Red Hat OpenStack Platform 13

保持 Red Hat OpenStack Platform 更新

执行 Red Hat OpenStack Platform 的小更新

Last Updated: 2023-04-20

Red Hat OpenStack Platform 13 保持 Red Hat OpenStack Platform 更新

执行 Red Hat OpenStack Platform 的小更新

OpenStack Team
rhos-docs@redhat.com

法律通告

摘要

本文档提供了更新 Red Hat OpenStack Platform 13 (Queens)环境的步骤。本文档假定您将更新安装在 Red Hat Enterprise Linux 7 上的容器化 OpenStack Platform 部署。

目录

第 1 章 简介

本文档提供了一个工作流，可帮助您使用最新的软件包和容器更新 Red Hat OpenStack Platform 13 环境。

本指南通过以下版本提供升级路径：

旧的 Overcloud 版本	新的 Overcloud 版本
Red Hat OpenStack Platform 13	Red Hat OpenStack Platform 13.z

1.1. 高级别工作流

下表概述了升级过程所需的步骤：

Step	Description
获取新容器镜像	创建一个新的环境文件，其中包含 OpenStack Platform 13 服务的最新容器镜像。
更新 undercloud	将 undercloud 更新至最新的 OpenStack Platform 13.z 版本。
更新 overcloud	将 overcloud 更新至最新的 OpenStack 平台 13.z 版本。
更新 Ceph Storage 节点	升级所有 Ceph Storage 3 服务。
完成升级	运行 convergence 命令以刷新 overcloud 堆栈。

1.2. 已知的可能会阻止更新的问题

查看以下可能影响到成功次版本更新的已知问题。

Red Hat Ceph Storage 3 的次要更新可能会导致 OSD 崩溃

Red Hat Ceph Storage 3 依赖于 docker 来进行在 EL7 上运行的容器化部署。BZ-10-0-830 的 ceph-ansible 修复更新控制 Ceph 容器的 systemd 单元，使 systemd 单元需要启动和运行 docker 服务来执行。这个要求是实施安全更新路径，并避免在 docker 软件包不受控制的更新时服务中断甚至数据损坏。
更新 ceph-ansible 软件包不足以使 ceph-ansible 修复有效。您还必须通过重新运行部署 playbook 来更新容器的 systemd 单元。有关解决 director 驱动的 Ceph Storage 部署中的问题的信息，请参阅红帽知识库解决方案 问题影响 Red Hat Had Ceph Storage 3 的次要更新可能会导致 OSD 崩溃 。

OSP13 更新可能会在最终成功时显示失败

openstack overcloud update run 命令中使用的 python **tripleo-client** 可能会在更新过程完成前超时。这会导致 **openstack overcloud update run** 命令返回失败，更新过程会继续在后台运行，直到完成为止。

要避免此失败，请编辑 **tripleo-client/plugin.py** 文件中的 **ttl** 参数的值，以便在更新 overcloud 节点前增加 **tripleo-client** 超时值。如需更多信息，请参阅红帽知识库解决方案 OSP 13 更新过程在更新过程在后台运行并成功完成时会出现失败。

在 rabbitmq 连接中，slight cut 在完全同步后触发数据平面丢失

如果要从 RHOSP 13 z10 之前的版本(2019 年 12 月 19 日)更新您的环境，以避免在 bug BZ remove5538 中描述的数据平面连接丢失，请参阅 OSP13 上的红帽知识库解决方案 Stale 命名空间可以在更新过程中创建数据平面 丢失。

在 ceph 升级所有 OSD （及其他 ceph 服务）的过程中，

如果您使用 Ceph，请参阅红帽知识库解决方案，在 OSP13/RHCS3 的次要更新期间，到最新的软件包 Ceph 服务离线，需要手动重启， 以便在完成以下步骤前手动重启 以避免 bug BZ -2021-20842：

- 更新所有 Controller 节点
- 更新所有 HCI Compute 节点
- 更新所有 Ceph Storage 节点

Octavia 和 LB 在 z11 升级后的问题

在更新过程中，因为缺少名为 **/var/lib/config-data/puppet-generated/octavia/etc/octavia/conf.d/common/post-deploy.conf** 的文件，所以负载均衡服务(Octavia)容器将持续重启。此文件是在 Red Hat OpenStack Platform 13 生命周期中引入的，以便在 Amphora 部署后配置 octavia 服务。此文件目前会在更新的 **openstack overcloud update converge** 步骤中生成。要临时解决这个问题，您必须继续更新。在运行 **openstack overcloud update converge** 命令后，octavia 容器会正常启动。Red Hat OpenStack Platform 工程团队目前正在调查这个问题的解决方法。

DBAPIError exception wrap from (pymysql.err.InternalError) （1054, u"Unknown column 'pool.tls_certificate_id' in 'field list'"

如果您使用负载均衡服务(octavia)，并希望从 RHOSP 13 z13 之前的版本更新(8 2020 年 10 月 8 日)以避免 bug BZheketi169，您必须以正确顺序运行升级负载均衡服务的数据库迁移。您必须更新 bootstrap Controller 节点，然后才能更新 control plane 的其余部分。

1. 要识别您当前的维护发行版本，请运行以下命令：

```
$ cat /etc/rhosp-release
```

2. 在 undercloud 节点上，要识别 bootstrap Controller 节点，请运行以下命令，并将 <**any_controller_node_IP_address**> 替换为部署中任何 Controller 节点的 IP 地址：

```
$ ssh heat-admin@<any_controller_node_IP_address> sudo hiera -c /etc/puppet/hiera.yaml octavia_api_short_bootstrap_node_name
```

3. 在 undercloud 节点上，运行 **openstack overcloud update run** 命令来更新 bootstrap Controller 节点：

```
$ openstack overcloud update run --nodes <bootstrap_node_name>
```

到 13z16 的次版本更新失败，并显示 "Unable to find constraint"

当您重启 Red Hat OpenStack Platform 13z16 overcloud 节点的更新时，您可能会遇到 **Unable to find constraint** 错误。发生此错误的原因是 RabbitMQ 版本在更新过程中存在差异。为确保新的 RabbitMQ 版本可以启动，您必须清除 overcloud 中可能存在的任何 pacemaker bans。
有关此问题的更多信息，请参阅红帽知识库解决方案 无法重启 OSP13z16 控制器。

无法在控制器上停止 ceph-monmon controller: No such container: ceph-mon controller-2

如果您使用 Red Hat Ceph Storage 版本 3.3 z5 或更早版本，并将 docker 软件包更新至 docker-1.13.1-209，RHOSP 13 更新会失败。RHOSP 13 更新不会在 docker 软件包更新前停止 ceph-mon 容器。这会造成一个孤立的 ceph-mon 进程，该进程会阻止新的 ceph-mon 容器启动。
有关此问题的更多信息，请参阅红帽知识库解决方案 更新 Red Hat OpenStack Platform 13.z12 及更早版本，在控制器更新过程中可能会失败。

