Red Hat Enterprise Linux 9

在公共云平台上部署 Red Hat Enterprise Linux 9

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: 2022-08-08

Red Hat Enterprise Linux 9 在公共云平台上部署 Red Hat Enterprise Linux 9

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

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律通告

摘要

您可以将自定义的 Red Hat Enterprise Linux 镜像创建并部署到各种云平台上，包括 Microsoft Azure、Amazon Web Services(AWS)和 Google Cloud Platform(GCP)。您也可以在每个云平台上创建和配置红帽高可用性集群。本文档包括创建 HA 集群的流程，包括安装必要的软件包和代理、配置隔离以及安装网络资源代理。每个云供应商都有自己的章节来描述创建和部署自定义镜像。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

让开源更具包容性

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

对红帽文档提供反馈

我们感谢您对文档提供反馈信息。请让我们了解如何改进文档。

- 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。
- 要通过 Bugzilla 提交反馈，请创建一个新的 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 上将 RED HAT ENTERPRISE LINUX 镜像部署为虚拟机

要在 Microsoft Azure 上部署 Red Hat Enterprise Linux 9(RHEL 9)镜像，请遵循以下信息。本章：

- 讨论您选择镜像的选项
- 列出或引用主机系统和虚拟机(VM)的系统要求。
- 提供从 ISO 镜像创建自定义虚拟机、将其上传到 Azure 以及启动 Azure 虚拟机实例的流程

重要

您可以从 ISO 镜像创建自定义的虚拟机，但红帽建议您使用 *Red Hat Image Builder* 产品来创建自定义的镜像以用于特定的云供应商。使用 Image Builder，您可以创建并上传 Azure 磁盘镜像（VHD 格式）。如需更多信息，请参阅生成自定义 RHEL 系统镜像。

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。

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。

1.1. 其他资源

- 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

1.2. AZURE 上的 RED HAT ENTERPRISE LINUX 镜像选项

下表列出了 Microsoft Azure 上 RHEL 9 的镜像选项，并记录镜像选项中的区别。

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。	使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。 红帽黄金镜像称为"云访问"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。	使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。 您移到 Azure 的自定义镜像是"云访问"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。
选择部署一个包含 RHEL 的现有 Azure 镜像。	Azure 镜像包括一个红帽产品。	在使用 Azure 控制台创建虚拟机时，或从 Azure Marketplace 中选择一个虚拟机时，请选择 RHEL 镜像。	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。 红帽提供了镜像的更新。Azure 通过红帽更新基础架构(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，请参阅生成自定义 RHEL 系统镜像。

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。

其他资源

- 在 Microsoft Azure 上使用红帽黄金镜像
- Red Hat Cloud Access 程序
- Azure Marketplace
- Azure Marketplace 中的账单选项
- Red Hat Enterprise Linux 在 Azure 中提供您自己的订阅黄金镜像

1.3. 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。

1.3.1. 使用自定义基础镜像

要手动配置虚拟机(VM)，首先创建一个基础（起步）虚拟机镜像。然后，您可以修改配置设置，并添加 VM 在云上操作所需的软件包。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 的云镜像，请按照以下部分中的说明操作。要准备 RHEL 的 Hyper-V 云镜像，请参阅 从 Hyper-V Manager 准备基于红帽的虚拟机。

1.3.2. 所需的系统软件包

本章中的流程假设您使用运行 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	软件仓库	描述
libvirt	rhel-9-for-x86_64-appstream-rpms	用于管理平台虚拟化的开源 API、守护进程和管理工具
virt-install	rhel-9-for-x86_64-appstream-rpms	用于构建虚拟机的命令行工具
libguestfs	rhel-9-for-x86_64-appstream-rpms	用于访问和修改虚拟机文件系统的库
guestfs-tools	rhel-9-for-x86_64-appstream-rpms	虚拟机的系统管理工具；包括 **virt-customize** 实用程序

1.3.3. 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。在进行操作时请记住这些设置。如有必要，请参阅它们。

表 1.3. 虚拟机配置设置

设置	建议
ssh	必须启用 SSH 来提供对 Azure 虚拟机的远程访问。
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)配置交换空间。
NIC	为主虚拟网络适配器选择 **virtio**。

设置	建议
encryption	对于自定义镜像，使用 Network Bound Disk Encryption(NBDE)在 Azure 上进行全磁盘加密。

1.3.4. 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。

先决条件

- 确保已为虚拟化启用主机机器。有关信息和流程，请参阅在 RHEL 9 中启用虚拟化。

步骤

1. 从红帽客户门户网站下载最新的 Red Hat Enterprise Linux 9 DVD ISO 镜像。
2. 创建并启动基本 Red Hat Enterprise Linux 虚拟机。有关说明，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 kvmtest --memory 2048 --vcpus 2 --disk rhel-9.0-x86_64-kvm.qcow2,bus=virtio --import --os-variant=rhel9.0
   ```

   b. 如果使用 Web 控制台来创建虚拟机，请按照 使用 web 控制台创建虚拟机 中的流程进行操作，包括以下注意事项：
      - 不要选择 **Immediately Start VM**。
      - 将 **Memory** 大小更改为你希望的设置。
      - 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. 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，在 **boot** 和 **root** 分区中使用 xfs、ext4 或 ext3。
     - 删除 swap 空间。交换空间通过 WALinuxAgent 在 Azure 上的物理刀片服务器上配置。

- 在 **Installation Summary** 屏幕中，选择 **Network and Host Name**。将 **Ethernet** 切换到 **On**。

4. 安装开始：
   - 创建 **root** 密码.
   - 创建管理用户帐户。
5. 安装完成后，重启虚拟机并登录到 root 帐户。
6. 以 **root** 身份登录后，您就可以配置镜像了。

1.4. 为 MICROSOFT AZURE 配置基础镜像

基础镜像需要配置更改才能作为 Azure 中的 RHEL 9 虚拟机镜像。以下小节提供 Azure 所需的其他配置更改。

1.4.1. 安装 Hyper-V 设备驱动程序

Microsoft 提供了网络和存储设备驱动程序，作为其 Hyper-V 软件包的 Linux 集成服务(LIS)的一部分。在将虚拟机镜像配置为 Azure虚拟机（VM）之前， Hyper-V 可能需要在其上安装 Hyper-V 设备驱动程序。使用 **lsinitrd | grep hv** 命令来验证是否已安装了驱动程序。

步骤

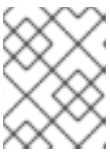
1. 输入以下 **grep** 命令，来确定是否已安装了所需的 Hyper-V 设备驱动程序：

```
# 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.el9.x86_64/kernel/drivers/hv
-rw-r--r--   1 root     root        31272 Aug 11 08:45 usr/lib/modules/3.10.0-
932.el9.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.el9.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.el9.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

环境中可能存在 **hv_vmbus** 驱动程序。即使存在这个驱动程序，请完成以下步骤。

2. 在 **/etc/dracut.conf.d** 中创建一个名为 **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 "
add_drivers+=" nvme "
```

注意

请注意引号前后的空格，例如 **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驱动程序。

4. 重新生成 **initramfs** 镜像。

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

验证

1. 重启机器。
2. 运行 **lsinitrd | grep hv** 命令来验证是否已安装了驱动程序。

1.4.2. 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 9 软件仓库。

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

3. 停止并删除 **cloud-init**
   a. 停止 **cloud-init** 服务（如果存在的话）。

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

   b. 删除 **cloud-init** 软件。

```
# dnf remove cloud-init
```

4. 完成其他虚拟机更改

a. 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 9 DVD ISO 镜像中没有 **ifcfg-eth0** 文件，必须创建。

```
DEVICE="eth0"
ONBOOT="yes"
BOOTPROTO="dhcp"
```

```
TYPE="Ethernet"
USERCTL="yes"
PEERDNS="yes"
IPV6INIT="no"
```

b. 如果存在，请删除以下持久性网络设备规则。

```
# 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
```

c. 创建一个新的网络设备规则 **/etc/udev/rules.d/68-azure-sriov-nm-unmanaged.rules**，并将以下行添加到其中：

```
SUBSYSTEM=="net", DRIVERS=="hv_pci", ACTION=="add", ENV{NM_UNMANAGED}="1"
```

d. 将 **ssh** 设为自动启动。

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

e. 修改内核引导参数。

i. 打开 **/etc/default/grub** 文件，并确保 **GRUB_TIMEOUT** 行具有以下值：

```
GRUB_TIMEOUT=10
```

ii. 如果存在，从 **GRUB_CMDLINE_LINUX** 行的末尾删除以下选项。

```
rhgb quiet
```

iii. 确保 **GRUB_CMDLINE_LINUX** 行包含下列所有选项：

```
GRUB_CMDLINE_LINUX="loglevel=3 crashkernel=auto console=tty1 console=ttyS0
earlyprintk=ttyS0 rootdelay=300"
```

iv. 在 **/etc/default/grub** 文件末尾添加以下行（如果不存在）。

```
GRUB_TIMEOUT_STYLE=countdown
GRUB_TERMINAL="serial console"
GRUB_SERIAL_COMMAND="serial --speed=115200 --unit=0 --word=8 --parity=no --
stop=1"
```

f. 重新生成 **grub.cfg** 文件。

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

g. 安装并启用 Windows Azure Linux Agent(WALinuxAgent)。Red Hat Enterprise Linux 9 Application Stream(AppStream)包括 WALinuxAgent。有关 AppStream 的更多信息，请参阅 Application stream。

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

h. 编辑 **/etc/waagent.conf** 文件中的以下行，以便为提供的虚拟机配置交换空间。为您置备的虚拟机设置 swap 空间。

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

1. 准备提供

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

```
# subscription-manager unregister
```

j. 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。这个命令会生成警告，这是预期的。

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

k. 清理 shell 历史记录并关闭虚拟机。

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

1.5. 将镜像转换为固定 VHD 格式

所有 Microsoft Azure VM 镜像都必须是固定的 **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。本节描述了如何将镜像从 **qcow2** 转换为固定的 **VHD** 格式，并在需要时对镜像进行调整。转换镜像后，您可以将其上传到 Azure。

步骤

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。本例使用名称 **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。

4. 使用以下命令来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。

5. 如果 **raw** 镜像不一致，请完成以下步骤使其保持一致。
   a. 使用运行验证脚本时显示的循环值，来调整 **raw** 文件的大小。

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

   b. 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。

1.6. 安装 AZURE CLI

完成以下步骤来安装 Azure 命令行界面(Azure CLI 2.1)。Azure CLI 2.1 是一个基于 Python 的工具，用来在 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 the **dnf** 软件包索引。

```
$ dnf check-update
```

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

```
$ sudo dnf install python3
```

5. 安装 Azure CLI。

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

6. 运行 Azure CLI。

