Red Hat OpenStack Platform 16.2

Compute 实例的高可用性

为 Compute 实例配置高可用性

Last Updated: 2024-07-30

Red Hat OpenStack Platform 16.2 Compute 实例的高可用性

为 Compute 实例配置高可用性

OpenStack Team
rhos-docs@redhat.com

法律通告

摘要

本指南介绍了如何管理实例高可用性（实例 HA）。使用 Instance HA 时，当 Compute 节点出现故障时，Red Hat OpenStack Platform (RHOSP)可以自动撤离和重新创建不同 Compute 节点上的实例。

目录

使开源包含更多 3

对红帽文档提供反馈 4

第 1 章 简介和规划实例 HA 部署 5
1.1. 实例 HA 的工作原理 5
1.2. 规划您的实例 HA 部署 5
1.3. 实例 HA 资源代理 6

第 2 章 安装和配置实例 HA 7
2.1. 配置 INSTANCE HA 角色、类别和配置集 7
2.2. 在带有实例 HA 的 OVERCLOUD 中启用隔离 8
2.3. 使用实例 HA 部署 OVERCLOUD 9
2.4. 测试实例 HA 撤离 10
2.5. 使用实例 HA 将实例设计为撤离 10
2.6. 其他资源 11

第 3 章 使用实例 HA 对 UNDERCLOUD 和 OVERCLOUD 执行维护 12
3.1. 前提条件 12
3.2. UNDERCLOUD 和 OVERCLOUD 关闭顺序 12
3.3. 执行系统维护 16
3.4. UNDERCLOUD 和 OVERCLOUD 启动顺序 16

第 4 章 在 COMPUTE 节点上通过 INSTANCE HA 执行维护 21

使开源包含更多

红帽致力于替换我们的代码、文档和 Web 属性中存在问题的语言。我们从这四个术语开始：master、slave、黑名单和白名单。由于此项工作十分艰巨，这些更改将在即将推出的几个发行版本中逐步实施。详情请查看 CTO Chris Wright 的信息。

对红帽文档提供反馈

我们感谢您对文档提供反馈信息。与我们分享您的成功秘诀。

在 JIRA 中提供文档反馈

使用 Create Issue 表单对文档提供反馈。JIRA 问题将在 Red Hat OpenStack Platform Jira 项目中创建，您可以在其中跟踪您的反馈进度。

1. 确保您已登录到 JIRA。如果您没有 JIRA 帐户，请创建一个帐户来提交反馈。
2. 点击以下链接打开 **Create Issue** 页面： Create Issue
3. 完成 **Summary** 和 **Description** 字段。在 **Description** 字段中，包含文档 URL、章节或章节号以及问题的详细描述。不要修改表单中的任何其他字段。
4. 点 **Create**。

第 1 章 简介和规划实例 HA 部署

Compute 实例(Instance HA)的高可用性是一个用于从故障 Compute 节点撤离实例的工具，并在不同的 Compute 节点上重新创建实例。

实例 HA 可以与共享存储或本地存储环境配合使用，这意味着撤离的实例维护相同的网络配置，如静态 IP 地址和浮动 IP 地址。重新创建的实例也会在新的 Compute 节点中维护相同的特征。

1.1. 实例 HA 的工作原理

当 Compute 节点出现故障时，overcloud 隔离代理会隔离该节点，那么 Instance HA 代理会将实例从故障 Compute 节点撤离到不同的 Compute 节点。

当 Compute 节点失败并触发 Instance HA 时，会出现以下事件：

1. 在出现故障时，**IPMI** 代理执行第一层隔离，其中包括物理重置节点，以确保它关闭并防止 overcloud 上的数据崩溃或多个相同实例。当节点离线时，它被视为被隔离。
2. 在物理 IPMI 隔离后，**fence-nova** 代理会自动执行第二个层隔离，并通过运行以下命令来使用 **"evacuate=yes"** 集群标记隔离的节点：