1.3. 故障排除

- 如果更新过程的时间比预期的要长，则可能会超时，并显示 error: **socket 已经关闭**。这可能是因为将 undercloud 的身份验证令牌设置为在设定持续时间后过期。如需更多信息，请参阅推荐大型部署。

第 2 章 更新容器镜像源

本章介绍了如何使用 Red Hat OpenStack Platform 的新 overcloud 容器镜像更新 registry 源。

更新容器镜像源前的注意事项

如果要将容器镜像源从一个 registry 类型改为另一个 registry，您必须更新当前容器镜像环境文件中的命名空间和前缀，并运行 **openstack overcloud deploy** 命令在完成任何以下任务前更改命名空间：

- 第 2.4 节 “使用红帽 registry 作为远程 registry 源”
- 第 2.5 节 “使用 undercloud 作为本地 registry”
- 第 2.6 节 “将 Satellite 服务器用作 registry”

2.1. REGISTRY 方法

Red Hat OpenStack Platform 支持以下 registry 类型：

远程 Registry

overcloud 直接从 **registry.redhat.io** 拉取容器镜像。此方法最容易生成初始配置。但是，每个 overcloud 节点直接从 Red Hat Container Catalog 拉取每个镜像，这可能会导致网络拥塞和较慢的部署。此外，所有 overcloud 节点都需要访问互联网。

本地 Registry

undercloud 使用 **docker-distribution** 服务来充当注册表。这允许 director 从 **registry.redhat.io** 同步镜像，并将它们推送到 **docker-distribution** registry。在创建 overcloud 时，overcloud 从 undercloud 的 **docker-distribution** 注册表中提取容器镜像。这个方法允许您在内部存储 registry，这可以加快部署并减少网络拥塞。但是，undercloud 仅充当一个基本的注册表，并为容器镜像提供有限的生命周期管理。

注意

docker-distribution 服务的工作方式与 **docker** 分开。Docker 用于将镜像拉取和推送到 **docker -distribution** 注册表，不为 overcloud 提供镜像。overcloud 从 **docker-distribution** 注册表中提取镜像。

Satellite Server

管理容器镜像的完整应用程序生命周期，并通过 Red Hat Satellite 6 服务器发布它们。overcloud 从 Satellite 服务器拉取镜像。此方法提供了一个企业级解决方案，用于存储、管理和部署 Red Hat OpenStack Platform 容器。

从列表中选择一个方法，并继续配置 registry 详情。

注意

在构建多架构云时，不支持本地 registry 选项。

2.2. 容器镜像准备命令使用

本节概述了如何使用 **openstack overcloud container image prepare** 命令，包括命令的各种选项的概念信息。

为 Overcloud 生成容器镜像环境文件

openstack overcloud container image prepare 命令的一个主要用途是创建一个包含 overcloud 使用的镜像列表的环境文件。您可以使用 overcloud 部署命令包含此文件，如 **openstack overcloud deploy**。**openstack overcloud container image prepare** 命令使用以下选项进行此功能：

--output-env-file

定义生成的环境文件名称。

以下片段是此文件的内容示例：

```
parameter_defaults:
  DockerAodhApiImage: registry.redhat.io/rhosp13/openstack-aodh-api:13.0-34
  DockerAodhConfigImage: registry.redhat.io/rhosp13/openstack-aodh-api:13.0-34
...
```

该环境文件还包含 **DockerInsecureRegistryAddress** 参数设置为 undercloud registry 的 IP 地址和端口。此参数配置 overcloud 节点，以在没有 SSL/TLS 认证的情况下从 undercloud registry 访问镜像。

为导入方法生成容器镜像列表

如果要将 OpenStack Platform 容器镜像导入到不同的 registry 源，您可以生成镜像列表。列表的语法主要用于将容器镜像导入到 undercloud 上的容器 registry，但您可以修改此列表的格式以适应其他导入方法，如 Red Hat Satellite 6。

openstack overcloud container image prepare 命令使用以下选项进行此功能：

--output-images-file

定义导入列表生成的文件名。

以下是此文件内容的示例：

```
container_images:
- imagename: registry.redhat.io/rhosp13/openstack-aodh-api:13.0-34
- imagename: registry.redhat.io/rhosp13/openstack-aodh-evaluator:13.0-34
...
```

为容器镜像设置命名空间

--output-env-file 和 **--output-images-file** 选项都需要命名空间来生成生成的镜像位置。**openstack overcloud container image prepare** 命令使用以下选项来设置要拉取的容器镜像的源位置：

--namespace

定义容器镜像的命名空间。这通常是带有目录的主机名或 IP 地址。

--prefix

定义在镜像名称之前添加的前缀。

因此，director 使用以下格式生成镜像名称：

- **[NAMESPACE]/[PREFIX][IMAGE NAME]**

设置容器镜像标签

将 **--tag** 和 **--tag-from-label** 选项一起使用为每个容器镜像设置标签。

--tag

为来自源的所有镜像设置特定标签。如果您只使用这个选项，director 会使用此标签拉取所有容器镜像。但是，如果您将这个选项与 **--tag-from-label** 结合使用，director 将使用 **--tag** 作为源镜像来根据标签识别特定版本标签。**--tag** 选项默认设置为 **latest**。

--tag-from-label

使用指定容器镜像标签的值来发现和拉取每个镜像的 versioned 标签。director 会检查使用您为 **--tag** 设置的值标记的每个容器镜像，然后使用容器镜像标签来构建一个新的标签，director 从 registry 中拉取新标签。例如，如果您设置了 **--tag-from-label {version}-{release}**，director 将使用 **version** 和 **release** 标签来构建新标签。对于一个容器，**version** 可能会设置为 **13.0**，**release** 可能会设置为 **34**，这会导致标签 **13.0-34**。

重要

Red Hat Container Registry 使用特定的版本格式来标记所有 Red Hat OpenStack Platform 容器镜像。此版本格式为 **{version}-{release}**，每个容器镜像都作为标签存储在容器元数据中。这个版本格式有助于减少从一个 **{release}** 到下一个版本的更新。因此，在运行 **openstack overcloud container image prepare** 命令时，您必须始终使用 **--tag-from-label {version}-{release}**。不要单独使用 **--tag** 来拉取容器镜像。例如，使用 **--tag latest** 会导致执行更新时出现问题，因为 director 需要更改标签来更新容器镜像。

2.3. 用于其他服务的容器镜像

director 只为核心 OpenStack Platform 服务准备容器镜像。其他一些功能使用需要额外容器镜像的服务。您可以使用环境文件启用这些服务。**openstack overcloud container image prepare** 命令使用以下选项包含环境文件及其对应的容器镜像：

-e

包含环境文件以启用其他容器镜像。

下表提供了使用容器镜像及其在 **/usr/share/openstack-tripleo-heat-templates** 目录中的相应环境文件位置的其他服务示例。