```
$ az
```

其它资源

- Azure CLI
- Azure CLI 命令参考

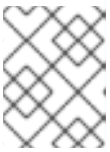
1.7. 在 AZURE 中创建资源

在上传 **VHD** 文件并创建 Azure 镜像之前，请完成以下流程来创建所需的 Azure 资源。

步骤

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：

```
[clouduser@localhost]$ 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：

```
[clouduser@localhost]$ 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;Endpoi
ntSuffix=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 Managed Disks 概述
- SKU 类型

1.8. 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。

步骤

1. 将 **VHD** 文件上传到存储容器。它可能需要几分钟时间。要获取存储容器的列表，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
```

Example:

```
[clouduser@localhost]$ az storage blob upload --account-name azrhelclistact --container-
name azrhelclistcont --type page --file rhel-image-9.vhd --name rhel-image-9.vhd
Percent complete: %100.0
```

2. 获取上传的 **VHD** 文件的 URL，以便在下面的步骤中使用。

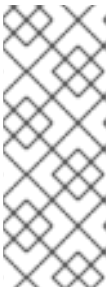
```
$ az storage blob url -c <container-name> -n <image-name>.vhd
```

Example:

```
$ az storage blob url -c azrhelclistcont -n rhel-image-9.vhd
"https://azrhelclistact.blob.core.windows.net/azrhelclistcont/rhel-image-9.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。您可以通过包含 **--hyper-v-generation V2** 选项来（可选）指定 V2 管理程序生成。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。

命令可能会返回错误"Only blobs formatted as VHDs can be imported." 此错误可能意味着在转换为 **VHD** 之前，镜像与最接近的 1 MB 边界不一致。

Example:

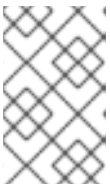
```
$ az image create -n rhel9 -g azrhelclirsgrp2 -l southcentralus --source
https://azrhelclistact.blob.core.windows.net/azrhelclistcont/rhel-image-9.vhd --os-type
linux
```

1.9. 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。



注意

选项 **--generate-ssh-keys** 创建私钥/公钥对。私钥和公钥文件创建在您系统上的 **~/.ssh** 中。公钥被添加到 **--admin-username** 选项所指定的用户的虚拟机上的 **authorized_keys** 文件中。如需更多信息，请参阅其他身份验证方法 。

```
$ 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>
```

Example:

```
[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 rhel9

{
  "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"
```

请注意 **publicIpAddress**。在以下步骤中，您需要这个地址来登录到虚拟机。

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.

[clouduser@rhel-azure-vm-1 ~]$
```

如果您看到用户提示，则代表成功部署了 Azure 虚拟机。

现在，您可以进到 Microsoft Azure 门户网站，并检查审计日志和资源的属性。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**，或参阅 Azure CLI 命令参考 来了解更多有关您在 Microsoft Azure 中管理虚拟机的命令。

1.10. 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：这些命令选项置备新虚拟机而不生成公钥文件。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，并使用现有的公钥文件进行 SSH 身份验证。

```
$ 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>
```

1.11. 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。如需更多信息，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。请参阅通过命令行附加和删除订阅。

其它资源

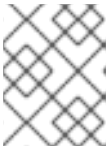
- 创建红帽客户门户网站激活码
- 通过命令行附加和删除订阅
- 使用并配置 Red Hat Subscription Manager

1.12. 对 AZURE 黄金镜像设置自动注册

要在 Micorsoft Azure 中更快地部署 RHEL 9 虚拟机，您可以设置 RHEL 9 的 Gold 镜像来自动注册到 Red Hat Subscription Manager(RHSM)中。

先决条件

- 您已在 Azure 上为 Cloud Access 启用了有效的红帽订阅，因此可在 Microsoft Azure 中使用 RHEL 9 Gold Images。具体说明请查看 在 Azure 上使用黄金镜像。



注意

Microsoft Azure 帐户一次只能附加到一个红帽帐户。因此，将其附加到您的红帽帐户之前，请确保其他用户不需要访问 Azure 帐户。

步骤

1. 使用 Gold Image 在 Azure 实例中创建 RHEL 9 虚拟机。具体步骤请参阅 在 Azure 中创建和启动虚拟机。
2. 启动创建的虚拟机。
3. 在 RHEL 9 虚拟机中，启用自动注册。

```
# subscription-manager config --rhsmcertd.auto_registration=1
```

4. 启用 **rhsmcertd** 服务。

```
# systemctl enable rhsmcertd.service
```

5. 禁用 **redhat.repo** 存储库。

```
# subscription-manager config --rhsm.manage_repos=0
```

6. 关闭虚拟机，并将它保存为 Azure 上的受管镜像。具体说明请查看 如何创建虚拟机的受管镜像或 VHD。

7. 使用受管映像创建虚拟机。他们将自动订阅 RHSM。

验证

- 在使用上述说明创建的 RHEL 9 虚拟机中，执行 **subscription-manager 身份** 命令验证系统是否注册到 RHSM。在成功注册的系统上，这会显示系统的 UUID。例如：

```
# subscription-manager identity
system identity: fdc46662-c536-43fb-a18a-bbcb283102b7
name: 192.168.122.222
org name: 6340056
org ID: 6340056
```

其他资源

- Azure 中的红帽黄金镜像
- Azure 中 RHEL 镜像概述
- 为红帽服务配置云资源

第 2 章 在 MICROSOFT AZURE 上配置红帽高可用性集群

本章提供了在 Azure 上配置红帽高可用性(HA)集群的信息和程序，其中使用 Azure 虚拟机(VM)实例作为集群节点。本章中的流程假设您要为 Azure 创建自定义镜像。您可以使用多个选项来获取用于集群的 RHEL 9 镜像。如需有关 Azure 镜像选项的信息，请参阅 Azure 上的 Red Hat Enterprise Linux Image 选项 。

本章包括：

- 为 Azure 设置环境的先决条件的流程。设置完环境后，您可以创建并配置 Azure 虚拟机实例。
- 本章还包括特定于创建 HA 集群的流程，该集群将各个节点转换到 Azure 上的 HA 节点集群之中。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 Azure 文档。如需了解更多信息，请参阅引用的 Azure 文档。

先决条件

- 注册 红帽客户门户网站帐户 。
- 注册具有管理员特权的 Microsoft Azure 帐户。
- 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。
  - 另外，也可使用 Azure Marketplace 按需获取 RHEL 镜像。

2.1. 其他资源

- RHEL 高可用性集群的支持政策 - 作为集群成员的 Microsoft Azure 虚拟机
- 配置和管理高可用性集群

2.2. 在 AZURE 中创建资源

完成以下流程来创建区域、资源组、存储帐户、虚拟网络和可用性集。您需要这些资源才能完成本章中的后续任务。

流程

1. 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。

例如：

```
[clouduser@localhost]$ az login

To sign in, use a web browser to open the page https://aka.ms/devicelogin and enter the code
FDMSCMETZ to authenticate.
 [
  {
   "cloudName": "AzureCloud",
   "id": "Subscription ID",
   "isDefault": true,
   "name": "MySubscriptionName",
   "state": "Enabled",
   "tenantId": "Tenant ID",
   "user": {
    "name": "clouduser@company.com",
    "type": "user"
   }
  }
 ]
```

2. 在 Azure 区域中创建资源组。

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

例如：

```
[clouduser@localhost]$ 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。

```
$ az storage account create -l azure-region -n storage-account-name -g resource-
group --sku sku_type --kind StorageV2
```

例如：

```
[clouduser@localhost]$ az storage account create -l southcentralus -n azrhelclistact -g
azrhelclirsgrp --sku Standard_LRS --kind StorageV2

{
  "accessTier": null,
  "creationTime": "2017-04-05T19:10:29.855470+00:00",
  "customDomain": null,
  "encryption": null,
  "id":
```

```
"/subscriptions//resourceGroups/azrhelclirsgrp/providers/Microsoft.Storage/storageAccounts/azrhelclistact",
  "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;Endpoi
ntSuffix=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。所有群集节点必须位于同一个虚拟网络中。

```
$ 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/azrhelclivnet1",
  "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/azrhelclivnet1/subnets/azrhelclisubnet1",
      "ipConfigurations": null,
      "name": "azrhelclisubnet1",
      "networkSecurityGroup": null,
      "provisioningState": "Succeeded",
      "resourceGroup": "azrhelclirsgrp",
      "resourceNavigationLinks": null,
      "routeTable": null
    }
  ],
  "tags": {},
```

```
  "type": "Microsoft.Network/virtualNetworks",
  "virtualNetworkPeerings": null
  }
}
```

8. 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

2.3. 高可用性所需的系统软件包

这个步骤假设您要使用 Red Hat Enterprise Linux 为 Azure HA 创建虚拟机镜像。要成功完成这个过程，必须安装以下软件包。

表 2.1. 系统软件包

软件包	软件仓库	描述
libvirt	rhel-9-for-x86_64-appstream-rpms	用于管理平台虚拟化的开源 API、守护进程和管理工具
virt-install	rhel-9-for-x86_64-appstream-rpms	用于构建虚拟机的命令行工具
libguestfs	rhel-9-for-x86_64-appstream-rpms	用于访问和修改虚拟机文件系统的库

软件包	软件仓库	描述
guestfs-tools	rhel-9-for-x86_64-appstream-rpms	虚拟机的系统管理工具；包括 **virt-customize** 实用程序

2.4. 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。在进行操作时请记住这些设置。如有必要，请参阅它们。

表 2.2. 虚拟机配置设置

设置	建议
ssh	必须启用 SSH 来提供对 Azure 虚拟机的远程访问。
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)配置交换空间。
NIC	为主虚拟网络适配器选择 **virtio**。
encryption	对于自定义镜像，使用 Network Bound Disk Encryption(NBDE)在 Azure 上进行全磁盘加密。

2.5. 安装 HYPER-V 设备驱动程序

Microsoft 提供了网络和存储设备驱动程序，作为其 Hyper-V 软件包的 Linux 集成服务(LIS)的一部分。在将虚拟机镜像配置为 Azure虚拟机（VM）之前， Hyper-V 可能需要在其上安装 Hyper-V 设备驱动程序。使用 **lsinitrd | grep hv** 命令来验证是否已安装了驱动程序。

步骤

1. 输入以下 **grep** 命令，来确定是否已安装了所需的 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.el9.x86_64/kernel/drivers/hv
-rw-r--r--   1 root     root        31272 Aug 11 08:45 usr/lib/modules/3.10.0-
932.el9.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.el9.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.el9.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

环境中可能存在 **hv_vmbus** 驱动程序。即使存在这个驱动程序，请完成以下步骤。

2. 在 **/etc/dracut.conf.d** 中创建一个名为 **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 "
add_drivers+=" nvme "
```

注意

请注意引号前后的空格，例如 **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驱动程序。

4. 重新生成 **initramfs** 镜像。

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

验证

1. 重启机器。
2. 运行 **lsinitrd | grep hv** 命令来验证是否已安装了驱动程序。

2.6. 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 9 软件仓库。

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

3. 停止并删除 **cloud-init**
   a. 停止 **cloud-init** 服务（如果存在的话）。

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

b. 删除 **cloud-init** 软件。

