합니다.

사전 요구 사항

- HA 인스턴스의 RHOSP 서버 그룹 구성에 설명된 대로 구성된 HA 클러스터 노드로 사용할 RHOSP 인스턴스의 서버 그룹 구성
- 각 HA 클러스터 노드의 RHOSP 인스턴스
  - 인스턴스는 서버 그룹의 멤버입니다.
  - 인스턴스는 RHEL 8.7 이상을 실행하는 노드로 구성됩니다.

절차

1. RHEL HA 리포지토리 및 RHOSP 툴 채널을 활성화합니다.

```
# subscription-manager repos --enable=rhel-8-for-x86_64-highavailability-rpms
# subscription-manager repos --enable=openstack-16-tools-for-rhel-8-x86_64-rpms
```

2. Red Hat High Availability Add-On 소프트웨어 패키지와 RHOSP 클러스터 리소스 에이전트 및 RHOSP 차단 에이전트에 필요한 패키지를 설치합니다.

```
# yum install pcs pacemaker python3-openstackclient python3-novaclient fence-agents-openstack
```

3. 각 노드에 pcs 및 pacemaker 패키지를 설치하면 pcs 관리 계정인 hacluster 사용자가 생성됩니다. 모든 클러스터 노드에서 사용자 hacluster 의 암호를 생성합니다. 모든 노드에 동일한 암호를 사용하면 클러스터 관리가 간소화됩니다.

```
# passwd hacluster
```

4. firewalld.service 가 설치된 경우 RHEL 방화벽에 고가용성 서비스를 추가합니다.

```
# firewall-cmd --permanent --add-service=high-availability
# firewall-cmd --add-service=high-availability
```

5. **pcs** 서비스를 시작하고 부팅 시 시작되도록 활성화합니다.

```
# systemctl start pcsd.service
# systemctl enable pcsd.service
```

6. **pcs** 서비스가 실행 중인지 확인합니다.

```
# systemctl status pcsd.service
pcsd.service - PCS GUI and remote configuration interface
Loaded: loaded (/usr/lib/systemd/system/pcsd.service; enabled; vendor preset:
disabled)
Active: active (running) since Thu 2018-03-01 14:53:28 UTC; 28min ago
Docs: man:pcsd(8)
man:pcs(8)
Main PID: 5437 (pcsd)
CGroup: /system.slice/pcsd.service
    └─5437 /usr/bin/ruby /usr/lib/pcsd/pcsd > /dev/null &
Mar 01 14:53:27 ip-10-0-0-48.ec2.internal systemd[1]: Starting PCS GUI and remote
configuration interface...
Mar 01 14:53:28 ip-10-0-0-48.ec2.internal systemd[1]: Started PCS GUI and remote
configuration interface.
```

7. **/etc/hosts** 파일을 편집하고 **RHEL** 호스트 이름과 내부 **IP** 주소를 추가합니다. **/etc/hosts** 에 대한 자세한 내용은 **Red Hat Knowledgebase** 솔루션을 참조하십시오. **RHEL** 클러스터 노드에서 **/etc/hosts** 파일을 설정하는 방법은 무엇입니까**?**.

추가 리소스

- **Red Hat** 고가용성 클러스터 구성 및 관리에 대한 자세한 내용은 고가용성 클러스터 구성 및 관리를 참조하십시오.

4장. RHOSP의 인증 방법 설정

고가용성 차단 에이전트 및 리소스 에이전트는 RHOSP와 통신하는 세 가지 인증 방법을 지원합니다.

- clouds.yaml 구성 파일로 인증
- OpenRC 환경 스크립트로 인증
- Pacemaker를 통한 사용자 이름 및 암호로 인증

클러스터에 사용할 인증 방법을 확인한 후 펜싱 또는 클러스터 리소스를 생성할 때 적절한 인증 매개변수를 지정합니다.

4.1. CLOUDS.YAML 파일을 사용하여 RHOSP로 인증

인증에 clouds.yaml 파일을 사용하는 이 문서의 절차에서는 이 절차에 표시된 clouds.yaml 파일을 사용합니다. 이러한 프로시저는 이 파일에 정의된 대로 cloud= 매개변수에 대한 ha-example 을 지정합니다.

