# Red Hat OpenStack Platform 16.1

## High Availability for Compute Instances

Configure High Availability for Compute Instances

Last Updated: 2023-11-20

# Red Hat OpenStack Platform 16.1 High Availability for Compute Instances

Configure High Availability for Compute Instances

OpenStack Team
rhos-docs@redhat.com

## Legal Notice



## Abstract

This guide describes how to manage Instance High Availability (Instance HA). With Instance HA, Red Hat OpenStack Platform (RHOSP) can automatically evacuate and re-create instances on a different Compute node when a Compute node fails.

# Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE ..... 3

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION ..... 4

CHAPTER 1. INTRODUCTION AND PLANNING AN INSTANCE HA DEPLOYMENT ..... 5
1.1. HOW INSTANCE HA WORKS 5
1.2. PLANNING YOUR INSTANCE HA DEPLOYMENT 5
1.3. INSTANCE HA RESOURCE AGENTS 6

CHAPTER 2. INSTALLING AND CONFIGURING INSTANCE HA ..... 7
2.1. CONFIGURING THE INSTANCE HA ROLE, FLAVOR, AND PROFILE 7
2.2. ENABLING FENCING ON AN OVERCLOUD WITH INSTANCE HA 8
2.3. DEPLOYING THE OVERCLOUD WITH INSTANCE HA 9
2.4. TESTING INSTANCE HA EVACUATION 10
2.5. DESIGNATING INSTANCES TO EVACUATE WITH INSTANCE HA 11
2.6. ADDITIONAL RESOURCES 11

CHAPTER 3. PERFORMING MAINTENANCE ON THE UNDERCLOUD AND OVERCLOUD WITH INSTANCE HA 12
3.1. PREREQUISITES 12
3.2. UNDERCLOUD AND OVERCLOUD SHUTDOWN ORDER 12
3.2.1. Shutting down instances on overcloud Compute nodes 12
3.2.2. Stopping instance HA services on overcloud Compute nodes 13
3.2.3. Shutting down Compute nodes 13
3.2.4. Stopping services on Controller nodes 14
3.2.5. Shutting down Ceph Storage nodes 14
3.2.6. Shutting down Controller nodes 15
3.2.7. Shutting down the undercloud 16
3.3. PERFORMING SYSTEM MAINTENANCE 16
3.4. UNDERCLOUD AND OVERCLOUD STARTUP ORDER 16
3.4.1. Starting the undercloud 17
3.4.2. Starting Controller nodes 17
3.4.3. Starting Ceph Storage nodes 18
3.4.4. Starting Compute nodes 19
3.4.5. Starting instance HA services on overcloud Compute nodes 19
3.4.6. Starting instances on overcloud Compute nodes 20

CHAPTER 4. PERFORMING MAINTENANCE ON COMPUTE NODES AND CONTROLLER NODES WITH INSTANCE HA ..... 22

# MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message.

# PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your input on our documentation. Tell us how we can make it better.

## Using the Direct Documentation Feedback (DDF) function

Use the **Add Feedback** DDF function for direct comments on specific sentences, paragraphs, or code blocks.

1. View the documentation in the *Multi-page HTML* format.
2. Ensure that you see the **Feedback** button in the upper right corner of the document.
3. Highlight the part of text that you want to comment on.
4. Click **Add Feedback**.
5. Complete the **Add Feedback** field with your comments.
6. Optional: Add your email address so that the documentation team can contact you for clarification on your issue.
7. Click **Submit**.

# CHAPTER 1. INTRODUCTION AND PLANNING AN INSTANCE HA DEPLOYMENT

High availability for Compute instances (Instance HA) is a tool that you can use to evacuate instances from a failed Compute node and re-create the instances on a different Compute node.

Instance HA works with shared storage or local storage environments, which means that evacuated instances maintain the same network configuration, such as static IP addresses and floating IP addresses. The re-created instances also maintain the same characteristics inside the new Compute node.