```
# dnf remove cloud-init
```

4. 完成其他虚拟机更改

a. 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 9 DVD ISO 镜像中没有 **ifcfg-eth0** 文件，必须创建。

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

b. 如果存在，请删除以下持久性网络设备规则。

```
# 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
```

c. 创建一个新的网络设备规则 **/etc/udev/rules.d/68-azure-sriov-nm-unmanaged.rules**，并将以下行添加到其中：

```
SUBSYSTEM=="net", DRIVERS=="hv_pci", ACTION=="add", ENV{NM_UNMANAGED}="1"
```

d. 将 **ssh** 设为自动启动。

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

e. 修改内核引导参数。

i. 打开 **/etc/default/grub** 文件，并确保 **GRUB_TIMEOUT** 行具有以下值：

```
GRUB_TIMEOUT=10
```

ii. 如果存在，从 **GRUB_CMDLINE_LINUX** 行的末尾删除以下选项。

```
rhgb quiet
```

iii. 确保 **GRUB_CMDLINE_LINUX** 行包含下列所有选项：

```
GRUB_CMDLINE_LINUX="loglevel=3 crashkernel=auto console=tty1 console=ttyS0
earlyprintk=ttyS0 rootdelay=300"
```

iv. 在 **/etc/default/grub** 文件末尾添加以下行（如果不存在）。

```
GRUB_TIMEOUT_STYLE=countdown
GRUB_TERMINAL="serial console"
GRUB_SERIAL_COMMAND="serial --speed=115200 --unit=0 --word=8 --parity=no --stop=1"
```

f. 重新生成 **grub.cfg** 文件。

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

g. 安装并启用 Windows Azure Linux Agent(WALinuxAgent)。Red Hat Enterprise Linux 9 Application Stream(AppStream)包括 WALinuxAgent。有关 AppStream 的更多信息，请参阅 Application stream。

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

h. 编辑 **/etc/waagent.conf** 文件中的以下行，以便为提供的虚拟机配置交换空间。为您置备的虚拟机设置 swap 空间。

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

1. 准备提供

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

```
# subscription-manager unregister
```

j. 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。这个命令会生成警告，这是预期的。

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

k. 清理 shell 历史记录并关闭虚拟机。

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

2.7. 创建 AZURE ACTIVE DIRECTORY 应用程序

完成以下流程来创建 Azure Active Directory AD Application。Azure AD Application Views 和自动访问集群中所有节点的 HA 操作。

先决条件

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，请在命令中添加 **--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

2.8. 将镜像转换为固定 VHD 格式

所有 Microsoft Azure VM 镜像都必须是固定的 **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。本节描述了如何将镜像从 **qcow2** 转换为固定的 **VHD** 格式，并在需要时对镜像进行调整。转换镜像后，您可以将其上传到 Azure。

步骤

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。本例使用名称 **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。

4. 使用以下命令来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。

5. 如果 **raw** 镜像不一致，请完成以下步骤使其保持一致。

a. 使用运行验证脚本时显示的循环值，来调整 **raw** 文件的大小。

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

b. 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。

2.9. 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。

步骤

1. 将 **VHD** 文件上传到存储容器。它可能需要几分钟时间。要获取存储容器的列表，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
```

Example:

```
[clouduser@localhost]$ az storage blob upload --account-name azrhelclistact --container-name azrhelclistcont --type page --file rhel-image-9.vhd --name rhel-image-9.vhd
Percent complete: %100.0
```

2. 获取上传的 **VHD** 文件的 URL，以便在下面的步骤中使用。

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

Example:

```
$ az storage blob url -c azrhelclistcont -n rhel-image-9.vhd
"https://azrhelclistact.blob.core.windows.net/azrhelclistcont/rhel-image-9.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。您可以通过包含 **--hyper-v-generation V2** 选项来（可选）指定 V2 管理程序生成。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。

命令可能会返回错误"Only blobs formatted as VHDs can be imported." 此错误可能意味着在转换为 **VHD** 之前，镜像与最接近的 1 MB 边界不一致。

Example:

```
$ az image create -n rhel9 -g azrhelclirsgrp2 -l southcentralus --source https://azrhelclistact.blob.core.windows.net/azrhelclistcont/rhel-image-9.vhd --os-type linux
```

2.10. 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. 启动 SSH 终端会话，并使用管理员名称和公共 IP 地址连接到虚拟机。

```
$ ssh administrator@PublicIP
```

要获取 Azure 虚拟机的公共 IP 地址，请在 Azure 门户网站中打开虚拟机属性，或者输入以下 Azure CLI 命令。

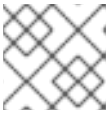
```
$ az vm list -g <resource-group> -d --output table
```

例如：

```
[clouduser@localhost ~] $ az vm list -g azrhelclirsgrp -d --output table
Name    ResourceGroup          PowerState      PublicIps       Location
------  ---------------------  --------------  -------------   --------------
node01  azrhelclirsgrp         VM running      192.98.152.251  southcentralus
```

2. 在红帽注册虚拟机。

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



注意

如果 **--auto-attach** 命令失败，请手动将虚拟机注册到您的订阅。

3. 禁用所有软件仓库。

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

4. 启用 RHEL 9 服务器 HA 软件仓库。

```
# subscription-manager repos --enable=rhel-9-for-x86_64-highavailability-rpms
```

5. 更新所有软件包。

```
# dnf update -y
```

6. 安装红帽高可用性附加组件软件包，以及高可用性通道的所有可用的隔离代理。

```
# dnf install pcs pacemaker fence-agents-azure-arm
```

7. 用户 **hacluster** 是在上一步中的 pcs 和 pacemaker 安装过程中创建的。在所有群集节点上为 **hacluster** 创建密码。所有节点都使用相同的密码。

```
# passwd hacluster
```

8. 如果安装了 **firewalld.service**，请在 RHEL Firewall 中添加 **high availability** 服务。

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

9. 启动 **pcs** 服务，并使其在引导时启动。

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

```
Created symlink from /etc/systemd/system/multi-user.target.wants/pcsd.service to
/usr/lib/systemd/system/pcsd.service.
```

验证

- 确保 **pcs** 服务正在运行。

```
# systemctl status pcsd.service
pcsd.service - PCS GUI and remote configuration interface
Loaded: loaded (/usr/lib/systemd/system/pcsd.service; enabled; vendor preset: disabled)
Active: active (running) since Fri 2018-02-23 11:00:58 EST; 1min 23s ago
Docs: man:pcsd(8)
        man:pcs(8)
Main PID: 46235 (pcsd)
  CGroup: /system.slice/pcsd.service
          └─46235 /usr/bin/ruby /usr/lib/pcsd/pcsd > /dev/null &
```

2.11. 创建集群

完成以下步骤以创建节点集群。

步骤

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

```
# pcs host auth hostname1 hostname2 hostname3
Username: hacluster
Password:
hostname1: Authorized
hostname2: Authorized
hostname3: Authorized
```

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 cluster-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

1. 启用集群。

```
[root@node01 clouduser]# pcs cluster enable --all
```

2. 启动集群。

```
[root@node01 clouduser]# 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...
```

2.12. 隔离（FENCING）概述

如果与集群中某个节点通信失败，那么集群中的其他节点必须能够限制或释放对故障集群节点可访问的资源的访问。这无法通过通过联系集群节点本身来实现，因为集群节点可能没有响应。反之，必须提供一个外部的方法来实现。这个方法为称为隔离（fencing）。

不响应的节点可能仍然在访问数据。确定您的数据安全的唯一方法是使用 STONITH 保护节点。STONITH 是"Shoot The Other Node In The Head"的缩写，它保护您的数据不受有问题的节点或并发访问的影响。使用 STONITH 可以确保，在允许从另一个节点访问数据前确定节点真正离线。

其它资源

- 在 Red Hat High Availability Cluster 中进行隔离

2.13. 创建隔离设备

完成以下步骤来配置隔离。在集群中的任何节点中完成这些命令

先决条件

您需要将集群属性**stonith-enabled** 设为 **true**。

步骤

步骤

1. 识别每个 RHEL 虚拟机的 Azure 节点名称。您可以使用 Azure 节点名称来配置隔离设备。

```
# fence_azure_arm -l AD-Application-ID -p AD-Password --resourceGroup
MyResourceGroup --tenantId Tenant-ID --subscriptionId Subscription-ID -o list
```

例如：

```
[root@node01 clouduser]# fence_azure_arm -l e04a6a49-9f00-xxxx-xxxx-a8bdda4af447 -
p z/a05AwCN0IzAjVwXXXXXXXEWIoeVp0xg7QT//JE= --resourceGroup azrhelclirsgrp --
tenantId 77ecefb6-cff0-XXXX-XXXX-757XXXX9485 --subscriptionId XXXXXXXX-38b4-
4527-XXXX-012d49dfc02c -o list
node01,
node02,
node03,
```

2. 查看 Azure ARM STONITH 代理的选项。

```
# pcs stonith describe fence_azure_arm
```

例如：

```
# pcs stonith describe fence_apc
Stonith options:
password: Authentication key
password_script: Script to run to retrieve password
```

警告

对于提供方法选项的隔离代理，请不要指定循环值，因为它不被支持，并可能导致数据崩溃。

有些隔离设备只能隔离一个节点，其他设备则可能隔离多个节点。您创建隔离设备时指定的参数取决于您的隔离设备的支持和要求。

您可以在创建隔离设备时使用 **pcmk_host_list** 参数，以指定由该隔离设备控制的所有机器。

在创建隔离设备时，您可以使用 **pcmk_host_map** 参数将主机名映射到包含隔离设备的规范。

3. 创建隔离设备。

```
# pcs stonith create clusterfence fence_azure_arm
```

4. 测试其他其中一个节点的隔离代理。

```
# pcs stonith fence azurenodename
```

例如：

```
[root@node01 clouduser]# pcs status
Cluster name: newcluster
Stack: corosync
Current DC: node01 (version 1.1.18-11.el7-2b07d5c5a9) - partition with quorum
Last updated: Fri Feb 23 11:44:35 2018
Last change: Fri Feb 23 11:21:01 2018 by root via cibadmin on node01

3 nodes configured
1 resource configured

Online: [ node01 node03 ]
OFFLINE: [ node02 ]