절차

1. 다음 예와 같이 클러스터의 일부인 각 노드에서 clouds.yaml 파일을 생성합니다. clouds.yaml 파일 생성에 대한 자세한 내용은 사용자 및 ID 관리 가이드를 참조하십시오.

```
$ cat .config/openstack/clouds.yaml
clouds:
  ha-example:
    auth:
      auth_url: https://<ip_address>:13000/
      project_name: rainbow
      username: unicorns
      password: <password>
      user_domain_name: Default
      project_domain_name: Default
  <. . . additional options . . .>
  region_name: regionOne
  verify: False
```

2. 인증을 성공적으로 수행하고 다음 기본 RHOSP 명령을 사용하여 RHOSP API에 액세스할 수

있는지 테스트하고 **ha-example** 용으로 생성한 **clouds.yaml** 파일에 지정한 클라우드의 이름을 대체합니다. 이 명령이 서버 목록을 표시하지 않으면 **RHOSP** 관리자에게 문의하십시오.

```
$ openstack --os-cloud=ha-example server list
```

3. 클러스터 리소스 또는 펜싱 리소스를 생성할 때 **cloud** 매개변수를 지정합니다.

4.2. OPENRC 환경 스크립트를 사용하여 RHOSP로 인증

OpenRC 환경 스크립트를 사용하여 **RHOSP**로 인증하려면 다음 단계를 수행합니다.

절차

1. 클러스터에 포함될 각 노드에서 **OpenRC** 환경 스크립트를 구성합니다. **OpenRC** 환경 스크립트 생성에 대한 자세한 내용은 **OpenStack RC** 파일을 사용하여 환경 변수 설정을 참조하십시오.

2. 다음 기본 **RHOSP** 명령을 사용하여 인증에 성공했는지 여부와 **RHOSP API**에 액세스할 수 있는지 테스트합니다. 이 명령이 서버 목록을 표시하지 않으면 **RHOSP** 관리자에게 문의하십시오.

```
$ openstack server list
```

3. 클러스터 리소스 또는 펜싱 리소스를 생성할 때 **openrc** 매개변수를 지정합니다.

4.3. 사용자 이름 및 암호를 사용하여 RHOSP로 인증

사용자 이름과 암호를 사용하여 **RHOSP**를 인증하려면 클러스터 리소스에 대한 사용자 이름 ,암호, **auth_url** 매개변수를 지정하거나 리소스를 생성할 때 펜싱 리소스를 지정합니다. **RHOSP** 구성에 따라 추가 인증 매개 변수가 필요할 수 있습니다. **RHOSP** 관리자는 사용할 인증 매개 변수를 제공합니다.

5장. RED HAT OPENSTACK PLATFORM에서 기본 클러스터 생성

이 절차에서는 펜싱 또는 리소스가 구성되지 않은 RHOSP 플랫폼에 고가용성 클러스터를 생성합니다.

사전 요구 사항

- RHOSP 인스턴스는 각 HA 클러스터 노드에 대해 구성됩니다.
- HA 클러스터 노드는 RHEL 8.7 이상을 실행하고 있습니다.
- 고가용성 및 RHOSP 패키지 및 에이전트 설치에 설명된 대로 고가용성 및 RHOSP 패키지는 각 노드에 설치됩니다.

절차

1. 클러스터 노드 중 하나에서 다음 명령을 입력하여 pcs 사용자 hacluster 를 인증합니다. 클러스터의 각 노드의 이름을 지정합니다. 이 예에서 클러스터의 노드는 node01,node02 및 node03 입니다.

```
[root@node01 ~]# pcs host auth node01 node02 node03
Username: hacluster
Password:
node01: Authorized
node02: Authorized
node03: Authorized
```

2. 클러스터를 생성합니다. 이 예제에서 클러스터 이름은 newcluster 입니다.

```
[root@node01 ~]# pcs cluster setup newcluster node01 node02 node03

...

Synchronizing pcsd certificates on nodes node01, node02, node03...
node02: Success
node03: Success
node01: Success
Restarting pcsd on the nodes in order to reload the certificates...
node02: Success
node03: Success
node01: Success
```

