# Red Hat Enterprise Linux 7

## 在公共云平台上部署 Red Hat Enterprise Linux 7

创建自定义 Red Hat Enterprise Linux 镜像并为公共云平台配置红帽高可用性集群

Last Updated: 2021-05-31

# Red Hat Enterprise Linux 7 在公共云平台上部署 Red Hat Enterprise Linux 7

创建自定义 Red Hat Enterprise Linux 镜像并为公共云平台配置红帽高可用性集群

## 法律通告



## 摘要

您可以将自定义 Red Hat Enterprise Linux 镜像创建并部署到各种云平台,包括 Microsoft Azure、Amazon Web Services(AWS)和 Google Cloud Platform(GCP)。 您还可以在每个云平台中创建和配置红帽高可用性集群。本文档描述了创建镜像的两个选择： Cloud Access image 和 on-demand(marketplace)镜像。它包括创建 HA 集群的步骤,包括安装所需的软件包和代理、配置隔离和安装网络资源代理。 每个云供应商都有自己的章节，介绍了创建和部署自定义镜像的信息。另外，还有一个单独章节用于为每个云供应商配置 HA 集群。

# 目录

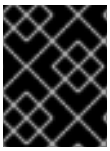
# 对红帽文档提供反馈

我们感谢您对文档提供反馈信息。请让我们了解如何改进文档。要做到这一点：

- 关于特定内容的简单评论：
  1. 请确定您使用 *Multi-page HTML* 格式查看文档。另外，确定 **Feedback** 按钮出现在文档页的右上方。
  2. 用鼠标指针高亮显示您想评论的文本部分。
  3. 点在高亮文本上弹出的 **Add Feedback**。
  4. 按照显示的步骤操作。
- 要提交更复杂的反馈，请创建一个 Bugzilla ticket：
  1. 进入 Bugzilla 网站。
  2. 在 Component 中选择 **Documentation**。
  3. 在 **Description** 中输入您要提供的信息。包括文档相关部分的链接。
  4. 点 **Submit Bug**。

# 第 1 章 在 MICROSOFT AZURE 上部署 RED HAT ENTERPRISE LINUX 7 镜像作为虚拟机

您有多个选项可在 Azure 上部署 Red Hat Enterprise Linux（RHEL）7 镜像。本章讨论选择镜像的选项,以及列出或引用主机系统和虚拟机(VM)的系统要求。本章还提供了从 ISO 镜像创建自定义虚拟机、将其上传到 Azure 以及启动 Azure 虚拟机实例的步骤。



**重要**

您可以从 ISO 镜像创建自定义虚拟机,但红帽建议您使用 Red Hat Image Builder 产品来创建自定义镜像,供特定云供应商使用。如需更多信息，请参阅镜像构建器指南。

本章在很多位置使用了 Azure 文档。如需了解更多详细信息，请参阅相关的 Azure 文档。

**注意**

有关您可以在 Azure 上安全使用的红帽产品列表，请参阅 Microsoft Azure 上的红帽产品。

**先决条件**

- 注册一个红帽客户门户网站（Red Hat Customer Portal） 帐户。
- 注册一个 Microsoft Azure 帐户。
- 在 Red Hat Cloud Access 项目中启用您的订阅。Red Hat Cloud Access 程序允许您在红帽完全支持的情况下将您的红帽订阅从物理系统或内部系统移到 Azure。

**其它资源**

- 公共云中的红帽
- Red Hat Cloud Access 参考指南
- Microsoft Azure 的常见问题解答及推荐实践

## 1.1. AZURE 上的 RED HAT ENTERPRISE LINUX 镜像选项

下表列出了镜像的不同选择并记录镜像选项的不同。

**表 1.1. 镜像选项**

| 镜像选项 | 订阅 | 示例情境 | 注意事项 |
|---|---|---|---|

| 镜像选项 | 订阅 | 示例情境 | 注意事项 |
| --- | --- | --- | --- |
| 选择部署一个 Red Hat Gold Image。 | 利用您现有的红帽订阅。 | 通过 Red Hat Cloud Access 程序 启用订阅，然后在 Azure 上选择 Red Hat Gold Image。如需了解有关 Gold Images 以及如何在 Azure 上访问它们的详细信息，请参阅 Red Hat Cloud Access 指南。 | 订阅包括了红帽产品的成本；您需要向 Microsoft 支付其他费用。<br><br>Red Hat Gold Images 被称为 "Cloud Access" 镜像，因为您使用现有的红帽订阅。红帽直接为 Cloud Access 镜像提供支持。 |
| 选择部署一个要移至 Azure 的自定义镜像。 | 利用您现有的红帽订阅。 | 通过 Red Hat Cloud Access 程序 启用订阅,上传您的自定义镜像并附加您的订阅。 | 订阅包括了红帽产品的成本；您需要向 Microsoft 支付其他费用。<br><br>移动到 Azure 的自定义镜像是 "Cloud Access" 镜像,因为您利用了您现有的红帽订阅。红帽直接为 Cloud Access 镜像提供支持。 |
| 选择部署一个包含 RHEL 的现有 Azure 镜像。 | Azure 镜像包括一个红帽产品。 | 使用 Azure 控制台创建虚拟机时选择 RHEL 镜像,或者从 Azure Marketplace 中选择虚拟机。 | 根据 pay-as-you-go 模式每小时向微软支付。这样的镜像名为 "on-demand"。 Azure 通过支持协议提供对 on-demand 镜像的支持。<br><br>红帽提供了镜像的更新。Azure 通过 Red Hat Update Infrastructure(RHUI)提供更新。 |

**注意**

您可以使用 Red Hat Image Builder 为 Azure 创建自定义镜像。如需更多信息，请参阅镜像构建器指南。

本章的以下部分包括了与 Red Hat Enterprise Linux 自定义镜像相关的信息和流程。

**其它资源**

- Microsoft Azure 上的 Red Hat Gold Images
- Red Hat Cloud Access 程序
- Azure Marketplace
- Azure Marketplace 中的账单选项

- Azure 中的 Red Hat Enterprise Linux Bring-Your-Own-Subscription Gold Images

## 1.2. 理解基础镜像

本节介绍使用预配置的基础镜像及其配置设置的信息。

### 1.2.1. 使用自定义基础镜像

要手动配置虚拟机，请以基础（启动程序）虚拟机镜像开始。创建基本虚拟机镜像后，您可以修改配置设置并添加虚拟机在云中操作的软件包。您可在上传镜像后为特定应用程序进行额外的配置更改。

要准备 RHEL 的 Hyper-V 云镜像，请参阅从 Hyper-V 管理器准备 RHEL 7 虚拟机 。

#### 其它资源

Red Hat Enterprise Linux

### 1.2.2. 所需的系统软件包

本章的步骤假设您使用运行 Red Hat Enterprise Linux 的主机系统。要成功完成这些操作，主机系统必须安装以下软件包。

**表 1.2. 系统软件包**

| 软件包 | 描述 | 命令 |
| --- | --- | --- |
| qemu-kvm | 这个软件包提供用户级别的 KVM 模拟器，并可方便主机和客户机虚拟机间的通信。 | **# yum install qemu-kvm libvirt** |
| qemu-img | 这个软件包为客户机虚拟机提供磁盘管理。qemu-img 软件包作为 qemu-kvm 软件包的依赖项安装。 | |
| libvirt | 这个软件包为服务器和主机端提供与虚拟机监控程序、主机系统和主机系统以及处理库调用、管理虚拟机和控制虚拟机监控程序的 libvirtd 守护进程交互的服务器和主机库。 | |

**表 1.3. 其他虚拟化软件包**

| 软件包 | 描述 | 命令 |
| --- | --- | --- |
| virt-install | 这个软件包提供从命令行创建虚拟机的 **virt-install** 命令。 | **# yum install virt-install libvirt-python virt-manager virt-install libvirt-client** |
| libvirt-python | 这个软件包包含一个模块，它允许使用 Python 编程语言编写的应用程序使用 libvirt API 提供的接口。 | |

| 软件包 | 描述 | 命令 |
| --- | --- | --- |
| virt-manager | 这个软件包提供 virt-manager 工具,也称 Virtual Machine Manager(VMM)。VMM 是一个图形化工具用于管理虚拟机。它使用 libvirt-client 库作为管理 API。 | |
| libvirt-client | 这个软件包为访问 libvirt 服务器提供客户端 API 和库。libvirt-client 软件包包含 **virsh** 命令行工具,用于从命令行或特殊虚拟化 shell 管理和控制虚拟机和虚拟机监控程序。 | |

**其它资源**

- 手动安装虚拟化软件包

### 1.2.3. Azure VM 配置设置

Azure 虚拟机必须具有以下配置设置。其中一些设置会在初始创建虚拟机期间启用。为 Azure 置备虚拟机镜像时会设置其他设置。在您进行操作时请注意这些设置 ; 如果您遇到任何错误，请参考它们。

**表 1.4. 虚拟机配置设置**

| 设置 | 建议 |
| --- | --- |
| ssh | **ssh** 必须启用才能提供对 Azure 虚拟机的远程访问权限。 |
| dhcp | 应该为 dhcp 配置主虚拟适配器（仅限 IPv4）。 |
| swap 空间 | 不要创建一个专用的交换文件或者交换分区。您可以使用 Windows Azure Linux Agent(WALinuxAgent)配置 swap 空间。 |
| NIC | 为主虚拟网络适配器选择 **virtio**。 |
| encryption | 对于自定义镜像,运行 RHEL 7.5 及更新的版本,请使用 Network Bound Disk Encryption(NBDE)在 Azure 上进行完整磁盘加密。只有 RHEL 7.5 及更新的版本支持 NBDE。 |

### 1.2.4. 从 ISO 镜像创建基础镜像

以下流程列出了创建自定义 ISO 镜像的步骤和初始配置要求。配置了镜像后，您可以使用镜像作为模板来创建额外的虚拟机实例。

**流程**

1. 从红帽客户门户网站下载最新的 Red Hat Enterprise Linux 7 Binary DVD ISO 镜像。
2. 确保已为虚拟化启用主机机器。有关相关信息和步骤，请参阅虚拟化入门指南。
3. 创建并启动基本 Red Hat Enterprise Linux 虚拟机。具体步骤请查看使用虚拟化命令行界面入门。
   a. 如果使用命令行创建虚拟机，请确保将默认内存和 CPU 设置为您所需的容量。将您的虚拟网络接口设置为 **virtio**。
   下面是一个基本的命令行示例。

   ```
   virt-install --name isotest --memory 2048 --vcpus 2 --disk size=8,bus=virtio --location rhel-7.0-x86_64-dvd.iso --os-variant=rhel7.0
   ```

   b. 如果您使用 VMM 应用程序创建虚拟机,请按照 虚拟机管理器入门 中的步骤进行操作：
      - 不要选择 **Immediately Start VM**。
      - 将 **Memory** 和 **Storage Size** 设置为您需要的值。
      - 在开始安装前，请确保将 **Virtual Network Interface Settings** 中的 **Model** 更改为 **virtio**，并将您的 **vCPU** 更改为您想要的虚拟机容量设置。
4. 查看以下额外的安装选择和修改。
   - 选择带有 **standard RHEL** 选项的 **Minimal Install**。
   - 对于 **Installation Destination**，选择 **Custom Storage Configuration**。使用以下配置信息进行选择。
     - 确保 **/boot** 验证至少 500 MB。
     - 对于文件系统，**boot** 和 **root** 分区都要使用 xfs、ext4 或 ext3。
     - 删除 swap 空间。swap 空间由 WALinuxAgent 在 Azure 中的物理 blade 服务器中配置。
   - 在 **Installation Summary** 屏幕中，选择 **Network and Host Name**。将 **Ethernet** 切换到 **On**。
5. 安装开始：
   - 创建 **root** 密码。
   - 创建管理用户帐户。
6. 安装完成后，重启虚拟机并登录到 root 帐户。
7. 登录后，您可以配置镜像 **root**。

## 1.3. 为 MICROSOFT AZURE 配置基础镜像

基础镜像需要更改配置，才能作为 Azure 中的 RHEL 7 虚拟机镜像。以下小节提供 Azure 所需的其他配置更改。

### 1.3.1. 安装 Hyper-V 设备驱动程序

Microsoft 为 Hyper-V 软件包提供网络和存储设备驱动程序作为其 Linux 集成服务(LIS)的一部分。在将 Hyper-V 设备驱动程序置备为 Azure 虚拟机之前，您可能需要在虚拟机镜像上安装 Hyper-V 设备驱动程序。使用 **lsinitrd | grep hv** 命令验证是否安装了驱动程序。

**流程**

1. 输入以下 **grep** 命令来确定是否安装了所需的 Hyper-V 设备驱动程序。

```
# lsinitrd | grep hv
```

在以下示例中安装了所有必需的驱动程序。

```
# lsinitrd | grep hv
drwxr-xr-x   2 root     root            0 Aug 12 14:21 usr/lib/modules/3.10.0-
932.el7.x86_64/kernel/drivers/hv
-rw-r--r--   1 root     root        31272 Aug 11 08:45 usr/lib/modules/3.10.0-
932.el7.x86_64/kernel/drivers/hv/hv_vmbus.ko.xz
-rw-r--r--   1 root     root        25132 Aug 11 08:46 usr/lib/modules/3.10.0-
932.el7.x86_64/kernel/drivers/net/hyperv/hv_netvsc.ko.xz
-rw-r--r--   1 root     root         9796 Aug 11 08:45 usr/lib/modules/3.10.0-
932.el7.x86_64/kernel/drivers/scsi/hv_storvsc.ko.xz
```

如果没有安装所有驱动程序，请完成剩余的步骤。

**注意**

环境中可能会存在 **hv_vmbus** 驱动程序。即使存在这个驱动程序，在虚拟机中完成以下步骤。

2. 在 **/etc/hv.conf.d** 中创建名为 **hv.conf** 的文件。

3. 在 **hv.conf** 文件中添加以下驱动程序参数。

```
add_drivers+=" hv_vmbus "
add_drivers+=" hv_netvsc "
add_drivers+=" hv_storvsc "
```

**注意**

请注意引号之前和之后的空格，如 **add_drivers+=" hv_vmbus "**。这样可确保在环境中存在其他 Hyper-V 驱动程序时载入唯一的驱动程序。

4. 重新生成 **initramfs** 镜像。

```
# dracut -f -v --regenerate-all
```

**验证步骤**

1. 重启机器。

2. 运行 **lsinitrd | grep hv** 命令验证是否安装了驱动程序。

### 1.3.2. 进行额外的配置更改

虚拟机需要进行进一步的配置更改才能在 Azure 中操作。执行以下步骤进行额外的更改。

**流程**

1. 如有必要，启动虚拟机。
2. 注册虚拟机并启用 Red Hat Enterprise Linux 7 软件仓库。

```
# subscription-manager register --auto-attach
```