## 1.1. HOW INSTANCE HA WORKS

When a Compute node fails, the overcloud fencing agent fences the node, then the Instance HA agents evacuate instances from the failed Compute node to a different Compute node.

The following events occur when a Compute node fails and triggers Instance HA:

1. At the time of failure, the **IPMI** agent performs first-layer fencing, which includes physically resetting the node to ensure that it shuts down and preventing data corruption or multiple identical instances on the overcloud. When the node is offline, it is considered fenced.

2. After the physical IPMI fencing, the **fence-nova** agent automatically performs second-layer fencing and marks the fenced node with the **"evacuate=yes"** cluster per-node attribute by running the following command:

   ```
   $ attrd_updater -n evacuate -A name="evacuate" host="FAILEDHOST" value="yes"
   ```

   **FAILEDHOST** is the name of the failed Compute node.

3. The **nova-evacuate** agent continually runs in the background and periodically checks the cluster for nodes with the **"evacuate=yes"** attribute. When **nova-evacuate** detects that the fenced node contains this attribute, the agent starts evacuating the node. The evacuation process is similar to the manual instance evacuation process that you can perform at any time.

4. When the failed node restarts after the IPMI reset, the **nova-compute** process on that node also starts automatically. Because the node was previously fenced, it does not run any new instances until Pacemaker un-fences the node.

5. When Pacemaker detects that the Compute node is online, it starts the **compute-unfence-trigger** resource agent on the node, which releases the node and so that it can run instances again.

**Additional resources**

- Evacuating an instance

## 1.2. PLANNING YOUR INSTANCE HA DEPLOYMENT

Before you deploy Instance HA, review the resource names for compliance and configure your storage and networking based on your environment.

- Compute node host names and Pacemaker remote resource names must comply with the W3C naming conventions. For more information, see Declaring Namespaces and Names and Tokens in the W3C documentation.

- Typically, Instance HA requires that you configure shared storage for disk images of instances. Therefore, if you attempt to use the **no-shared-storage** option, you might receive an **InvalidSharedStorage** error during evacuation, and the instances will not start on another Compute node.
  However, if all your instances are configured to boot from an OpenStack Block Storage (**cinder**) volume, you do not need to configure shared storage for the disk image of the instances, and you can evacuate all instances using the **no-shared-storage** option.

  During evacuation, if your instances are configured to boot from a Block Storage volume, any evacuated instances boot from the same volume on another Compute node. Therefore, the evacuated instances immediately restart their jobs because the OS image and the application data are stored on the OpenStack Block Storage volume.

- If you deploy Instance HA in a Spine-Leaf environment, you must define a single **internal_api** network for the Controller and Compute nodes. You can then define a subnet for each leaf. For more information about configuring Spine-Leaf networks, see Creating a roles data file in the *Spine Leaf Networking* guide.

- From Red Hat OpenStack Platform 13 and later, you use director to upgrade Instance HA as a part of the overcloud upgrade. For more information about upgrading the overcloud, see Keeping Red Hat OpenStack Platform Updated guide.

- Disabling Instance HA with the director after installation is not supported. For a workaround to manually remove Instance HA components from your deployment, see the article How can I remove Instance HA components from the controller nodes? .

IMPORTANT

This workaround is not verified for production environments. You must verify the procedure in a test environment before you implement it in a production environment.

## 1.3. INSTANCE HA RESOURCE AGENTS

Instance HA uses the **fence_compute**, **NovaEvacuate**, and **comput-unfence-trigger** resource agents to evacuate and re-created instance if a Compute node fails.

| Agent name | Name inside cluster | Role |
|---|---|---|
| **fence_compute** | **fence-nova** | Marks a Compute node for evacuation when the node becomes unavailable. |
| **NovaEvacuate** | **nova-evacuate** | Evacuates instances from failed nodes. This agent runs on one of the Controller nodes. |
| **Dummy** | **compute-unfence-trigger** | Releases a fenced node and enables the node to run instances again. |

# CHAPTER 2. INSTALLING AND CONFIGURING INSTANCE HA

Red Hat OpenStack Platform (RHOSP) director deploys Instance High Availability (HA). However, you must perform additional steps to configure a new Instance HA deployment on a new overcloud. After you complete the steps, Instance HA will run on a subset of Compute nodes with a custom role.