Full list of resources:

 clusterfence  (stonith:fence_azure_arm):  Started node01

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

5. 启动上一步中隔离的节点。

```
# pcs cluster start hostname
```

6. 检查状态以验证节点已启动。

```
# pcs status
```

例如：

```
[root@node01 clouduser]# pcs status
Cluster name: newcluster
Stack: corosync
Current DC: node01 (version 1.1.18-11.el7-2b07d5c5a9) - partition with quorum
Last updated: Fri Feb 23 11:34:59 2018
Last change: Fri Feb 23 11:21:01 2018 by root via cibadmin on node01

3 nodes configured
1 resource configured

Online: [ node01 node02 node03 ]

Full list of resources:

clusterfence    (stonith:fence_azure_arm):  Started node01

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

其它资源

- 在 Red Hat High Availability Cluster 中进行隔离
- 隔离设备的常规属性

2.14. 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，并包含用于自定义 HA 负载均衡器的设置。

先决条件

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（如 SAP HANA 和 SQL Server），您可能需要与 Microsoft 合作以确定要使用的正确端口。
4. 创建负载均衡规则。要创建负载均衡规则，默认值会预先填充。确保将 **Floating IP (direct server return)** 设置为 **Enabled**。

2.15. 配置负载均衡资源代理

创建健康探测后，您必须配置 **负载均衡** 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。

步骤

1. 在所有节点上安装 **nmap-ncat** 资源代理。

```
# dnf install nmap-ncat resource-agents
```

   在单个节点上执行以下步骤。

2. 创建 **pcs** 资源和组。将负载均衡器 FrontendIP 用于 IPaddr2 地址。

```
# pcs resource create resource-name IPaddr2 ip="10.0.0.7" --group cluster-resources-group
```

3. 配置 **负载平衡** 资源代理。

```
# pcs resource create resource-loadbalancer-name azure-lb port=port-number --group cluster-resources-group
```

验证

- 运行 **pcs status** 来查看结果。

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

输出示例：

```
Cluster name: clusterfence01
Stack: corosync
Current DC: node02 (version 1.1.16-12.el7_4.7-94ff4df) - partition with quorum
Last updated: Tue Jan 30 12:42:35 2018
Last change: Tue Jan 30 12:26:42 2018 by root via cibadmin on node01

3 nodes configured
3 resources configured

Online: [ node01 node02 node03 ]

Full list of resources:

clusterfence (stonith:fence_azure_arm):      Started node01
Resource Group: g_azure
    vip_azure  (ocf::heartbeat:IPaddr2):       Started node02
    lb_azure   (ocf::heartbeat:azure-lb):      Started node02

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

2.16. 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。这个流程假设三个带有 1 TB 共享磁盘的 Azure 虚拟机（一个三节点集群）。

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。
- 您必须在 Azure 中创建了集群环境，其中包括创建以下资源。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

步骤

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. 验证您已使用 Azure 命令 **az disk show** 创建了共享块卷。

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

例如，以下命令显示资源组 **sharedblock-rg** 中共享块卷 **shared-block-volume.vhd** 的详细信息。

```
$ 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。对每个网络接口使用不同的 **<nic_name>** 运行以下命令三次。

```
$ 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** 来连接共享块卷。每个虚拟机的选项值都相同，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，以下命令创建名为 **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-9.3.0-20200713.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

1. 对于集群中的每个虚拟机，使用带有 VM **<ip_address>** 的 **ssh** 命令来验证块设备是否可用。

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

例如，以下命令列出了虚拟机 IP **198.51.100.3** 的详细信息，其中包括主机名和块设备。

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

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

2. 使用 **ssh** 命令来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> "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 IP 地址为 **198.51.100.3** 的实例的详细信息，其中包括主机名和共享磁盘卷 ID 。

```
# *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。

其他资源

- 在集群中配置 GFS2 文件系统
- 配置 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

您可以在 Amazon Web Services(AWS)上将 Red Hat Enterprise Linux(RHEL)9 镜像部署为 EC2 实例。本章讨论了您选择镜像的选项，并列出或引用了主机系统和虚拟机(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 EC2 以及启动 EC2 实例的步骤。

要将 Red Hat Enterprise Linux 9(RHEL 9)作为 Amazon Web Services(AWS)上的 EC2 实例部署，请遵循以下信息。本章：

- 讨论您选择镜像的选项
- 列出或引用主机系统和虚拟机(VM)的系统要求。
- 提供从 ISO 镜像创建自定义虚拟机、将其上传到 EC2，并启动 EC2 实例的流程

重要

虽然您可以从 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，以便用于特定的云供应商。使用 Image Builder，您可以创建并上传 **ami** 格式的 Amazon Machine Image(AMI)。如需更多信息，请参阅 生成自定义 RHEL 系统镜像。

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。

3.1. 其它资源

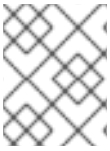
- Red Hat Cloud Access 参考指南
- 公共云中的红帽
- Amazon EC2 上的 Red Hat Enterprise Linux - FAQ
- Setting Up with Amazon EC2
- Red Hat on Amazon Web Services

3.2. 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
选择部署一个 Red Hat Gold Image。	利用您现有的红帽订阅。	通过 红帽云访问计划 启用订阅，然后选择用于 AWS 的红帽黄金镜像。	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。 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。 您移到 AWS 的自定义镜像是"云访问"镜像，因为您使用了现有的红帽订阅。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。 红帽提供了镜像的更新。AWS 通过红帽更新基础架构(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，请参阅生成自定义 RHEL 系统镜像。

重要

您无法将 on-demand 实例转换为 Red Hat Cloud Access 实例。要从按需镜像改为红帽云访问自带订阅(BYOS)镜像，请创建一个新的红帽云访问实例，并将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

- Red Hat Cloud Access 程序
- 编写自定义 RHEL 系统镜像

- AWS Management Console
- AWS Marketplace

3.3. 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。

3.3.1. 使用自定义基础镜像

要手动配置虚拟机(VM)，首先创建一个基础（起步）虚拟机镜像。然后，您可以修改配置设置，并添加 VM 在云上操作所需的软件包。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

3.3.2. 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	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 RHEL 9 基础镜像。

先决条件

- 虚拟化已在您的主机上启用。
- 您已 从红帽客户门户网站下载 了最新的 Red Hat Enterprise Linux ISO 镜像，并将该镜像移到 **/var/lib/libvirt/images** 中。

3.4.1. 从 RHEL ISO 镜像创建虚拟机

流程

1. 确保已为虚拟化启用主机机器。有关信息和流程，请参阅在 RHEL 9 中启用虚拟化。
2. 创建并启动基本 Red Hat Enterprise Linux 虚拟机。有关说明，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 kvmtest --memory 2048 --vcpus 2 --disk rhel-9.0-x86_64-
kvm.qcow2,bus=virtio --import --os-variant=rhel9.0
```

b. 如果使用 Web 控制台来创建虚拟机，请按照 使用 web 控制台创建虚拟机 中的流程进行操作，包括以下注意事项：

- 不要选择 **Immediately Start VM**。
- 将 **Memory** 大小更改为你希望的设置。
- 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。

3.4.2. 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**。
   - 验证 **/boot** 至少 500 MB。将剩余空间用于根 /。
   - 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**。根据情况创建其他用户。
5. 重新启动虚拟机，并在安装完成后以 **root** 身份登录。
6. 配置镜像。
   a. 注册虚拟机并启用 Red Hat Enterprise Linux 9 软件仓库。

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

   b. 确保已安装并启用了 **cloud-init** 软件包。

```
# dnf install cloud-init
# systemctl enable --now cloud-init.service
```

7. **重要：此步骤只适用于您要上传到 AWS 的虚拟机。**
   a. 对于 AMD64 或 Intel 64(x86_64)VM，安装 **nvme-xen-netfront** 和 **xen-blkfront** 驱动程

a. 对于 AMD64 或 Intel 64(x86_64)VM，安装 **nvme**,**xen-netfront** 和 **xen-blkfront** 驱动程序。

```
# dracut -f --add-drivers "nvme xen-netfront xen-blkfront"
```

b. 对于 ARM 64(aarch64)虚拟机，安装 **nvme** 驱动程序。

```
# dracut -f --add-drivers "nvme"
```

包括这些驱动程序会删除 dracut 超时的可能性。

或者，您可以将驱动程序添加到 **/etc/dracut.conf.d/**，然后输入 **dracut -f** 来覆盖现有的 **initramfs** 文件。

8. 关闭虚拟机。

其它资源

- 了解客户门户网站上的自动附加订阅
- cloud-init 简介

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

按照本节中的步骤将您的镜像上传到 AWS。

3.5.1. 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。

步骤

1. 安装 Python 3 和 **pip** 工具。

```
# dnf install python3
# dnf 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，来验证您是否安装了 AWS CLI。

```
$ aws --version
aws-cli/1.19.77 Python/3.6.15 Linux/5.14.16-201.fc34.x86_64 botocore/1.20.77
```

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

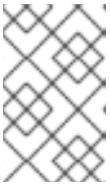
- 快速配置 AWS CLI
- AWS 命令行工具

3.5.2. 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。

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. 输入 **Region** 。
   c. 点击 **Next**。
4. 在 **Configure options** 视图中，选择所需的选项，然后单击 **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** 命令会创建一个名为 **my-new-bucket** 的 S3 存储桶。有关 **mb** 命令的更多信息，请参阅 AWS CLI 命令参考。

其他资源

- Amazon S3 Console
- AWS CLI Command Reference

3.5.3. 创建 vmimport 角色

执行以下流程来创建 **vmimport** 角色，这是 VM 导入所要求的。如需更多信息，请参阅 Amazon 文档中的 VM Import Service Role 部分。

步骤

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** 文件所在位置的完整路径。为该路径加上前缀 **file://**。例如：

```
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所创建的角色。指定 **role-policy.json** 文件的完整路径。例如：

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

其他资源

- VM 导入服务角色
- 所需的服务角色

3.5.4. 将镜像转换为 S3

完成以下步骤，将您的镜像转换为 S3。示例具有代表性；它们将格式为 **qcow2** 文件格式的镜像转换为 **raw** 格式的文件。Amazon 接受 **OVA**、**VHD**、**VHDX**、**VMDK** 和 **raw** 格式的镜像。如需有关 Amazon 接受的镜像格式的更多信息，请参阅 VM Import/Export Works。

步骤

1. 运行 **qemu-img** 命令来转换您的镜像。例如：

```
qemu-img convert -f qcow2 -O raw rhel-9.0-sample.qcow2 rhel-9.0-sample.raw
```

2. 将镜像推送到 S3。

```
aws s3 cp rhel-9.0-sample.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

3.5.5. 将您的镜像导入为快照

执行以下步骤将镜像导入为快照。

流程

1. 创建文件来指定镜像的存储桶和路径。将文件命名为 **containers.json**。在下面的示例中，将 **s3-bucket-name** 替换为您 的存储桶名称，将 **s3-key** 替换为您的密钥。您可以使用 Amazon S3 控制台获取镜像的密钥。

```
{
    "Description": "rhel-9.0-sample.raw",
    "Format": "raw",
    "UserBucket": {
        "S3Bucket": "s3-bucket-name",
        "S3Key": "s3-key"
    }
}
```

2. 将镜像导入为快照。本例使用公有 Amazon S3 文件；您可以使用 Amazon S3 控制台 来更改存储桶的权限设置。