검증

1. 클러스터를 활성화합니다.

```
[root@node01 ~]# pcs cluster enable --all
node01: Cluster Enabled
node02: Cluster Enabled
node03: Cluster Enabled
```

2. 클러스터를 시작합니다. 명령의 출력은 클러스터가 각 노드에서 시작되었는지 여부를 나타냅니다.

```
[root@node01 ~]# pcs cluster start --all
node02: Starting Cluster…
node03: Starting Cluster…
node01: Starting Cluster...
```

6장. RED HAT OPENSTACK PLATFORM에서 HA 클러스터에 대한 펜싱 구성

펜싱 구성을 사용하면 HA 클러스터에서 오작동하는 노드가 자동으로 격리됩니다. 이렇게 하면 노드가 클러스터 리소스를 사용하거나 클러스터의 기능을 손상시키지 않습니다.

fence_openstack 차단 에이전트를 사용하여 RHOSP에서 HA 클러스터의 차단 장치를 구성합니다. 다음 명령을 사용하여 RHOSP 차단 에이전트의 옵션을 볼 수 있습니다.

```
# pcs stonith describe fence_openstack
```

사전 요구 사항

- RHOSP에서 실행되는 구성된 HA 클러스터
- RHOSP의 인증 방법 설정에 설명된 대로 클러스터 구성에 사용할 RHOSP 인증 방법을 사용하여 RHOSP API에 액세스할수 있습니다.
- 클러스터 속성 stonith-enabled 를 true 로 설정합니다. 이 값이 기본값입니다. Red Hat은 프로덕션 환경에 적합하지 않으므로 펜싱이 비활성화된 경우 클러스터를 지원하지 않습니다. 다음 명령을 실행하여 펜싱이 활성화되었는지 확인합니다.

```
# pcs property config --all
Cluster Properties:
. . .
stonith-enabled: true
```

절차

클러스터의 모든 노드에서 다음 단계를 완료합니다.

1. 클러스터의 각 노드의 UUID를 확인합니다.

   다음 명령은 ha-example 프로젝트 내의 모든 RHOSP 인스턴스 이름 전체 목록과 해당 RHOSP 인스턴스와 연결된 클러스터 노드의 UUID를 제목 ID 아래에 표시합니다. 노드 호스트 이름이 RHOSP 인스턴스 이름과 일치하지 않을 수 있습니다.

```
# openstack --os-cloud="ha-example" server list
…
| ID                                   | Name          |...
```

```
| 6d86fa7d-b31f-4f8a-895e-b3558df9decb|testnode-node03-vm|...
| 43ed5fe8-6cc7-4af0-8acd-a4fea293bc62|testnode-node02-vm|...
| 4df08e9d-2fa6-4c04-9e66-36a6f002250e|testnode-node01-vm|...
```

2. 클러스터의 각 노드를 해당 노드 의 **UUID**에 매핑하려면 **fencing** 장치를 **IRQmk_host_map** 매개변수 를 사용하여 생성합니다. 다음 예제 **fence** 장치 생성 명령은 각각 다른 인증 방법을 사용합니다.

   a. 다음 명령은 인증을 위해 **clouds.yaml** 구성 파일을 사용하여 **3-**노드 클러스터의 **fence_openstack** 펜싱 장치를 생성합니다. **cloud=** 매개변수 의 경우 **clouds.yaml'** 파일에서 클라우드 이름을 지정합니다.

   ```
   # pcs stonith create fenceopenstack fence_openstack
   pcmk_host_map="node01:4df08e9d-2fa6-4c04-9e66-
   36a6f002250e;node02:43ed5fe8-6cc7-4af0-8acd-a4fea293bc62;node03:6d86fa7d-
   b31f-4f8a-895e-b3558df9decb" power_timeout="240" pcmk_reboot_timeout="480"
   pcmk_reboot_retries="4" cloud="ha-example"
   ```

   b. 다음 명령은 인증에 **OpenRC** 환경 스크립트를 사용하여 **fence_openstack** 펜싱 장치를 생성합니다.

