Red Hat Enterprise Linux 9

在 Amazon Web Services 上部署 RHEL 9

获取 RHEL 系统镜像并在 AWS 上创建 RHEL 实例

Last Updated: 2024-06-26

Red Hat Enterprise Linux 9 在 Amazon Web Services 上部署 RHEL 9

获取 RHEL 系统镜像并在 AWS 上创建 RHEL 实例

法律通告

摘要

要在公共云环境中使用 Red Hat Enterprise Linux (RHEL)，您可以在各种云平台（包括 Amazon Web Services (AWS)）上创建和部署 RHEL 系统镜像。您还可以在 AWS 上创建和配置红帽高可用性(HA)集群。 以下章节提供了在 AWS 上创建云 RHEL 实例和 HA 集群的说明。这些进程包括安装所需的软件包和代理、配置隔离以及安装网络资源代理。

目录

对红帽文档提供反馈

我们感谢您对我们文档的反馈。让我们了解如何改进它。

通过 Jira 提交反馈（需要帐户）

1. 登录到 Jira 网站。
2. 在顶部导航栏中点 **Create**
3. 在 **Summary** 字段中输入描述性标题。
4. 在 **Description** 字段中输入您对改进的建议。包括文档相关部分的链接。
5. 点对话框底部的 **Create**。

第 1 章 公有云平台上的 RHEL 简介

公有云平台提供计算资源即服务。除了使用内部硬件外，您可以运行您的 IT 工作负载，包括 Red Hat Enterprise Linux (RHEL)系统，作为公共云实例。

1.1. 在公有云中使用 RHEL 的好处

RHEL 作为公有云平台上的云实例与内部物理系统或虚拟机(VM)上的 RHEL 相比有以下优点：

- **灵活精细的资源分配**
  RHEL 的云实例作为虚拟机在云平台上运行，这通常意味着由云服务提供商维护的远程服务器集群。因此，给实例分配硬件资源，如特定类型的 CPU 或存储，发生在软件层面上，可轻松地自定义。

  与本地 RHEL 系统相比，您也不会受到物理主机的功能的限制。相反，您可以根据云提供商提供的选择，从各种功能中进行选择。

- **空间及成本效率**
  您不需要拥有任何内部服务器来托管您的云工作负载。这可避免与物理硬件关联的空间、电源和维护要求。

  相反，在公有云平台上，您直接支付给云提供商使用云实例的费用。成本通常根据分配给实例的硬件以及您使用的时间。因此，您可以根据要求优化成本。

- **软件控制的配置**
  云实例的整个配置都作为数据保存在云平台上，并由软件控制。因此，您可以轻松地创建、删除、克隆或迁移实例。云实例也在云供应商控制台中远程操作，默认连接到远程存储。

  另外，您可以随时将云实例的当前状态备份为快照。之后，您可以加载快照，将实例恢复到保存的状态。

- **与主机和软件兼容性分离**
  与本地虚拟机类似，云实例上的 RHEL 客户机操作系统运行在虚拟化内核上。这个内核与主机操作系统以及用来连接实例的*客户端*系统分开。

  因此，任何操作系统都可以安装在云实例上。这意味着，在 RHEL 公有云实例中，您可以运行无法在本地操作系统上使用的特定于 RHEL 的应用程序。

  另外，即使实例的操作系统变得不稳定或被破坏，您的客户端系统也不会受到任何影响。

其他资源

- 什么是公有云？
- 什么是 Hyperscaler？
- 云计算的类型
- RHEL 的公有云用例
- 为公有云部署获取 RHEL
- 为什么在 AWS 上运行 Linux？

1.2. RHEL 的公有云用例

在公有云上部署会带来许多好处，但可能并非是每种场景中最有效的解决方案。如果您要评估是否将您的 RHEL 部署迁移到公有云，请考虑您的用例是否将从公共云的好处中受益。