服务	环境文件
Ceph Storage	**environments/ceph-ansible/ceph-ansible.yaml**
collectd	**environments/services-docker/collectd.yaml**
Congress	**environments/services-docker/congress.yaml**
Fluentd	**environments/services-docker/fluentd.yaml**
OpenStack Bare Metal (ironic)	**environments/services-docker/ironic.yaml**
OpenStack 数据处理(sahara)	**environments/services-docker/sahara.yaml**
OpenStack EC2-API	**environments/services-docker/ec2-api.yaml**
OpenStack Key Manager (barbican)	**environments/services-docker/barbican.yaml**

服务	环境文件
OpenStack Load Balancing-as-a-Service (octavia)	**environments/services-docker/octavia.yaml**
OpenStack Shared File System Storage (manila)	**environments/manila-{backend-name}-config.yaml** 注意：如需更多信息，请参阅 OpenStack 共享文件系统服务(manila)。
开放虚拟网络(OVN)	**environments/services-docker/neutron-ovn-dvr-ha.yaml**
Sensu	**environments/services-docker/sensu-client.yaml**

接下来的几个部分提供了包括其他服务的示例。

Ceph Storage

如果使用 overcloud 部署 Red Hat Ceph Storage 集群，则需要包含 **/usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-ansible.yaml** 环境文件。此文件支持 overcloud 中的可组合容器化服务，并且 director 需要知道这些服务才能准备其镜像。

除了此环境文件外，您还需要定义 Ceph Storage 容器位置，它与 OpenStack Platform 服务不同。使用 **--set** 选项设置特定于 Ceph Storage 的以下参数：

--set ceph_namespace

定义 Ceph Storage 容器镜像的命名空间。这个功能与 **--namespace** 选项类似。

--set ceph_image

定义 Ceph Storage 容器镜像的名称。通常，这是 **rhceph-3-rhel7**。

--set ceph_tag

定义用于 Ceph Storage 容器镜像的标签。这个功能与 **--tag** 选项类似。当指定 **--tag-from-label** 时，从此标签开始会发现版本化的标签。

以下是在容器镜像文件中包含 Ceph Storage 的示例：

```
$ openstack overcloud container image prepare \
  ...
  -e /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-ansible.yaml \
  --set ceph_namespace=registry.redhat.io/rhceph \
  --set ceph_image=rhceph-3-rhel7 \
  --tag-from-label {version}-{release} \
  ...
```

OpenStack Bare Metal (ironic)

如果在 overcloud 中部署 OpenStack Bare Metal (ironic)，您需要包含 **/usr/share/openstack-tripleo-heat-templates/environments/services-docker/ironic.yaml** 环境文件，以便 director 可以准备镜像。以下片段是一个如何包含此环境文件的示例：

```
$ openstack overcloud container image prepare \
  ...
  -e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/ironic.yaml \
  ...
```

-

OpenStack 数据处理(sahara)

如果在 overcloud 中部署 OpenStack Data Processing (sahara)，您需要包含 **/usr/share/openstack-tripleo-heat-templates/environments/services-docker/sahara.yaml** 环境文件，以便 director 可以准备镜像。以下片段是一个如何包含此环境文件的示例：

```
$ openstack overcloud container image prepare \
  ...
  -e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/sahara.yaml \
  ...
```

OpenStack Neutron SR-IOV

如果在 overcloud 中部署 OpenStack Neutron SR-IOV，请包含 **/usr/share/openstack-tripleo-heat-templates/environments/services-docker/neutron-sriov.yaml** 环境文件，以便 director 可以准备镜像。默认 Controller 和 Compute 角色不支持 SR-IOV 服务，因此您必须使用 **-r** 选项包括包含 SR-IOV 服务的自定义角色文件。以下片段是一个如何包含此环境文件的示例：

```
$ openstack overcloud container image prepare \
  ...
  -r ~/custom_roles_data.yaml
  -e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/neutron-sriov.yaml \
  ...
```

OpenStack Load Balancing-as-a-Service (octavia)

如果在 overcloud 中部署 OpenStack Load Balancing-as-a-Service，请包含 **/usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml** 环境文件，以便 director 可以准备镜像。以下片段是一个如何包含此环境文件的示例：

```
$ openstack overcloud container image prepare \
  ...
  -e /usr/share/openstack-tripleo-heat-templates/environments/services-docker/octavia.yaml
\
  ...
```

OpenStack Shared File System (manila)

使用 **manila-{backend-name}-config.yaml** 格式，您可以选择使用该后端部署共享文件系统服务的后端。可以通过包括以下环境文件来准备共享文件系统服务容器：

```
environments/manila-isilon-config.yaml
environments/manila-netapp-config.yaml
environments/manila-vmax-config.yaml
environments/manila-cephfsnative-config.yaml
environments/manila-cephfsganesha-config.yaml
environments/manila-unity-config.yaml
environments/manila-vnx-config.yaml
```

有关自定义和部署环境文件的更多信息，请参阅以下资源：

- *对于共享文件系统服务，通过 NFS 后端指南在 CephFS 中部署更新的环境*

- 使用 NetApp 后端指南中的使用 NetApp 后端部署共享文件系统服务，作为 *共享文件系统服务的 NetApp 后端指南*
- 使用 CephFS Back End Guide for the Shared File System Service for the Shared File System Service with a *CephFS Back End* Guide for the Shared File System Service

2.4. 使用红帽 REGISTRY 作为远程 REGISTRY 源

红帽在 **registry.redhat.io** 上托管 overcloud 容器镜像。从远程 registry 拉取镜像是最简单的方法，因为 registry 已配置，所有您需要都是您要拉取的镜像的 URL 和命名空间。但是，在创建 overcloud 的过程中，overcloud 节点都会从远程存储库中拉取镜像，该仓库可编排外部连接。因此，不建议在生产环境中使用这个方法。对于生产环境，请使用以下方法之一：

- 设置本地 registry
- 在 Red Hat Satellite 6 上托管镜像

流程

1. 要直接从 overcloud 部署中的 **registry.redhat.io** 拉取镜像，需要环境文件来指定镜像参数。运行以下命令以生成容器镜像环境文件：

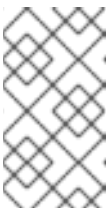
```
(undercloud) $ sudo openstack overcloud container image prepare \
  --namespace=registry.redhat.io/rhosp13 \
  --prefix=openstack- \
  --tag-from-label {version}-{release} \
  --output-env-file=/home/stack/templates/overcloud_images.yaml
```

   - 使用 **-e** 选项包括可选服务的任何环境文件。
   - 使用 **-r** 选项包含自定义角色文件。
   - 如果使用 Ceph Storage，请包含额外的参数来定义 Ceph Storage 容器镜像位置：-- **set ceph_namespace ,--set ceph _image,--set ceph_tag**。

2. 修改 **overcloud_images.yaml** 文件，并包含以下参数以在部署期间与 **registry.redhat.io** 进行身份验证：

```
ContainerImageRegistryLogin: true
ContainerImageRegistryCredentials:
  registry.redhat.io:
    <USERNAME>: <PASSWORD>
```

   - 将 &**lt;USERNAME&** gt **；和 <PASSWORD** > 替换为您的 **registry.redhat.io** 的凭证。**overcloud_images.yaml** 文件包含 undercloud 上的镜像位置。将此文件与您的部署一起包含。