**停止和删除 cloud-init（如果存在）**

1. 停止 **cloud-init** 服务。

```
# systemctl stop cloud-init
```

2. 删除 **cloud-init** 软件。

```
# yum remove cloud-init
```

**完成其他虚拟机更改**

1. 编辑 **/etc/ssh/sshd_config** 文件并启用密码验证。

```
PasswordAuthentication yes
```

2. 设置通用主机名。

```
# hostnamectl set-hostname localhost.localdomain
```

3. 编辑（或创建） **/etc/sysconfig/network-scripts/ifcfg-eth0** 文件。仅使用以下列出的参数。

**注意**

RHEL 7 DVD ISO 镜像中没有 **ifcfg-eth0** 文件，它必须被创建。

```
DEVICE="eth0"
ONBOOT="yes"
BOOTPROTO="dhcp"
TYPE="Ethernet"
USERCTL="yes"
PEERDNS="yes"
IPV6INIT="no"
```

4. 删除所有持久的网络设备规则（如果存在）。

```
# rm -f /etc/udev/rules.d/70-persistent-net.rules
# rm -f /etc/udev/rules.d/75-persistent-net-generator.rules
# rm -f /etc/udev/rules.d/80-net-name-slot-rules
```

5. 将 **ssh** 设置为自动启动。

```
# systemctl enable sshd
# systemctl is-enabled sshd
```

6. 修改内核引导参数。

   a. 在 **/etc/default/grub** 文件中的 **GRUB_CMDLINE_LINUX** 行的开头添加 **crashkernel=256M**。如果 **crashkernel=auto** 存在，请将其改为 **crashkernel=256M**。

   b. 在 **GRUB_CMDLINE_LINUX** 行末尾添加以下行（如果不存在）。

```
earlyprintk=ttyS0
console=ttyS0
rootdelay=300
```

   c. 删除以下选项（如果存在）。

```
rhgb
quiet
```

7. 重新生成 **grub.cfg** 文件。

```
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

8. 安装并启用 Windows Azure Linux Agent(WALinuxAgent)。

```
# yum install WALinuxAgent -y
# systemctl enable waagent
```

**注意**

如果您看到 **No package WALinuxAgent available** 错误信息，安装 **rhel-7-server-extras-rpms** 仓库。在再次尝试安装前运行 **# subscription-manager repos --enable=rhel-7-server-extras-rpms** 命令。

9. 编辑 **/etc/waagent.conf** 文件中的以下行，为置备的虚拟机配置 swap 空间。为您置备的虚拟机设置 swap 空间。

```
Provisioning.DeleteRootPassword=n
ResourceDisk.Filesystem=ext4
ResourceDisk.EnableSwap=y
ResourceDisk.SwapSizeMB=2048
```

**准备置备**

1. 从 Red Hat Subscription Manager 取消注册虚拟机。

```
# subscription-manager unregister
```

2. 通过清理现有置备详情来准备 Azure 置备。Azure 在 Azure 中重新置备虚拟机。这个命令会生成数据丢失警告信息，这是预期的。

```
# waagent -force -deprovision
```

3. 清理 shell 历史记录并关闭虚拟机。

```
# export HISTSIZE=0
# poweroff
```

## 1.4. 将镜像转换为固定 VHD 格式

所有 Microsoft Azure VM 镜像都必须采用固定的 **VHD** 格式。镜像必须在将镜像转换为 VHD 之前被对齐到 1 MB 边界。本节论述了如何将镜像从 **qcow2** 转换为固定的 **VHD** 格式，并在需要时匹配镜像。转换镜像后，您可以将其上传到 Azure。

**流程**

1. 将镜像从 **qcow2** 转换为 **raw** 格式。

```
$ qemu-img convert -f qcow2 -O raw <image-name>.qcow2 <image-name>.raw
```

2. 使用以下内容创建一个 shell 脚本。

```
#!/bin/bash
MB=$((1024 * 1024))
size=$(qemu-img info -f raw --output json "$1" | gawk 'match($0, /"virtual-size": ([0-9]+),/, val)
{print val[1]}')
rounded_size=$((($size/$MB + 1) * $MB))
if [ $(($size % $MB)) -eq  0 ]
then
 echo "Your image is already aligned. You do not need to resize."
 exit 1
fi
echo "rounded size = $rounded_size"
export rounded_size
```

3. 运行脚本。这个示例使用名称 **align.sh**。

```
$ sh align.sh <image-xxx>.raw
```

- 如需显示 *"Your image is already aligned.You do not need to resize."*，执行以下步骤。
- 如果显示了一个值，代表您的镜像没有被对齐。在继续执行下一步前，按照 **Aligning the image** 部分中的步骤重新定义镜像大小。

4. 使用以下命令将文件转换为固定的 **VHD** 格式。
**示例使用 qemu-img 版本 2.12.0。**

```
$ qemu-img convert -f raw -o subformat=fixed,force_size -O vpc <image-xxx>.raw
<image.xxx>.vhd
```

转换后，**VHD** 文件就可以上传到 Azure。

**对齐镜像**

只有在 **raw** 文件没有被对齐时才执行以下步骤。

1. 使用在运行验证脚本时显示的舍入值重新定义 **raw** 文件大小。

```
$ qemu-img resize -f raw <image-xxx>.raw <rounded-value>
```

2. 将 **raw** 镜像文件转换为 **VHD** 格式。
**示例使用 qemu-img 版本 2.12.0。**

```
$ qemu-img convert -f raw -o subformat=fixed,force_size -O vpc <image-xxx>.raw
<image.xxx>.vhd
```

转换后，**VHD** 文件就可以上传到 Azure。

## 1.5. 安装 AZURE CLI

完成以下步骤,在主机上安装 Azure 命令行界面(Azure CLI 2.1)。Azure CLI 2.1 是基于 Python 的实用程序,可在 Azure 中创建和管理虚拟机。

**先决条件**

- 在使用 Azure CLI 之前，您需要具有 Microsoft Azure 帐户。
- Azure CLI 安装需要 Python 3.x。

**流程**

1. 导入 Microsoft 软件仓库密钥。

```
$ sudo rpm --import https://packages.microsoft.com/keys/microsoft.asc
```

2. 创建本地 Azure CLI 存储库条目。

```
$ sudo sh -c 'echo -e "[azure-cli]\nname=Azure
CLI\nbaseurl=https://packages.microsoft.com/yumrepos/azure-
cli\nenabled=1\ngpgcheck=1\ngpgkey=https://packages.microsoft.com/keys/microsoft.asc" >
/etc/yum.repos.d/azure-cli.repo'
```

3. 更新 **yum** 软件包索引。

```
$ yum check-update
```

4. 检查您的 Python 版本（**python --version**），如果需要安装 Python 3.x。

```
$ sudo yum install python3
```

5. 安装 Azure CLI。

```
$ sudo yum install -y azure-cli
```

6. 运行 Azure CLI。

```
$ az
```

**其它资源**

- Azure CLI
- Azure CLI 命令参考

## 1.6. 在 AZURE 中创建资源

完成以下步骤，在上传 **VHD** 文件并创建 Azure 镜像前创建所需的 Azure 资源。

**流程**

1. 输入以下命令使用 Azure 验证您的系统并登录。

```
$ az login
```

**注意**

如果在您的环境中有浏览器，则 CLI 会打开浏览器到 Azure 登录页面。如需更多信息和选项，请参阅使用 Azure CLI 登录。

2. 在 Azure 区域中创建资源组。

```
$ az group create --name <resource-group> --location <azure-region>
```

例如：

```
$ az group create --name azrhelclirsgrp --location southcentralus
{
  "id": "/subscriptions//resourceGroups/azrhelclirsgrp",
  "location": "southcentralus",
  "managedBy": null,
  "name": "azrhelclirsgrp",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null
}
```

3. 创建存储帐户。有关有效 SKU 值的更多信息，请参阅 SKU Types。

```
$ az storage account create -l <azure-region> -n <storage-account-name> -g <resource-group> --sku <sku_type>
```

例如：

```
$ az storage account create -l southcentralus -n azrhelclistact -g azrhelclirsgrp --sku Standard_LRS
{
  "accessTier": null,
```

```
  "creationTime": "2017-04-05T19:10:29.855470+00:00",
  "customDomain": null,
  "encryption": null,
  "id":
"/subscriptions//resourceGroups/azrhelclirsgrp/providers/Microsoft.Storage/storageAccounts/azr
helclistact",
  "kind": "StorageV2",
  "lastGeoFailoverTime": null,
  "location": "southcentralus",
  "name": "azrhelclistact",
  "primaryEndpoints": {
    "blob": "https://azrhelclistact.blob.core.windows.net/",
    "file": "https://azrhelclistact.file.core.windows.net/",
    "queue": "https://azrhelclistact.queue.core.windows.net/",
    "table": "https://azrhelclistact.table.core.windows.net/"
},
"primaryLocation": "southcentralus",
"provisioningState": "Succeeded",
"resourceGroup": "azrhelclirsgrp",
"secondaryEndpoints": null,
"secondaryLocation": null,
"sku": {
  "name": "Standard_LRS",
  "tier": "Standard"
},
"statusOfPrimary": "available",
"statusOfSecondary": null,
"tags": {},
  "type": "Microsoft.Storage/storageAccounts"
}
```

4. 获取存储帐户连接字符串。

```
$ az storage account show-connection-string -n <storage-account-name> -g <resource-
group>
```

例如：

```
[clouduser@localhost]$ az storage account show-connection-string -n azrhelclistact -g
azrhelclirsgrp
{
  "connectionString":
"DefaultEndpointsProtocol=https;EndpointSuffix=core.windows.net;AccountName=azrhelclistact
AccountKey=NreGk...=="
}
```

5. 通过复制连接字符串并将其粘贴到以下命令来导出连接字符串。这个字符串将您的系统连接到存储帐户。

```
$ export AZURE_STORAGE_CONNECTION_STRING="<storage-connection-string>"
```

例如：

```
[clouduser@localhost]$ export
AZURE_STORAGE_CONNECTION_STRING="DefaultEndpointsProtocol=https;EndpointSuffi
x=core.windows.net;AccountName=azrhelclistact;AccountKey=NreGk...=="
```

6. 创建存储容器。

```
$ az storage container create -n <container-name>
```

例如：

```
[clouduser@localhost]$ az storage container create -n azrhelclistcont
{
  "created": true
}
```

7. 创建虚拟网络。

```
$ az network vnet create -g <resource group> --name <vnet-name> --subnet-name <subnet-
name>
```

例如：

```
[clouduser@localhost]$ az network vnet create --resource-group azrhelclirsgrp --name
azrhelclivnet1 --subnet-name azrhelclisubnet1
{
  "newVNet": {
    "addressSpace": {
      "addressPrefixes": [
      "10.0.0.0/16"
      ]
  },
  "dhcpOptions": {
    "dnsServers": []
  },
  "etag": "W/\"\"",
  "id":
"/subscriptions//resourceGroups/azrhelclirsgrp/providers/Microsoft.Network/virtualNetworks/azr
helclivnet1",
  "location": "southcentralus",
  "name": "azrhelclivnet1",
  "provisioningState": "Succeeded",
  "resourceGroup": "azrhelclirsgrp",
  "resourceGuid": "0f25efee-e2a6-4abe-a4e9-817061ee1e79",
  "subnets": [
    {
      "addressPrefix": "10.0.0.0/24",
      "etag": "W/\"\"",
      "id":
"/subscriptions//resourceGroups/azrhelclirsgrp/providers/Microsoft.Network/virtualNetworks/azr
helclivnet1/subnets/azrhelclisubnet1",
      "ipConfigurations": null,
      "name": "azrhelclisubnet1",
      "networkSecurityGroup": null,
      "provisioningState": "Succeeded",
```

```
      "resourceGroup": "azrhelclirsgrp",
      "resourceNavigationLinks": null,
      "routeTable": null
    }
  ],
  "tags": {},
  "type": "Microsoft.Network/virtualNetworks",
  "virtualNetworkPeerings": null
  }
}
```

**其它资源**

- Azure geographies
- Sign in with Azure CLI
- Azure Managed Disks 概述
- SKU 类型

## 1.7. 上传并创建 AZURE 镜像

完成以下步骤，将 **VHD** 文件上传到容器并创建 Azure 自定义镜像。

**注意**

系统重启后，导出的存储连接字符串不会保留。如果以下步骤中的任何命令失败，请再次导出连接字符串。

**流程**

1. 将 **VHD** 文件上传到存储容器，可能需要几分钟时间。要获取存储容器列表，请输入 **az storage container list** 命令。

```
$ az storage blob upload --account-name <storage-account-name> --container-name
<container-name> --type page --file <path-to-vhd> --name <image-name>.vhd
```

例如：

```
$ az storage blob upload --account-name azrhelclistact --container-name azrhelclistcont --
type page --file rhel-image-7.vhd --name rhel-image-7.vhd
Percent complete: %100.0
```

2. 获取上传的 **VHD** 文件的 URL，以便在以下步骤中使用。

```
$ az storage blob url -c <container-name> -n <image-name>.vhd
```

例如：

```
$ az storage blob url -c azrhelclistcont -n rhel-image-7.vhd
"https://azrhelclistact.blob.core.windows.net/azrhelclistcont/rhel-image-7.vhd"
```

3. 创建 Azure 自定义镜像。

```
$ az image create -n <image-name> -g <resource-group> -l <azure-region> --source <URL>
--os-type linux
```

**注意**

虚拟机的默认 hypervisor 系列为 V1。您可以通过包括选项 **--hyper-v-generation V2** 来将 hypervisor 系列指定为 V2。第二代虚拟机使用基于 UEFI 的引导架构。如需有关生成 2 代虚拟机的信息，请参阅 Azure 生成 2 个虚拟机的支持。

该命令可能会返回错误 *"Only blobs formatted as VHDs can be imported ."* 这个错误可能意味着，在镜像转换为 **VHD** 前，镜像没有与最接近的 1 MB 边界一致。

例如：

```
$ az image create -n rhel7 -g azrhelclirsgrp2 -l southcentralus --source
https://azrhelclistact.blob.core.windows.net/azrhelclistcont/rhel-image-7.vhd --os-type linux
```

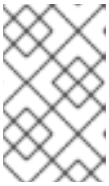
## 1.8. 在 AZURE 中创建并启动虚拟机

以下步骤提供从镜像创建 managed-disk Azure 虚拟机的最低命令选项。如需了解更多选项，请参阅 az vm create。

**流程**

1. 输入以下命令来创建虚拟机。

```
$ az vm create -g <resource-group> -l <azure-region> -n <vm-name> --vnet-name <vnet-
name> --subnet <subnet-name> --size Standard_A2 --os-disk-name <simple-name> --
admin-username <administrator-name> --generate-ssh-keys --image <path-to-image>
```



**注意**

选项 **--generate-ssh-keys** 会创建一个私有/公钥对。私钥和公钥文件会在您的系统 **~/.ssh** 中创建。公钥添加到虚拟机上由 **--admin-username** 选项指定的用户的 **authorized_keys** 文件中。如需更多信息，请参阅其他验证方法。

例如：