有益的用例

- 部署公有云实例对于灵活地增加和减少部署的活跃计算能力（也称为 *扩展* 和 *缩减*）非常有效。因此，在以下情况下，建议使用公有云上的 RHEL：
  - 具有高峰值工作负载和一般性能要求的集群。在资源成本方面，根据您的需求扩展和缩减可能是非常高效的。
  - 快速建立或扩展集群。这可避免设置本地服务器的高前期成本。
- 云实例不受本地环境中发生的情况的影响。因此，您可以使用它们进行备份和灾难恢复。

有潜在问题的用例

- 您正在运行一个无法调整的现有环境。与您当前的主机平台相比，自定义云实例以适应现有部署的特定需求可能不划算。
- 你的预算受到严格限制。在本地数据中心中维护您的部署通常提供较少的灵活性，但比公有云提供更多对最大资源成本的控制。

后续步骤

- 为公有云部署获取 RHEL

其他资源

- 我是否应该将应用程序迁移到云？以下是如何决定。

1.3. 迁移到公有云时的常见关注

将 RHEL 工作负载从本地环境移到公有云平台可能会引起对涉及的变化的担忧。以下是最常见的问题。

我的 RHEL 作为云实例的工作方式是否与本地虚拟机的工作方式不同？

在大部分方面，公有云平台上 RHEL 实例的工作方式与本地主机上 RHEL 虚拟机的工作方式相同，如内部服务器。主要例外包括：

- 公有云实例使用特定于提供商的控制台接口来管理云资源，而不是使用私有编排接口。
- 某些功能，如嵌套虚拟化，可能无法正常工作。如果特定功能对部署至关重要，请提前与您选择的公共云提供商检查该功能的兼容性。

与本地服务器相比，我的数据在公有云中是否安全？

RHEL 云实例中的数据归您所有，您的公有云提供商没有任何访问权限。此外，主要的云提供商支持传输中的数据加密，这提高了将虚拟机迁移到公有云时数据的安全性。

RHEL 公有云实例的一般安全性按如下进行管理：

- 您的公有云供应商负责云 hypervisor 的安全性
- 红帽在您的实例中提供 RHEL 客户机操作系统的安全功能
- 您可以在云基础架构中管理特定的安全设置和实践

我的地理区域对 RHEL 公共云实例的功能有何影响？

无论您所在的地理位置如何，您都可以在公有云平台上使用 RHEL 实例。因此，您可以在与内部服务器同样的区域运行实例。

但是，将您的实例托管在物理上较远的区域可能会导致操作它们时出现高延迟。此外，取决于公有云提供商，某些区域可能会提供额外的功能或更具成本效益。在创建 RHEL 实例前，请查看您选择的云提供商提供的托管区域的属性。

1.4. 为公有云部署获取 RHEL

要在公有云环境中部署 RHEL 系统，您需要：

1. 根据您的需求和当前市场上提供的，为您的使用案例选择最佳云提供商。
   当前认证的可以运行 RHEL 实例的云提供商有：
   - Amazon Web Services (AWS)
   - Google Cloud Platform (GCP)
   - Microsoft Azure

   注意

   本文档专门讨论在 AWS 上部署 RHEL。
2. 在您选择的云平台上创建 RHEL 云实例。如需更多信息，请参阅 创建 RHEL 云实例的方法。
3. 要让您的 RHEL 部署保持最新状态，请使用 红帽更新基础设施 (RHUI)。

其他资源

- RHUI 文档
- Red Hat Open Hybrid Cloud

1.5. 创建 RHEL 云实例的方法

要在公有云平台上部署 RHEL 实例，您可以使用以下方法之一：

创建 RHEL 的系统镜像，并将其导入到云平台。

- 要创建系统镜像，您可以使用 RHEL 镜像构建器，也可以手动构建镜像。
- 此方法使用您现有的 RHEL 订阅，也称为 *自带订阅* (BYOS)。
- 预付费一年，您可以使用红帽客户折扣。
- 您的客户服务由红帽提供。
- 要有效地创建多个镜像，您可以使用 **cloud-init** 工具。