IMPORTANT

Instance HA is not supported on RHOSP hyperconverged infrastructures (HCI) environments. To use Instance HA in your RHOSP HCI environment, you must designate a subset of the Compute nodes with the ComputeInstanceHA role to use the Instance HA. Red Hat Ceph Storage services must not be hosted on the Compute nodes that host Instance HA.

IMPORTANT

To enable instance HA in a different environment, such as an existing overcloud that uses standard or custom roles, perform only the procedures that are relevant to your deployment and adapt your templates accordingly.

## 2.1. CONFIGURING THE INSTANCE HA ROLE, FLAVOR, AND PROFILE

Before deploying Instance HA, add the Instance HA role to your **roles-data.yaml** file, create the Instance HA flavor, tag each Comput node that you want to manage with Instanc HA with the Instance HA profile, and map the Instance HA role to the Instance HA flavor.

NOTE

You can modify the example file and role names in this procedure according to your environment.

**Procedure**

1. Add the **ComputeInstanceHA** role to your **roles-data.yaml** file and regenerate the file.

```
$ openstack overcloud roles generate -o ~/my_roles_data.yaml Controller Compute
ComputeInstanceHA
```

   The **ComputeInstanceHA** role includes all the services in the default **Compute** role, the **ComputeInstanceHA** services, and the **PacemakerRemote** services.

2. Create the **compute-instance-ha** flavor to tag the Compute nodes to manage with Instance HA.

```
$ source ~/stackrc
$ openstack flavor create --id auto --ram 6144 --disk 40 --vcpus 4 compute-instance-ha
$ openstack flavor set --property "cpu_arch"="x86_64" --property
"capabilities:boot_option"="local" --property "capabilities:profile"="compute-instance-ha"
compute-instance-ha
$ openstack flavor set --property resources:VCPU=0 --property resources:MEMORY_MB=0 -
-property resources:DISK_GB=0 --property resources:CUSTOM_BAREMETAL=1 compute-
instance-ha
```

3. Tag each Compute node that you want to manage with Instance HA with the **compute-instance-ha** profile, and replace **<NODE UUID>** with the actual UUID:

```
$ openstack baremetal node set --property capabilities='profile:compute-instance-ha,boot_option:local' <NODE UUID>
```

4. Map the **ComputeInstanceHA** role to the **compute-instance-ha** flavor by creating an environment file with the following parameter:

```
parameter_defaults:
  OvercloudComputeInstanceHAFlavor: compute-instance-ha
```

**Additional resources**

- Roles

## 2.2. ENABLING FENCING ON AN OVERCLOUD WITH INSTANCE HA

Enable fencing on all Controller and Compute nodes in the overcloud by creating an environment file with fencing information.

**Procedure**

1. Create the environment file in an accessible location, such as ***~/templates***, and include the following content:

```
parameter_defaults:
  EnableFencing: true
  FencingConfig:
    devices:
    - agent: fence_ipmilan
      host_mac: 00:ec:ad:cb:3c:c7
      params:
        login: admin
        ipaddr: 192.168.24.1
        ipport: 6230
        passwd: password
        lanplus: 1
    - agent: fence_ipmilan
      host_mac: 00:ec:ad:cb:3c:cb
      params:
        login: admin
        ipaddr: 192.168.24.1
        ipport: 6231
        passwd: password
        lanplus: 1
    - agent: fence_ipmilan
      host_mac: 00:ec:ad:cb:3c:cf
      params:
        login: admin
        ipaddr: 192.168.24.1
        ipport: 6232
        passwd: password
        lanplus: 1
```

```
  - agent: fence_ipmilan
    host_mac: 00:ec:ad:cb:3c:d3
    params:
      login: admin
      ipaddr: 192.168.24.1
      ipport: 6233
      passwd: password
      lanplus: 1
  - agent: fence_ipmilan
    host_mac: 00:ec:ad:cb:3c:d7
    params:
      login: admin
      ipaddr: 192.168.24.1
      ipport: 6234
      passwd: password
      lanplus: 1
```