注意

在运行 **openstack overcloud deploy** 命令前，您必须登录到远程 registry：

```
(undercloud) $ sudo docker login registry.redhat.io
```

registry 配置已就绪。

2.5. 使用 UNDERCLOUD 作为本地 REGISTRY

您可以在 undercloud 上配置本地 registry 以存储 overcloud 容器镜像。

您可以使用 director 从 **registry.redhat.io** 中拉取每个镜像，并将每个镜像推送到 undercloud 上运行的 **docker-distribution** registry。在使用 director 创建 overcloud 时，节点会从 undercloud **docker-distribution** registry 中拉取相关的镜像。

这会为内部网络中的容器镜像保留网络流量，这不会限制您的外部网络连接，并可加快部署过程。

流程

1. 查找本地 undercloud registry 的地址。地址使用以下模式：

```
<REGISTRY_IP_ADDRESS>:8787
```

使用之前通过 **undercloud.conf** 文件中的 **local_ip** 参数设置的 undercloud 的 IP 地址。对于下面的命令，地址假定为 **192.168.24.1:8787**。

2. 登录到 **registry.redhat.io** ：

```
(undercloud) $ docker login registry.redhat.io --username $RH_USER --password
$RH_PASSWD
```

3. 创建模板以将镜像上传到本地 registry，环境文件来引用这些镜像：

```
(undercloud) $ openstack overcloud container image prepare \
  --namespace=registry.redhat.io/rhosp13 \
  --push-destination=192.168.24.1:8787 \
  --prefix=openstack- \
  --tag-from-label {version}-{release} \
  --output-env-file=/home/stack/templates/overcloud_images.yaml \
  --output-images-file /home/stack/local_registry_images.yaml
```

- 使用 **-e** 选项包括可选服务的任何环境文件。
- 使用 **-r** 选项包含自定义角色文件。
- 如果使用 Ceph Storage，请包含额外的参数来定义 Ceph Storage 容器镜像位置：-- **set ceph_namespace ,--set ceph _image,--set ceph_tag**。

4. 验证是否已创建以下两个文件：

- **local_registry_images.yaml**，其中包含来自远程源的容器镜像信息。使用此文件将镜像从 Red Hat Container Registry (**registry.redhat.io**)拉取镜像到 undercloud。
- **overcloud_images.yaml**，其中包含 undercloud 上的最终镜像位置。您可以在部署中包含此文件。

5. 从远程 registry 中拉取容器镜像并将其推送到 undercloud registry：

```
(undercloud) $ openstack overcloud container image upload \
  --config-file  /home/stack/local_registry_images.yaml \
  --verbose
```

拉取所需的镜像可能需要一些时间，具体取决于您的网络和 undercloud 磁盘的速度。

注意

容器镜像消耗大约 10 GB 磁盘空间。

6. 镜像现在存储在 undercloud 的 **docker-distribution** 注册表中。要查看 undercloud 的 **docker-distribution** registry 上的镜像列表，请运行以下命令：

```
(undercloud) $  curl http://192.168.24.1:8787/v2/_catalog | jq .repositories[]
```

注意

_catalog 资源本身仅显示 100 个镜像。要显示更多镜像，请使用带有 **_catalog** 资源的 **?n=<interger** > 查询字符串来显示更多镜像：

```
(undercloud) $  curl http://192.168.24.1:8787/v2/_catalog?n=150 | jq
.repositories[]
```

要查看特定镜像的标签列表，请使用 **skopeo** 命令：

```
(undercloud) $ curl -s http://192.168.24.1:8787/v2/rhosp13/openstack-keystone/tags/list | jq
.tags
```

要验证标记的镜像，请使用 **skopeo** 命令：

```
(undercloud) $ skopeo inspect --tls-verify=false
docker://192.168.24.1:8787/rhosp13/openstack-keystone:13.0-44
```

registry 配置已就绪。

2.6. 将 SATELLITE 服务器用作 REGISTRY

Red Hat Satellite 6 提供了注册表同步功能。通过该功能可将多个镜像提取到 Satellite 服务器中，作为应用程序生命周期的一部分加以管理。Satellite 也可以作为 registry 供其他启用容器功能的系统使用。有关管理容器镜像的更多信息，请参阅 *Red Hat Satellite 6 内容管理指南* 中的"管理容器镜像"。

以下操作过程示例中使用了 Red Hat Satellite 6 的 **hammer** 命令行工具和一个名为 **ACME** 的示例组织。请将该组织替换为您自己 Satellite 6 中的组织。

流程

1. 创建模板以将镜像拉取到本地 registry:

```
$ source ~/stackrc
(undercloud) $ openstack overcloud container image prepare \
  --namespace=rhosp13 \
  --prefix=openstack- \
  --output-images-file /home/stack/satellite_images
```

- 使用 **-e** 选项包括可选服务的任何环境文件。

- 使用 **-r** 选项包含自定义角色文件。
- 如果使用 Ceph Storage，请包含额外的参数来定义 Ceph Storage 容器镜像位置：-- **set ceph_namespace ,--set ceph _image,--set ceph_tag**。

注意

此版本的 **openstack overcloud container image prepare** 命令以 registry 上的 **registry** 为目标，以生成镜像列表。它使用与后续步骤中使用的 **openstack overcloud container image prepare** 命令不同的值。

2. 这会创建一个名为 **satellite_images** 的文件，以及您的容器镜像信息。您将使用此文件将容器镜像同步到 Satellite 6 服务器。
3. 从 **satellite_images** 文件中删除 YAML 特定信息，并将其转换为仅包含镜像列表的平面文件。以下 **sed** 命令完成此操作：

```
(undercloud) $ awk -F ':' '{if (NR!=1) {gsub("[[:space:]]", ""); print $2}}' ~/satellite_images > ~/satellite_images_names
```

这提供了您拉取到 Satellite 服务器的镜像列表。

4. 将 **satellite_images_names** 文件复制到包含 Satellite 6 **hammer** 工具的系统中。或者，根据 *Hammer CLI 指南* 中的说明将 **hammer** 工具安装到 undercloud 中。
5. 运行以下 **hammer** 命令，为您的 Satellite 组织创建新产品（**OSP13 容器**）：

```
$ hammer product create \
  --organization "ACME" \
  --name "OSP13 Containers"
```

该定制产品将会包含我们的镜像。

6. 为产品添加基本容器镜像：

```
$ hammer repository create \
  --organization "ACME" \
  --product "OSP13 Containers" \
  --content-type docker \
  --url https://registry.redhat.io \
  --docker-upstream-name rhosp13/openstack-base \
  --name base
```

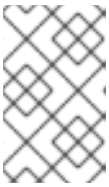
7. 添加 **satellite_images** 文件中的 overcloud 容器镜像。

```
$ while read IMAGE; do \
  IMAGENAME=$(echo $IMAGE | cut -d"/" -f2 | sed "s/openstack-//g" | sed "s/:.*//g") ; \
  hammer repository create \
  --organization "ACME" \
  --product "OSP13 Containers" \
  --content-type docker \
  --url https://registry.redhat.io \
  --docker-upstream-name $IMAGE \
  --name $IMAGENAME ; done < satellite_images_names
```