直接从云提供商市场购买 RHEL 实例。

- 使用该服务您需要按小时付费。因此，此方法也称为 *随用随付*(PAYG)。
- 您的客户服务由云平台提供商提供。

注意

有关使用各种方法在 Amazon Web Services 上部署 RHEL 实例的详细信息，请参阅本文档中的以下章节。

其他资源

- 什么是黄金镜像？
- 为 RHEL 9 配置和管理 cloud-init

第 2 章 创建并上传 AWS AMI 镜像

要在 Amazon Web Services (AWS)云中使用自定义的 RHEL 系统镜像，请使用相应的输出类型，使用镜像构建器创建系统镜像，配置您的系统以上传镜像，并将镜像上传到 AWS 帐户。

2.1. 准备上传 AWS AMI 镜像

在上传 AWS AMI 镜像前，您必须配置系统来上传镜像。

先决条件

- 您必须在 AWS IAM account manager 中配置了一个 Access Key ID。
- 您必须具有一个可写的 S3 存储桶。

流程

1. 安装 Python 3 和 **pip** 工具：

```
# dnf install python3 python3-pip
```

2. 使用 **pip** 安装 AWS 命令行工具 ：

```
# pip3 install awscli
```

3. 设置您的配置集。终端提示您提供凭证、地区和输出格式：

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

4. 为存储桶定义名称并创建存储桶：

```
$ BUCKET=bucketname
$ aws s3 mb s3://$BUCKET
```

使用实际存储桶名称替换 *bucketname*。它必须是全局唯一的名称。因此，您的存储桶会被创建。

5. 要授予访问 S3 存储桶的权限，如果您还没有这样做，请在 AWS Identity and Access Management (IAM) 中创建一个 **vmimport** S3 角色：

   a. 创建一个 JSON 格式的带有信任策略配置的 **trust-policy.json** 文件。例如：

```
{
   "Version": "2022-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Principal": {
         "Service": "vmie.amazonaws.com"
      },
```

```
        "Action": "sts:AssumeRole",
        "Condition": {
            "StringEquals": {
                "sts:Externalid": "vmimport"
            }
        }
    }]
}
```

b. 创建一个 JSON 格式的带有角色策略配置的 **role-policy.json** 文件。例如：

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

c. 使用 **trust-policy.json** 文件为您的 Amazon Web Services 帐户创建一个角色：

```
$ aws iam create-role --role-name vmimport --assume-role-policy-document file://trust-
policy.json
```

d. 使用 **role-policy.json** 文件嵌入一个内联策略文档：

```
$ aws iam put-role-policy --role-name vmimport --policy-name vmimport --policy-
document file://role-policy.json
```

其他资源

- 使用 AWS CLI 中的高级(s3)命令

2.2. 使用 CLI 将 AMI 镜像上传到 AWS

您可以使用 RHEL 镜像构建器构建 **ami** 镜像，并使用 CLI 将它们直接推送到 Amazon AWS Cloud 服务提供商。

先决条件

- 您已在 AWS IAM 账号管理器中配置了一个 **Access Key ID**。
- 您已准备好了一个可写的 S3 存储桶。
- 您有一个定义的蓝图。

流程

1. 使用文本编辑器，使用以下内容创建配置文件：

```
provider = "aws"

[settings]
accessKeyID = "AWS_ACCESS_KEY_ID"
secretAccessKey = "AWS_SECRET_ACCESS_KEY"
bucket = "AWS_BUCKET"
region = "AWS_REGION"
key = "IMAGE_KEY"
```

将字段中的值替换为您的 **accessKeyID**、**secretAccessKey**、**bucket** 和 **region** 的凭证。**IMAGE_KEY** 值是要上传到 EC2 的虚拟机镜像的名称。

2. 将文件保存为 *CONFIGURATION-FILE*.toml，然后关闭文本编辑器。

3. 启动 compose 来将其上传到 AWS：

```
# composer-cli compose start blueprint-name image-type image-key configuration-file.toml
```