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

   FAILEDHOST 是故障 Compute 节点的名称。
3. **nova-evacuate** 代理持续在后台运行，并使用 **"evacuate=yes"** 属性定期检查集群是否有节点。当 **nova-evacuate** 检测到隔离的节点包含此属性时，代理会启动撤离节点。撤离过程与可随时执行的手动实例撤离过程类似。
4. 当失败的节点在 IPMI 重置后重新启动时，该节点上的 **nova-compute** 进程也会自动启动。因为该节点之前被隔离，所以它不会运行任何新实例，直到 Pacemaker 取消隔离节点为止。
5. 当 Pacemaker 检测到 Compute 节点在线时，它会在节点上启动 **compute-unfence-trigger** 资源代理，这会释放该节点，以便可以再次运行实例。

其他资源

- 清空实例

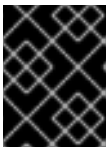
1.2. 规划您的实例 HA 部署

在部署 Instance HA 前，请查看资源名称是否合规，并根据您的环境配置存储和网络。

- Compute 节点主机名和 Pacemaker 远程资源名称必须符合 W3C 命名约定。如需更多信息，请参阅 W3C 文档中的 Declaring Namespaces 和 Names and Tokens。
- 通常，instance HA 要求您为实例的磁盘镜像配置共享存储。因此，如果您尝试使用 **no-shared-storage** 选项，您可能会在撤离过程中收到 **InvalidSharedStorage** 错误，且实例不会在另一个 Compute 节点上启动。
  但是，如果所有实例都被配置为从 OpenStack Block Storage (**cinder**)卷引导，则不需要为实例的磁盘镜像配置共享存储，您可以使用 **no-shared-storage** 选项驱除所有实例。

  在撤离过程中，如果您的实例被配置为从块存储卷引导，则任何撤离的实例都从另一个 Compute 节点上的同一卷启动。因此，撤离的实例会立即重新启动其作业，因为 OS 镜像和应用程序数据存储在 OpenStack Block Storage 卷上。

- 如果在 Spine-Leaf 环境中部署 Instance HA，您必须为 Controller 和 Compute 节点定义一个 **internal_api** 网络。然后，您可以为每个叶定义一个子网。有关配置 Spine-Leaf 网络的更多信息，请参阅 *Spine Leaf Networking 指南*中的创建角色数据文件。
- 从 Red Hat OpenStack Platform 13 及更高版本，您可以使用 director 升级实例 HA 作为 overcloud 升级的一部分。有关升级 overcloud 的更多信息，请参阅 Red Hat OpenStack Platform Updated 指南。
- 在安装后，不支持使用 director 禁用实例 HA。有关从部署中手动删除实例 HA 组件的临时解决方案，请参阅 如何从控制器节点中删除 Instance HA 组件？。



重要

在生产环境中不会验证这个临时解决方案。您必须在生产环境中实施测试环境中前验证测试环境中的步骤。

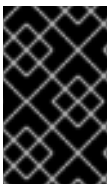
1.3. 实例 HA 资源代理

实例 HA 使用 **fence_compute**、**NovaEvacuate**，以及 **comput-unfence-trigger** 资源代理在 Compute 节点失败时撤离和重新创建实例。

代理名称	集群内的名称	角色
fence_compute	**fence-nova**	当节点不可用时，标记 Compute 节点以进行撤离。
NovaEvacuate	**nova-evacuate**	从故障节点撤离实例。此代理在其中一个 Controller 节点上运行。
dummy	**compute-unfence-trigger**	释放隔离的节点，并让节点再次运行实例。

第 2 章 安装和配置实例 HA

Red Hat OpenStack Platform (RHOSP) director 部署实例高可用性(HA)。但是，您必须执行额外的步骤，以在新的 overcloud 上配置新的实例 HA 部署。完成这些步骤后，instance HA 将运行在具有自定义角色的 Compute 节点的子集。