2. If you do not use shared storage for your Compute instance, add the following parameter to the environment file that you created:

```
parameter_defaults:
  ExtraConfig:
    tripleo::instanceha::no_shared_storage: true
```

**Additional resources**

- Section 1.2, “Planning your Instance HA deployment”
- Fencing Controller Nodes with STONITH

## 2.3. DEPLOYING THE OVERCLOUD WITH INSTANCE HA

If you already deployed the overcloud, rerun the **openstack overcloud deploy** command with the additional Instance HA files you created. You can configure Instance HA for your overcloud at any time after you create the undercloud.

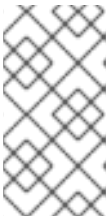
**Prerequisites**

- Instance HA role, flavor, and profile is configured.
- Fencing is enabled on the overcloud.

**Procedure**

- Use the **openstack overcloud deploy** command with the **-e** option for each environment file that you created and with the **compute-instanceha.yaml** environment file. Replace **<FLAVOR_ENV_FILE>** and **<FENCING_ENV_FILE>** with the appropriate file names in your environment:

```
$ openstack overcloud deploy --templates \
  -e <FLAVOR_ENV_FILE> \
  -e <FENCING_ENV_FILE> \
  -r my_roles_data.yaml \
  -e /usr/share/openstack-tripleo-heat-templates/environments/compute-instanceha.yaml
```



NOTE

- Do not modify the **compute-instanceha.yaml** environment file.
- Include the full path to each environment file that you want to include in the overcloud deployment.

After the deployment is complete, each Compute node includes a **STONITH** device and a **GuestNode** service.

## 2.4. TESTING INSTANCE HA EVACUATION

To test that Instance HA evacuates instances correctly, you trigger evacuation on a Compute node and check that the Instance HA agents successfully evacuate and re-create the instance on a different Compute node.

WARNING

The following procedure involves deliberately crashing a Compute node, which triggers the automated evacuation of instances with Instance HA.

**Prerequisites**

- Instance HA is deployed on the Compute node.

**Procedure**

1. Start one or more instances on the overcloud.

```
stack@director $ . overcloudrc
stack@director $ openstack server create --image cirros --flavor 2 test-failover
stack@director $ openstack server list -c Name -c Status
```

2. Log in to the Compute node that hosts the instances and change to the **root** user. Replace **compute-n** with the name of the Compute node:

```
stack@director $ . stackrc
stack@director $ ssh -l heat-admin compute-n
heat-admin@compute-n $ su -
```

3. Crash the Compute node.

```
root@compute-n $ echo c > /proc/sysrq-trigger
```

4. Wait a few minutes for the node to restart, and then verify that the instances from the Compute node that you crash are re-created on another Compute node:

```
stack@director $ openstack server list -c Name -c Status
stack@director $ openstack compute service list
```

## 2.5. DESIGNATING INSTANCES TO EVACUATE WITH INSTANCE HA

By default, Instance HA evacuates all instances from a failed node. You can configure Instance HA to only evacuate instances with specific images or flavors.

**Prerequisites**

- Instance HA is deployed on the overcloud.