```
aws ec2 import-snapshot --disk-container file://containers.json
```

终端会显示如下信息。注意消息中的 **ImportTaskID**。

```
{
    "SnapshotTaskDetail": {
        "Status": "active",
        "Format": "RAW",
        "DiskImageSize": 0.0,
        "UserBucket": {
            "S3Bucket": "s3-bucket-name",
            "S3Key": "rhel-9.0-sample.raw"
        },
        "Progress": "3",
        "StatusMessage": "pending"
    },
    "ImportTaskId": "import-snap-06cea01fa0f1166a8"
}
```

3. 使用 **describe-import-snapshot-tasks** 命令来跟踪导入的进度。包含 **ImportTaskID**。

```
aws ec2 describe-import-snapshot-tasks --import-task-ids import-snap-06cea01fa0f1166a8
```

返回的消息显示任务的当前状态。完成后，**Status** 显示为 **completed**。在状态中记录快照 ID。

其它资源

- Amazon S3 Console
- 使用 VM Import/Export 将 Disk 导入为快照

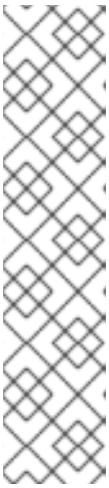
3.5.6. 从上传的快照创建 AMI

在 EC2 中，在启动实例时，您必须选择一个 Amazon Machine Image(AMI)。执行以下步骤从上传的快照中创建 AMI。

流程

1. 进入 AWS EC2 仪表板。
2. 在 **Elastic Block Store** 下，选择 **Snapshots**。

3. 搜索快照 ID（例如，**snap-0e718930bd72bcda0**）。
4. 右键点击快照并选择 **Create image**。
5. 为您的镜像命名。
6. 在 **Virtualization type** 中，选择 **Hardware-assisted virtualization**。
7. 点 **Create**。在关于镜像创建的备注中，有一个到您镜像的链接。
8. 点击镜像链接。您的镜像显示在 **Images>AMIs** 下。



注意

另外，您可以使用 AWS CLI **register-image** 命令来从快照创建 AMI。如需更多信息，请参阅 register-image。下面是一个示例。

```
$ aws ec2 register-image --name "myimagename" --description
"myimagedescription" --architecture x86_64  --virtualization-type hvm --root-
device-name "/dev/sda1" --block-device-mappings "{\"DeviceName\":
\"/dev/sda1\",\"Ebs\": {\"SnapshotId\": \"snap-0ce7f009b69ab274d\"}}" --ena-
support
```

您必须将根设备卷 **/dev/sda1** 指定为 **root-device-name**。有关 AWS 设备映射的概念信息，请参阅块设备映射示例。

3.5.7. 从 AMI 启动实例

执行以下步骤从 AMI 启动和配置实例。

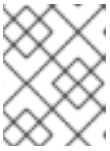
流程

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. 选择一个满足或超过工作负载要求的 **Instance Type**。
   如需有关实例类型的信息，请参阅 Amazon EC2 实例类型。
4. 点 **Next：配置实例详情**。
   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 **Next：添加 Storage**。验证默认存储是否足够。
6. 点 **Next：添加 Tags**。



注意

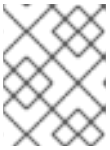
标签可帮助您管理 AWS 资源。有关标记的信息，请参阅标记您的 Amazon EC2 资源。

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。如有必要，使用命令选项 **chmod 400 <keyname>.pem** 来更改权限。

10. 点 **Launch Instances**。
11. 点击 **View Instances**。您可以命名实例。
现在，您可以通过选择一个实例并单击 **Connect** 来启动与实例的 SSH 会话。使用为 **独立的 SSH 客户端** 提供的示例。



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

3.5.8. 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。如需更多信息，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。请参阅通过命令行附加和删除订阅。

其它资源

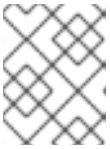
- 创建红帽客户门户网站激活码
- 通过命令行附加和删除订阅
- 使用并配置 Red Hat Subscription Manager

3.5.9. 对 AWS 黄金镜像设置自动注册

要在 Amazon Web Services(AWS)上更快、更舒适地部署 RHEL 8 虚拟机，您可以将 RHEL 8 的黄金镜像设置为自动注册到 Red Hat Subscription Manager(RHSM)。

先决条件

- 您已下载了 AWS 的最新 RHEL 8 黄金镜像。有关说明，请参阅 在AWS 上使用黄金镜像。



注意

一个 AWS 帐户一次只能附加到一个红帽帐户。因此，在将其附加到您的红帽帐户之前，请确保其他用户不需要访问 AWS 帐户。

步骤

1. 将黄金镜像上传到 AWS。具体步骤请参阅 将 Red Hat Enterprise Linux 镜像上传到 AWS 。
2. 使用上传的镜像创建虚拟机。他们将自动订阅 RHSM。

验证

- 在使用上述说明创建的 RHEL 9 虚拟机中，执行 **subscription-manager 身份** 命令验证系统是否注册到 RHSM。在成功注册的系统上，这会显示系统的 UUID。例如：

```
# subscription-manager identity
system identity: fdc46662-c536-43fb-a18a-bbcb283102b7
name: 192.168.122.222
org name: 6340056
org ID: 6340056
```

其他资源

- AWS Management Console
- 为红帽服务配置云资源

第 4 章 在 AWS 上配置红帽高可用性集群

本章提供了在 Amazon Web Services(AWS)上配置红帽高可用性(HA)集群的信息和流程，其中使用 EC2 实例作为集群节点。请注意，您有多个选项可用来获取用于集群的 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 。

本章包括：

- 为 AWS 设置环境的先决条件的流程。设置环境后，您可以创建并配置 EC2 实例。
- 特定于创建 HA 集群的流程，其将单个节点转换为 AWS 上的 HA 节点的集群。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。

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。

4.1. 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

4.2. 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. 点击您的 AWS 帐户 ID 来显示下拉菜单，并选择 **My Security Credentials**。
3. 点 **Users**。
4. 选择用户并打开 **Summary** 屏幕。

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

4.3. 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。

步骤

1. 安装 Python 3 和 **pip** 工具。

```
# dnf install python3
# dnf 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，来验证您是否安装了 AWS CLI。

```
$ aws --version
aws-cli/1.19.77 Python/3.6.15 Linux/5.14.16-201.fc34.x86_64 botocore/1.20.77
```

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

4.4. 创建 HA EC2 实例

完成以下步骤以创建用作 HA 集群节点的实例。请注意，您有几个选项可用于获取用于集群的 RHEL 镜像。有关 AWS 的镜像选项的信息，请参阅 AWS 上的 Red Hat Enterprise Linux 镜像选项。

您可以创建和上传用于集群节点的自定义镜像，也可以选择黄金镜像（云访问镜像）或按需镜像。

先决条件

- 您需要已设置 AWS 环境。如需更多信息，请参阅使用 Amazon EC2 设置。

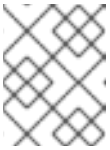
流程

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. 选择一个满足或超过工作负载要求的 **Instance Type**。根据您的 HA 应用，每个实例可能需要具有更高的容量。
   如需有关实例类型的信息，请参阅 Amazon EC2 实例类型。
4. 点 **Next：配置实例详情**。
   a. 输入您要为集群创建的 **Number of instances**。本章示例使用三个集群节点。

注意

不要启动自动缩放组。

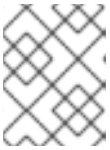
   b. 对于 **Network**，请选择您在 Set up AWS environment 中创建的 VPC。选择实例子网以创建新子网。
   c. 为 Auto-assign Public IP 选择 **Enable**。以下是 **Configure Instance Details** 所需的最小选择。根据您的特定 HA 应用，您可能需要进行其他选择。



注意

这些是创建基本实例所需的最小配置选项。根据您的 HA 应用程序要求查看其他选项。

5. 点 **Next：添加 Storage** 并验证默认存储是否已经足够。除非您的 HA 应用程序需要其他存储选项，您不需要修改这些设置。
6. 点 **Next：添加 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**。选择在 Setting the AWS environment 中创建的现有安全组。
8. 点击 **Review and Launch**，并验证您的选择。
9. 点击 **Launch**。此时会提示您选择现有密钥对或创建新密钥对。选择在 Setting up the AWS environment 时创建的密钥对。
10. 点击 **Launch Instances**。
11. 点击 **View Instances**。您可以命名实例。

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

4.5. 配置私钥

在 SSH 会话中使用私有 SSH 密钥文件（**.pem**）之前，请完成以下配置任务来使用该文件。

步骤

1. 将密钥文件从 **Downloads** 目录移到您的 **主** 目录或 **~/.ssh 目录** 。
2. 输入以下命令更改密钥文件的权限，以便只有 root 用户才能读取它。

```
# chmod 400 KeyName.pem
```

4.6. 连接到 EC2 实例

在所有节点上完成以下步骤来连接 EC2 实例。

步骤

1. 启动 AWS Console ， 并选择 EC2 实例。
2. 点击 **Connect** ， 并选择 **A standalone SSH client**。
3. 在 SSH 终端会话中，使用弹出窗口中提供的 AWS 示例连接到实例。如果示例中没有显示路径，请为您的 **KeyName.pem** 文件添加正确的路径。

4.7. 安装高可用性软件包和代理

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. 输入以下命令删除 AWS Red Hat Update Infrastructure（RHUI）客户端。由于您要使用红帽云访问订阅，因此除了订阅之外，您不应该使用 AWS RHUI。

```
$ sudo -i
# dnf -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 9 服务器 HA 软件仓库。

```
# subscription-manager repos --enable=rhel-9-for-x86_64-highavailability-rpms
```

5. 更新 RHEL AWS 实例。

```
# dnf update -y
```

6. 安装红帽高可用性附加组件软件包，以及高可用性通道的所有可用的隔离代理。

```
# dnf install pcs pacemaker fence-agents-aws
```

7. 在上一步中的 **pcs** 和 **pacemaker** 安装过程中，创建了用户 **hacluster** 。在所有群集节点上为 **hacluster** 创建密码。所有节点都使用相同的密码。

```
# passwd hacluster
```

8. 如果安装了 **firewalld.service**，请在 RHEL Firewall 中添加 **high availability** 服务。

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

9. 启动 **pcs** 服务，并使其在引导时启动。

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

10. 编辑 **/etc/hosts** ，并添加 RHEL 主机名和内部 IP 地址。详情请参阅 如何在 RHEL 集群节点上设置 /etc/hosts 文件？

验证

- 确保 **pcs** 服务正在运行。

```
# systemctl status pcsd.service

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

```
configuration interface…
Mar 01 14:53:28 ip-10-0-0-48.ec2.internal systemd[1]: Started PCS GUI and remote
configuration interface.
```