重要

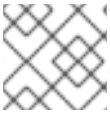
RHOSP 超融合基础架构(HCI)环境不支持实例 HA。要在 RHOSP HCI 环境中使用实例 HA，您必须使用 ComputeInstanceHA 角色指定 Compute 节点的子集以使用实例 HA。Red Hat Ceph Storage 服务不能托管在托管 Instance HA 的 Compute 节点上。

重要

要在不同的环境中启用实例 HA，如使用标准或自定义角色的现有 overcloud，请仅执行与部署相关的步骤并相应地调整模板。

2.1. 配置 INSTANCE HA 角色、类别和配置集

在部署 Instance HA 之前，将 Instance HA 角色添加到 **roles-data.yaml** 文件中，创建 Instance HA 类别，为您要通过 Instance HA 配置集管理的 Instanc HA 标记每个 Comput 节点，并将 Instance HA 角色映射到 Instance HA 类别。



注意

您可以根据您的环境修改此流程中的示例文件和角色名称。

流程

1. 将 **ComputeInstanceHA** 角色添加到 **roles-data.yaml** 文件中，并重新生成该文件。

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

ComputeInstanceHA 角色包含默认 **Compute** 角色、**ComputeInstanceHA** 服务和 **PacemakerRemote** 服务中的所有服务。

2. 创建 **compute-instance-ha** 类别，以标记要通过 Instance HA 管理的 Compute 节点。

```
$ 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. 将您要管理的每个 Compute 节点标记为 Instance HA，使用 **compute-instance-ha** 配置集，并将 < **NODE UUID&gt**; 替换为实际 UUID：

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

4. 通过创建具有以下参数的环境文件，将 **ComputeInstanceHA** 角色映射到 **compute-instance-ha** 类别：

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

其他资源

- 角色

2.2. 在带有实例 HA 的 OVERCLOUD 中启用隔离

通过创建带有隔离信息的环境文件，在 overcloud 中的所有 Controller 和 Compute 节点上启用隔离。

流程

1. 在可访问的位置创建环境文件，如 *~/templates*，并包含以下内容：

```
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. 如果没有将共享存储用于 Compute 实例，请在您创建的环境文件中添加以下参数：

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

其他资源

- 第 1.2 节 “规划您的实例 HA 部署”
- 使用 STONITH 的隔离 Controller 节点

2.3. 使用实例 HA 部署 OVERCLOUD

如果您已经部署了 overcloud，请使用您创建的额外实例 HA 文件重新运行 **openstack overcloud deploy** 命令。您可以在创建 undercloud 后随时为 overcloud 配置实例 HA。

前提条件

- 实例 HA 角色、类别和配置集已配置。
- overcloud 上启用了隔离功能。

流程

- 将 **openstack overcloud deploy** 命令与您创建的每个环境文件以及 **compute-instanceha.yaml** 环境文件的 **-e** 选项一起使用。将 **<FLAVOR_ENV_FILE** &gt **; 和 <FENCING_ENV_FILE** > 替换为环境中的适当文件名：

```
$ 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
```