```
[clouduser@localhost]$ az vm create -g azrhelclirsgrp2 -l southcentralus -n rhel-azure-vm-1 -
-vnet-name azrhelclivnet1 --subnet azrhelclisubnet1  --size Standard_A2 --os-disk-name vm-
1-osdisk --admin-username clouduser --generate-ssh-keys --image rhel7
{
  "fqdns": "",
  "id":
"/subscriptions//resourceGroups/azrhelclirsgrp/providers/Microsoft.Compute/virtualMachines/rhe
l-azure-vm-1",
  "location": "southcentralus",
  "macAddress": "",
  "powerState": "VM running",
```

```
"privateIpAddress": "10.0.0.4",
"publicIpAddress": "<public-IP-address>",
"resourceGroup": "azrhelclirsgrp2"
```

2. 启动 SSH 会话并登录到虚拟机。

```
[clouduser@localhost]$ ssh  -i /home/clouduser/.ssh/id_rsa clouduser@<public-IP-address>.
The authenticity of host,  '<public-IP-address>' can't be established.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '<public-IP-address>' (ECDSA) to the list of known hosts.
```

如果您看到用户提示，则代表成功部署了 Azure 虚拟机。

现在，您可以进入 Azure 门户，检查审计日志和资源属性。您可以在此门户中直接管理虚拟机。如果要管理多个虚拟机，您应该使用 Azure CLI。Azure CLI 为您在 Azure 中的资源提供了一个强大的接口。在 CLI 中输入 **az --help** 命令，或者参阅 Azure CLI 命令来了解您在 Microsoft Azure 中用来管理虚拟机的更多信息。

## 1.9. 其他验证方法

虽然使用 Azure 生成的密钥对不是必须的，但为了提高安全性，推荐使用。以下示例演示了使用两个 SSH 验证方法。

**示例 1:** 这些命令选项在不生成公钥文件的情况下置备新虚拟机。它们允许使用密码进行 SSH 验证。

```
$ az vm create -g <resource-group> -l <azure-region> -n <vm-name> --vnet-name <vnet-name> --subnet <subnet-name> --size Standard_A2 --os-disk-name <simple-name> --authentication-type password --admin-username <administrator-name> --admin-password <ssh-password> --image <path-to-image>
```

```
$ ssh <admin-username>@<public-ip-address>
```

**示例 2:** 这些命令选项置备一个新的 Azure 虚拟机,并允许使用现有公钥文件进行 SSH 验证。

```
$ az vm create -g <resource-group> -l <azure-region> -n <vm-name> --vnet-name <vnet-name> --subnet <subnet-name> --size Standard_A2 --os-disk-name <simple-name> --admin-username <administrator-name> --ssh-key-value <path-to-existing-ssh-key> --image <path-to-image>
```

```
$ ssh -i <path-to-existing-ssh-key> <admin-username>@<public-ip-address>
```

## 1.10. 附加红帽订阅

完成以下步骤以附加您之前通过 Red Hat Cloud Access 程序启用的订阅。

### 先决条件

您必须已启用您的订阅。

### 流程

1. 注册您的系统。

```
subscription-manager register --auto-attach
```

2. 附加您的订阅。

   - 您可以使用激活码来附加订阅。请参阅创建红帽客户门户网站激活码。
   - 或者,您可以使用订阅池（池 ID）的 ID 手动附加订阅。请参阅通过 command Line 来附加和删除订阅。

**其它资源**

- 创建红帽客户门户网站激活码
- 通过命令行附加和删除订阅
- 使用并配置 Red Hat Subscription Manager

# 第 2 章 在 MICROSOFT AZURE 上配置红帽高可用性集群

红帽支持 Red Hat Enterprise Linux(RHEL)7.4 及更新的版本中的高可用性(HA)。本章包含使用虚拟机(VM)实例作为集群节点在 Microsoft Azure 上配置 Red Hat HA 集群的信息和步骤。本章中的流程假设您要为 Azure 创建自定义镜像。您有多个选项来获取用于集群的 RHEL 7 镜像。如需有关 Azure 镜像选项的更多信息,请参阅 Azure 的 Red Hat Enterprise Linux Image Options。

本章包含为 Azure 设置环境的先决条件。设置环境后,您可以创建并配置 Azure VM 实例。

本章还包含与创建 HA 集群相关的流程,该集群将单个虚拟机节点转换为 Azure 上的 HA 节点集群。这包括在每个集群节点上安装高可用性软件包和代理、配置隔离和安装 Azure 网络资源代理的步骤。

本章在很多位置使用了 Microsoft Azure 文档。如需了解更多信息，请参阅引用的 Azure 文档。

**先决条件**

- 您需要安装 Azure 命令行界面（CLI）。如需更多信息，请参阅安装 Azure CLI。
- 在 Red Hat Cloud Access 程序中启用您的订阅 。Red Hat Cloud Access 程序允许您在红帽完全支持的情况下将您的红帽订阅从物理系统或内部系统移到 Azure。

**其它资源**

- RHEL 高可用性集群的支持政策 - 作为集群成员的 Microsoft Azure 虚拟机
- 高可用性附加组件概述

## 2.1. 在 AZURE 中创建资源

完成以下步骤以创建可用性集。您需要这些资源才能完成本章中的后续任务。

**流程**

- 创建可用性集。所有集群节点都必须处于相同的可用性集。

```
$ az vm availability-set create --name _MyAvailabilitySet_ --resource-group
_MyResourceGroup_
```

例如：

```
[clouduser@localhost]$ az vm availability-set create --name rhelha-avset1 --resource-group
azrhelclirsgrp
{
  "additionalProperties": {},
    "id":
"/subscriptions/.../resourceGroups/azrhelclirsgrp/providers/Microsoft.Compute/availabilitySets/rh
elha-avset1",
    "location": "southcentralus",
    "name": “rhelha-avset1",
    "platformFaultDomainCount": 2,
    "platformUpdateDomainCount": 5,

...omitted
```

**其它资源**

- Sign in with Azure CLI
- SKU 类型
- Azure Managed Disks 概述

## 2.2. 创建 AZURE ACTIVE DIRECTORY 应用程序

完成以下步骤以创建 Azure Active Directory(AD)应用程序。Azure AD Application Views 和自动访问集群中所有节点的 HA 操作。

**先决条件**

您需要安装 Azure 命令行界面(CLI)。

**流程**

1. 确保您是 Microsoft Azure 订阅的管理员或所有者。您需要此授权来创建 Azure AD 应用程序。
2. 登录到您的 Azure 帐户。

```
$ az login
```

3. 输入以下命令来创建 Azure AD 应用程序。要使用您自己的密码，在该命令中添加 **--password** 选项。确保您创建一个强大密码。

```
$ az ad sp create-for-rbac --name _FencingApplicationName_ --role owner --scopes
"/subscriptions/_SubscriptionID_/resourceGroups/_MyResourseGroup_"
```

例如：

```
[clouduser@localhost ~] $ az ad sp create-for-rbac --name FencingApp --role owner --
scopes "/subscriptions/2586c64b-xxxxxx-xxxxxxx-xxxxxxx/resourceGroups/azrhelclirsgrp"
Retrying role assignment creation: 1/36
Retrying role assignment creation: 2/36
Retrying role assignment creation: 3/36
{
  "appId": "1a3dfe06-df55-42ad-937b-326d1c211739",
  "displayName": "FencingApp",
  "name": "http://FencingApp",
  "password": "43a603f0-64bb-482e-800d-402efe5f3d47",
  "tenant": "77ecefb6-xxxxxxxxxx-xxxxxxx-757a69cb9485"
}
```

4. 在继续操作前，保存以下信息。您需要这些信息来设置隔离代理。
   - Azure AD 应用 ID
   - Azure AD 应用程序密码
   - 租户 ID
   - Microsoft Azure Subscription ID

**其它资源**

查看用户对 Azure 资源的访问

## 2.3. 配置 HA 服务

在所有节点上完成以下步骤。

**流程**

1. 用户 **hacluster** 在上一步中的 **pcs** 和 **pacemaker** 安装中创建。在所有集群节点上为 **hacluster** 创建密码。所有节点都使用相同的密码。

```
# passwd hacluster
```

2. 如果启用了 **firewalld.service**，在 RHEL 防火墙中添加 **high availability** 服务。

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

3. 启动 **pcs** 服务并在引导时启用它。

```
# systemctl enable pcsd.service --now
```

**验证步骤**

- 确定 **pcs** 服务正在运行。

```
# systemctl is-active pcsd.service
```

## 2.4. 安装 RED HAT HA 软件包和代理

在所有节点上完成以下步骤。

**流程**

1. 在红帽注册虚拟机。

```
$ sudo -i
# subscription-manager register --auto-attach
```

2. 禁用所有软件仓库。

```
# subscription-manager repos --disable=*
```

3. 启用 RHEL 7 Server 和 RHEL 7 Server HA 软件仓库。

```
# subscription-manager repos --enable=rhel-7-server-rpms
# subscription-manager repos --enable=rhel-ha-for-rhel-7-server-rpms
```

4. 更新所有软件包。

```
# yum update -y
```

5. 如果内核被更新，则需要重启。

```
# reboot
```

6. 安装 **pcs**、**pacemaker**、**fence agent resource agent** 和 **nmap-ncat**。

```
# yum install -y pcs pacemaker fence-agents-azure-arm resource-agents nmap-ncat
```

## 2.5. 创建集群

完成以下步骤以创建节点集群。

**流程**

1. 在其中一个节点上，输入以下命令验证 pcs 用户 **hacluster**。指定集群中的每个节点的名称。

```
# pcs host auth  _hostname1_ _hostname2_ _hostname3_
```

例如：

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

2. 创建集群。

```
# pcs cluster setup --name _hostname1_ _hostname2_ _hostname3_
```

例如：

```
[root@node01 clouduser]# pcs cluster setup --name newcluster node01 node02 node03

...omitted

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
```

**验证步骤**

1. 启用集群。

```
# pcs cluster enable --all
```

2. 启动集群。

```
# pcs cluster start --all
```

例如：

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

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

## 2.6. 创建隔离设备

完成以下步骤，从集群中的任何节点配置隔离。

**流程**

1. 找到可隔离的可用实例。

```
# fence_azure_arm -l [appid] -p [authkey] --resourceGroup=[name] --subscriptionId=[name] -
-tenantId=[name] -o list
```

例如：

```
[root@node1 ~]# fence_azure_arm -l XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX -p
XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX --resourceGroup=hacluster-rg --
subscriptionId=XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX --tenantId=XXXXXXXX-
XXXX-XXXX-XXXX-XXXXXXXXXXXX -o list
node01-vm,
node02-vm,
node03-vm,
```

2. 创建隔离设备。使用 **pcmk_host_map** 命令将 RHEL 主机名映射到实例 ID。

```
# pcs stonith create _clusterfence_ fence_azure_arm login=_AD-Application-ID_
passwd=_AD-passwd_ pcmk_host_map="_pcmk-host-map_ resourcegroup=
_myresourcegroup_ tenantid=_tenantid_ subscriptionid=_subscriptionid_
```

**验证步骤**

1. 测试其他其中一个节点的隔离代理。

```
# pcs stonith fence _azurenodename_
```

例如：

```
[root@node01 ~]# pcs stonith fence fenceazure
 Resource: fenceazure (class=stonith type=fence_azure_arm)
  Attributes: login=XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX passwd=XXXXXXXX-
XXXX-XXXX-XXXX-XXXXXXXXXXXX pcmk_host_map=nodea:nodea-vm;nodeb:nodeb-
vm;nodec:nodec-vm pcmk_reboot_retries=4 pcmk_reboot_timeout=480 power_timeout=240
resourceGroup=rg subscriptionId=XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
tenantId=XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
  Operations: monitor interval=60s (fenceazure-monitor-interval-60s)
[root@node01 ~]# pcs stonith
 fenceazure     (stonith:fence_azure_arm):      Started nodea
```

2. 检查状态以验证节点已启动。

```
# watch pcs status
```

例如：

```
[root@node01 ~]# watch pcs status
 fenceazure     (stonith:fence_azure_arm):      Started nodea
```

**其它资源**

- 在 Red Hat High Availability Cluster 中进行隔离
- High Availability Add-On 管理

## 2.7. 创建 AZURE 内部负载均衡器

Azure 内部负载均衡器会删除对健康探测请求没有做出响应的集群节点。

执行以下步骤来创建 Azure 内部负载均衡器。每个步骤都引用特定的 Microsoft 流程,并包括为 HA 自定义负载均衡器的设置。

**先决条件**

访问 Azure 控制面板

**流程**

1. 创建基本负载均衡器。为 IP 地址分配类型选择 **Internal load balancer**、**Basic SKU** 和 **Dynamic**。
2. 创建后端地址池。将后端池与在 HA 中创建 Azure 资源时创建的可用性集关联。不要设置任何目标网络 IP 配置。
3. 创建健康探测。对于健康探测，选择 **TCP** 并输入端口 **61000**。您可以使用不会影响到另一个服务的 TCP 端口号。对于某些 HA 产品应用程序,如 SAP HANA 和 SQL Server,您可能需要与 Microsoft 合作来识别要使用的正确端口。
4. 创建负载均衡器规则。要创建负载均衡规则，请使用预先填充的默认值。确保将 **Floating IP (direct server return)** 设置为 **Enabled**。

## 2.8. 配置 AZURE 负载均衡器资源代理

创建健康探测后，您必须配置 **load balancer** 资源代理。此资源代理运行一个服务，它回答来自 Azure 负载均衡器的健康探测请求，并删除不回答请求的集群节点。

**流程**

1. 输入 **Azure id** 命令来查看 Azure 负载均衡器资源代理描述。这显示了这个代理的选项和默认操作。

```
# pcs resource describe _azure-id_
```

2. 创建用于管理节点上 IP 的 **Ipaddr2** 资源。

```
# pcs resource create _resource-id_ IPaddr2 ip=_virtual/floating-ip_
cidr_netmask=_virtual/floating-mask_ --group _group-id_ nic=_network-interface_ op monitor
interval=30s
```

例如：

```
[root@node01 ~]# pcs resource create ClusterIP ocf:heartbeat:IPaddr2 ip=172.16.66.99
cidr_netmask=24 --group CloudIP nic=eth0 op monitor interval=30s
```

3. 配置 **load balancer** 资源代理。

```
# pcs resource create _resource-loadbalancer-name_ azure-lb port=_port-number_ --group
_cluster-resources-group_
```

**验证步骤**

- 运行 **pcs status** 命令查看结果。