**Procedure**

1. Log in to the undercloud as the **stack** user.
2. Source the **overcloudrc** file:

```
$ source ~/overcloudrc
```

3. Use one of the following options:
   - Tag an image:

     ```
     (overcloud) $ openstack image set --tag evacuable <image_id>
     ```

     Replace **<image_id>** with the ID of the image that you want to evacuate.
   - Tag a flavor:

     ```
     (overcloud) $ openstack flavor set --property evacuable=true <flavor_id>
     ```

     Replace **<flavor_id>** with the ID of the flavor that you want to evacuate.

## 2.6. ADDITIONAL RESOURCES

- Director Installation and Usage
- Composable Services and Custom Roles

# CHAPTER 3. PERFORMING MAINTENANCE ON THE UNDERCLOUD AND OVERCLOUD WITH INSTANCE HA

To perform maintenance on the undercloud and overcloud, you must shut down and start up the undercloud and overcloud nodes in a specific order to ensure minimal issues when your start your overcloud. You can also perform maintenance on a specific Compute or Controller node by stopping the node and disabling the Pacemaker resources on the node.

## 3.1. PREREQUISITES

- A running undercloud and overcloud with Instance HA enabled.

## 3.2. UNDERCLOUD AND OVERCLOUD SHUTDOWN ORDER

To shut down the Red Hat OpenStack Platform environment, you must shut down the overcloud and undercloud in the following order:

1. Shut down instances on overcloud Compute nodes
2. Shut down Compute nodes
3. Stop all high availability and OpenStack Platform services on Controller nodes
4. Shut down Ceph Storage nodes
5. Shut down Controller nodes
6. Shut down the undercloud

### 3.2.1. Shutting down instances on overcloud Compute nodes

As a part of shutting down the Red Hat OpenStack Platform environment, shut down all instances on Compute nodes before shutting down the Compute nodes.

**Prerequisites**

- An overcloud with active Compute services

**Procedure**

1. Log in to the undercloud as the **stack** user.
2. Source the credentials file for your overcloud:

```
$ source ~/overcloudrc
```

3. View running instances in the overcloud:

```
$ openstack server list --all-projects
```

4. Stop each instance in the overcloud:

```
$ openstack server stop <INSTANCE>
```

-

Repeat this step for each instance until you stop all instances in the overcloud.

### 3.2.2. Stopping instance HA services on overcloud Compute nodes

As a part of shutting down the Red Hat OpenStack Platform environment, you must shut down all Instance HA services that run on Compute nodes before stopping the instances and shutting down the Compute nodes.

**Prerequisites**

- An overcloud with active Compute services
- Instance HA is enabled on Compute nodes

**Procedure**

1. Log in as the **root** user to an overcloud node that runs Pacemaker.
2. Disable the Pacemaker remote resource on each Compute node:
   a. Identify the Pacemaker Remote resource on Compute nodes:

   ```
   # pcs resource status
   ```

   These resources use the **ocf::pacemaker:remote** agent and are usually named after the Compute node host format, such as **overcloud-novacomputeiha-0**.

   b. Disable each Pacemaker Remote resource. The following example shows how to disable the resource for **overcloud-novacomputeiha-0**:

   ```
   # pcs resource disable overcloud-novacomputeiha-0
   ```

3. Disable the Compute node STONITH devices:
   a. Identify the Compute node STONITH devices:

   ```
   # pcs stonith status
   ```

   b. Disable each Compute node STONITH device:

   ```
   # pcs stonith disable <STONITH_DEVICE>
   ```

### 3.2.3. Shutting down Compute nodes

As a part of shutting down the Red Hat OpenStack Platform environment, log in to and shut down each Compute node.

**Prerequisites**

- Shut down all instances on the Compute nodes

**Procedure**

1. Log in as the **root** user to a Compute node.
2. Shut down the node:

```
# shutdown -h now
```

3. Perform these steps for each Compute node until you shut down all Compute nodes.

### 3.2.4. Stopping services on Controller nodes

As a part of shutting down the Red Hat OpenStack Platform environment, stop services on the Controller nodes before shutting down the nodes. This includes Pacemaker and systemd services.