注意

  - 不要修改 **compute-instanceha.yaml** 环境文件。
  - 包含您要包含在 overcloud 部署中的每个环境文件的完整路径。

部署完成后，每个 Compute 节点都包含 **STONITH** 设备和 **GuestNode** 服务。

2.4. 测试实例 HA 撤离

要测试实例 HA 正确撤离实例，您可以在 Compute 节点上触发撤离，并检查实例 HA 代理是否成功撤离并重新创建实例。

警告

以下流程涉及严重崩溃 Compute 节点，该节点会触发使用 Instance HA 自动清空实例。

前提条件

- 实例 HA 部署在 Compute 节点上。

流程

1. 启动 overcloud 上的一个或多个实例。

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

2. 登录托管实例的 Compute 节点，并更改到 **root** 用户。将 **compute-n** 替换为 Compute 节点的名称：

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

3. 崩溃 Compute 节点。

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

4. 等待几分钟，以便节点重启，然后验证您在另一个 Compute 节点上重新创建您崩溃的 Compute 节点上的实例：

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

2.5. 使用实例 HA 将实例设计为撤离

默认情况下，instance HA 从故障节点驱除所有实例。您可以将实例 HA 配置为仅撤离具有特定镜像或类别的实例。

前提条件

- 实例 HA 部署在 overcloud 上。

流程

1. 以 **stack** 用户身份登录 undercloud。
2. 获取 **overcloudrc** 文件：

```
$ source ~/overcloudrc
```

3. 使用以下选项之一：
   - 标记镜像：

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

     将 **<image_id** > 替换为您要撤离的镜像 ID。
   - 标记类别：

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

     将 **<flavor_id** > 替换为您要撤离的类别 ID。

2.6. 其他资源

- director 的安装和使用
- 可组合服务和自定义角色

第 3 章 使用实例 HA 对 UNDERCLOUD 和 OVERCLOUD 执行维护

要对 undercloud 和 overcloud 执行维护，您必须按照特定顺序关闭并启动 undercloud 和 overcloud 节点，以确保启动 overcloud 时的最小问题。您还可以通过停止节点并禁用节点上的 Pacemaker 资源，在特定 Compute 或 Controller 节点上执行维护。

3.1. 前提条件

- 正在运行的 undercloud 和启用了实例 HA 的 overcloud。

3.2. UNDERCLOUD 和 OVERCLOUD 关闭顺序

要关闭 Red Hat OpenStack Platform 环境，您必须按照以下顺序关闭 overcloud 和 undercloud：

1. 关闭 overcloud Compute 节点上的实例
2. 关闭 Compute 节点
3. 停止 Controller 节点上的所有高可用性和 OpenStack Platform 服务
4. 关闭 Ceph Storage 节点
5. 关闭 Controller 节点
6. 关闭 undercloud

3.2.1. 关闭 overcloud Compute 节点上的实例

作为关闭 Red Hat OpenStack Platform 环境的一部分，在关闭 Compute 节点之前关闭 Compute 节点上的所有实例。

先决条件

- 具有活跃 Compute 服务的 overcloud

步骤

1. 以 **stack** 用户身份登录 undercloud。
2. 提供 overcloud 的凭据文件：

```
$ source ~/overcloudrc
```

3. 查看 overcloud 中运行的实例：

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

4. 停止 overcloud 中的每个实例：

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

对每个实例重复这一步，直到停止 overcloud 中的所有实例。

3.2.2. 在 overcloud Compute 节点上停止实例 HA 服务

作为关闭 Red Hat OpenStack Platform 环境的一部分，您必须关闭在 Compute 节点上运行的所有实例 HA 服务，然后才能停止实例并关闭 Compute 节点。

前提条件

- 具有活跃 Compute 服务的 overcloud
- Compute 节点上启用了实例 HA

流程

1. 以 **root** 用户身份登录运行 Pacemaker 的 overcloud 节点。
2. 在每个 Compute 节点上禁用 Pacemaker 远程资源：
   a. 识别 Compute 节点上的 Pacemaker 远程资源：

   ```
   # pcs resource status
   ```

   这些资源使用 **ocf::pacemaker:remote** 代理，通常以 Compute 节点主机格式命名，如 **overcloud-novacomputeiha-0**。