```
[root@node01 clouduser]# pcs status
```

例如：

```
[root@node01 ~]# pcs status
Cluster name: hacluster

WARNINGS:
No stonith devices and stonith-enabled is not false

Stack: corosync
Current DC: nodeb (version 1.1.22-1.el7-63d2d79005) - partition with quorum
Last updated: Wed Sep  9 16:47:07 2020
Last change: Wed Sep  9 16:44:32 2020 by hacluster via crmd on nodeb

3 nodes configured
0 resource instances configured

Online: [ node01 node02 node03 ]

No resources
```

```
Daemon Status:
  corosync: active/enabled
  pacemaker: active/enabled
  pcsd: active/enabled
```

**其它资源**

- Cluster Operation

## 2.9. 配置共享块存储

本节介绍了使用 Microsoft Azure Shared Disks 为 Red Hat High Availability 集群配置共享块存储的可选流程。该流程假设三个 Azure VM（一个三节点集群）带有一个 1TB 共享磁盘。

**注意**

这是配置块存储的独立示例步骤。该流程假设您还没有创建集群。

**先决条件**

- 您必须已在主机系统中安装了 Azure CLI，并创建了 SSH 密钥。
- 您必须已在 Azure 中创建了集群环境，其中包括创建以下内容。Microsoft Azure 文档链接。
  - 资源组
  - 虚拟网络
  - 网络安全组
  - 网络安全组规则
  - 子网
  - 负载均衡器（可选）
  - 存储帐户
  - 代理放置组
  - 可用性集

**流程**

1. 使用 Azure 命令 **az disk create** 创建共享块卷。

```
$ az disk create -g resource_group -n shared_block_volume_name --size-gb disk_size --max-shares number_vms -l location
```

例如，以下命令在 Azure Availability Zone **westcentralus** 的资源组 **sharedblock** 中创建了一个名为 **shared-block-volume.vhd** 的共享块卷。

```
$ az disk create -g sharedblock-rg -n shared-block-volume.vhd --size-gb 1024 --max-shares 3 -l westcentralus
```

```
{
  "creationData": {
    "createOption": "Empty",
    "galleryImageReference": null,
    "imageReference": null,
    "sourceResourceId": null,
    "sourceUniqueId": null,
    "sourceUri": null,
    "storageAccountId": null,
    "uploadSizeBytes": null
  },
  "diskAccessId": null,
  "diskIopsReadOnly": null,
  "diskIopsReadWrite": 5000,
  "diskMbpsReadOnly": null,
  "diskMbpsReadWrite": 200,
  "diskSizeBytes": 1099511627776,
  "diskSizeGb": 1024,
  "diskState": "Unattached",
  "encryption": {
    "diskEncryptionSetId": null,
    "type": "EncryptionAtRestWithPlatformKey"
  },
  "encryptionSettingsCollection": null,
  "hyperVgeneration": "V1",
  "id": "/subscriptions/12345678910-12345678910/resourceGroups/sharedblock-
rg/providers/Microsoft.Compute/disks/shared-block-volume.vhd",
  "location": "westcentralus",
  "managedBy": null,
  "managedByExtended": null,
  "maxShares": 3,
  "name": "shared-block-volume.vhd",
  "networkAccessPolicy": "AllowAll",
  "osType": null,
  "provisioningState": "Succeeded",
  "resourceGroup": "sharedblock-rg",
  "shareInfo": null,
  "sku": {
    "name": "Premium_LRS",
    "tier": "Premium"
  },
  "tags": {},
  "timeCreated": "2020-08-27T15:36:56.263382+00:00",
  "type": "Microsoft.Compute/disks",
  "uniqueId": "cd8b0a25-6fbe-4779-9312-8d9cbb89b6f2",
  "zones": null
}
```


2. 使用 Azure 命令 **az disk show** 创建共享块卷。

```
$ az disk show -g resource_group -n shared_block_volume_name
```

例如，以下命令显示资源组 **sharedblock-rg** 中的共享块卷 **shared-block-volume.vhd** 的详情。

```
$ az disk show -g sharedblock-rg -n shared-block-volume.vhd
```

```
{
  "creationData": {
    "createOption": "Empty",
    "galleryImageReference": null,
    "imageReference": null,
    "sourceResourceId": null,
    "sourceUniqueId": null,
    "sourceUri": null,
    "storageAccountId": null,
    "uploadSizeBytes": null
  },
  "diskAccessId": null,
  "diskIopsReadOnly": null,
  "diskIopsReadWrite": 5000,
  "diskMbpsReadOnly": null,
  "diskMbpsReadWrite": 200,
  "diskSizeBytes": 1099511627776,
  "diskSizeGb": 1024,
  "diskState": "Unattached",
  "encryption": {
    "diskEncryptionSetId": null,
    "type": "EncryptionAtRestWithPlatformKey"
  },
  "encryptionSettingsCollection": null,
  "hyperVgeneration": "V1",
  "id": "/subscriptions/12345678910-12345678910/resourceGroups/sharedblock-
rg/providers/Microsoft.Compute/disks/shared-block-volume.vhd",
  "location": "westcentralus",
  "managedBy": null,
  "managedByExtended": null,
  "maxShares": 3,
  "name": "shared-block-volume.vhd",
  "networkAccessPolicy": "AllowAll",
  "osType": null,
  "provisioningState": "Succeeded",
  "resourceGroup": "sharedblock-rg",
  "shareInfo": null,
  "sku": {
    "name": "Premium_LRS",
    "tier": "Premium"
  },
  "tags": {},
  "timeCreated": "2020-08-27T15:36:56.263382+00:00",
  "type": "Microsoft.Compute/disks",
  "uniqueId": "cd8b0a25-6fbe-4779-9312-8d9cbb89b6f2",
  "zones": null
}
```


3. 使用 Azure 命令 **az network nic create** 创建三个网络接口。运行以下命令三次，每次使用不同的 **<nic_name>**。

```
$ az network nic create -g resource_group -n nic_name --subnet subnet_name --vnet-name
virtual_network --location location --network-security-group network_security_group --
private-ip-address-version IPv4
```

例如：以下命令创建一个名为 **shareblock-nodea-vm-nic-protected** 的网络接口。

```
$ az network nic create -g sharedblock-rg -n sharedblock-nodea-vm-nic-protected --subnet
sharedblock-subnet-protected --vnet-name sharedblock-vn --location westcentralus --
network-security-group sharedblock-nsg --private-ip-address-version IPv4
```

4. 创建三个虚拟机并使用 Azure 命令 **az vm create** 附加共享块卷。每个虚拟机的选项值都是一样的，但每个虚拟机都有自己的 **<vm_name>**、**<new_vm_disk_name>** 和 **<nic_name>**。

```
$ az vm create -n vm_name -g resource_group --attach-data-disks
shared_block_volume_name --data-disk-caching None --os-disk-caching ReadWrite --os-
disk-name new-vm-disk-name --os-disk-size-gb disk_size --location location --size
virtual_machine_size --image image_name --admin-username vm_username --
authentication-type ssh --ssh-key-values ssh_key --nics -nic_name_ --availability-set
availability_set --ppg proximity_placement_group
```

例如，以下命令会创建一个名为 **sharedblock-nodea-vm** 的虚拟机。

```
$ az vm create -n sharedblock-nodea-vm -g sharedblock-rg --attach-data-disks shared-
block-volume.vhd --data-disk-caching None --os-disk-caching ReadWrite --os-disk-name
sharedblock-nodea-vm.vhd --os-disk-size-gb 64 --location westcentralus --size
Standard_D2s_v3 --image /subscriptions/12345678910-
12345678910/resourceGroups/sample-
azureimagesgroupwestcentralus/providers/Microsoft.Compute/images/sample-azure-rhel-
7.0-20200713.n.0.x86_64 --admin-username sharedblock-user --authentication-type ssh --
ssh-key-values @sharedblock-key.pub --nics sharedblock-nodea-vm-nic-protected --
availability-set sharedblock-as --ppg sharedblock-ppg

{
  "fqdns": "",
  "id": "/subscriptions/12345678910-12345678910/resourceGroups/sharedblock-
rg/providers/Microsoft.Compute/virtualMachines/sharedblock-nodea-vm",
  "location": "westcentralus",
  "macAddress": "00-22-48-5D-EE-FB",
  "powerState": "VM running",
  "privateIpAddress": "198.51.100.3",
  "publicIpAddress": "",
  "resourceGroup": "sharedblock-rg",
  "zones": ""
}
```

**验证步骤**

1. 对于集群中的每个虚拟机，使用虚拟机的 SSH 命令 **<ip_address>** 验证块设备是否可用。

```
# ssh ip_address "hostname ; lsblk -d | grep ' 1T '"
```

例如,以下命令列出了包括虚拟机 IP 地址 **198.51.100.3** 的主机名和块设备的详细信息。

```
# ssh 198.51.100.3 "hostname ; lsblk -d | grep ' 1T '"

nodea
sdb    8:16   0    1T  0 disk
```

2. 使用 SSH 命令验证集群中的每个虚拟机是否使用相同的共享磁盘。

```
# ssh _ip_address_s "hostname ; lsblk -d | grep ' 1T ' | awk '{print \$1}' | xargs -i udevadm info
--query=all --name=/dev/{} | grep '^E: ID_SERIAL='"
```

例如，以下命令列出了包括实例 IP 地址 **198.51.100.3** 的主机名和共享磁盘卷 ID 的详情。

```
# ssh 198.51.100.3 "hostname ; lsblk -d | grep ' 1T ' | awk '{print \$1}' | xargs -i udevadm info -
-query=all --name=/dev/{} | grep '^E: ID_SERIAL='"

nodea
E: ID_SERIAL=3600224808dd8eb102f6ffc5822c41d89
```

在确认将共享磁盘附加到每个虚拟机后，您可以为集群配置弹性存储。有关为红帽高可用性集群配置弹性存储的详情，请参考在集群中配置 GFS2 文件系统。有关 GFS2 文件系统的常规信息，请参考配置和管理 GFS2 文件系统。

# 第 3 章 在 AMAZON WEB SERVICES 上将 RED HAT ENTERPRISE LINUX 镜像部署为 EC2 实例

您有多个选项,可在 Amazon Web Services(AWS)上将 Red Hat Enterprise Linux(RHEL)7 镜像部署为 EC2 实例。本章讨论选择镜像的选项,以及列出或引用主机系统和虚拟机(VM)的系统要求。本章还提供了从 ISO 镜像创建自定义虚拟机并将其上传到 EC2 和启动 EC2 实例的步骤。

**重要**

您可以从 ISO 镜像创建自定义虚拟机,但红帽建议您使用 Red Hat Image Builder 产品来创建自定义镜像,供特定云供应商使用。使用镜像构建器,您可以使用 **ami** 格式创建并上传 AMI(Amazon Machine Image)。如需更多信息，请参阅镜像构建器指南。

本章在很多位置使用了 Amazon 文档。如需很多流程，请参阅引用的 Amazon 文档了解更多详情。

**注意**

如需可以在 AWS 上安全使用的红帽产品列表，请参阅 Amazon Web Services。

**先决条件**

- 注册一个红帽客户门户网站（Red Hat Customer Portal） 帐户。
- 注册 AWS 并设置 AWS 资源。如需更多信息，请参阅使用 Amazon EC2 设置。
- 在 Red Hat Cloud Access 程序中启用您的订阅 。Red Hat Cloud Access 程序允许您在红帽的完全支持下将红帽订阅从物理或内部系统移到 AWS。

**其它资源**

- Red Hat Cloud Access 参考指南
- 公共云中的红帽
- Amazon EC2 上的 Red Hat Enterprise Linux - FAQ
- Setting Up with Amazon EC2
- Red Hat on Amazon Web Services

## 3.1. AWS 上的 RED HAT ENTERPRISE LINUX 镜像选项

下表列出了镜像的不同选择并记录镜像选项的不同。

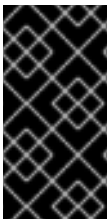
**表 3.1. 镜像选项**

| 镜像选项 | 订阅 | 示例情境 | 注意事项 |
|---|---|---|---|

| 镜像选项 | 订阅 | 示例情境 | 注意事项 |
| --- | --- | --- | --- |
| 选择部署一个 Red Hat Gold Image。 | 利用您现有的红帽订阅。 | 通过 Red Hat Cloud Access 程序 启用订阅,然后在 AWS 上选择 Red Hat Gold Image。 | 订阅包括红帽产品成本；您可以为 Amazon 提供所有其他实例成本。<br><br>Red Hat Gold Images 被称为 "Cloud Access" 镜像，因为您使用现有的红帽订阅。红帽直接为 Cloud Access 镜像提供支持。 |
| 选择部署移动到 AWS 的自定义镜像。 | 利用您现有的红帽订阅。 | 通过 Red Hat Cloud Access 程序 启用订阅,上传您的自定义镜像并附加您的订阅。 | 订阅包括红帽产品成本；您可以为 Amazon 提供所有其他实例成本。<br><br>移动到 AWS 的自定义镜像是 "Cloud Access" 镜像,因为您利用了您现有的红帽订阅。红帽直接为 Cloud Access 镜像提供支持。 |
| 选择部署包含 RHEL 的现有 Amazon 镜像。 | AWS EC2 镜像包括红帽产品。 | 在 AWS 管理控制台上启动实例时，选择 RHEL 镜像，或者从 AWS Marketplace 中选择镜像。 | 根据 pay-as-you-go 模式每小时向 Amazon 支付。这样的镜像称为 "on-demand" 镜像。Amazon 支持 on-demand 镜像。<br><br>红帽提供了镜像的更新。AWS 通过 Red Hat Update Infrastructure(RHUI)提供更新。 |

**注意**

您可以使用 Red Hat Image Builder 为 AWS 创建自定义镜像。如需更多信息，请参阅镜像构建器指南。



**重要**

您无法将 on-demand 实例转换为 Red Hat Cloud Access 实例。要从 on-demand 镜像改为 Red Hat Cloud Access bring-your-own-subscription(BYOS)镜像,请创建新的 Red Hat Cloud Access 实例并从您的按需实例迁移数据。在迁移数据后取消您的 on-demand 实例以避免出现重复账单。

本章的剩余部分包含与自定义镜像相关的信息和流程。

**其它资源**

- 使用 Red Hat Gold 镜像
- Red Hat Cloud Access 程序
- 镜像构建器指南
- AWS Management Console
- AWS Marketplace

## 3.2. 安装 AWS CLI

本章的许多流程包括使用 AWS CLI。完成以下步骤以安装 AWS CLI。

**先决条件**

您需要已创建并有权访问 AWS 访问密钥 ID 和 AWS Secret 访问密钥。有关信息和说明，请参阅快速配置 AWS CLI。

**流程**

1. 安装 Python 3 和 **pip** 工具。

```
# yum install python3
# yum install python3-pip
```

2. 使用 **pip** 命令安装 AWS 命令行工具。