**Prerequisites**

- An overcloud with active Pacemaker services

**Procedure**

1. Log in as the **root** user to a Controller node.
2. Stop the Pacemaker cluster.

```
# pcs cluster stop --all
```

   This command stops the cluster on all nodes.

3. Wait until the Pacemaker services stop and check that the services stopped.
   a. Check the Pacemaker status:

```
# pcs status
```

   b. Check that no Pacemaker services are running in Podman:

```
# podman ps --filter "name=.*-bundle.*"
```

4. Stop the Red Hat OpenStack Platform services:

```
# systemctl stop 'tripleo_*'
```

5. Wait until the services stop and check that services are no longer running in Podman:

```
# podman ps
```

### 3.2.5. Shutting down Ceph Storage nodes

As a part of shutting down the Red Hat OpenStack Platform environment, disable Ceph Storage services then log in to and shut down each Ceph Storage node.

**Prerequisites**

- A healthy Ceph Storage cluster

- Ceph MON services are running on standalone Ceph MON nodes or on Controller nodes

**Procedure**

1. Log in as the **root** user to a node that runs Ceph MON services, such as a Controller node or a standalone Ceph MON node.
2. Check the health of the cluster. In the following example, the **podman** command runs a status check within a Ceph MON container on a Controller node:

```
# sudo podman exec -it ceph-mon-controller-0 ceph status
```

   Ensure that the status is **HEALTH_OK**.

3. Set the **noout**, **norecover**, **norebalance**, **nobackfill**, **nodown**, and **pause** flags for the cluster. In the following example, the **podman** commands set these flags through a Ceph MON container on a Controller node:

```
# sudo podman exec -it ceph-mon-controller-0 ceph osd set noout
# sudo podman exec -it ceph-mon-controller-0 ceph osd set norecover
# sudo podman exec -it ceph-mon-controller-0 ceph osd set norebalance
# sudo podman exec -it ceph-mon-controller-0 ceph osd set nobackfill
# sudo podman exec -it ceph-mon-controller-0 ceph osd set nodown
# sudo podman exec -it ceph-mon-controller-0 ceph osd set pause
```

4. Shut down each Ceph Storage node:
   a. Log in as the **root** user to a Ceph Storage node.
   b. Shut down the node:

```
# shutdown -h now
```

   c. Perform these steps for each Ceph Storage node until you shut down all Ceph Storage nodes.

5. Shut down any standalone Ceph MON nodes:
   a. Log in as the **root** user to a standalone Ceph MON node.
   b. Shut down the node:

```
# shutdown -h now
```

   c. Perform these steps for each standalone Ceph MON node until you shut down all standalone Ceph MON nodes.

**Additional resources**

- "What is the procedure to shutdown and bring up the entire ceph cluster?"

### 3.2.6. Shutting down Controller nodes

As a part of shutting down the Red Hat OpenStack Platform environment, log in to and shut down each Controller node.

**Prerequisites**

- Stop the Pacemaker cluster
- Stop all Red Hat OpenStack Platform services on the Controller nodes

**Procedure**

1. Log in as the **root** user to a Controller node.
2. Shut down the node:

```
# shutdown -h now
```

3. Perform these steps for each Controller node until you shut down all Controller nodes.

### 3.2.7. Shutting down the undercloud

As a part of shutting down the Red Hat OpenStack Platform environment, log in to the undercloud node and shut down the undercloud.

**Prerequisites**

- A running undercloud

**Procedure**

1. Log in to the undercloud as the **stack** user.
2. Shut down the undercloud:

```
$ sudo shutdown -h now
```

## 3.3. PERFORMING SYSTEM MAINTENANCE

After you completely shut down the undercloud and overcloud, perform any maintenance to the systems in your environment and then start up the undercloud and overcloud.