   b. 禁用每个 Pacemaker 远程资源。以下示例演示了如何禁用 **overcloud-novacomputeiha-0** 的资源：

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

3. 禁用 Compute 节点 STONITH 设备：
   a. 确定 Compute 节点 STONITH 设备：

   ```
   # pcs stonith status
   ```

   b. 禁用每个 Compute 节点 STONITH 设备：

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

3.2.3. 关闭 Compute 节点

作为关闭 Red Hat OpenStack Platform 环境的一部分，登录并关闭每个 Compute 节点。

先决条件

- 关闭 Compute 节点上的所有实例：

步骤

1. 以 **root** 用户身份登录 Compute 节点。
2. 关闭该节点：

   ```
   # shutdown -h now
   ```

3. 对每个 Compute 节点执行这些步骤，直到关闭所有 Compute 节点。

3.2.4. 停止 Controller 节点上的服务

作为关闭 Red Hat OpenStack Platform 环境的一部分，在关闭 Controller 节点前停止节点上的服务。这包括 Pacemaker 和 systemd 服务。

先决条件

- 具有活跃 Pacemaker 服务的 overcloud

步骤

1. 以 **root** 用户身份登录 Controller 节点。
2. 停止 Pacemaker 集群。

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

此命令停止所有节点上的集群。

3. 等待 Pacemaker 服务停止并检查服务是否已停止。
   a. 检查 Pacemaker 状态：

```
# pcs status
```

   b. 检查 Podman 中没有 Pacemaker 服务在运行：

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

4. 停止 Red Hat OpenStack Platform 服务：

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

5. 等待服务停止，检查 Podman 中服务不再运行：

```
# podman ps
```

3.2.5. 关闭 Ceph Storage 节点

作为关闭 Red Hat OpenStack Platform 环境的一部分，禁用 Ceph Storage 服务，然后登录并关闭每个 Ceph Storage 节点。

先决条件

- 正常运行的 Ceph Storage 集群
- Ceph MON 服务在单机 Ceph MON 节点或 Controller 节点上运行

步骤

1. 以 **root** 用户身份登录运行 Ceph MON 服务的节点，如 Controller 节点或单机 Ceph MON 节点。

2. 检查集群的运行状况。在以下示例中，**podman** 命令在 Controller 节点上的 Ceph MON 容器中运行状态检查：

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

确保状态为 **HEALTH_OK**。

3. 为集群设置 **noout**、**norecover**、**norebalance**、**nobackfill**、**nodown** 和 **pause** 标志。在以下示例中，**podman** 命令通过 Controller 节点上的 Ceph MON 容器设置这些标志：

```
# 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. 关闭每个 Ceph Storage 节点：

   a. 以 **root** 用户身份登录 Ceph Storage 节点。

   b. 关闭该节点：

```
# shutdown -h now
```

   c. 对每个 Ceph Storage 节点执行这些步骤，直到关闭所有 Ceph Storage 节点。

5. 关闭任何单机 Ceph MON 节点：

   a. 以 **root** 用户身份登录单机 Ceph MON 节点。

   b. 关闭该节点：

```
# shutdown -h now
```

   c. 对每个单机 Ceph MON 节点执行这些步骤，直到关闭所有单机 Ceph MON 节点。

其他资源

- “关闭并启动整个 Ceph 集群的步骤是什么？”

3.2.6. 关闭 Controller 节点

作为关闭 Red Hat OpenStack Platform 环境的一部分，登录并关闭每个 Controller 节点。

先决条件

- 停止 Pacemaker 集群
- 停止 Controller 节点上的所有 Red Hat OpenStack Platform 服务

步骤

1. 以 **root** 用户身份登录 Controller 节点。

2. 关闭该节点：