4.8. 创建集群

完成以下步骤以创建节点集群。

步骤

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

```
# pcs host auth hostname1 hostname2 hostname3
Username: hacluster
Password:
hostname1: Authorized
hostname2: Authorized
hostname3: Authorized
```

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 cluster-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

1. 启用集群。

```
[root@node01 clouduser]# pcs cluster enable --all
```

2. 启动集群。

```
[root@node01 clouduser]# 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...
```

4.9. 配置隔离

隔离配置可确保 AWS 集群上的故障节点被自动隔离，这样可防止节点消耗集群的资源或影响集群的功能。

您可以使用多种方法在 AWS 集群上配置隔离功能。本节提供以下内容：

- 默认配置的标准过程。
- 另一种配置过程，用于更高级的配置，专注于自动化。

标准流程

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. 输入以下命令配置隔离设备。使用 **pcmk_host_map** 命令将 RHEL 主机名映射到实例 ID。使用您之前设置的 AWS 访问密钥和 AWS Secret 访问密钥。

```
# pcs stonith create name 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" power_timeout=240 pcmk_reboot_timeout=480 pcmk_reboot_retries=4
```

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
```

备用步骤

1. 获取集群的 VPC ID。

```
# aws ec2 describe-vpcs --output text --filters "Name=tag:Name,Values=clustername-
vpc" --query 'Vpcs[*].VpcId'
vpc-06bc10ac8f6006664
```

2. 使用集群的 VPC ID，获取 VPC 实例。

```
$ aws ec2 describe-instances --output text --filters "Name=vpc-id,Values=vpc-
06bc10ac8f6006664" --query 'Reservations[].Instances[].{Name:Tags[? Key==Name]|
[0].Value,Instance:InstanceId}' | grep "\-node[a-c]"
i-0b02af8927a895137     clustername-nodea-vm
i-0cceb4ba8ab743b69     clustername-nodeb-vm
i-0502291ab38c762a5     clustername-nodec-vm
```

3. 使用获得的实例 ID 在集群的每个节点中配置隔离。例如：

```
[root@nodea ~]# CLUSTER=clustername && pcs stonith create fence${CLUSTER}
fence_aws access_key=XXXXXXXXXXXXXXXXXXXX pcmk_host_map=$(for NODE \ in
node{a..c}; do ssh ${NODE} "echo -n \${HOSTNAME}:\$(curl -s
http://169.254.169.254/latest/meta-data/instance-id)\;"; done) \ pcmk_reboot_retries=4
pcmk_reboot_timeout=480 power_timeout=240 region=xx-xxxx-x
secret_key=XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

[root@nodea ~]# pcs stonith config fence${CLUSTER}
Resource: clustername (class=stonith type=fence_aws)
Attributes: access_key=XXXXXXXXXXXXXXXXXXXX pcmk_host_map=nodea:i-
0b02af8927a895137;nodeb:i-0cceb4ba8ab743b69;nodec:i-0502291ab38c762a5;
pcmk_reboot_retries=4 pcmk_reboot_timeout=480 power_timeout=240 region=xx-xxxx-x
secret_key=XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
Operations: monitor interval=60s (clustername-monitor-interval-60s)
```

验证

1. 测试其中一个群集节点的隔离代理。

```
# pcs stonith fence awsnodename
```

注意

这个命令的响应可能需要几分钟时间来显示。如果您监视节点被隔离的活跃终端会话，您会在进入 fence 命令后马上终止终端连接。

Example:

```
[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。

```
# pcs status
```

Example:

```
[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 19:55:41 2018
Last change: Fri Mar  2 19:24:59 2018 by root via cibadmin on ip-10-0-0-46

3 nodes configured
1 resource configured

Online: [ ip-10-0-0-46 ip-10-0-0-48 ]
OFFLINE: [ 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
```

3. 启动上一步中隔离的节点。

```
# pcs cluster start awshostname
```

4. 检查状态以验证节点已启动。

```
# pcs status
```

Example:

```
[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
```

4.10. 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。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。

步骤

1. 安装 AWS CLI。具体步骤请参阅 安装 AWS CLI。
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
```

4.11. 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。如果节点在一定时间内没有响应心跳检查，则该节点会被隔离，操作会切换到集群中的其它节点。需要配置网络资源代理才能正常工作。

将两个资源添加到 same group，以强制 **order** 和 **colocation** 约束。

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。

步骤

1. 输入以下命令来查看 **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 secondary Private IP 地址。

```
# pcs resource create privip awsvip secondary_private_ip=Unused-IP-Address --group
group-name
```

Example:

```
[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
```

Example:

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

验证

- 输入 **pcs status** 命令来验证资源是否正在运行。

```
# pcs status
```

Example:

```
[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重新映射到故障转移节点，从而屏蔽隔离节点的故障。

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 same group 中，来强制执行 **order** 和 **colocation** 约束。
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. 输入以下命令来查看 AWS Secondary Elastic IP Address 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
```

Example:

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

验证

- 输入 **pcs status** 命令来验证资源是否正在运行。

```
# pcs status
```

Example:

```
[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

输入以下命令来验证虚拟 IP(awsvip)和弹性 IP(awseip)资源是否正常工作。

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
```

Example:

```
$ 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
- 配置和管理高可用性集群

4.12. 配置共享块存储

本节提供了为带有 Amazon Elastic Block Storage(EBS) Multi-Attach 卷的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）。

先决条件

- 您必须使用 基于 AWS Nitro 系统的 Amazon EC2 实例 。

步骤

1. 使用 AWS 命令 create-volume 创建共享块卷。

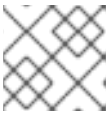
```
$ aws ec2 create-volume --availability-zone <availability_zone> --no-encrypted --size
1024 --volume-type io1 --iops 51200 --multi-attach-enabled
```

例如，以下命令在 **us-east-1a** 可用区域中创建一个卷。

```
$ 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

在下一步中您需要 **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 实**例 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

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

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

例如，以下命令列出了实例 IP **198.51.100.3** 的详细信息，其中包括主机名和块设备。

```
# 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都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 IP 地址为 **198.51.100.3** 的实例的详细信息，其中包括主机名和共享磁盘卷 ID 。

```
# 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
```

其他资源

- 在集群中配置 GFS2 文件系统
- 配置 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

要在 Google Cloud Platform(GCP)上将 Red Hat Enterprise Linux 9(RHEL 9)作为 Google Compute Engine(GCE)实例部署，请遵循以下信息。本章：

- 讨论您选择镜像的选项
- 列出或引用主机系统和虚拟机(VM)的系统要求。
- 提供从 ISO 镜像创建自定义虚拟机的流程，将其上传到 GCE，并启动实例

注意

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

重要

您可以从 ISO 镜像创建自定义的虚拟机，但红帽建议您使用 *Red Hat Image Builder* 产品来创建自定义的镜像以用于特定的云供应商。如需更多信息，请参阅生成自定义 RHEL 系统镜像。

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。

5.1. 其它资源

- 公共云中的红帽
- Google Cloud

5.2. GCP 上的 RED HAT ENTERPRISE LINUX 镜像选项

下表列出了 Google Cloud Platform 上的 RHEL 9 和镜像选项的不同镜像选择。

表 5.1. 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。	使用您现有的红帽订阅。	通过 Red Hat Cloud Access 程序 启用订阅，然后在 Google Cloud Platform 上选择 Red Hat Gold Image。如需了解有关 Gold Images 以及如何在 Google Cloud Platform 上访问它们的详细信息，请参阅 Red Hat Cloud Access 参考指南。	订阅包括红帽产品成本；您需要为 Google 支付其他费用。 红帽黄金镜像称为"云访问"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。
选择部署移至 GCP 的自定义镜像。	使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。 移到 GCP 的自定义镜像称为"云访问"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。要从按需镜像改为红帽云访问自带订阅(BYOS)镜像，请创建一个新的红帽云访问实例，并将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

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

5.3. 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。

5.3.1. 使用自定义基础镜像

要手动配置虚拟机(VM)，首先创建一个基础（起步）虚拟机镜像。然后，您可以修改配置设置，并添加 VM 在云上操作所需的软件包。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

5.3.2. 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	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。

5.4. 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 RHEL 9 基础镜像。

先决条件

- 虚拟化已在您的主机上启用。
- 您已 从红帽客户门户网站下载 了最新的 Red Hat Enterprise Linux ISO 镜像，并将该镜像移到 **/var/lib/libvirt/images** 中。

5.4.1. 从 RHEL ISO 镜像创建虚拟机

流程

1. 确保已为虚拟化启用主机机器。有关信息和流程，请参阅在 RHEL 9 中启用虚拟化。
2. 创建并启动基本 Red Hat Enterprise Linux 虚拟机。有关说明，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 kvmtest --memory 2048 --vcpus 2 --disk rhel-9.0-x86_64-kvm.qcow2,bus=virtio --import --os-variant=rhel9.0
   ```

   b. 如果使用 Web 控制台来创建虚拟机，请按照 使用 web 控制台创建虚拟机 中的流程进行操作，包括以下注意事项：
      - 不要选择 **Immediately Start VM**。

- 将 **Memory** 大小更改为你希望的设置。
- 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。

5.4.2. 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**。
      - 验证 **/boot** 至少 500 MB。将剩余空间用于根 /。
      - 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**。根据情况创建其他用户。
5. 重新启动虚拟机，并在安装完成后以 **root** 身份登录。
6. 配置镜像。
   a. 注册虚拟机并启用 Red Hat Enterprise Linux 9 软件仓库。

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

   b. 确保已安装并启用了 **cloud-init** 软件包。

   ```
   # dnf install cloud-init
   # systemctl enable --now cloud-init.service
   ```

7. 关闭虚拟机。

其他资源

- 了解客户门户网站上的自动附加订阅
- cloud-init 简介

5.5. 将 RHEL 镜像上传到 GCP

要将 RHEL 9 镜像上传到 Google Cloud Platform(GCP)，请遵循本节中的步骤。

5.5.1. 在 GCP 上创建新项目

完成以下步骤，来在 Google Cloud Platform(GCP)上创建一个新项目。

先决条件

- 您必须拥有 GCP 帐户。如果没有，请参阅 Google Cloud 了解更多信息。

步骤

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. 检查 **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 命令来创建项目。下面是一个简单的例子。

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

该示例创建了一个 项目 ID 为 **my-gcp-project3**，项目名称为 **project3** 的项目。如需更多信息，请参阅 gcloud 项目创建。

其他资源

- 在 Google Cloud 中创建和管理资源

5.5.2. 安装 Google Cloud SDK

完成以下步骤以安装 Google Cloud SDK。

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 命令来执行任务，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>** 命令来显示项目信息。

其他资源

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

5.5.3. 为 Google Compute Engine 创建 SSH 密钥

执行以下流程，通过 GCE 生成 SSH 密钥并注册 SSH 密钥，以便您可以使用其公共 IP 地址直接 SSH 到实例。

步骤

1. 使用 **ssh-keygen** 命令来生成用于 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 banner** 左侧的 **Navigation** 菜单，并选择 **Compute Engine**，然后选择 **Metadata**。
3. 点 **SSH Keys**，然后点 **Edit**。
4. 输入 **~/.ssh/google_compute_engine.pub** 文件中生成的结果，然后单击 **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，使用实例的别名来填充配置文件。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。

其他资源

- gcloud 计算 config-ssh
- 连接到实例

5.5.4. 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。

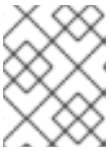
流程

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

5.5.5. 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。示例具有代表性；它们将 **qcow2** 镜像转换为 **raw** 格式，然后对该映像进行 tar 操作以便上传。

步骤

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