## 3.4. UNDERCLOUD AND OVERCLOUD STARTUP ORDER

To start the Red Hat OpenStack Platform environment, you must start the undercloud and overcloud in the following order:

1. Start the undercloud
2. Start Controller nodes
3. Start Ceph Storage nodes
4. Start Compute nodes
5. Start instances on overcloud Compute nodes

### 3.4.1. Starting the undercloud

As a part of starting the Red Hat OpenStack Platform environment, power on the undercloud node, log in to the undercloud, and check the undercloud services.

#### Prerequisites

- A powered down undercloud

#### Procedure

1. Power on the undercloud and wait until the undercloud boots.

#### Verification

1. Log in to the undercloud as the **stack** user.
2. Check the services on the undercloud:

```
$ systemctl list-units 'tripleo_*'
```

3. Source the credential file for your undercloud and run the validation command to check that all services and containers are active and healthy.

```
$ source stackrc
$ openstack tripleo validator run --validation service-status --limit undercloud
```

#### Additional resources

- Using the validation framework

### 3.4.2. Starting Controller nodes

As a part of starting the Red Hat OpenStack Platform environment, power on each Controller node and check the non-Pacemaker services on the node.

#### Prerequisites

- Powered down Controller nodes

#### Procedure

1. Power on each Controller node.

#### Verification

1. Log in to each Controller node as the **root** user.
2. Check the services on the Controller node:

```
$ systemctl -t service
```

Only non-Pacemaker based services are running.

3. Wait until the Pacemaker services start and check that the services started:

```
$ pcs status
```

NOTE

If your environment uses Instance HA, the Pacemaker resources do not start until you start the Compute nodes or perform a manual unfence operation with the **pcs stonith confirm <compute_node>** command. You must run this command on each Compute node that uses Instance HA.

### 3.4.3. Starting Ceph Storage nodes

As a part of starting the Red Hat OpenStack Platform environment, power on the Ceph MON and Ceph Storage nodes and enable Ceph Storage services.

**Prerequisites**

- A powered down Ceph Storage cluster
- Ceph MON services are enabled on powered down standalone Ceph MON nodes or on powered on Controller nodes

**Procedure**

1. If your environment has standalone Ceph MON nodes, power on each Ceph MON node.
2. Power on each Ceph Storage node.
3. Log in as the **root** user to a node that runs Ceph MON services, such as a Controller node or a standalone Ceph MON node.
4. Check the status of the cluster nodes. In the following example, the **podman** command runs a status check within a Ceph MON container on a Controller node:

```
# sudo podman exec -it ceph-mon-controller-0 ceph status
```

   Ensure that each node is powered on and connected.

5. Unset the **noout**, **norecover**, **norebalance**, **nobackfill**, **nodown** and **pause** flags for the cluster. In the following example, the **podman** commands unset these flags through a Ceph MON container on a Controller node:

```
# sudo podman exec -it ceph-mon-controller-0 ceph osd unset noout
# sudo podman exec -it ceph-mon-controller-0 ceph osd unset norecover
# sudo podman exec -it ceph-mon-controller-0 ceph osd unset norebalance
# sudo podman exec -it ceph-mon-controller-0 ceph osd unset nobackfill
# sudo podman exec -it ceph-mon-controller-0 ceph osd unset nodown
# sudo podman exec -it ceph-mon-controller-0 ceph osd unset pause
```

**Verification**

1. Check the health of the cluster. In the following example, the **podman** command runs a status check within a Ceph MON container on a Controller node:

```
# sudo podman exec -it ceph-mon-controller-0 ceph status
```

Ensure the status is **HEALTH_OK**.

**Additional resources**

- "What is the procedure to shutdown and bring up the entire ceph cluster?"

### 3.4.4. Starting Compute nodes

As a part of starting the Red Hat OpenStack Platform environment, power on each Compute node and check the services on the node.