```
# shutdown -h now
```

3. 对每个 Controller 节点执行这些步骤，直到关闭所有 Controller 节点。

3.2.7. 关闭 undercloud

作为关闭 Red Hat OpenStack Platform 环境的一部分，登录到 undercloud 节点并关闭 undercloud。

先决条件

- 正在运行的 undercloud

步骤

1. 以 **stack** 用户身份登录 undercloud。
2. 关闭 undercloud：

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

3.3. 执行系统维护

在完全关闭 undercloud 和 overcloud 后，对环境中的系统执行任何维护，然后启动 undercloud 和 overcloud。

3.4. UNDERCLOUD 和 OVERCLOUD 启动顺序

要启动 Red Hat OpenStack Platform 环境，您必须按照以下顺序启动 undercloud 和 overcloud：

1. 启动 undercloud
2. 启动 Controller 节点
3. 启动 Ceph Storage 节点
4. 启动 Compute 节点。
5. 启动 overcloud Compute 节点上的实例

3.4.1. 启动 undercloud

作为启动 Red Hat OpenStack Platform 环境的一部分，启动 undercloud 节点，登录到 undercloud，再检查 undercloud 服务。

先决条件

- undercloud 已关机。

流程

- 打开 undercloud 并等待 undercloud 引导。

验证

1. 以 **stack** 用户身份登录 undercloud 主机。
2. 查找 **stackrc** undercloud 凭证文件：

```
$ source ~/stackrc
```

3. 检查 undercloud 上的服务：

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

4. 创建并验证名为 **inventory.yaml** 的静态清单文件：

```
$ tripleo-ansible-inventory --static-yaml-inventory inventory.yaml
$ openstack tripleo validator run --group pre-introspection \
 -i inventory.yaml
```

5. 检查所有服务和容器是否活跃且健康：

```
$ openstack tripleo validator run --validation service-status \
 --limit undercloud -i inventory.yaml
```

其他资源

- 使用验证框架

3.4.2. 启动 Controller 节点

作为启动 Red Hat OpenStack Platform 环境的一部分，打开每个 Controller 节点电源，并检查节点上的非 Pacemaker 服务。

先决条件

- 关闭 Controller 节点电源

步骤

1. 打开每个 Controller 节点电源。

验证

1. 以 **root** 用户身份登录每个 Controller 节点。
2. 检查 Controller 节点上的服务：

```
$ systemctl -t service
```

只有基于非 Pacemaker 的服务正在运行。

3. 等待 Pacemaker 服务启动并检查服务是否已启动：

```
$ pcs status
```

注意

如果您的环境使用 Instance HA，Pacemaker 资源不会启动，直到您启动 Compute 节点，或使用 **pcs stonith confirm <compute_node>** 命令执行手动取消隔离操作。您必须在使用 Instance HA 的每个 Compute 节点上运行此命令。

3.4.3. 启动 Ceph Storage 节点

作为启动 Red Hat OpenStack Platform 环境的一部分，打开 Ceph MON 和 Ceph Storage 节点电源，并启用 Ceph Storage 服务。

先决条件

- 已关闭电源的 Ceph Storage 集群
- Ceph MON 服务在已关闭电源的单机 Ceph MON 节点或已打开电源的 Controller 节点上启用

步骤

1. 如果您的环境有单机 Ceph MON 节点，请打开每个 Ceph MON 节点电源。
2. 打开每个 Ceph Storage 节点电源。
3. 以 **root** 用户身份登录运行 Ceph MON 服务的节点，如 Controller 节点或单机 Ceph MON 节点。
4. 检查集群节点的状态：在以下示例中，**podman** 命令在 Controller 节点上的 Ceph MON 容器中运行状态检查：

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

   确保每个节点都已打开电源并连接。

5. 为集群取消设置 **noout**、**norecover**、**norebalance**、**nobackfill**、**nodown** 和 **pause** 标志。在以下示例中，**podman** 命令通过 Controller 节点上的 Ceph MON 容器取消设置这些标志：

```
# 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
```

验证

1. 检查集群的运行状况。在以下示例中，**podman** 命令在 Controller 节点上的 Ceph MON 容器中运行状态检查：

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

   确保状态为 **HEALTH_OK**。

其他资源

- "关闭并启动整个 Ceph 集群的步骤是什么？"

3.4.4. 启动 Compute 节点

作为启动 Red Hat OpenStack Platform 环境的一部分，打开每个 Compute 节点电源并检查节点上的服务。

先决条件

- 关闭 Compute 节点电源

步骤

1. 打开每个 Compute 节点电源。

验证

1. 以 **root** 用户身份登录每个 Compute。
2. 检查 Compute 节点上的服务：