```
# pip3 install awscli
```

3. 运行 **aws --version** 命令验证您是否安装了 AWS CLI。

```
$ aws --version
aws-cli/1.16.182 Python/2.7.5 Linux/3.10.0-957.21.3.el7.x86_64 botocore/1.12.172
```

4. 根据 AWS 访问详情配置 AWS 命令行客户端。

```
$ aws configure
AWS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [None]:
Default output format [None]:
```

**其它资源**

- 快速配置 AWS CLI
- AWS 命令行工具

## 3.3. 虚拟机配置设置

云虚拟机必须具有以下配置设置。

表 3.2. 虚拟机配置设置

| 设置 | 建议 |
| --- | --- |
| ssh | **ssh** 必须启用 SSH 来提供虚拟机的远程访问。 |
| dhcp | 应该为 dhcp 配置主虚拟适配器。 |

## 3.4. 从 ISO 镜像创建基本虚拟机

按照本节中的步骤从 ISO 镜像创建基础镜像。

### 先决条件

按照 Virtualization 部署和管理指南，为您的 Red Hat Enterprise Linux 7 主机机器启用虚拟化。

### 3.4.1. 下载 ISO 镜像

#### 流程

1. 从红帽客户门户网站下载最新的 Red Hat Enterprise Linux ISO 镜像。
2. 将镜像移动到 **/var/lib/libvirt/images** 目录中。

### 3.4.2. 从 ISO 镜像创建虚拟机

#### 流程

1. 确保已为虚拟化启用主机机器。有关安装可升级软件包的信息和步骤，请参阅在现有 Red Hat Enterprise Linux 系统上安装虚拟化软件包
2. 创建并启动基本 Red Hat Enterprise Linux 虚拟机。有关创建虚拟机的说明，请参阅创建虚拟机。
   a. 如果使用命令行创建虚拟机，请确保将默认内存和 CPU 设置为您所需的容量。将您的虚拟网络接口设置为 **virtio**。
   下面是一个基本的命令行示例。

   ```
   virt-install --name _vmname_ --memory 2048 --vcpus 2 --disk size=8,bus=virtio --location rhel-7.0-x86_64-dvd.iso --os-variant=rhel7.0
   ```

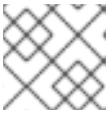
   b. 如果您使用 virt-manager 应用程序创建虚拟机，请按照使用 virt-manager 创建虚拟机的步骤进行以下操作：
      - 不要选择 **Immediately Start VM**。
      - 将 **Memory** 和 **Storage Size** 设置为您需要的值。
      - 在开始安装前，请确保将 **Virtual Network Interface Settings** 中的 **Model** 更改为 **virtio**，并将您的 **vCPU** 更改为您想要的虚拟机容量设置。

### 3.4.3. 完成 RHEL 安装

执行以下步骤完成安装并在虚拟机启动后启用 root 访问。

**流程**

1. 选择您要在安装过程中使用的语言。
2. 在 **Installation Summary** 视图中：
   a. 点 **Software Selection**，选择 **Minimal Install**。
   b. 点 **Done**。
   c. 点击 **Installation Destination** 并检查 **Storage Configuration** 中的 **Custom**。
      - 确保 **/boot** 验证至少有 500 MB。您可以使用剩余空间作为 root /。
      - 建议使用标准分区,但您可以使用逻辑卷管理(LVM)。
      - 您可以将 xfs、ext4 或者 ext3 用于文件系统。
      - 完成更改后点 **Done**。
3. 点 **Begin Installation**。
4. 设置 **Root 密码**。
5. 重启虚拟机，然后在安装完成后以 **root** 身份登录。
6. 配置镜像。



**注意**

确定安装并启用 **cloud-init** 软件包。

7. 关闭虚拟机。

## 3.5. 将 RED HAT ENTERPRISE LINUX 镜像上传到 AWS

按照本节中的步骤，在主机机器中将您的镜像上传到 AWS。

### 3.5.1. 创建 S3 存储桶

导入到 AWS 需要 Amazon S3 存储桶。Amazon S3 存储桶是一个 Amazon 资源用于存储对象。作为上传镜像过程的一部分，您可以创建一个 S3 存储桶，然后将镜像移到存储桶。完成以下步骤以创建存储桶。

**先决条件**

- 您需要安装 AWS CLI。如需更多信息，请参阅安装 AWS CLI。

**流程**

1. 启动 Amazon S3 控制台。
2. 点 **Create Bucket**。此时会出现 **Create Bucket** 对话框。

3. 在 **Name and region** 视图中：

   a. 输入 **Bucket name**。

   b. 选择 **Region**。在字段中输入您的区域,或者点击下拉并从所有可用区域选择您的区域。

   c. 点 **Next**。

4. 在 **Configure options** 视图中，选择所需选项并点 **Next**。

5. 在 **Set permissions** 视图中，更改或者接受默认选项并点 **Next**。

6. 查看存储桶配置。

7. 点 **Create bucket**。

**注意**

另外，您可以使用 AWS CLI 创建存储桶。例如： **aws s3 mb s3://my-new-bucket** 创建一个名为 **my-new-bucket** 的 S3 存储桶。有关 **mb** 命令的详情，请参考 AWS CLI 命令参考。

**其它资源**

- Amazon S3 Console
- AWS CLI Command Reference

### 3.5.2. 创建 vmimport 角色

执行以下步骤创建 VM 导入所需的 **vmimport** 角色。如需更多信息，请参阅 Amazon 文档中的 VM Import Service Role 部分。

**流程**

1. 创建名为 **trust-policy.json** 的文件并包含以下策略。在您的系统中保存该文件并记录其位置。

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Principal": { "Service": "vmie.amazonaws.com" },
         "Action": "sts:AssumeRole",
         "Condition": {
            "StringEquals":{
               "sts:Externalid": "vmimport"
            }
         }
      }
   ]
}
```

2. 使用 **create role** 命令创建 **vmimport** 角色。指定 **trust-policy.json** 文件位置的完整路径。使用 **file://** 作为路径的前缀。下面是一个示例。

```
aws iam create-role _--role-name_ vmimport --assume-role-policy-document
file:///home/sample/ImportService/trust-policy.json
```

3. 创建名为 **role-policy.json** 的文件并包含以下策略。将 **s3-bucket-name** 替换为 S3 存储桶的名称。

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetBucketLocation",
            "s3:GetObject",
            "s3:ListBucket"
         ],
         "Resource":[
            "arn:aws:s3:::s3-bucket-name",
            "arn:aws:s3:::s3-bucket-name/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "ec2:ModifySnapshotAttribute",
            "ec2:CopySnapshot",
            "ec2:RegisterImage",
            "ec2:Describe*"
         ],
         "Resource":"*"
      }
   ]
}
```

4. 使用 **put-role-policy** 命令将策略附加到您创建的角色。指定 **role-policy.json** 文件的完整路径。下面是一个示例。

```
aws iam put-role-policy _--role-name_ vmimport _--policy-name_ vmimport --policy-
document file:///home/sample/ImportService/role-policy.json
```

**其它资源**

- VM 导入服务角色
- 所需的服务角色

### 3.5.3. 将 AMI 转换为 S3

完成以下步骤,将 Amazon Machine Image(AMI)转换为 S3。示例是代表 ; 它们将 **qcow2** 文件格式格式的镜像转换为 **raw** 格式。Amazon 接受 **OVA**、**VHD**、**VHDX VMDK**、和 **raw** 格式的镜像。如需更多信息,请参阅 Amazon 接受的镜像格式的 VM Import/Export Works。

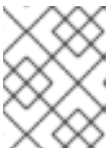
**流程**

1. 运行 **qemu-img** 命令转换您的镜像。下面是一个示例。

```
qemu-img convert -f qcow2 -O raw rhel-server-7.7-1-x86_64-kvm.qcow2 rhel-server-7.7-1-x86_64-kvm.raw
```

2. 将镜像推送到 S3。

```
aws s3 cp rhel-server-7.7.1-x86_64-kvm.raw s3://s3-_bucket-name_
```



**注意**

这个过程可能需要几分钟时间。完成后，您可以使用 AWS S3 控制台检查您的镜像是否已成功上传到 S3 存储桶。

**其它资源**

- VM 导入/导出的工作方式
- AWS S3 控制台

### 3.5.4. 从原始镜像创建 AMI

执行以下步骤从原始镜像创建 AMI。

**先决条件**

- 您需要安装 AWS CLI。如需更多信息，请参阅安装 AWS CLI。

**流程**

- 您可以在 AWS CLI 上运行 **aws ec2 import-image** 命令从原始镜像创建 AMI。

```
# aws ec2 import-image --platform Linux --license-type BYOL --no-encrypted --description _imagedescription_ --architecture x86_64 --disk-containers Format=Raw,UserBucket="{S3Bucket=virtqes1,S3Key=rhel-server-ec2-7.9-30.x86_64.raw}" --region _regionname_
```

**其它资源**

- 将您的虚拟机导入为镜像

### 3.5.5. 从 AMI 启动实例

执行以下步骤从 AMI 启动和配置实例。

**流程**

1. 在 AWS EC2 Dashboard 中选择 **Images**，然后选择 **AMI**。
2. 右键点击您的镜像并选择 **Launch**。
3. 选择满足或超过工作负载要求的 **实例类型**。
   如需有关实例类型的信息，请参阅 Amazon EC2 实例类型。

4. 点 **Next: Configure Instance Details**。

   a. 输入您要创建的实**例数量**。

   b. 对于 **Network**，选择您在设置 AWS 环境时创建的 VPC。为实例选择子网或创建新子网。

   c. 为 Auto-assign Public IP 选择 **Enable**。

**注意**

这些是创建基本实例所需的最小配置选项。根据您的应用程序要求查看其他选项。

5. 点击 **Next: Add Storage**。验证默认存储是否足够。

6. 点击 **Next: Add Tags**。

**注意**

标签可帮助您管理 AWS 资源。有关标记的信息，请参阅标记您的 Amazon EC2 资源。

7. 点 **Next: 配置安全组**。选择设置 AWS 环境时创建的安全组。

8. 点 **Review and Launch**。验证您的选择。

9. 点 **Launch**。此时会提示您选择现有密钥对或创建新密钥对。选择设置 AWS 环境时创建的密钥对。

**注意**

验证您的私钥权限是否正确。如果需要，使用命令选项 **chmod 400 <keyname>.pem** 更改权限。

10. 点 **Launch Instances**。

11. 点 **View Instances**。您可以命名实例。
现在,您可以通过选择实例并点击 **连接** 来启动 SSH 会话到实例。使用为 **独立 SSH 客户端**提供的示例命令。

**注意**

另外，您可以使用 AWS CLI 启动实例。如需更多信息，请参阅 Amazon 文档中的启动、列出和终止 Amazon EC2 实例。

**其它资源**

- AWS Management Console
- Setting Up with Amazon EC2
- Amazon EC2 实例
- Amazon EC2 实例类型

### 3.5.6. 附加红帽订阅

完成以下步骤以附加您之前通过 Red Hat Cloud Access 程序启用的订阅。

#### 先决条件

您必须已启用您的订阅。

#### 流程

1. 注册您的系统。

```
subscription-manager register --auto-attach
```

2. 附加您的订阅。
   - 您可以使用激活码来附加订阅。请参阅创建红帽客户门户网站激活码。
   - 或者，您可以使用订阅池（池 ID）的 ID 手动附加订阅。请参阅通过 command Line 来附加和删除订阅。

#### 其它资源

- 创建红帽客户门户网站激活码
- 通过命令行附加和删除订阅
- 使用并配置 Red Hat Subscription Manager

# 第 4 章 在 AWS 上配置红帽高可用性集群

本章包含使用 EC2 实例作为集群节点在 Amazon Web Services(AWS)上配置红帽高可用性(HA)集群的信息和步骤。您有多个选项来获取您用于集群的 Red Hat Enterprise Linux(RHEL)镜像。有关 AWS 镜像选项的详情，请查看 AWS 的 Red Hat Enterprise Linux 镜像选项 。

本章包含为 AWS 设置环境的先决条件。设置环境后，您可以创建并配置 EC2 实例。

本章还包含与创建 HA 集群相关的流程，该集群将单个节点转换为 AWS 上的一个 HA 节点集群。这包括在每个集群节点上安装高可用性软件包和代理、配置隔离以及安装 AWS 网络资源代理的步骤。

本章在很多位置使用了 Amazon 文档。更多信息，请参阅引用的 Amazon 文档来获得更多信息。

**先决条件**

- 您需要安装 AWS 命令行界面(CLI)。有关安装 AWS CLI 的更多信息，请参阅安装 AWS CLI。
- 在 Red Hat Cloud Access 程序中启用您的订阅 。Red Hat Cloud Access 程序允许您在红帽的完全支持下将红帽订阅从物理或内部系统移到 AWS。

**其它资源**

- Red Hat Cloud Access 参考指南
- 公共云中的红帽
- Amazon EC2 上的 Red Hat Enterprise Linux - FAQ
- 设置 Amazon EC2
- Red Hat on Amazon Web Services
- RHEL 高可用性集群的支持政策

## 4.1. 创建 AWS 访问密钥和 AWS SECRET 访问密钥

在安装 AWS CLI 前，您需要创建一个 AWS 访问密钥和 AWS Secret 访问密钥。隔离和资源代理 API 使用 AWS 访问密钥和 Secret 访问密钥连接到集群中的每个节点。

完成以下步骤以创建这些密钥。

**先决条件**

您的 IAM 用户帐户必须具有 Programmatic 访问权限。如需更多信息，请参阅设置 AWS 环境。

**流程**

1. 启动 AWS 控制台。
2. 点击 AWS Account ID 以显示下拉菜单并选择 **My Security Credentials**。
3. 点 **Users**。
4. 选择用户以打开 **Summary** 屏幕。
5. 点 **Security credentials** 选项卡。

6. 点 **Create access key**。
7. 下载 **.csv** 文件（或者保存这两个密钥）。创建隔离设备时需要输入这些密钥。

## 4.2. 安装 HA 软件包和代理

在所有节点上完成以下步骤以安装 HA 软件包和代理。

**流程**

1. 输入以下命令删除 AWS Red Hat Update Infrastructure（RHUI）客户端。由于您要使用 Red Hat Cloud Access 订阅,所以您不应该在订阅之外使用 AWS RHUI。

```
$ sudo -i
# yum -y remove rh-amazon-rhui-client*
```

2. 在红帽注册虚拟机。

```
# subscription-manager register --auto-attach
```

3. 禁用所有软件仓库。

```
# subscription-manager repos --disable=*
```

4. 启用 RHEL 7 Server 和 RHEL 7 Server HA 软件仓库。

```
# subscription-manager repos --enable=rhel-7-server-rpms
# subscription-manager repos --enable=rhel-ha-for-rhel-7-server-rpms
```

5. 更新所有软件包。