   ```
   # pcs stonith create fenceopenstack fence_openstack
   pcmk_host_map="node01:4df08e9d-2fa6-4c04-9e66-
   36a6f002250e;node02:43ed5fe8-6cc7-4af0-8acd-a4fea293bc62;node03:6d86fa7d-
   b31f-4f8a-895e-b3558df9decb" power_timeout="240" pcmk_reboot_timeout="480"
   pcmk_reboot_retries="4" openrc="/root/openrc"
   ```

   c. 다음 명령은 인증에 사용자 이름과 암호를 사용하여 **fence_openstack** 펜싱 장치를 생성합니다. 사용자 이름 **,**암호 **,project_name, auth_url** 등의 인증 매개 변수는 **RHOSP** 관리자가 제공합니다.

   ```
   # pcs stonith create fenceopenstack fence_openstack
   pcmk_host_map="node01:4df08e9d-2fa6-4c04-9e66-
   36a6f002250e;node02:43ed5fe8-6cc7-4af0-8acd-a4fea293bc62;node03:6d86fa7d-
   b31f-4f8a-895e-b3558df9decb" power_timeout="240" pcmk_reboot_timeout="480"
   pcmk_reboot_retries="4" username="XXX" password="XXX"
   project_name="rhelha" auth_url="XXX" user_domain_name="Default"
   ```

검증

1. 클러스터의 한 노드에서 클러스터의 다른 노드를 펜싱하고 클러스터 상태를 확인합니다. 펜싱된 노드가 오프라인 상태인 경우 펜싱 작업에 성공했습니다.

```
[root@node01 ~] # pcs stonith fence node02
[root@node01 ~] # pcs status
```

2. 펜싱한 노드를 재시작하고 상태를 확인하여 노드가 시작되었는지 확인합니다.

```
[root@node01 ~] # pcs cluster start node02
[root@node01 ~] # pcs status
```

7장. RED HAT OPENSTACK PLATFORM에서 HA 클러스터 리소스 구성

다음 표에는 RHOSP에서 HA 클러스터의 리소스를 구성하는 데 사용하는 RHOSP별 리소스 에이전트가 나열되어 있습니다.

openstack-info (필수)	RHOSP별 리소스 에이전트를 지원합니다. **fence_openstack** 차단 에이전트 이외의 다른 RHOSP별 리소스 에이전트를 실행하려면 **openstack-info** 리소스를 클러스터의 복제 리소스로 구성해야 합니다.**openstack-info** 리소스 구성에 대한 자세한 내용은Red Hat OpenStack Platform의 HA 클러스터에서 **openstack-info** 리소스 구성 을 참조하십시오.
openstack-virtual-ip	가상 IP 주소 리소스를 구성합니다. **openstack-virtual-ip** 리소스 구성에 대한 자세한 내용은 Red Hat Openstack Platform의 HA 클러스터에서 가상 IP 주소 구성 을 참조하십시오.
openstack-floating-ip	유동 IP 주소 리소스를 구성합니다. **openstack-floating-ip** 리소스 구성에 대한 자세한 내용은 Red Hat OpenStack Platform의 HA 클러스터에서 유동 IP 주소 구성 을 참조하십시오.
openstack-cinder-volume	블록 스토리지 리소스를 구성합니다. **openstack-cinder-volume** 리소스를 구성하는 방법에 대한 자세한 내용은 Red Hat OpenStack Platform의 HA 클러스터에서 블록 스토리지 리소스 구성 을 참조하십시오.

다른 클러스터 리소스를 구성할 때 표준 **Pacemaker** 리소스 에이전트를 사용합니다.

7.1. RED HAT OPENSTACK PLATFORM의 HA 클러스터에서 OPENSTACK-INFO 리소스 구성 (필수)

fence_openstack 차단 에이전트를 제외한 다른 **RHOSP**별 리소스 에이전트를 실행하려면 **openstack-info** 리소스를 구성해야 합니다.

openstack-info 리소스를 생성하는 다음 절차에서는 **RHOSP** 인증을 위해 **clouds.yaml** 파일을 사용합니다.

사전 요구 사항

- **RHOSP**에서 실행되는 구성된 **HA** 클러스터
- **RHOSP**의 인증 방법 설정에 설명된 대로 클러스터 구성에 사용할 **RHOSP** 인증 방법을 사용하여 **RHOSP API**에 액세스할수 있습니다.

절차

클러스터의 모든 노드에서 다음 단계를 완료합니다.