8. 同步容器镜像：

```
$ hammer product synchronize \
  --organization "ACME" \
  --name "OSP13 Containers"
```

等待 Satellite 服务器完成同步。



注意

根据具体配置情况，**hammer** 可能会询问您的 Satellite 服务器用户名和密码。您可以使用配置文件将 **hammer** 配置为自动登录。请参阅 *Hammer CLI 指南中的* "身份验证" 部分。

9. 如果您的 Satellite 6 服务器使用内容视图，请创建一个新的内容视图版本来包含镜像。

10. 检查可用于 **基础镜像** 的标签：

```
$ hammer docker tag list --repository "base" \
  --organization "ACME" \
  --product "OSP13 Containers"
```

这将显示 OpenStack Platform 容器镜像的标签。

11. 返回到 undercloud，再为卫星服务器上的镜像生成环境文件。以下是生成环境文件的示例命令：

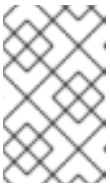
```
(undercloud) $ openstack overcloud container image prepare \
  --namespace=satellite6.example.com:5000 \
  --prefix=acme-osp13_containers- \
  --tag-from-label {version}-{release} \
  --output-env-file=/home/stack/templates/overcloud_images.yaml
```

注意

此版本的 **openstack overcloud container image prepare** 命令以 Satellite 服务器为目标。它使用与上一步中使用的 **openstack overcloud container image prepare** 命令不同的值。

在运行这个命令时，包括以下数据：

- **--namespace** - Satellite 服务器上 registry 的 URL 和端口。Red Hat Satellite 上的 registry 端口是 5000。例如，-- **namespace=satellite6.example.com:5000**。



注意

如果您使用 Red Hat Satellite 版本 6.10，则不需要指定端口。使用 **443** 的默认端口。如需更多信息，请参阅 "如何将 RHOSP13 部署到 Red Hat Satellite 6.10?"。

- **--prefix=** - 前缀基于标签的 Satellite 6 约定，它使用小写字符并替换下划线的空格。根据您是否使用内容视图，前缀会有所不同：
  - 如果您使用了内容视图，则前缀的结构为 **[组织]-[环境]-[内容视图]-[产品]-**。例如：

acme-production-myosp13-osp13_containers-。

- 如果不使用内容视图，则前缀的结构为 **[组织]-[产品]-**。例如： **acme-osp13_containers-**。

- **--tag-from-label {version}-{release}** - 标识每个镜像的 latest 标签。
- **-e** - 包含任何用于可选服务的环境文件。
- **-r** - 包含自定义角色文件。
- **--set ceph_namespace**,**--set ceph_image**,**--set ceph_tag** - 如果使用 Ceph Storage，请包含额外的参数来定义 Ceph Storage 容器镜像位置。请注意，**ceph_image** 现包含特定于 Satellite 的前缀。这个前缀与 **--prefix** 选项的值相同。例如：

```
--set ceph_image=acme-osp13_containers-rhceph-3-rhel7
```

这可确保 overcloud 使用采用卫星命名约定的 Ceph 容器镜像。

12. **overcloud_images.yaml** 文件包含 Satellite 服务器上的镜像位置。将此文件与您的部署一起包含。

registry 配置已就绪。

2.7. 后续步骤

现在，您有一个新的 **overcloud_images.yaml** 环境文件，其中包含容器镜像源列表。包含所有将来的升级和部署操作，包括此文件。

现在，您可以为更新准备 overcloud。

第 3 章 升级 UNDERCLOUD

此流程将 undercloud 及其 overcloud 镜像升级到 **Red Hat OpenStack Platform 13**。

3.1. 执行 UNDERCLOUD 的次要更新

director 提供了更新 undercloud 节点上的软件包的命令。这样，您可以在当前版本的 OpenStack Platform 环境中执行次要更新。

流程

1. 以 **stack** 用户身份登录 director。
2. 更新 **python-tripleoclient** 软件包及其依赖项，以确保具有次版本更新的最新脚本：

```
$ sudo yum update -y python-tripleoclient
```

3. director 使用 **openstack undercloud upgrade** 命令更新 Undercloud 环境。运行以下命令：

```
$ openstack undercloud upgrade
```

4. 等待 undercloud 升级过程完成。
5. 重新引导 undercloud 以更新操作系统的内核和其他系统软件包：

```
$ sudo reboot
```

6. 稍等片刻，直到节点启动。

3.2. 更新 OVERCLOUD 镜像

您需要将当前的 overcloud 镜像替换为新版本。新镜像确保 director 可以使用最新版本的 OpenStack Platform 软件内省和置备节点。

前提条件

- 您已将 undercloud 更新至最新版本。

流程

1. 从 stack 用户主目录(/**home**/ **stack** / **images**)上的 images 目录中删除任何现有的镜像：

```
$ rm -rf ~/images/*
```

2. 解压存档：

```
$ cd ~/images
$ for i in /usr/share/rhosp-director-images/overcloud-full-latest-13.0.tar /usr/share/rhosp-director-images/ironic-python-agent-latest-13.0.tar; do tar -xvf $i; done
$ cd ~
```

3. 在 undercloud 节点上，提供 undercloud 凭证：

```
$ source ~/stackrc
```

4. 将最新的镜像导入到 director：

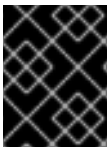
```
$ openstack overcloud image upload --update-existing --image-path /home/stack/images/
```

5. 将节点配置为使用新镜像：

```
$ openstack overcloud node configure $(openstack baremetal node list -c UUID -f value)
```

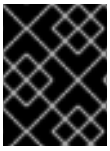
6. 验证新镜像是否存在：

```
$ openstack image list
$ ls -l /httpboot
```



重要

在部署 overcloud 节点时，确保 overcloud 镜像版本对应于对应的 heat 模板版本。例如，仅使用带有 OpenStack Platform 13 heat 模板的 OpenStack Platform 13 镜像。



重要

新的 **overcloud-full** 镜像替换旧的 **overcloud-full** 镜像。如果对旧镜像进行了更改，您必须对新镜像重复更改，特别是未来要部署新节点时。

3.3. UNDERCLOUD POST-UPGRADE NOTES

- 如果在 **stack** 用户主目录中使用一组本地核心模板，请确保使用" 自定义核心 Heat 模板"中的推荐工作流更新模板。在升级 overcloud 之前，您必须更新本地副本。

3.4. 后续步骤

undercloud 升级已完成。现在，您可以为升级准备 overcloud。

第 4 章 更新 OVERCLOUD

此过程更新 overcloud。

前提条件

- 您已将 undercloud 更新至最新版本。

4.1. 加快 OVERCLOUD 更新的速度

为加快 overcloud 更新过程，您可以配置 **DockerPuppetProcessCount** heat 参数，归档已删除的数据库条目，并在执行更新前在 overcloud 节点上下载所需的软件包。