```
# yum update -y
```

6. 如果内核被更新，则需要重启。

```
# reboot
```

7. 安装 pcs、pacemaker、隔离代理和资源代理。

```
# yum -y install pcs pacemaker fence-agents-aws resource-agents
```

8. 用户 **hacluster** 在上一步中的 **pcs** 和 **pacemaker** 安装中创建。在所有集群节点上为 **hacluster** 创建密码。所有节点都使用相同的密码。

```
# passwd hacluster
```

9. 如果启用了 **firewalld.service**，在 RHEL 防火墙中添加 **high availability** 服务。

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

10. 启动 **pcs** 服务并在引导时启用它。

```
# systemctl enable pcsd.service --now
```

**验证步骤**

确定 **pcs** 服务正在运行。

```
# systemctl is-active pcsd.service
```

## 4.3. 创建集群

完成以下步骤以创建节点集群。

**流程**

1. 在其中一个节点上，输入以下命令验证 pcs 用户 **hacluster**。指定集群中的每个节点的名称。

   ```
   # pcs host auth  _hostname1_ _hostname2_ _hostname3_
   ```

   例如：

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

2. 创建集群。

   ```
   # pcs cluster setup --name _hostname1_ _hostname2_ _hostname3_
   ```

   例如：

   ```
   [root@node01 clouduser]# pcs cluster setup --name newcluster node01 node02 node03

   ...omitted

   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
   ```

**验证步骤**

1. 启用集群。

   ```
   # pcs cluster enable --all
   ```

2. 启动集群。

```
# pcs cluster start --all
```

例如：

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

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

## 4.4. 创建隔离设备

完成以下步骤来配置隔离。

**流程**

1. 输入以下 AWS 元数据查询以获取每个节点的实例 ID。您需要这些 ID 来配置隔离设备。如需更多信息，请参阅实例元数据和用户数据。

```
# echo $(curl -s http://169.254.169.254/latest/meta-data/instance-id)
```

例如：

```
[root@ip-10-0-0-48 ~]# echo $(curl -s http://169.254.169.254/latest/meta-data/instance-id) i-07f1ac63af0ec0ac6
```

2. 创建隔离设备。使用 **pcmk_host_map** 命令将 RHEL 主机名映射到实例 ID。使用之前在 创建 AWS 访问密钥和 AWS Secret 访问密钥中设置的 AWS 访问密钥和 AWS Secret 访问密钥。

```
# pcs stonith create cluster_fence fence_aws access_key=access-key secret_key=_secret-access-key_ region=_region_ pcmk_host_map="rhel-hostname-1:Instance-ID-1;rhel-hostname-2:Instance-ID-2;rhel-hostname-3:Instance-ID-3"
```

例如：

```
[root@ip-10-0-0-48 ~]# pcs stonith create clusterfence fence_aws access_key=AKIAI*******6MRMJA secret_key=a75EYIG4RVL3h*******K7koQ8dzaDyn5yoIZ/ region=us-east-1 pcmk_host_map="ip-10-0-0-48:i-07f1ac63af0ec0ac6;ip-10-0-0-46:i-063fc5fe93b4167b2;ip-10-0-0-58:i-08bd39eb03a6fd2c7" power_timeout=240 pcmk_reboot_timeout=480 pcmk_reboot_retries=4
```

**验证步骤**

1. 测试其他其中一个节点的隔离代理。

```
# pcs stonith fence _awsnodename_
```

例如：

```
[root@ip-10-0-0-48 ~]# pcs stonith fence ip-10-0-0-58
Node: ip-10-0-0-58 fenced
```

2. 检查状态以验证该节点是否已隔离。

```
# watch pcs status
```

例如：

```
[root@ip-10-0-0-48 ~]# pcs status
Cluster name: newcluster
Stack: corosync
Current DC: ip-10-0-0-46 (version 1.1.18-11.el7-2b07d5c5a9) - partition with quorum
Last updated: Fri Mar  2 20:01:31 2018
Last change: Fri Mar  2 19:24:59 2018 by root via cibadmin on ip-10-0-0-48

3 nodes configured
1 resource configured

Online: [ ip-10-0-0-46 ip-10-0-0-48 ip-10-0-0-58 ]

Full list of resources:

 clusterfence  (stonith:fence_aws):    Started ip-10-0-0-46

Daemon Status:
  corosync: active/disabled
  pacemaker: active/disabled
  pcsd: active/enabled
```

## 4.5. 在集群节点上安装 AWS CLI

在以前的版本中，您在主机系统中安装了 AWS CLI。现在，在配置网络资源代理前，您需要在集群节点上安装 AWS CLI。

在每个集群节点上完成以下步骤。

### 先决条件

您必须已创建了 AWS Access Key 和 AWS Secret 访问密钥。如需更多信息，请参阅创建 AWS 访问密钥和 AWS Secret 访问密钥。

### 流程

1. 执行安装 AWS CLI 的步骤。
2. 输入以下命令验证 AWS CLI 是否已正确配置。应该会显示实例 ID 和实例名称。
   例如：

```
[root@ip-10-0-0-48 ~]# aws ec2 describe-instances --output text --query
'Reservations[*].Instances[*].[InstanceId,Tags[?Key==`Name`].Value]'
i-07f1ac63af0ec0ac6
```

```
ip-10-0-0-48
i-063fc5fe93b4167b2
ip-10-0-0-46
i-08bd39eb03a6fd2c7
ip-10-0-0-58
```

## 4.6. 安装网络资源代理

要使 HA 操作正常工作,集群使用 AWS 网络资源代理来启用故障切换功能。如果节点在设定的时间里不响应 heartbeat 检查,则该节点会被隔离,操作切换到集群中的额外节点。需要配置网络资源代理才能正常工作。

将两个资源添加到 同一组 以强制 **order** 和 **colocation** 限制。

**创建二级私有 IP 资源及虚拟 IP 资源**

完成以下步骤以添加二级专用 IP 地址并创建虚拟 IP。您可以从集群中的任何节点完成此步骤。

**流程**

1. 输入以下命令查看 **AWS Secondary Private IP Address** 资源代理(awsvip)描述。这显示了这个代理的选项和默认操作。

```
# pcs resource describe awsvip
```

2. 输入以下命令使用 **VPC CIDR** 块中未使用的私有 IP 地址创建二级私有 IP 地址。

```
# pcs resource create privip awsvip secondary_private_ip=_Unused-IP-Address_ --group
_group-name_
```

例如：

```
[root@ip-10-0-0-48 ~]# pcs resource create privip awsvip secondary_private_ip=10.0.0.68 --
group networking-group
```

3. 创建虚拟 IP 资源。这是一个 VPC IP 地址，可以从隔离的节点快速迁移到故障切换节点，从而使子网中隔离的节点失败。

```
# pcs resource create vip IPaddr2 ip=_secondary-private-IP_ --group _group-name_
```

例如：

```
root@ip-10-0-0-48 ~]# pcs resource create vip IPaddr2 ip=10.0.0.68 --group networking-
group
```

**验证步骤**

输入 **pcs status** 命令来验证资源是否正在运行。

```
# pcs status
```

例如：

```
[root@ip-10-0-0-48 ~]# pcs status
Cluster name: newcluster
Stack: corosync
Current DC: ip-10-0-0-46 (version 1.1.18-11.el7-2b07d5c5a9) - partition with quorum
Last updated: Fri Mar  2 22:34:24 2018
Last change: Fri Mar  2 22:14:58 2018 by root via cibadmin on ip-10-0-0-46

3 nodes configured
3 resources configured

Online: [ ip-10-0-0-46 ip-10-0-0-48 ip-10-0-0-58 ]

Full list of resources:

clusterfence    (stonith:fence_aws):    Started ip-10-0-0-46
 Resource Group: networking-group
     privip (ocf::heartbeat:awsvip):    Started ip-10-0-0-48
     vip    (ocf::heartbeat:IPaddr2):   Started ip-10-0-0-58

Daemon Status:
  corosync: active/disabled
  pacemaker: active/disabled
  pcsd: active/enabled
```

**创建弹性 IP 地址**

弹性 IP 地址是一个公共 IP 地址,可以从隔离的节点快速迁移到故障转移节点,从而使隔离的节点失败。

请注意，这与之前创建的虚拟 IP 资源不同。弹性 IP 地址用于面向公共的互联网连接，而不是子网连接。

1. 将两个资源添加到之前创建的用来强制 **order** 和 **colocation** 约束的 同一组 中。
2. 输入以下 AWS CLI 命令来创建弹性 IP 地址。

   ```
   [root@ip-10-0-0-48 ~]# aws ec2 allocate-address --domain vpc --output text
   eipalloc-4c4a2c45    vpc 35.169.153.122
   ```

3. 输入以下命令查看 AWS 二级 Elastic IP 地址资源代理(awseip)描述。这显示了这个代理的选项和默认操作。

   ```
   # pcs resource describe awseip
   ```

4. 使用步骤 1 中创建的分配的 IP 地址创建二级 Elastic IP 地址资源。

   ```
   # pcs resource create elastic awseip elastic_ip=_Elastic-IP-Address_allocation_id=_Elastic-IP-Association-ID_ --group networking-group
   ```

   例如：

   ```
   # pcs resource create elastic awseip elastic_ip=35.169.153.122 allocation_id=eipalloc-4c4a2c45 --group networking-group
   ```

**验证步骤**

输入 **pcs status** 命令来验证资源是否正在运行。

```
# pcs status
```

例如：

```
[root@ip-10-0-0-58 ~]# pcs status
Cluster name: newcluster
Stack: corosync
Current DC: ip-10-0-0-58 (version 1.1.18-11.el7-2b07d5c5a9) - partition with quorum
Last updated: Mon Mar  5 16:27:55 2018
Last change: Mon Mar  5 15:57:51 2018 by root via cibadmin on ip-10-0-0-46

3 nodes configured
4 resources configured

Online: [ ip-10-0-0-46 ip-10-0-0-48 ip-10-0-0-58 ]

Full list of resources:

 clusterfence   (stonith:fence_aws):    Started ip-10-0-0-46
 Resource Group: networking-group
     privip (ocf::heartbeat:awsvip):  Started ip-10-0-0-48
     vip    (ocf::heartbeat:IPaddr2):    Started ip-10-0-0-48
     elastic (ocf::heartbeat:awseip):    Started ip-10-0-0-48

Daemon Status:
  corosync: active/disabled
  pacemaker: active/disabled
  pcsd: active/enabled
```

**测试弹性 IP 地址**

输入以下命令验证虚拟 IP(awsvip)和弹性 IP(awseip)资源是否可以正常工作。

**流程**

1. 从本地工作站启动 SSH 会话到之前创建的弹性 IP 地址。

   ```
   $ ssh -l ec2-user -i ~/.ssh/<KeyName>.pem elastic-IP
   ```

   例如：

   ```
   $ ssh -l ec2-user -i ~/.ssh/cluster-admin.pem 35.169.153.122
   ```

2. 验证您通过 SSH 连接到的主机是否与创建的弹性资源关联。

**其它资源**

- 高可用性附加组件概述
- High Availability Add-On 管理
- High Availability Add-On 参考

## 4.7. 配置共享块存储

本节提供了使用 Amazon EBS Multi-Attach 卷为红帽高可用性集群配置共享块存储的可选步骤。此流程假设三个带有 1TB 共享磁盘的实例（三节点集群）。

**流程**

1. 使用 AWS 命令 create-volume 创建共享块卷。

```
$ aws ec2 create-volume --availability-zone availability_zone --no-encrypted --size 1024 --volume-type io1 --iops 51200 --multi-attach-enabled
```

例如，以下命令在 **us-east-1a** 可用区中创建卷。

```
$ aws ec2 create-volume --availability-zone us-east-1a --no-encrypted --size 1024 --volume-type io1 --iops 51200 --multi-attach-enabled

{
    "AvailabilityZone": "us-east-1a",
    "CreateTime": "2020-08-27T19:16:42.000Z",
    "Encrypted": false,
    "Size": 1024,
    "SnapshotId": "",
    "State": "creating",
    "VolumeId": "vol-042a5652867304f09",
    "Iops": 51200,
    "Tags": [ ],
    "VolumeType": "io1"
}
```

**注意**

下一步需要 **VolumeId**。

2. 对于集群中的每个实例，使用 AWS 命令 attach-volume 附加一个共享块卷。使用您的 **<instance_id>** 和 **<volume_id>**。

```
$ aws ec2 attach-volume --device /dev/xvdd --instance-id instance_id --volume-id volume_id
```

例如：以下命令将共享块卷 **vol-042a5652867304f09** 附加到 **instance i-0eb803361c2c887f2**。

```
$ aws ec2 attach-volume --device /dev/xvdd --instance-id i-0eb803361c2c887f2 --volume-id vol-042a5652867304f09

{
    "AttachTime": "2020-08-27T19:26:16.086Z",
    "Device": "/dev/xvdd",
    "InstanceId": "i-0eb803361c2c887f2",
    "State": "attaching",
    "VolumeId": "vol-042a5652867304f09"
}
```

**验证步骤**

1. 对于集群中的每个实例，使用带有实例 **<ip_address>** 的 SSH 命令验证块设备是否可用。

```
# ssh <ip_address> "hostname ; lsblk -d | grep ' 1T '"
```

例如：以下命令列出了包括实例 IP **198.51.100.3** 的主机名和块设备的详情。

```
# ssh 198.51.100.3 "hostname ; lsblk -d | grep ' 1T '"

nodea
nvme2n1 259:1    0   1T  0 disk
```

2. 使用 **ssh** 命令验证集群中的每个实例是否使用相同的共享磁盘。

```
# ssh ip_address "hostname ; lsblk -d | grep ' 1T ' | awk '{print \$1}' | xargs -i udevadm info --query=all --name=/dev/{} | grep '^E: ID_SERIAL='"
```

例如，以下命令列出了包括实例 IP 地址 **198.51.100.3** 的主机名和共享磁盘卷 ID 的详情。