```
# qemu-img convert -f qcow2 -O raw rhel-{ProductNumber}.0-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
```

4. 在 **Google Cloud Platform** 主屏幕中，单击折叠菜单图标，并选择 **Storage**，然后选择 **Browser**。

5. 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。为此，可单击存储桶的名称，然后单击 **Upload files**。

其它资源

- 手动导入虚拟磁盘
- 选择导入方法

5.5.6. 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

1. 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。

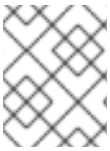
2. 另外，还可在 GCP Console 中找到该镜像。
   a. 单击 **Google Cloud Console** 标语左侧的 **Navigation** 菜单。
   b. 选择 **Compute Engine**，然后选择 **Images**。

其它资源

- 创建、删除和弃用自定义镜像
- gcloud 计算镜像创建

5.5.7. 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 banner** 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 traffic** 或 **Allow HTTPS traffic**。

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 banner** 左侧的 **Navigation** 菜单，选择 **Compute Engine**，然后选择 **VM instances**。

注意

或者，您可以使用 **gcloud compute instances create** CLI 命令来通过镜像创建 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
```

该示例根据现有的 **test-iso2-image** 映像，在区域 **us-central1-a** 中创建名为 **myinstance3** 的虚拟机实例。如需更多信息，请参阅 gcloud 计算实例创建。

5.5.8. 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。

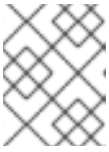
流程

1. 运行以下命令以确保您的实例正在运行。命令列出有关 GCE 实例的信息，包括实例是否正在运行，如果正在运行，则列出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。该示例是使用之前创建的 **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。您还可以使用之前设置的 root 帐户和密码连接到您的实例。

其它资源

- gcloud 计算实例列表
- 连接到实例

5.5.9. 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

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。如需更多信息，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。请参阅通过命令行附加和删除订阅。

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

本章提供了使用 Google Compute Engine(GCE)虚拟机实例作为集群节点，在 Google Cloud Platform(GCP)上配置红帽高可用性(HA)集群的信息和流程。

本章包括：

- 为 GCP 设置环境的先决条件的流程。设置完环境后，您可以创建并配置虚拟机实例。
- 特定于创建 HA 集群的流程，其将单个节点转换为 GCP 上的 HA 节点的集群。这包括在每个集群节点上安装高可用性软件包和代理、配置隔离以及安装网络资源代理的步骤。

先决条件

- 您必须在 Red Hat Cloud Access 程序 中注册，且没有使用的 RHEL 订阅。附加的订阅必须包括为每个 GCP 实例访问以下软件仓库。
  - Red Hat Enterprise Linux 9 Server: rhel-9-server-rpms/8Server/x86_64
  - Red Hat Enterprise Linux 9 服务器（高可用性）：rhel-9-server-ha-rpms/8Server/x86_64
- 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

6.1. 其它资源

- RHEL 高可用性集群的支持策略 - 传输协议
- 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

6.2. 所需的系统软件包

本章中的流程假设您使用运行 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 6.1. 系统软件包

软件包	软件仓库	描述
libvirt	rhel-9-for-x86_64-appstream-rpms	用于管理平台虚拟化的开源 API、守护进程和管理工具
virt-install	rhel-9-for-x86_64-appstream-rpms	用于构建虚拟机的命令行工具
libguestfs	rhel-9-for-x86_64-appstream-rpms	用于访问和修改虚拟机文件系统的库
guestfs-tools	rhel-9-for-x86_64-appstream-rpms	虚拟机的系统管理工具；包括 **virt-customize** 实用程序

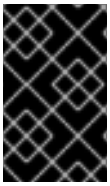
6.3. GCP 上的 RED HAT ENTERPRISE LINUX 镜像选项

下表列出了 Google Cloud Platform 上的 RHEL 9 和镜像选项的不同镜像选择。

表 6.2. 镜像选项

镜像选项	订阅	示例情境	注意事项
选择部署一个 Red Hat Gold Image。	使用您现有的红帽订阅。	通过 Red Hat Cloud Access 程序 启用订阅，然后在 Google Cloud Platform 上选择 Red Hat Gold Image。如需了解有关 Gold Images 以及如何在 Google Cloud Platform 上访问它们的详细信息，请参阅 Red Hat Cloud Access 参考指南。	订阅包括红帽产品成本；您需要为 Google 支付其他费用。 红帽黄金镜像称为"云访问"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。
选择部署移至 GCP 的自定义镜像。	使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。 移到 GCP 的自定义镜像称为"云访问"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。

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。要从按需镜像改为红帽云访问自带订阅(BYOS)镜像，请创建一个新的红帽云访问实例，并将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

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

6.4. 安装 GOOGLE CLOUD SDK

完成以下步骤以安装 Google Cloud SDK。

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 命令来执行任务，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>** 命令来显示项目信息。

其他资源

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

6.5. 创建 GCP 镜像存储桶

以下文档包含在默认位置创建 multi-regional 存储桶的最低要求。

先决条件

- GCP 存储工具(gsutil)

流程

1. 如果您还没有登录到 Google Cloud Platform，请使用以下命令登录。

```
# gcloud auth login
```

2. 创建存储桶。

```
$ gsutil mb gs://BucketName
```

例如：

```
$ gsutil mb gs://rhel-ha-bucket
```

其它资源

- 生成存储桶

6.6. 创建自定义虚拟私有云网络和子网

完成以下步骤来创建自定义虚拟私有云(VPC)网络和子网。

流程

1. 启动 GCP 控制台。
2. 在左侧导航窗格中，选择 **Networking** 下的 **VPC networks**。
3. 点 **Create VPC Network**。
4. 输入 VPC 网络的名称。
5. 在 **New subnet** 下，在您要创建集群的区域中创建 **Custom subnet**。
6. 点 **Create**。

6.7. 准备并导入基本 GCP 镜像

完成以下步骤，为 GCP 准备 Red Hat Enterprise Linux 9 镜像。

步骤

1. 输入以下命令来转换该文件。上传到 GCP 的镜像必须是 **raw** 格式，并命名为 **disk.raw**。

```
$ qemu-img convert -f qcow2 ImageName.qcow2 -O raw disk.raw
```

2. 输入以下命令来压缩 **raw** 文件。上传到 GCP 的镜像必须被压缩。

```
$ tar -Sczf ImageName.tar.gz disk.raw
```

3. 将压缩镜像导入到之前创建的存储桶。

```
$ gsutil cp ImageName.tar.gz gs://BucketName
```

6.8. 创建并配置基本 GCP 实例

完成以下步骤，创建并配置符合 GCP 操作和安全要求的 GCP 实例。

流程

1. 输入以下命令从存储桶中压缩文件创建镜像。

```
$ gcloud compute images create BaseImageName --source-uri
gs://BucketName/BaseImageName.tar.gz
```

例如：

```
[admin@localhost ~] $ gcloud compute images create rhel-76-server --source-uri
gs://user-rhelha/rhel-server-76.tar.gz
Created [https://www.googleapis.com/compute/v1/projects/MyProject/global/images/rhel-
server-76].
NAME          PROJECT                FAMILY  DEPRECATED  STATUS
rhel-76-server  rhel-ha-testing-on-gcp                    READY
```

2. 输入以下命令从镜像创建模板实例。基本 RHEL 实例所需的最小值为 n1-standard-2。如需了解更多配置选项，请参阅 gcloud 计算实例创建。

```
$ gcloud compute instances create BaseInstanceName --can-ip-forward --machine-
type n1-standard-2 --image BaseImageName --service-account ServiceAccountEmail
```

例如：

```
[admin@localhost ~] $ gcloud compute instances create rhel-76-server-base-instance --
can-ip-forward --machine-type n1-standard-2 --image rhel-76-server --service-account
account@project-name-on-gcp.iam.gserviceaccount.com
Created [https://www.googleapis.com/compute/v1/projects/rhel-ha-testing-on-gcp/zones/us-
east1-b/instances/rhel-76-server-base-instance].
NAME   ZONE   MACHINE_TYPE   PREEMPTIBLE  INTERNAL_IP  EXTERNAL_IP
STATUS
rhel-76-server-base-instance  us-east1-bn1-standard-2          10.10.10.3   192.227.54.211
RUNNING
```

3. 通过 SSH 终端会话连接到实例。

```
$ ssh root@PublicIPaddress
```

4. 更新 RHEL 软件。

a. 使用红帽订阅管理器(RHSM)注册。

b. 启用订阅池 ID（或使用 **--auto-attach** 命令）。

c. 禁用所有软件仓库。

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

d. 启用以下软件仓库。

```
# subscription-manager repos --enable=rhel-9-server-rpms
```

e. 运行 the **dnf update** 命令。

```
# dnf update -y
```

5. 在运行的实例（原位安装）中安装 GCP Linux 客户机环境。
具体步骤请参阅原位安装客户机环境。

6. 选择 **CentOS/RHEL** 选项。

7. 复制命令脚本，并将它粘贴到命令提示符处，来立即运行脚本。

8. 对实例进行以下配置更改。这些更改基于自定义镜像的 GCP 建议。如需更多信息，请参阅 gcloudcompute 镜像列表。

a. 编辑 **/etc/chrony.conf** 文件，并删除所有 NTP 服务器。

b. 添加以下 NTP 服务器。

```
metadata.google.internal iburst Google NTP server
```

c. 删除任何持久的网络设备规则。

```
# rm -f /etc/udev/rules.d/70-persistent-net.rules

# rm -f /etc/udev/rules.d/75-persistent-net-generator.rules
```

d. 将网络服务设置为自动启动。

```
# chkconfig network on
```

e. 将 **sshd 服务** 设置为自动启动。

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

f. 输入以下命令将时区设置为 UTC。

```
# ln -sf /usr/share/zoneinfo/UTC /etc/localtime
```

g. （可选）编辑 **/etc/ssh/ssh_config** 文件，并将以下行添加到文件的末尾：这将在较长的不活跃时间段内使您的 SSH 会话保持活跃。

```
-
```