```
$ systemctl -t service
```

3.4.5. 启动 overcloud Compute 节点上的实例 HA 服务

作为启动 Red Hat OpenStack Platform 环境的一部分，启动 Compute 节点上的所有实例 HA 服务。

前提条件

- 具有运行 Compute 节点的 overcloud
- Compute 节点上启用了实例 HA

流程

1. 以 **root** 用户身份登录运行 Pacemaker 的 overcloud 节点。
2. 为 Compute 节点启用 STONITH 设备：
   a. 确定 Compute 节点 STONITH 设备：

```
# pcs stonith status
```

   b. 清除 Compute 节点的 STONITH 错误：

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

   这个命令将节点返回到干净的 STONITH 状态。

   c. 启用 Compute 节点 STONITH 设备：

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

   d. 使用 STONITH 为每个 Compute 节点执行这些步骤。
3. 在每个 Compute 节点上启用 Pacemaker 远程资源：

a. 识别 Compute 节点上的 Pacemaker 远程资源：

```
# pcs resource status
```

这些资源使用 **ocf::pacemaker:remote** 代理，通常以 Compute 节点主机格式命名，如 **overcloud-novacomputeiha-0**。

b. 启用每个 Pacemaker 远程资源。以下示例演示了如何为 **overcloud-novacomputeiha-0** 启用资源：

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

c. 使用 Pacemaker 远程管理为每个 Compute 节点执行这些步骤。

4. 等待 Pacemaker 服务启动并检查服务是否已启动：

```
# pcs status
```

5. 如果任何 Pacemaker 资源在启动过程中无法启动，请重置资源的状态和失败计数：

```
# pcs resource cleanup
```

注意

有些服务可能需要更长的时间才能启动，如 **fence_compute** 和 **fence_kdump**。

3.4.6. 启动 overcloud Compute 节点上的实例

作为启动 Red Hat OpenStack Platform 环境的一部分，启动 Compute 节点上的实例。

先决条件

- 具有活跃节点的活跃 overcloud

步骤

1. 以 **stack** 用户身份登录 undercloud。
2. 提供 overcloud 的凭据文件：

```
$ source ~/overcloudrc
```

3. 查看 overcloud 中运行的实例：

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

4. 启动 overcloud 中的实例：

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

第 4 章 在 COMPUTE 节点上通过 INSTANCE HA 执行维护

要在 Compute 节点上或带有 Instance HA 的 Controller 节点上执行维护，请通过将其设置为 **待机模式** 并禁用节点上的 Pacemaker 资源来停止节点。完成维护工作后，您将启动该节点并检查 Pacemaker 资源是否健康。

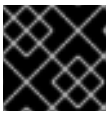
前提条件

- 已启用实例 HA 的运行的 overcloud

流程

1. 登录到 Controller 节点并停止 Compute 或 Controller 节点：

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



重要

您必须从您要停止的节点登录到不同的节点。

2. 在节点上禁用 Pacemaker 资源：

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

3. 在节点上执行任何维护工作。

4. 恢复 IPMI 连接并启动节点。在继续操作前，等待节点就绪。

5. 在节点上启用 Pacemaker 资源并启动节点：

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

6. 如果将节点设置为维护模式，请提供 overcloud 的凭据文件，并从维护模式取消设置节点：

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

验证

1. 检查 Pacemaker 资源是否活跃且健康：

```
# pcs status
```

2. 如果任何 Pacemaker 资源在启动过程中无法启动，请运行 **pcs resource cleanup** 命令来重置资源的状态和故障计数。

3. 如果您在停止节点前从 Compute 节点撤离实例，请检查实例是否已迁移到不同的节点：

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