```
 # ssh 198.51.100.3 "hostname ; lsblk -d | grep ' 1T ' | awk '{print \$1}' | xargs -i udevadm info --query=all --name=/dev/{} | grep '^E: ID_SERIAL='"

nodea
E: ID_SERIAL=Amazon Elastic Block Store_vol0fa5342e7aedf09f7
```

在确认将共享磁盘附加到每个实例后，您可以为集群配置弹性存储。有关为红帽高可用性集群配置弹性存储的详情，请参考在集群中配置 GFS2 文件系统。有关 GFS2 文件系统的常规信息，请参考配置和管理 GFS2 文件系统。

# 第 5 章 在 GOOGLE CLOUD PLATFORM 上将 RED HAT ENTERPRISE LINUX 镜像部署为 GOOGLE COMPUTE ENGINE 实例

您有多个选项,可在 Google Cloud Platform(GCP)上将 Red Hat Enterprise Linux(RHEL)7 镜像部署为 Google Compute Engine(GCE)实例。本章讨论选择镜像的选项,以及列出或注明主机系统和虚拟机的系统要求。本章介绍了从 ISO 镜像创建自定义虚拟机、上传到 GCE 并启动实例的步骤。

本章在很多位置使用了 Google 文档。如需更多信息，请参阅引用的 Google 文档。

**注意**

有关 GCP 红帽产品认证列表，请参阅 Google Cloud Platform 上的红帽。

**先决条件**

- 您需要一个红帽客户门户网站帐户才能完成本章中的步骤。
- 使用 GCP 创建帐户来访问 Google Cloud Platform 控制台。如需更多信息，请参阅 Google Cloud。
- 通过 Red Hat Cloud Access 程序 启用您的红帽订阅。Red Hat Cloud Access 程序允许您在红帽的完全支持下将红帽订阅从物理或内部系统移到 GCP。

**其它资源**

- 公共云中的红帽
- Google Cloud

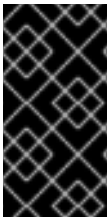
## 5.1. GCP 上的 RED HAT ENTERPRISE LINUX 镜像选项

下表列出了镜像选择和镜像选项的不同。

**表 5.1. 镜像选项**

| 镜像选项 | 订阅 | 示例情境 | 注意事项 |
| --- | --- | --- | --- |
| 选择部署移至 GCP 的自定义镜像。 | 利用您现有的红帽订阅。 | 通过 Red Hat Cloud Access 程序 启用订阅,上传您的自定义镜像并附加您的订阅。 | 订阅只包括红帽产品的成本；您还需要支付其他成本。<br><br>移至 GCP 的自定义镜像名为 "Cloud Access" 镜像，因为您利用了您现有的红帽订阅。红帽直接为 Cloud Access 镜像提供支持。 |

| 镜像选项 | 订阅 | 示例情境 | 注意事项 |
| --- | --- | --- | --- |
| 选择部署包含 RHEL 的现有 GCP 镜像。 | GCP 镜像包括一个红帽产品。 | 在 GCP Compute Engine 上启动实例时选择 RHEL 镜像，或者从 Google Cloud Platform Marketplace 中选择镜像。 | 根据 pay-as-you-go 模式每小时向 GCP 支付。这样的镜像称为 "on-demand" 镜像。GCP 通过支持协议支持 on-demand 镜像。 |



**重要**

您无法将 on-demand 实例转换为 Red Hat Cloud Access 实例。要从 on-demand 镜像改为 Red Hat Cloud Access bring-your-own subscription(BYOS)镜像,创建一个新的 Red Hat Cloud Access 实例,并从您的 on-demand 实例迁移数据。在迁移数据后取消您的 on-demand 实例以避免出现重复账单。

本章的剩余部分包含与自定义镜像相关的信息和流程。

**其它资源**

- 公共云中的红帽
- 镜像
- Red Hat Cloud Access 参考指南
- 从自定义镜像创建实例

## 5.2. 理解基础镜像

本节介绍使用预配置的基础镜像及其配置设置的信息。

### 5.2.1. 使用自定义基础镜像

要手动配置虚拟机，请以基础（启动程序）虚拟机镜像开始。创建基本虚拟机镜像后，您可以修改配置设置并添加虚拟机在云中操作的软件包。您可在上传镜像后为特定应用程序进行额外的配置更改。

**其它资源**

Red Hat Enterprise Linux

### 5.2.2. 虚拟机配置设置

云虚拟机必须具有以下配置设置。

**表 5.2. 虚拟机配置设置**

| 设置 | 建议 |
| --- | --- |
| ssh | **ssh** 必须启用 SSH 来提供虚拟机的远程访问。 |

| 设置 | 建议 |
| --- | --- |
| dhcp | 应该为 dhcp 配置主虚拟适配器。 |

## 5.3. 从 ISO 镜像创建基本虚拟机

按照本节中的步骤从 ISO 镜像创建基础镜像。

### 先决条件

按照 Virtualization 部署和管理指南，为您的 Red Hat Enterprise Linux 7 主机机器启用虚拟化。

### 5.3.1. 下载 ISO 镜像

#### 流程

1. 从红帽客户门户网站下载最新的 Red Hat Enterprise Linux ISO 镜像。
2. 将镜像移动到 **/var/lib/libvirt/images** 目录中。

### 5.3.2. 从 ISO 镜像创建虚拟机

#### 流程

1. 确保已为虚拟化启用主机机器。有关安装可升级软件包的信息和步骤，请参阅在现有 Red Hat Enterprise Linux 系统上安装虚拟化软件包
2. 创建并启动基本 Red Hat Enterprise Linux 虚拟机。有关创建虚拟机的说明，请参阅创建虚拟机。
   a. 如果使用命令行创建虚拟机，请确保将默认内存和 CPU 设置为您所需的容量。将您的虚拟网络接口设置为 **virtio**。
   下面是一个基本的命令行示例。

   ```
   virt-install --name _vmname_ --memory 2048 --vcpus 2 --disk size=8,bus=virtio --location rhel-7.0-x86_64-dvd.iso --os-variant=rhel7.0
   ```

   b. 如果您使用 virt-manager 应用程序创建虚拟机，请按照使用 virt-manager 创建虚拟机的步骤进行以下操作：
      - 不要选择 **Immediately Start VM**。
      - 将 **Memory** 和 **Storage Size** 设置为您需要的值。
      - 在开始安装前，请确保将 **Virtual Network Interface Settings** 中的 **Model** 更改为 **virtio**，并将您的 **vCPU** 更改为您想要的虚拟机容量设置。

### 5.3.3. 完成 RHEL 安装

执行以下步骤完成安装并在虚拟机启动后启用 root 访问。

#### 流程

1. 选择您要在安装过程中使用的语言。
2. 在 **Installation Summary** 视图中：
   a. 点 **Software Selection**，选择 **Minimal Install**。
   b. 点 **Done**。
   c. 点击 **Installation Destination** 并检查 **Storage Configuration** 中的 **Custom**。
      - 确保 **/boot** 验证至少有 500 MB。您可以使用剩余空间作为 root /。
      - 建议使用标准分区，但您也可以使用逻辑卷管理（LVM）。
      - 您可以将 xfs、ext4 或者 ext3 用于文件系统。
      - 完成更改后点 **Done**。
3. 点 **Begin Installation**。
4. 设置 **Root 密码**。
5. 重启虚拟机，然后在安装完成后以 **root** 身份登录。
6. 配置镜像。

**注意**

确定安装并启用 **cloud-init** 软件包。

7. 关闭虚拟机。

## 5.4. 将 RHEL 镜像上传到 GCP

按照本节中的步骤，在主机机器中将您的镜像上传到 GCP

### 5.4.1. 在 GCP 上创建新项目

完成以下步骤，在 GCP 上创建新项目。

**先决条件**

您必须使用 GCP 创建了一个帐户。如果没有，请参阅 Google Cloud。

**流程**

1. 启动 GCP 控制台。
2. 点击 **Google Cloud Platform** 右侧的下拉菜单。
3. 在弹出菜单中点击 **NEW PROJECT**。
4. 在 **New Project** 窗口中输入新项目的名称。
5. 选择 **Organization**。如果需要，点击下拉菜单更改机构。

6. 确认您的父机构或文件夹的 **位置**。如果需要，点 **Browse** 搜索并更改这个值。

7. 点击 **CREATE** 创建新 GCP 项目。

**注意**

安装 Google Cloud SDK 后,您可以使用 **gcloud projects create** CLI 命令创建项目。下面是一个简单的例子。

```
gcloud projects create my-gcp-project3 --name project3
```

这个示例创建项目 ID 为 **my-gcp-project3**，项目名称为 **project3**。如需更多信息，请参阅 gcloud 项目创建。

**其它资源**

创建和管理资源

### 5.4.2. 安装 Google Cloud SDK

完成以下步骤以安装 Google Cloud SDK。

**先决条件**

- 如果您还没有这样做，在 GCP 上创建一个项目。如需更多信息，请参阅在 Google Cloud Platform 上创建新项目。
- 确定您的主机系统包含 Python 2.7 或更高版本。如果没有安装，安装 Python 2.7。

**流程**

1. 按照下载和提取 Google Cloud SDK 归档的 GCP 说明。详情请查看 GCP 文档中的 Linux Quickstart。
2. 按照初始化 Google Cloud SDK 的说明。

**注意**

初始化 Google Cloud SDK 后，您可以使用 **gcloud** CLI 命令执行任务并获取有关项目和实例的信息。例如，您可以使用 **gcloud compute project-info describe --project <project-name>** 命令显示项目信息。

**其它资源**

- Linux 快速入门
- gcloud 命令参考
- gcloud 命令行工具概述

### 5.4.3. 为 Google Compute Engine 创建 SSH 密钥

执行以下步骤使用 GCE 生成并注册 SSH 密钥,以便您可以使用它的公共 IP 地址直接将 SSH 连接到实例。

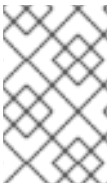
**流程**

1. 使用 **ssh-keygen** 命令生成用于 GCE 的 SSH 密钥对。

```
# ssh-keygen -t rsa -f ~/.ssh/google_compute_engine
```

2. 在 GCP Console Dashboard 页面 中,点击 Google **Cloud Console** 标题 左侧的 **Navigation** 菜单, 然后选择 **Compute Engine** 并选择 **Metadata**。
3. 点 **SSH Keys**，然后点 **Edit**。
4. 输入从 **~/.ssh/google_compute_engine.pub** 文件生成的输出并点 **Save**。
现在，您可以使用标准 SSH 连接到实例。

```
# ssh -i ~/.ssh/google_compute_engine <username>@<instance_external_ip>
```



**注意**

您可以运行 **gcloud compute config-ssh** 命令来使用实例的别名填充配置文件。别名允许按实例名称简单的 SSH 连接。有关 **gcloud compute config-ssh** 命令的详情，请参考 gcloud compute config-ssh。

**其它资源**

- gcloud 计算 config-ssh
- 连接到实例

### 5.4.4. 在 GCP Storage 中创建存储桶

导入到 GCP 需要 GCP Storage Bucket。完成以下步骤以创建存储桶。

**流程**

1. 如果您还没有登录到 GCP，请使用以下命令登录。

```
# gcloud auth login
```

2. 创建存储桶。

```
# gsutil mb gs://bucket_name
```

**注意**

另外，您可以使用 Google Cloud Console 创建存储桶。如需更多信息，请参阅创建存储桶。

**其它资源**

创建存储桶

### 5.4.5. 转换并上传您的镜像到您的 GCP 存储桶

完成以下步骤，将您的镜像转换并上传到您的 GCP 存储桶。示例是代表,它们将 **qcow2** 镜像转换为 **raw** 格式,然后将该镜像 tar 用于上传。

**流程**

1. 运行 **qemu-img** 命令转换您的镜像。转换的镜像必须具有名称 **disk.raw**。

```
# qemu-img convert -f qcow2 -O raw rhel-sample.qcow2 disk.raw
```

2. 打包镜像。

```
# tar --format=oldgnu -Sczf disk.raw.tar.gz disk.raw
```

3. 将镜像上传到之前创建的存储桶。上传可能需要几分钟时间。

```
# gsutil cp disk.raw.tar.gz gs://bucket_name
```

**验证步骤**

1. 在 **Google Cloud Platform** 主屏幕中点击折叠菜单图标并选择 **Storage**,然后选择 **Browser**。
2. 点存储桶的名称。
   打包的镜像列在存储桶名称下。

**注意**

您还可以使用 **GCP 控制台**上传您的镜像。要做到这一点,请点击存储桶的名称,然后点击 **Upload 文件**。

**其它资源**

- 手动导入虚拟磁盘
- 选择导入方法

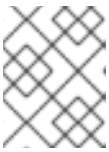
### 5.4.6. 从 GCP 存储桶中创建镜像

执行以下步骤从 GCP 存储桶中的对象创建镜像。

**流程**

- 运行以下命令来为 GCE 创建镜像。指定您要创建的镜像的名称、存储桶名称和打包的镜像的名称。

```
# gcloud compute images create my-image-name --source-uri gs://my-bucket-
name/disk.raw.tar.gz
```



**注意**

另外，您可以使用 Google Cloud Console 创建镜像。如需更多信息，请参阅创建、删除和弃用自定义镜像。

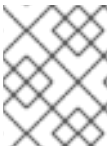
- 另外，还可在 GCP Console 中找到该镜像。
  a. 点击 **Google Cloud Console** 标题左侧的 **导航** 菜单。
  b. 选择 **Compute Engine ,然后选择 Images**。

**其它资源**

- 创建、删除和弃用自定义镜像
- gcloud 计算镜像创建

### 5.4.7. 从镜像创建 Google Compute Engine 实例

完成以下步骤，使用 GCP 控制台配置 GCE 虚拟机实例。



**注意**

以下流程提供了使用 GCP 控制台创建基本虚拟机实例的说明。如需有关 GCE 虚拟机实例及其配置选项的更多信息，，参阅创建并启动虚拟机实例。

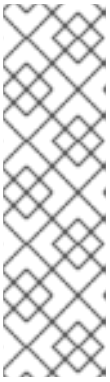
**流程**

1. 在 GCP Console Dashboard 页面 中,点击 Google **Cloud Console 标题** 左侧的 **Navigation** 菜单,选择 **Compute Engine**,然后选择 **Images**。
2. 选择您的镜像。
3. 点 **Create Instance**。
4. 在 **Create an instance** 页面中，为您的实例输入一个 **Name**。
5. 选择一个 **Region** 和 **Zone**。
6. 选择满足或超过工作负载要求的**机器配置**。
7. 确保**引导磁盘**指定了您的镜像名称。
8. （可选）在 **Firewall** 下,选择 **Allow HTTP 流量** 或 **Allow HTTPS 流量**。
9. 点 **Create**。

**注意**

这些是创建基本实例所需的最小配置选项。根据您的应用程序要求查看其他选项。

10. 在**虚拟机实例**中查找您的镜像。
11. 在 GCP Console Dashboard 中点击 Google **Cloud Console 标题** 左侧的 **Navigation** 菜单,选择 **Compute Engine**,然后选择 **VM 实例**。



**注意**

另外，您可以使用 **gcloud compute instances create** CLI 命令从镜像创建 GCE 虚拟机实例。下面是一个简单的例子。

```
gcloud compute instances create myinstance3 --zone=us-central1-a --image
test-iso2-image
```

这个示例基于现有镜像 **test-iso2-image** 在区 **us-central1-a** 中创建一个名为 **myinstance3** 的虚拟机实例。如需更多信息，请参阅 gcloud 计算实例创建。

### 5.4.8. 连接到您的实例

执行以下步骤使用其公共 IP 地址连接到 GCE 实例。

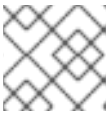
**流程**

1. 运行以下命令以确保您的实例正在运行。该命令列出 GCE 实例的信息,包括实例是否在运行,以及正在运行的实例的公共 IP 地址。