有关加快大型 OpenStack 部署更新过程的更多信息，请参阅红帽知识库文章 Openstack Director 节点性能调优。

流程

1. 以 **stack** 用户的身份登录 undercloud。
2. Source **stackrc** 文件：

```
$ source ~/stackrc
```

3. 要增加 **container-puppet** 用来生成配置文件的并发进程数量，您必须配置 **DockerPuppetProcessCount** 参数。
   a. 在 **templates** 目录中创建一个名为 **updates-environment.yaml** 的环境文件：

```
$ touch ~/templates/updates-environment.yaml
```

   b. 编辑该文件并添加以下内容：

```
parameter_defaults:
  DockerPuppetProcessCount: 8
```

   c. 在运行 **openstack overcloud update prepare**、**openstack overcloud ceph-upgrade run**, 和 **openstack overcloud update converge** 命令时，使用 **-e** 选项包含此环境文件。
4. 在 Controller 节点上，归档您的已删除的数据库条目：
   a. 从 overcloud，列出 Controller 节点的所有实例：

```
$ source ~/overcloudrc
$ openstack server list
```

   b. 登录到运行 **nova_api_cron** 容器的 Controller 节点：

```
ssh heat-admin@<controller_ip>
```

   - 将 **<controller name 或 IP** > 替换为 Controller 节点的 IP 地址。

   c. 归档已删除的数据库条目：

```
$ sudo docker exec -u 42436 -ti nova_api_cron bash
$ nova-manage db archive_deleted_rows --max_rows 1000
$ exit
```

5. 要下载所有 overcloud 节点上更新所需的所有软件包，请完成以下步骤：

   a. 创建 overcloud 的静态清单文件：

   ```
   $ tripleo-ansible-inventory \
   --ansible_ssh_user heat-admin \
   --static-yaml-inventory ~/inventory.yaml
   ```

   b. 创建以下 Ansible playbook：

   ```
   $ cat > ~/yum-download-only.yaml <<'EOF'
   - hosts: all
     gather_facts: false
     tasks:
       - name: Pre-download all packages on all overcloud nodes
         shell:
           yum upgrade -y --downloadonly
         become: true
   EOF
   ```

   c. 运行 **yum-download-only.yaml** Ansible playbook:

   ```
   $ ansible-playbook \
   -i ~/inventory.yaml \
   -f 20 ~/yum-download-only.yaml \
   --limit Controller,Compute,CephStorage
   ```

4.2. 自定义角色考虑

如果您的部署包含自定义角色，请检查角色文件中的以下值：

- 将您的自定义角色文件与 **/usr/share/openstack-tripleo-heat-templates/roles** 目录中的最新文件进行比较。将您环境的 **RoleParametersDefault** 部分中的任何新参数添加到自定义角色文件中等同的角色。
- 如果您使用 Data Plane Development Kit (DPDK)并从 13.4 或更早版本升级，请确保包含 OVS-DPDK 服务的角色也包含以下强制参数：

  ```
  RoleParametersDefault:
    VhostuserSocketGroup: "hugetlbfs"
    TunedProfileName: "cpu-paritioning"
    NovaLibvirtRxQueueSize: 1024
    NovaLibvirtTxQueueSize: 1024
  ```

4.3. 运行 OVERCLOUD 更新准备

更新需要运行 **openstack overcloud update prepare** 命令，该命令执行以下任务：

- 将 overcloud 计划更新为 OpenStack Platform 13

- 为更新准备节点

流程

1. Source **stackrc** 文件：

```
$ source ~/stackrc
```

2. 运行更新准备命令：

```
$ openstack overcloud update prepare \
  --templates \
  -r <ROLES DATA FILE> \
  -n <NETWORK DATA FILE> \
  -e /home/stack/templates/overcloud_images.yaml \
  -e /home/stack/templates/updates-environment.yaml \
  -e <ENVIRONMENT FILE> \
  -e <ENVIRONMENT FILE> \
  --stack <STACK_NAME>
  ...
```

包含以下与您的环境相关的选项：

- 自定义配置环境文件(**例如**)。
- 带有新容器镜像位置的环境文件(**-e**)。请注意，update 命令可能会显示关于使用 **--container-registry-file** 的警告。您可以忽略这个警告，因为此选项已弃用，而是对容器镜像环境文件使用 **-e**。
- 如果您使用自己的自定义角色，请包含您的自定义角色(**roles_data**)文件(**-r**)。
- 如果您使用自定义网络，请包含可组合网络(**network_data**)文件(**-n**)。
- 如果您部署高可用性集群，请在 update preparation 命令中包含 **--ntp-server** 选项，或者在环境文件中包含 **NtpServer** 参数和值。
- 如果 **overcloud** 堆栈的名称与默认名称 overcloud 不同，请在更新准备命令中包含 **--stack** 选项，并将 < **STACK_NAME&** gt; 替换为堆栈的名称。

3. 等待更新准备完成。

4.4. 更新 CEPH STORAGE 集群

此过程更新 Ceph Storage 集群。这个过程涉及运行 **openstack overcloud ceph-upgrade run** 命令，以对 Red Hat Ceph Storage 3 集群执行更新。

注意

支持以下 Ansible 与 **ceph-ansible** 的组合：

- **ansible-2.6** 带有 **ceph-ansible-3.2**
- 使用 **ceph-ansible-3.1**的 **ansible-2.4**

如果您的环境有带有 **ceph-ansible-3.1** 的 **ansible-2.6**，请将 **ceph-ansible** 更新至最新版本：

```
# subscription-manager repos --enable=rhel-7-server-rhceph-3-tools-rpms
# subscription-manager repos --enable=rhel-7-server-ansible-2.6-rpms
# yum update ceph-ansible
```

流程

1. Source **stackrc** 文件：

```
$ source ~/stackrc
```

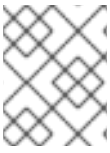
2. 运行 Ceph Storage update 命令。例如：

```
$ openstack overcloud ceph-upgrade run \
  --templates \
  -e <ENVIRONMENT FILE> \
  -e /home/stack/templates/overcloud_images.yaml \
  -e /home/stack/templates/updates-environment.yaml
```

包含以下与您的环境相关的选项：

- 自定义配置环境文件(**-e**)
- 带有容器镜像位置的环境文件(**-e**)。请注意，update 命令可能会显示关于使用 **--container-registry-file** 的警告。您可以忽略这个警告，因为此选项已弃用，而使用 **-e** 作为容器镜像环境文件。
- 如果适用，您的自定义角色(**roles_data**)文件(**--roles-file**)
- 如果适用，您的可组合网络(**network_data**)文件(**--networks-file**)

3. 等待 Ceph Storage 节点更新完成。



注意

如果 Heat 堆栈在流程期间超时，请参阅红帽知识库文章 在 Ceph 节点后续更新期间红帽知识库文章 **openstack overcloud ceph-upgrade run** 似乎超时。