1. **openstack-info** 리소스 에이전트의 옵션을 보려면 다음 명령을 실행합니다.

```
# pcs resource describe openstack-info
```

2. **openstack-info** 리소스를 복제 리소스로 생성합니다. 이 예에서 리소스의 이름은 **openstack-info** 이기도 합니다. 이 예에서는 **clouds.yaml** 구성 파일을 사용하고 **cloud=** 매개변수는 **clouds.yaml** 파일의 클라우드 이름으로 설정됩니다.

```
# pcs resource create openstack-info openstack-info cloud="ha-example" clone
```

3. 클러스터 상태를 확인하여 리소스가 실행 중인지 확인합니다.

```
# pcs status

Full List of Resources:

    * Clone Set: openstack-info-clone [openstack-info]:
      * Started: [ node01 node02 node03 ]
```

7.2. RED HAT OPENSTACK PLATFORM의 HA 클러스터에서 가상 IP 주소 구성

RHOSP 플랫폼에서 **HA** 클러스터에 대한 **RHOSP** 가상 **IP** 주소 리소스를 생성하는 다음 절차에서는 **RHOSP** 인증을 위해 **clouds.yaml** 파일을 사용합니다.

RHOSP 가상 **IP** 리소스는 **IPaddr2** 클러스터 리소스와 함께 작동합니다. **RHOSP** 가상 **IP** 주소 리소스를 구성하면 리소스 에이전트는 **RHOSP** 인프라가 가상 **IP** 주소를 네트워크의 클러스터 노드와 연결하도록 합니다. 이를 통해 **IPaddr2** 리소스가 해당 노드에서 작동할 수 있습니다.

사전 요구 사항

- **RHOSP**에서 실행되는 구성된 **HA** 클러스터
- 가상 **IP** 주소로 사용할 할당된 **IP** 주소

- RHOSP의 인증 방법 설정에 설명된 대로 클러스터 구성에 사용할 RHOSP 인증 방법을 사용하여 RHOSP API에 액세스할수 있습니다.

절차

클러스터의 모든 노드에서 다음 단계를 완료합니다.

1. openstack-virtual-ip 리소스 에이전트의 옵션을 보려면 다음 명령을 실행합니다.

```
# pcs resource describe openstack-virtual-ip
```

2. 다음 명령을 실행하여 사용 중인 가상 IP 주소의 서브넷 ID를 확인합니다. 이 예에서 가상 IP 주소는 172.16.0.119입니다.

```
# openstack --os-cloud=ha-example subnet list
+--------------------------------------+ ...  +---------------+
| ID                                   | ...  | Subnet        |
+--------------------------------------+ ...  +---------------+
| 723c5a77-156d-4c3b-b53c-ee73a4f75185 | ...  | 172.16.0.0/24  |
+--------------------------------------+ ...  +---------------+
```

3. RHOSP 가상 IP 주소 리소스를 생성합니다.

다음 명령은 IP 주소 172.16.0.119에 대한 RHOSP 가상 IP 주소 리소스를 생성하여 이전 단계에서 결정한 서브넷 ID를 지정합니다.

```
# pcs resource create ClusterIP-osp ocf:heartbeat:openstack-virtual-ip cloud=ha-example ip=172.16.0.119 subnet_id=723c5a77-156d-4c3b-b53c-ee73a4f75185
```

4. 순서 및 위치 제약 조건을 구성합니다.

- 가상 IP 주소 리소스 전에 openstack-info 리소스가 시작되었는지 확인합니다.
- 가상 IP 주소 리소스가 openstack-info 리소스와 동일한 노드에서 실행되는지 확인합니다.

```
# pcs constraint order start openstack-info-clone then ClusterIP-osp
Adding openstack-info-clone ClusterIP-osp (kind: Mandatory) (Options: first-
```

```
action=start then-action=start)
# pcs constraint colocation add ClusterIP-osp with openstack-info-clone
score=INFINITY
```

5. 가상 IP 주소에 대한 IPaddr2 리소스를 생성합니다.

```
# pcs resource create ClusterIP ocf:heartbeat:IPaddr2 ip=172.16.0.119
```