```
# gcloud compute instances list
```

2. 使用标准 SSH 连接到您的实例。这个示例使用之前创建的 **google_compute_engine** 密钥。

```
# ssh -i ~/.ssh/google_compute_engine <user_name>@<instance_external_ip>
```



**注意**

GCP 提供了多种 SSH 到您的实例的方法。如需更多信息，请参阅连接到实例。

**其它资源**

- gcloud 计算实例列表
- 连接到实例

### 5.4.9. 附加红帽订阅

完成以下步骤以附加您之前通过 Red Hat Cloud Access 程序启用的订阅。

**先决条件**

您必须已启用您的订阅。

**流程**

1. 注册您的系统。

```
subscription-manager register --auto-attach
```

2. 附加您的订阅。
   - 您可以使用激活码来附加订阅。请参阅创建红帽客户门户网站激活码。

- 或者，您可以使用订阅池（池 ID）的 ID 手动附加订阅。请参阅通过 command Line 来附加和删除订阅。

**其它资源**

- 创建红帽客户门户网站激活码
- 通过命令行附加和删除订阅
- 使用并配置 Red Hat Subscription Manager

# 第 6 章 在 GOOGLE CLOUD PLATFORM 上配置红帽高可用性集群

本章包含使用 Google Compute Engine(GCE)虚拟机(VM)实例在 Google Cloud Platform(GCP)上配置红帽高可用性(HA)集群的信息和步骤。

本章包含为 GCP 设置环境的先决条件步骤。设置环境后，您可以创建并配置 GCP VM 实例。

本章还包含与创建 HA 集群相关的流程，该集群将单个节点转换为 GCP 上的 HA 节点集群。这包括在每个集群节点上安装高可用性软件包和代理、配置隔离以及安装 GCP 网络资源代理的步骤。

本章在很多位置使用了 GCP 文档。如需更多信息，请参阅引用的 GCP 文档。

**先决条件**

- 您需要安装 GCP 软件开发组件(SDK)。如需更多信息，请参阅安装 Google cloud SDK 。
- 在 Red Hat Cloud Access 程序中启用您的订阅 。Red Hat Cloud Access 程序允许您在红帽的完全支持下将红帽订阅从物理或内部系统移到 GCP。
- 您必须属于活跃的 GCP 项目，并有足够的权限在项目中创建资源。
- 您的项目应具有属于虚拟机实例而非单独的用户的服务帐户。有关使用默认服务帐户而不是创建单独服务帐户的信息，请参阅使用 Compute Engine 默认服务帐户。

如果您或项目管理员创建自定义服务帐户，则应该为以下角色配置服务帐户。

- Cloud Trace Agent
- Compute Admin
- Compute Network Admin
- Cloud Datastore User
- Logging Admin
- Monitoring Editor
- Monitoring Metric Writer
- Service Account Administrator
- Storage Admin

**其它资源**

- Support Policies for RHEL High Availability Clusters - Google Cloud Platform Virtual Machines as Cluster Members
- Support Policies for RHEL High Availability clusters - Transport Protocols
- VPC 网络概述
- 查看 RHEL 高可用性的组件、概念和功能 - 传输概述
- RHEL 高可用性集群设计指南 - 选择传输协议

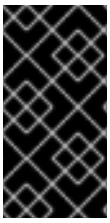
- Quickstart for Red Hat 和 Centos

## 6.1. GCP 上的 RED HAT ENTERPRISE LINUX 镜像选项

下表列出了镜像选择和镜像选项的不同。

**表 6.1. 镜像选项**

| 镜像选项 | 订阅 | 示例情境 | 注意事项 |
| --- | --- | --- | --- |
| 选择部署移至 GCP 的自定义镜像。 | 利用您现有的红帽订阅。 | 通过 Red Hat Cloud Access 程序启用订阅，上传您的自定义镜像并附加您的订阅。 | 订阅只包括红帽产品的成本；您还需要支付其他成本。<br><br>移至 GCP 的自定义镜像名为 "Cloud Access" 镜像，因为您利用了您现有的红帽订阅。红帽直接为 Cloud Access 镜像提供支持。 |
| 选择部署包含 RHEL 的现有 GCP 镜像。 | GCP 镜像包括一个红帽产品。 | 在 GCP Compute Engine 上启动实例时选择 RHEL 镜像，或者从 Google Cloud Platform Marketplace 中选择镜像。 | 根据 pay-as-you-go 模式每小时向 GCP 支付。这样的镜像称为 "on-demand" 镜像。GCP 通过支持协议支持 on-demand 镜像。 |



**重要**

您无法将 on-demand 实例转换为 Red Hat Cloud Access 实例。要从 on-demand 镜像改为 Red Hat Cloud Access bring-your-own subscription(BYOS)镜像,创建一个新的 Red Hat Cloud Access 实例,并从您的 on-demand 实例迁移数据。在迁移数据后取消您的 on-demand 实例以避免出现重复账单。

本章的剩余部分包含与自定义镜像相关的信息和流程。

**其它资源**

- 公共云中的红帽
- 镜像
- Red Hat Cloud Access 参考指南
- 从自定义镜像创建实例

## 6.2. 所需的系统软件包

本章的步骤假设您使用运行 Red Hat Enterprise Linux 的主机系统。要成功完成这些操作，主机系统必须安装以下软件包。

**表 6.2. 系统软件包**

| 软件包 | 描述 | 命令 |
| --- | --- | --- |
| qemu-kvm | 这个软件包提供用户级别的 KVM 模拟器，并可方便主机和客户机虚拟机间的通信。 | **# yum install qemu-kvm libvirt** |
| qemu-img | 这个软件包为客户机虚拟机提供磁盘管理。qemu-img 软件包作为 qemu-kvm 软件包的依赖项安装。 | |
| libvirt | 这个软件包为服务器和主机端提供与虚拟机监控程序、主机系统和主机系统以及处理库调用、管理虚拟机和控制虚拟机监控程序的 libvirtd 守护进程交互的服务器和主机库。 | |

**表 6.3. 其他虚拟化软件包**

| 软件包 | 描述 | 命令 |
| --- | --- | --- |
| virt-install | 这个软件包提供从命令行创建虚拟机的 **virt-install** 命令。 | **# yum install virt-install libvirt-python virt-manager virt-install libvirt-client** |
| libvirt-python | 这个软件包包含一个模块，它允许使用 Python 编程语言编写的应用程序使用 libvirt API 提供的接口。 | |
| virt-manager | 这个软件包提供 virt-manager 工具，也称为 Virtual Machine Manager（VMM）。VMM 是一个图形化工具用于管理虚拟机。它使用 libvirt-client 库作为管理 API。 | |
| libvirt-client | 这个软件包为访问 libvirt 服务器提供客户端 API 和库。libvirt-client 软件包包含 **virsh** 命令行工具,用于从命令行或特殊虚拟化 shell 管理和控制虚拟机和虚拟机监控程序。 | |

**其它资源**

- 手动安装虚拟化软件包

## 6.3. 安装 HA 软件包和代理

在所有节点上完成以下步骤，安装高可用性软件包和代理。

**流程**

1. 禁用所有软件仓库。

```
# subscription-manager repos --disable=*
```

2. 启用 RHEL 7 服务器和 RHEL 7 服务器 HA 软件仓库。

```
# subscription-manager repos --enable=rhel-7-server-rpms
# subscription-manager repos --enable=rhel-ha-for-rhel-7-server-rpms
```

3. 更新所有软件包。

```
# yum update -y
```

4. 安装 **pcs pacemaker** 隔离代理和资源代理。

```
# yum install -y pcs pacemaker fence-agents-gce resource-agents-gcp
```

5. 如果内核已更新，重启机器。

```
# reboot
```

## 6.4. 配置 HA 服务

在所有节点上完成以下步骤以配置高可用性服务。

**流程**

1. 用户 **hacluster** 在上一步中的 **pcs** 和 **pacemaker** 安装中创建。在所有集群节点上为用户 **hacluster** 创建密码。所有节点都使用相同的密码。

```
# passwd hacluster
```

2. 如果启用了 **firewalld** 服务，在 RHEL 中添加高可用性服务。

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

# firewall-cmd --reload
```

3. 启动 **pcs** 服务并在引导时启用它。

```
# systemctl enable pcsd.service --now
```

**验证步骤**

1. 确定 **pcs** 服务正在运行。

```
# systemctl is-active pcsd.service
```

## 6.5. 创建集群

完成以下步骤以创建节点集群。

**流程**

1. 在其中一个节点上，输入以下命令验证 pcs 用户 **ha cluster**。指定集群中的每个节点的名称。

```
# pcs cluster auth  _hostname1_ _hostname2_ _hostname3_ -u hacluster
```

例如：

```
[root@node01 ~]# pcs cluster auth node01 node02 node03 -u hacluster
node01: Authorized
node02: Authorized
node03: Authorized
```

2. 创建集群。

```
# pcs cluster setup --name cluster-name _hostname1_ _hostname2_  _hostname3_
```

**验证步骤**

1. 启用集群。

```
# pcs cluster enable --all
```

2. 启动集群。

```
# pcs cluster start --all
```

## 6.6. 创建隔离设备

对于大多数默认配置，GCP 实例名称和 RHEL 主机名是相同的。

完成以下步骤，从集群中的任何节点配置隔离。

**流程**

1. 从集群中的任何节点获取 GCP 实例名称。请注意，输出还显示实例的内部 ID。

```
# fence_gce --zone _gcp_ _region_ --project= _gcp_ _project_ -o list
```

例如：

```
[root@rhel71-node-01 ~]# fence_gce --zone us-west1-b --project=rhel-ha-testing-on-gcp -o
list
44358**********3181,InstanceName-3
40819**********6811,InstanceName-1
71736**********3341,InstanceName-2
```

2. 创建隔离设备。使用 **pcmk_host-name** 命令将 RHEL 主机名与实例 ID 映射。

```
# pcs stonith create _clusterfence_ fence_gce pcmk_host_map=_pcmk-hpst-map_
fence_gce zone=_gcp-zone_ project=_gcpproject_
```

例如：

```
[root@node01 ~]# pcs stonith create fencegce fence_gce
pcmk_host_map="node01:node01-vm;node02:node02-vm;node03:node03-vm"
project=hacluster zone=us-east1-b
```

**验证步骤**

1. 测试其他其中一个节点的隔离代理。

```
# pcs stonith fence gcp nodename
```

2. 检查状态以验证该节点是否已隔离。

```
# watch pcs status
```

例如：

```
[root@node01 ~]# watch pcs status
Cluster name: gcp-cluster
Stack: corosync
Current DC: rhel71-node-02 (version 1.1.18-11.el7_5.3-2b07d5c5a9) - partition with quorum
Last updated: Fri Jul 27 12:53:25 2018
Last change: Fri Jul 27 12:51:43 2018 by root via cibadmin on rhel71-node-01

3 nodes configured
3 resources configured

Online: [ rhel71-node-01 rhel71-node-02 rhel71-node-03 ]

Full list of resources:

us-east1-b-fence    (stonith:fence_gce):    Started rhel71-node-01

Daemon Status:
corosync: active/enabled
pacemaker: active/enabled
pcsd: active/enabled
```

## 6.7. 配置 GCP 节点授权

配置 cloud SDK 工具，使用您的帐户凭证访问 GCP。

**流程**

在每个节点上输入以下命令，使用项目 ID 和帐户凭证初始化每个节点。

```
# gcloud-ra init
```

## 6.8. 配置 GCP 网络资源代理

集群使用附加到二级 IP 地址（从 IP）的 GCP 网络资源代理到正在运行的实例。这是一个浮动 IP 地址，可在集群中的不同节点间传递。

### 流程

输入以下命令查看 GCP 虚拟 IP 地址资源代理(gcp-vpc-move-vip)描述。这显示了这个代理的选项和默认操作。

```
# pcs resource describe gcp-vpc-move-vip
```

您可以将资源代理配置为使用主子网地址范围或二级子网地址范围。本节包含了这两者的步骤。

#### 主子网地址范围

### 流程

完成以下步骤，为主 VPC 子网配置资源。

1. 创建 **aliasip** 资源。包括一个未使用的内部 IP 地址。在命令中包含 CIDR 块。

   ```
   # pcs resource create aliasip gcp-vpc-move-vip  alias_ip=_UnusedIPaddress/CIDRblock_ --group _group-name_ --group _networking-group_
   ```

2. 创建用于管理节点上 IP 的 **IPaddr2** 资源。

   ```
   # pcs resource create vip IPaddr2 nic=_interface_ ip=_AliasIPaddress_ cidr_netmask=32 --group _group-name_ --group _networking-group_
   ```

3. 对 **vipgrp** 下的网络资源进行分组。

   ```
   # pcs resource group add vipgrp aliasip vip
   ```

### 验证步骤

1. 验证资源是否已启动，并分组到 **vipgrp** 下。

   ```
   # pcs status
   ```

2. 验证资源是否可以移到另一个节点。

   ```
   # pcs resource move vip _Node_
   ```

   例如：

   ```
   # pcs resource move vip rhel71-node-03
   ```

3. 验证 **vip** 是否在不同节点上成功启动。

   ```
   # pcs status
   ```

#### 二级子网地址范围

完成以下步骤，为二级子网地址范围配置资源。

### 先决条件

创建自定义网络和子网

### 流程

1. 创建二级子网地址范围。

```
# gcloud-ra compute networks subnets update _SubnetName_ --region _RegionName_ --add-secondary-ranges _SecondarySubnetName_=_SecondarySubnetRange_
```

例如：

```
# gcloud-ra compute networks subnets update range0 --region us-west1 --add-secondary-ranges range1=10.10.20.0/24
```

2. 创建 **aliasip** 资源。在二级子网地址范围内创建一个未使用的内部 IP 地址。在命令中包含 CIDR 块。

```
# pcs resource create aliasip gcp-vpc-move-vip alias_ip=_UnusedIPaddress/CIDRblock_ --group _group-name_ --group _networking-group_
```

3. 创建用于管理节点上 IP 的 **IPaddr2** 资源。

```
# pcs resource create vip IPaddr2 nic=_interface_ ip=_AliasIPaddress_ cidr_netmask=32 --group _group-name_ --group _networking-group_
```

### 验证步骤

1. 验证资源是否已启动，并分组到 **vipgrp** 下。

```
# pcs status
```

2. 验证资源是否可以移到另一个节点。

```
# pcs resource move vip _Node_
```

例如：

```
[root@rhel71-node-01 ~]# pcs resource move vip rhel71-node-03
```

3. 验证 **vip** 是否在不同节点上成功启动。

```
# pcs status
```