4.5. 更新所有 CONTROLLER 节点

要将 Controller 节点更新至最新的 Red Hat OpenStack Platform (RHOSP) 13 版本，请在 **openstack overcloud update run** 命令中包含 **--nodes Controller** 选项。**--nodes Controller** 选项仅将更新操作限制为 Controller 节点。

警告

如果使用 Ceph，请参阅红帽知识库解决方案，在 OSP13/RHCS3 的次要更新期间，到最新的软件包 Ceph 服务处于离线状态，并在更新 Controller 节点前手动重启，以避免 bug BZ-2021-3084 2。

前提条件

- 如果您使用负载均衡服务(octavia)，并希望从 RHOSP 13 z13 之前的版本更新(8 2020 年 10 月 8 日)以避免 bug BZheketi169，您必须以正确顺序运行升级负载均衡服务的数据库迁移。您必须更新 bootstrap Controller 节点，然后才能更新 control plane 的其余部分。

  1. 要识别您当前的维护发行版本，请运行以下命令：

     ```
     $ cat /etc/rhosp-release
     ```

  2. 在 undercloud 节点上，要识别 bootstrap Controller 节点，请运行以下命令，并将 <**any_controller_node_IP_address**> 替换为部署中任何 Controller 节点的 IP 地址：

     ```
     $ ssh heat-admin@<any_controller_node_IP_address> sudo hiera -c
     /etc/puppet/hiera.yaml octavia_api_short_bootstrap_node_name
     ```

  3. 在 undercloud 节点上，运行 **openstack overcloud update run** 命令来更新 bootstrap Controller 节点：

     ```
     $ openstack overcloud update run --nodes <bootstrap_node_name>
     ```

流程

1. Source **stackrc** 文件：

   ```
   $ source ~/stackrc
   ```

2. 运行更新命令：

   ```
   $ openstack overcloud update run --nodes Controller
   ```

3. 等待 Controller 节点更新完成。

4.6. 更新所有 COMPUTE 节点

此过程会将所有 Compute 节点更新至最新的 OpenStack Platform 13 版本。此过程涉及运行 **openstack overcloud update run** 命令，并包括 **--nodes Compute** 选项，以限制仅向 Compute 节点限制操作。

并行化注意事项

当您更新大量 Compute 节点以提高性能时，您可以在 20 个节点时并行使用 **--nodes Compute** 选项运行 **openstack overcloud update run** 命令。例如，如果您的部署中有 80 个 Compute 节点，您可以运行以下命令来并行更新 Compute 节点：

```
$ openstack overcloud update run --nodes 'Compute[0:19]' > update-compute-0-19.log 2>&1 &
$ openstack overcloud update run --nodes 'Compute[20:39]' > update-compute-20-39.log 2>&1 &
$ openstack overcloud update run --nodes 'Compute[40:59]' > update-compute-40-59.log 2>&1 &
$ openstack overcloud update run --nodes 'Compute[60:79]' > update-compute-60-79.log 2>&1 &
```

'Compute[0:19]', **'Compute[20:39]'**, **'Compute[40:59]'**, 和 **'Compute[60:79]'** 方法对节点进行分区是随机的，您没有控制每个批处理中更新节点的顺序。

要更新特定的 Compute 节点，请列出您要在以逗号分隔的批处理中更新的节点：

```
$ openstack overcloud update run --nodes <Compute0>,<Compute1>,<Compute2>,<Compute3>
```

流程

1. Source **stackrc** 文件：

```
$ source ~/stackrc
```

2. 运行更新命令：

```
$ openstack overcloud update run --nodes Compute
```

3. 等待 Compute 节点更新完成。

4.7. 更新所有 HCI COMPUTE 节点

此过程更新超融合基础架构(HCI) Compute 节点。这个过程涉及运行 **openstack overcloud update run** 命令，包括 **--nodes ComputeHCI** 选项来将操作限制为 HCI 节点。

警告

如果使用 Ceph，请参阅红帽知识库解决方案，在 OSP13/RHCS3 的次要更新期间，升级到最新的软件包 Ceph 服务，并在按照本节操作前手动重启，以避免以下 bug BZ-2022-0842。

流程

1. Source **stackrc** 文件：

```
$ source ~/stackrc
```

2. 运行更新命令：

```
$ openstack overcloud update run --nodes ComputeHCI
```

3. 等待节点更新完成。

4.8. 更新所有 CEPH STORAGE 节点

此过程更新 Ceph Storage 节点。这个过程涉及运行 **openstack overcloud update run** 命令，包括 **--nodes CephStorage** 选项，仅限制对 Ceph Storage 节点的操作。

警告

如果使用 Ceph，请参阅红帽知识库解决方案，在 OSP13/RHCS3 的次要更新期间，升级到最新的软件包 Ceph 服务，并在按照本节操作前手动重启，以避免以下 bug BZ-2022-0842。

流程

1. Source **stackrc** 文件：

```
$ source ~/stackrc
```

2. 更新组节点。
更新组中的所有节点：

```
$ openstack overcloud update run --nodes <GROUP_NAME>
```

更新组中的单个节点：

```
$ openstack overcloud update run --nodes <GROUP_NAME> [NODE_INDEX]
```

注意

如果您选择单独更新节点，请确保您更新所有节点。

组中第一个节点的索引为零(0)。例如，要更新名为 **CephStorage** 的组中的第一个节点，该命令是：

OpenStack overcloud update run --nodes CephStorage[0]

3. 等待节点更新完成。

4.9. 对更新进行最终大小

更新需要最后一步来更新 overcloud 堆栈。这样可确保堆栈资源结构与常规部署 Red Hat OpenStack Platform 13 保持一致，您可以在以后执行标准的 **openstack overcloud deploy** 功能。

流程

1. Source **stackrc** 文件：

```
$ source ~/stackrc
```

2. 运行更新最终化命令：

```
$ openstack overcloud update converge \
  --templates \
  --stack <STACK_NAME> \
  -e /home/stack/templates/overcloud_images.yaml \
  -e /home/stack/templates/updates-environment.yaml \
  -e <ENVIRONMENT FILE>
  ...
```

包含以下与您的环境相关的选项：

- 自定义配置环境文件(**-e**)
- 带有新容器镜像位置的环境文件(**-e**)。请注意，update 命令可能会显示关于使用 **--container-registry-file** 的警告。您可以忽略这个警告，因为此选项已弃用，而是对容器镜像环境文件使用 **-e**。

- 如果适用，您的自定义角色(**roles_data**)文件(**--roles-file**)
- 如果适用，您的可组合网络(**network_data**)文件(**--networks-file**)
- 如果 **overcloud** 堆栈的名称与默认名称 overcloud 不同，则必须在 update preparation 命令中包含 **--stack** 选项，并将 < **STACK_NAME&** gt; 替换为 overcloud 堆栈的名称。

3. 等待更新最终完成。

第 5 章 重新引导 OVERCLOUD

在次要 Red Hat OpenStack 版本更新后，重启您的 overcloud。重启会更新带有任何关联的内核、系统级别和容器组件更新的节点。这些更新可能会提供性能和安全性优势。