6. IPaddr2 리소스 전에 openstack-virtual-ip 리소스가 시작되고 IPaddr2 리소스가 openstack-virtual-ip 리소스와 동일한 노드에서 실행되도록 순서 및 위치 제약 조건을 구성합니다.

```
# pcs constraint order start ClusterIP-osp then ClusterIP
Adding ClusterIP-osp ClusterIP (kind: Mandatory) (Options: first-action=start then-
action=start)
# pcs constraint colocation add ClusterIP with ClusterIP-osp
```

검증

1. 리소스 제약 조건 구성을 확인합니다.

```
# pcs constraint config
Location Constraints:
Ordering Constraints:
  start ClusterIP-osp then start ClusterIP (kind:Mandatory)
  start openstack-info-clone then start ClusterIP-osp (kind:Mandatory)
Colocation Constraints:
  ClusterIP with ClusterIP-osp (score:INFINITY)
  ClusterIP-osp with openstack-info-clone (score:INFINITY)
```

2. 클러스터 상태를 확인하여 리소스가 실행 중인지 확인합니다.

```
# pcs status
. . .

Full List of Resources:
  * fenceopenstack     (stonith:fence_openstack):      Started node01
  * Clone Set: openstack-info-clone [openstack-info]:
    * Started: [ node01 node02 node03 ]
  * ClusterIP-osp      (ocf::heartbeat:openstack-virtual-ip):   Started node03
  * ClusterIP   (ocf::heartbeat:IPaddr2):       Started node03
```

7.3. RED HAT OPENSTACK PLATFORM의 HA 클러스터에서 유동 IP 주소 구성

다음 절차에서는 RHOSP의 HA 클러스터에 대한 유동 IP 주소 리소스를 생성합니다. 다음 절차에서는 RHOSP 인증에 clouds.yaml 파일을 사용합니다.

사전 요구 사항

- RHOSP에서 실행되는 구성된 HA 클러스터
- RHOSP 관리자가 할당한 유동 IP 주소로 사용할 공용 네트워크의 IP 주소
- RHOSP의 인증 방법 설정에 설명된 대로 클러스터 구성에 사용할 RHOSP 인증 방법을 사용하여 RHOSP API에 액세스할수 있습니다.

절차

클러스터의 모든 노드에서 다음 단계를 완료합니다.

1. openstack-floating-ip 리소스 에이전트의 옵션을 보려면 다음 명령을 실행합니다.

```
# pcs resource describe openstack-floating-ip
```

2. 유동 IP 주소 리소스를 만드는 데 사용할 공용 네트워크에서 주소의 서브넷 ID를 찾습니다.

a. 공용 네트워크는 일반적으로 기본 게이트웨이가 있는 네트워크입니다. 다음 명령을 실행하여 기본 게이트웨이 주소를 표시합니다.

```
# route -n | grep ^0.0.0.0 | awk '{print $2}'
172.16.0.1
```

b. 다음 명령을 실행하여 공용 네트워크의 서브넷 ID를 찾습니다. 이 명령은 ID 및 서브넷 제목이 있는 테이블을 생성합니다.

```
# openstack --os-cloud=ha-example subnet list
+--------------------------------------+---+--------------+
| ID                                   |   | Subnet
+--------------------------------------+---+--------------+
| 723c5a77-156d-4c3b-b53c-ee73a4f75185 |   | 172.16.0.0/24 |
+--------------------------------------+-----------------+
```

3. 리소스의 공용 IP 주소와 해당 주소의 서브넷 ID를 지정하여 유동 IP 주소 리소스를 만듭니다. 유동 IP 주소 리소스를 구성하면 리소스 에이전트는 공용 네트워크에서 가상 IP 주소를 구성하고 이를 클러스터 노드와 연결합니다.

```
# pcs resource create float-ip openstack-floating-ip cloud="ha-example"
ip_id="10.19.227.211" subnet_id="723c5a77-156d-4c3b-b53c-ee73a4f75185"
```

4. 유동 IP 주소 리소스보다 먼저 openstack-info 리소스가 시작되도록 순서 제한 조건을 구성합니다.

```
# pcs constraint order start openstack-info-clone then float-ip
Adding openstack-info-clone float-ip (kind: Mandatory) (Options: first-action=start
then-action=start
```