**Prerequisites**

- Powered down Compute nodes

**Procedure**

1. Power on each Compute node.

**Verification**

1. Log in to each Compute as the **root** user.
2. Check the services on the Compute node:

```
$ systemctl -t service
```

### 3.4.5. Starting instance HA services on overcloud Compute nodes

As a part of starting the Red Hat OpenStack Platform environment, start all Instance HA services on the Compute nodes.

**Prerequisites**

- An overcloud with running Compute nodes
- Instance HA is enabled on Compute nodes

**Procedure**

1. Log in as the **root** user to an overcloud node that runs Pacemaker.
2. Enable the STONITH device for a Compute node:
   a. Identify the Compute node STONITH device:

```
# pcs stonith status
```

   b. Clear any STONITH errors for the Compute node:

```
# pcs stonith confirm <COMPUTE_NODE>
```

This command returns the node to a clean STONITH state.

c. Enable the Compute node STONITH device:

```
# pcs stonith enable <STONITH_DEVICE>
```

d. Perform these steps for each Compute node with STONITH.

3. Enable the Pacemaker remote resource on each Compute node:

a. Identify the Pacemaker remote resources on Compute nodes:

```
# pcs resource status
```

These resources use the **ocf::pacemaker:remote** agent and are usually named after the Compute node host format, such as **overcloud-novacomputeiha-0**.

b. Enable each Pacemaker Remote resource. The following example shows how to enable the resource for **overcloud-novacomputeiha-0**:

```
# pcs resource enable overcloud-novacomputeiha-0
```

c. Perform these steps for each Compute node with Pacemaker remote management.

4. Wait until the Pacemaker services start and check that the services started:

```
# pcs status
```

5. If any Pacemaker resources fail to start during the startup process, reset the status and the fail count of the resource:

```
# pcs resource cleanup
```

NOTE

Some services might require more time to start, such as **fence_compute** and **fence_kdump**.

### 3.4.6. Starting instances on overcloud Compute nodes

As a part of starting the Red Hat OpenStack Platform environment, start the instances on on Compute nodes.

**Prerequisites**

- An active overcloud with active nodes

**Procedure**

1. Log in to the undercloud as the **stack** user.
2. Source the credentials file for your overcloud:

```
$ source ~/overcloudrc
```

3. View running instances in the overcloud:

```
$ openstack server list --all-projects
```

4. Start an instance in the overcloud:

```
$ openstack server start <INSTANCE>
```

# CHAPTER 4. PERFORMING MAINTENANCE ON COMPUTE NODES AND CONTROLLER NODES WITH INSTANCE HA

To perform maintenance on a Compute node or a Controller node with Instance HA, stop the node by setting it in **standby** mode and disabling the Pacemaker resources on the node. After you complete the maintenance work, you start the node and check that the Pacemaker resources are healthy.

### Prerequisites

- A running overcloud with Instance HA enabled

### Procedure

1. Log in to a Controller node and stop the Compute or Controller node:

```
# pcs node standby <node UUID>
```

**IMPORTANT**

You must log in to a different node from the node you want to stop.

2. Disable the Pacemaker resources on the node:

```
# pcs resource disable <ocf::pacemaker:remote on the node>
```

3. Perform any maintenance work on the node.

4. Restore the IPMI connection and start the node. Wait until the node is ready before proceeding.

5. Enable the Pacemaker resources on the node and start the node:

```
# pcs resource enable <ocf::pacemaker:remote on the node>
# pcs node unstandby <node UUID>
```

6. If you set the node to maintenance mode, source the credential file for your overcloud and unset the node from maintenance mode:

```
# source stackrc
# openstack baremetal node maintenance unset <baremetal node UUID>
```

### Verification

1. Check that the Pacemaker resources are active and healthy:

```
# pcs status
```

2. If any Pacemaker resources fail to start during the startup process, run the **pcs resource cleanup** command to reset the status and the fail count of the resource.

3. If you evacuated instances from a Compute node before you stopped the node, check that the instances are migrated to a different node:

```
# openstack server list --long
# nova migration-list
```