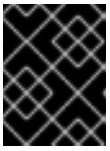
```
# Server times out connections after several minutes of inactivity.
# Keep alive ssh connections by sending a packet every 7 minutes.
ServerAliveInterval 420
```

h. 编辑 **/etc/ssh/sshd_config** 文件，并根据需要进行以下更改：**ClientAliveInterval 420** 设置是可选的；这会使 SSH 会话在长时间处于非活动状态期间保持活跃状态。

```
PermitRootLogin no
PasswordAuthentication no
AllowTcpForwarding yes
X11Forwarding no
PermitTunnel no
# Compute times out connections after 10 minutes of inactivity.
# Keep ssh connections alive by sending a packet every 7 minutes.
ClientAliveInterval 420
```

9. 输入以下命令来禁用密码访问。编辑 **/etc/cloud/cloud.cfg** 文件。

```
ssh_pwauth from 1 to 0.
ssh_pwauth: 0
```



重要

在以前的版本中，您可以启用密码访问来允许 SSH 会话访问来配置实例。您必须禁用密码访问。所有 SSH 会话访问都必须是无密码的。

10. 输入以下命令从订阅管理器取消注册实例。

```
# subscription-manager unregister
```

11. 输入以下命令清理 shell 历史记录。为下一个流程保留实例运行。

```
# export HISTSIZE=0
```

6.9. 创建快照镜像

完成以下步骤以保留实例配置设置并创建快照。

流程

1. 在正在运行的实例中，输入以下命令将数据同步到磁盘。

```
# sync
```

2. 在您的主机系统中输入以下命令来创建快照。

```
$ gcloud compute disks snapshot InstanceName --snapshot-names SnapshotName
```

3. 在主机系统中，输入以下命令从快照创建配置的镜像。

```
$ gcloud compute images create ConfiguredImageFromSnapshot --source-snapshot
SnapshotName
```

其它资源

- 创建持久性磁盘快照

6.10. 创建 HA 节点模板实例和 HA 节点

从快照配置了镜像后，您可以创建节点模板。使用此模板来创建所有 HA 节点。完成以下步骤以创建模板和 HA 节点。

流程

1. 运行以下命令来创建实例模板。

```
$ gcloud compute instance-templates create InstanceTemplateName --can-ip-forward --machine-type n1-standard-2 --image ConfiguredImageFromSnapshot --service-account ServiceAccountEmailAddress
```

例如：

```
[admin@localhost ~] $ gcloud compute instance-templates create rhel-91-instance-template --can-ip-forward --machine-type n1-standard-2 --image rhel-91-gcp-image --service-account account@project-name-on-gcp.iam.gserviceaccount.com
Created [https://www.googleapis.com/compute/v1/projects/project-name-on-gcp/global/instanceTemplates/rhel-91-instance-template].
NAME  MACHINE_TYPE   PREEMPTIBLE  CREATION_TIMESTAMP
rhel-91-instance-template   n1-standard-2          2018-07-25T11:09:30.506-07:00
```

2. 输入以下命令在一个区中创建多个节点。

```
# gcloud compute instances create NodeName01 NodeName02 --source-instance-template InstanceTemplateName --zone RegionZone --network=NetworkName --subnet=SubnetName
```

例如：

```
[admin@localhost ~] $ gcloud compute instances create rhel81-node-01 rhel81-node-02 rhel81-node-03 --source-instance-template rhel-91-instance-template --zone us-west1-b --network=projectVPC --subnet=range0
Created [https://www.googleapis.com/compute/v1/projects/project-name-on-gcp/zones/us-west1-b/instances/rhel81-node-01].
Created [https://www.googleapis.com/compute/v1/projects/project-name-on-gcp/zones/us-west1-b/instances/rhel81-node-02].
Created [https://www.googleapis.com/compute/v1/projects/project-name-on-gcp/zones/us-west1-b/instances/rhel81-node-03].
NAME          ZONE        MACHINE_TYPE  PREEMPTIBLE  INTERNAL_IP  EXTERNAL_IP
STATUS
rhel81-node-01  us-west1-b  n1-standard-2              10.10.10.4   192.230.25.81   RUNNING
rhel81-node-02  us-west1-b  n1-standard-2              10.10.10.5   192.230.81.253  RUNNING
rhel81-node-03  us-east1-b  n1-standard-2              10.10.10.6   192.230.102.15  RUNNING
```

6.11. 安装 HA 软件包和代理

在所有节点上完成以下步骤。

流程

1. 在 Google Cloud Console 中，选择 **Compute Engine**，然后选择 **VM instances**。
2. 选择实例，单击 **SSH** 旁边的箭头，然后选择 **View** gcloud 命令选项。
3. 在命令提示符下粘贴此命令，以进行免密码访问实例。
4. 启用 sudo 帐户访问，并通过 Red Hat Subscription Manager 注册。
5. 启用订阅池 ID（或使用 **--auto-attach** 命令）。
6. 禁用所有软件仓库。

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

7. 启用以下软件仓库。

```
# subscription-manager repos --enable=rhel-9-server-rpms
# subscription-manager repos --enable=rhel-9-for-x86_64-highavailability-rpms
```

8. 安装 **pcs pacemaker**、隔离代理和资源代理。

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

9. 更新所有软件包。

```
# dnf update -y
```

6.12. 配置 HA 服务

在所有节点上完成以下步骤以配置 HA 服务。

步骤

1. 在上一步中的 **pcs** 和 **pacemaker** 安装过程中，创建了用户 **hacluster** 。在所有群集节点上为用户 **hacluster** 创建密码。所有节点都使用相同的密码。

```
# passwd hacluster
```

2. 如果安装了 **firewalld** 服务，请输入以下命令来添加 HA 服务。

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

# firewall-cmd --reload
```

3. 输入以下命令来启动 **pcs** 服务，并使其在引导时启动：

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

```
# systemctl enable pcsd.service

Created symlink from /etc/systemd/system/multi-user.target.wants/pcsd.service to
/usr/lib/systemd/system/pcsd.service.
```

验证

1. 确保 **pcsd** 服务正在运行。

```
# systemctl status pcsd.service

pcsd.service - PCS GUI and remote configuration interface
Loaded: loaded (/usr/lib/systemd/system/pcsd.service; enabled; vendor preset: disabled)
Active: active (running) since Mon 2018-06-25 19:21:42 UTC; 15s ago
Docs: man:pcsd(8)
man:pcs(8)
Main PID: 5901 (pcsd)
CGroup: /system.slice/pcsd.service
└─5901 /usr/bin/ruby /usr/lib/pcsd/pcsd > /dev/null &
```

2. 编辑 **/etc/hosts** 文件。为所有节点添加 RHEL 主机名和内部 IP 地址。

其它资源

- 如何在 RHEL 集群节点上设置 /etc/hosts 文件？

6.13. 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。在该命令中指定集群中每个节点的名称。

```
# pcs host auth hostname1 hostname2 hostname3
Username: hacluster
Password:
hostname1: Authorized
hostname2: Authorized
hostname3: Authorized
```

2. 运行以下命令来创建集群。

```
# pcs cluster setup cluster-name hostname1 hostname2 hostname3
```

验证

1. 运行以下命令，以便在启动时自动加入集群。

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

2. 输入以下命令启动集群。

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

6.14. 创建隔离设备

完成以下步骤以创建隔离设备。

请注意，对于大多数默认配置，GCP 实例名称和 RHEL 主机名是一样的。

步骤

1. 输入以下命令获取 GCP 实例名称。请注意，输出还显示实例的内部 ID。

```
# fence_gce --zone us-west1-b --project=rhel-ha-on-gcp -o list
```

例如：

```
[root@rhel81-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. 输入以下命令来创建隔离设备。

```
# pcs stonith create FenceDeviceName fence_gce zone=Region-Zone
project=MyProject
```

验证

- 验证隔离设备是否已启动。

```
# pcs status
```

例如：

```
[root@rhel81-node-01 ~]# pcs status
Cluster name: gcp-cluster
Stack: corosync
Current DC: rhel81-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 rhel81-node-01

3 nodes configured
3 resources configured

Online: [ rhel81-node-01 rhel81-node-02 rhel81-node-03 ]

Full list of resources:

us-west1-b-fence    (stonith:fence_gce):    Started rhel81-node-01

Daemon Status:
```

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

6.15. 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
```

6.16. 配置 GCP-VCP-MOVE-VIP 资源代理

gcp-vpc-move-vip 资源代理将辅助 IP 地址（别名 IP）附加到正在运行的实例。这是一个浮动 IP 地址，可在集群中的不同节点间传递。

输入以下命令来显示有关此资源的更多信息。

```
# 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。本节包含这两个范围的步骤。

主子网地址范围

完成以下步骤，为主 VPC 子网配置资源。

步骤

1. 输入以下命令来创建 **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
   ```

   Example:

   ```
   [root@rhel81-node-01 ~]# pcs resource create aliasip gcp-vpc-move-vip
   alias_ip=10.10.10.200/32
   ```

2. 输入以下命令来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
   ```

   Example:

   ```
   [root@rhel81-node-01 ~]# pcs resource create vip IPaddr2 nic=eth0 ip=10.10.10.200
   cidr_netmask=32
   ```

3. 输入以下命令对 **vipgrp** 下的网络资源进行分组。

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

验证

1. 输入以下命令来验证资源是否已启动，并分组在 **vipgrp** 下。

```
[root@rhel81-node-01 ~]# pcs status
```

2. 输入以下命令验证资源是否可移至其他节点。

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

Example:

```
[root@rhel81-node-01 ~]# pcs resource move vip rhel81-node-03
```

3. 输入以下命令来验证 **vip** 是否在不同节点上成功启动。

```
[root@rhel81-node-01 ~]# 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. 输入以下命令来创建二级子网地址范围。

```
# gcloud-ra compute networks subnets update SubnetName --region RegionName --add-secondary-ranges SecondarySubnetName=SecondarySubnetRange
```

Example:

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

2. 输入以下命令来创建 **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
```

Example:

```
[root@rhel81-node-01 ~]# pcs resource create aliasip gcp-vpc-move-vip alias_ip=10.10.20.200/32
```

3. 输入以下命令来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
```

Example:

```
[root@rhel81-node-01 ~]# pcs resource create vip IPaddr2 nic=eth0 ip=10.10.20.200
cidr_netmask=32
```

4. 将网络资源分组到 **vipgrp** 下。

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

验证步骤

1. 输入以下命令来验证资源是否已启动，并分组在 **vipgrp** 下。

```
[root@rhel81-node-01 ~]# pcs status
```

2. 输入以下命令验证资源是否可移至其他节点。

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

Example:

```
[root@rhel81-node-01 ~]# pcs resource move vip rhel81-node-03
```

3. 输入以下命令来验证 **vip** 是否在不同节点上成功启动。

```
[root@rhel81-node-01 ~]# pcs status
```