5. 유동 IP 주소 리소스가 openstack-info 리소스와 동일한 노드에서 실행되도록 위치 제약 조건을 구성합니다.

```
# pcs constraint colocation add float-ip with openstack-info-clone score=INFINITY
```

검증

1. 리소스 제약 조건 구성을 확인합니다.

```
# pcs constraint config
Location Constraints:
Ordering Constraints:
  start openstack-info-clone then start float-ip (kind:Mandatory)
Colocation Constraints:
  float-ip with openstack-info-clone (score:INFINITY)
```

2. 클러스터 상태를 확인하여 리소스가 실행 중인지 확인합니다.

```
# pcs status
. . .
Full List of Resources:
  * fenceopenstack     (stonith:fence_openstack):      Started node01
  * Clone Set: openstack-info-clone [openstack-info]:
    * Started: [ node01 node02 node03 ]
  * float-ip    (ocf::heartbeat:openstack-floating-ip):  Started node02
```

7.4. RED HAT OPENSTACK PLATFORM의 HA 클러스터에서 블록 스토리지 리소스 구성

다음 절차에서는 RHOSP의 HA 클러스터에 대한 블록 스토리지 리소스를 생성합니다. 다음 절차에서는 RHOSP 인증에 clouds.yaml 파일을 사용합니다.

사전 요구 사항

- RHOSP에서 실행되는 구성된 HA 클러스터
- RHOSP 관리자가 생성한 블록 스토리지 볼륨
- RHOSP의 인증 방법 설정에 설명된 대로 클러스터 구성에 사용할 RHOSP 인증 방법을 사용하여 RHOSP API에 액세스할수 있습니다.

절차

클러스터의 모든 노드에서 다음 단계를 완료합니다.

1. openstack-cinder-volume 리소스 에이전트의 옵션을 보려면 다음 명령을 실행합니다.

```
# pcs resource describe openstack-cinder-volume
```

2. 클러스터 리소스로 구성 중인 블록 스토리지 볼륨의 볼륨 ID를 결정합니다.

다음 명령을 실행하여 각 볼륨의 UUID 및 이름이 포함된 사용 가능한 볼륨 테이블을 표시합니다.

```
# openstack --os-cloud=ha-example volume list
| ID                                   | Name                    |
| 23f67c9f-b530-4d44-8ce5-ad5d056ba926| testvolume-cinder-data-disk |
```

볼륨 이름을 이미 알고 있는 경우 다음 명령을 실행하여 구성 중인 볼륨을 지정할 수 있습니다. ID 필드가 있는 테이블이 표시됩니다.

```
# openstack --os-cloud=ha-example volume show testvolume-cinder-data-disk
```

3. 볼륨 ID를 지정하여 블록 스토리지 리소스를 생성합니다.

```
# pcs resource create cinder-vol openstack-cinder-volume volume_id="23f67c9f-b530-
4d44-8ce5-ad5d056ba926" cloud="ha-example"
```

4. 블록 스토리지 리소스보다 먼저 **openstack-info** 리소스가 시작되도록 순서 제한 조건을 구성합니다.

```
# pcs constraint order start openstack-info-clone then cinder-vol
Adding openstack-info-clone cinder-vol (kind: Mandatory) (Options: first-action=start
then-action=start
```

5. 블록 스토리지 리소스가 **openstack-info** 리소스와 동일한 노드에서 실행되도록 위치 제약 조건을 구성합니다.

```
# pcs constraint colocation add cinder-vol with openstack-info-clone score=INFINITY
```

검증

1. 리소스 제약 조건 구성을 확인합니다.

```
# pcs constraint config
Location Constraints:
Ordering Constraints:
  start openstack-info-clone then start cinder-vol (kind:Mandatory)
Colocation Constraints:
  cinder-vol with openstack-info-clone (score:INFINITY)
```

2. 클러스터 상태를 확인하여 리소스가 실행 중인지 확인합니다.

```
# pcs status
. . .
Full List of Resources:
  * Clone Set: openstack-info-clone [openstack-info]:
    * Started: [ node01 node02 node03 ]
  * cinder-vol  (ocf::heartbeat:openstack-cinder-volume):       Started node03
  * fenceopenstack      (stonith:fence_openstack):       Started node01
```