计划停机时间来执行以下重启过程。

5.1. 重新引导控制器和可组合节点

以下流程根据可组合角色重新引导控制器节点和独立节点。这不包括 Compute 节点和 Ceph Storage 节点。

流程

1. 登录您要重新引导的节点。
2. 可选：如果节点使用 Pacemaker 资源，请停止集群：

```
[heat-admin@overcloud-controller-0 ~]$ sudo pcs cluster stop
```

3. 重新引导节点：

```
[heat-admin@overcloud-controller-0 ~]$ sudo reboot
```

4. 稍等片刻，直到节点启动。
5. 检查服务。例如：
   a. 如果该节点使用 Pacemaker 服务，请检查该节点是否已重新加入集群：

```
[heat-admin@overcloud-controller-0 ~]$ sudo pcs status
```

   b. 如果该节点使用 Systemd 服务，请检查是否所有服务都已启用：

```
[heat-admin@overcloud-controller-0 ~]$ sudo systemctl status
```

   c. 对所有 Controller 和可组合节点重复这些步骤。

5.2. 重新引导 CEPH STORAGE (OSD) 集群

以下流程重启 Ceph Storage (OSD)节点集群。

流程

1. 登录 Ceph MON 或 Controller 节点，并暂时禁用 Ceph Storage 集群重新平衡：

```
$ sudo ceph osd set noout
$ sudo ceph osd set norebalance
```

2. 选择第一个 Ceph Storage 节点以重新引导并登录。
3. 重新引导节点：

```
$ sudo reboot
```

4. 稍等片刻，直到节点启动。

5. 登录 Ceph MON 或 Controller 节点，并检查集群状态：

```
$ sudo ceph -s
```

确认 **pgmap** 报告的所有 **pgs** 的状态是否都正常 (**active+clean**)。

6. 从 Ceph MON 或 Controller 节点注销，重新引导下一个 Ceph Storage 节点，再检查其状态。重复此流程，直到您已重新引导所有 Ceph 存储节点。

7. 完成之后，登录 Ceph MON 或 Controller 节点，然后重新启用集群重新平衡：

```
$ sudo ceph osd unset noout
$ sudo ceph osd unset norebalance
```

8. 执行最后的状态检查，确认集群报告 **HEALTH_OK**：

```
$ sudo ceph status
```

5.3. 重新引导 COMPUTE 节点

重新引导 Compute 节点涉及以下工作流：

- 选择一个 Compute 节点来重新引导并禁用它，使其不置备新实例。
- 将实例迁移到另一个 Compute 节点，以最小化实例停机时间。
- 重新引导空的 Compute 节点并启用它。

流程

1. 以 **stack** 用户的身份登录 undercloud。

2. 要识别您要重新引导的 Compute 节点，请列出所有 Compute 节点：

```
$ source ~/stackrc
(undercloud) $ openstack server list --name compute
```

3. 从 overcloud 中，选择一个 Compute 节点并禁用它：

```
$ source ~/overcloudrc
(overcloud) $ openstack compute service list
(overcloud) $ openstack compute service set <hostname> nova-compute --disable
```

4. 列出 Compute 节点上的所有实例：

```
(overcloud) $ openstack server list --host <hostname> --all-projects
```

5. 迁移您的实例。有关迁移策略的更多信息，请参阅在 Compute 节点之间迁移虚拟机。

6. 登录到 Compute 节点并重新引导它：

```
[heat-admin@overcloud-compute-0 ~]$ sudo reboot
```

7. 稍等片刻，直到节点启动。

8. 启用 Compute 节点：

```
$ source ~/overcloudrc
(overcloud) $ openstack compute service set <hostname> nova-compute --enable
```

9. 验证 Compute 节点是否已启用：

```
(overcloud) $ openstack compute service list
```

5.4. 重新引导计算 HCI 节点

以下流程重启计算超融合基础架构(HCI)节点。

流程

1. 登录 Ceph MON 或 Controller 节点，并暂时禁用 Ceph Storage 集群重新平衡：

```
$ sudo ceph osd set noout
$ sudo ceph osd set norebalance
```

2. 以 **stack** 用户的身份登录 undercloud。

3. 列出所有的 Compute 节点及其 UUID：

```
$ source ~/stackrc
(undercloud) $ openstack server list --name compute
```

确定您要重新引导的 Compute 节点的 UUID。

4. 在 undercloud 中，选择 Compute 节点并禁用它：

```
$ source ~/overcloudrc
(overcloud) $ openstack compute service list
(overcloud) $ openstack compute service set [hostname] nova-compute --disable
```

5. 列出 Compute 节点上的所有实例：

```
(overcloud) $ openstack server list --host [hostname] --all-projects
```

6. 使用以下命令之一迁移您的实例：

   a. 将实例迁移到您选择的特定主机：

   ```
   (overcloud) $ openstack server migrate [instance-id] --live [target-host]--wait
   ```

   b. 让 **nova-scheduler** 自动选择目标主机：

   -

```
(overcloud) $ nova live-migration [instance-id]
```

c. 一次性实时迁移所有实例：

```
$ nova host-evacuate-live [hostname]
```

注意

nova 命令可能会引发一些弃用警告，这些警告信息可以被安全忽略。

7. 等待迁移完成。
8. 确认迁移成功完成：

```
(overcloud) $ openstack server list --host [hostname] --all-projects
```

9. 继续迁移实例，直到所选 Compute 节点中不剩任何实例。
10. 登录到 Ceph MON 或 Controller 节点并检查集群状态：

```
$ sudo ceph -s
```

确认 **pgmap** 报告的所有 **pgs** 的状态是否都正常 (**active+clean**)。

11. 重新引导 Compute HCI 节点：

```
$ sudo reboot
```

12. 稍等片刻，直到节点启动。
13. 再次启用 Compute 节点：

```
$ source ~/overcloudrc
(overcloud) $ openstack compute service set [hostname] nova-compute --enable
```

14. 验证 Compute 节点是否已启用：

```
(overcloud) $ openstack compute service list
```

15. 注销节点，重新引导下一个节点，并检查其状态。重复此流程，直到您已重新引导所有 Ceph 存储节点。
16. 完成后，登录 Ceph MON 或 Controller 节点，然后再次启用集群重新平衡：

```
$ sudo ceph osd unset noout
$ sudo ceph osd unset norebalance
```

17. 执行最后的状态检查，确认集群报告 **HEALTH_OK**：

```
$ sudo ceph status
```

第 6 章 执行更新后的任务

在 Red Hat OpenStack 次版本更新后，您必须执行与更新后的任务，以确保您的环境被完全支持，并准备好将来的操作。

6.1. OCTAVIA 部署的注意事项

如果您的部署使用 Octavia 服务，则必须使用新镜像更新正在运行的负载均衡 amphora 实例。

要更新 amphora 镜像，您必须通过负载均衡器失败，然后等待负载均衡器重新获得活跃状态。当负载均衡器再次处于活动状态时，它会运行新镜像。

如需更多信息，请参阅 *使用 Octavia for Load Balancing-as-a-Service 指南中的更新运行负载均衡服务实例*。