替换：

- 使用您创建的蓝图名称替换 *blueprint-name*
- 使用 **ami** 镜像类型替换 *image-type*。
- 使用要上传到 EC2 的虚拟机镜像的名称替换 *image-key*。
- *configuration-file*.toml，使用云供应商的配置文件的名称。

注意

您必须有要将自定义镜像发送到的存储桶的正确 AWS 身份和访问管理(IAM)设置。在将镜像上传到存储桶前，您必须对存储桶设置策略。

4. 检查镜像构建的状态：

```
# composer-cli compose status
```

完成镜像上传过程后，您可以看到"FINISHED"状态。

验证

要确认镜像上传成功：

1. 访问菜单中的 EC2，并在 AWS 控制台中选择正确的区域。镜像必须具有 **available** 状态，以指示它已被成功上传。

2. 在控制面板中，选择您的镜像并点 **Launch**。

其它资源

- 导入虚拟机所需的服务角色

2.3. 将镜像推送到 AWS CLOUD AMI

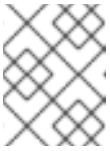
您可以使用 RHEL 镜像构建器创建一个 **(.raw)** 镜像，并选择 **Upload to AWS** 复选框，来将您创建的结果镜像直接推送到 **Amazon AWS Cloud AMI**服务提供商。

先决条件

- 您必须有访问系统的 **root** 或 **wheel** 组用户权限。
- 您已在浏览器中打开了 RHEL web 控制台的 RHEL 镜像构建器界面。
- 您已创建了一个蓝图。请参阅 在 web 控制台界面中创建一个蓝图。
- 您必须在 AWS IAM account manager 中配置了一个 Access Key ID。
- 您必须具有一个可写的 S3 存储桶。

流程

1. 在 RHEL 镜像构建器仪表盘中，点之前创建的 **blueprint name**。
2. 选择选项卡 **Images**。
3. 点 **Create Image** 创建自定义镜像。
   Create Image 窗口打开。
   a. 在 **Type** 下拉菜单中选择 **Amazon Machine Image Disk (.raw)**。
   b. 选中 **Upload to AWS** 复选框，来将您的镜像上传到 AWS 云，然后点 **Next**。
   c. 要验证您是否可以访问 AWS，请在对应的字段中输入您的**"AWS access key ID"**和**"AWS secret access key"**。点击 **Next**。

   注意

   您只能在创建新 Access Key ID 时查看 AWS secret access key 。如果您不知道您的 Secret Key，请生成一个新的 Access Key ID。

   d. 在 **Image name** 字段中输入镜像名称，在 **Amazon S3 bucket name** 字段中输入 Amazon bucket 名称，并为您要添加自定义镜像的存储桶输入 **AWS region** 字段。点击 **Next**。
   e. 查看信息并点 **Finish**。
   可选，点 **Back** 来修改任何不正确的详情。

   注意

   您必须具有要发送自定义镜像的存储桶的正确 IAM 设置。此流程使用 IAM 导入和导出，因此您必须在将镜像上传到存储桶前为存储桶设置 **策略**。如需更多信息，请参阅 IAM 用户所需的权限。

4. 右上角的弹出窗口告诉您保存的进度。它还告知镜像创建过程、创建此镜像的过程以及后续的上传到 AWS Cloud。
   完成这个过程后，您可以看到 **镜像构建完成**状态。
5. 在浏览器中，访问 Service→EC2。

a. 在 AWS 控制台仪表盘菜单中，选择 correct region。镜像必须具有 **Available** 状态，以指示它已被上传。

b. 在 AWS 仪表盘中，选择您的镜像并点 **Launch**。

6. 此时会打开一个新窗口。根据启动镜像所需的资源选择实例类型。点击 **Review and Launch**。

7. 查看您的实例启动详情。如果需要进行任何更改，您可以编辑任何部分。点 **Launch**

8. 在启动实例之前，选择一个访问它的公钥。
您可以使用您已有的密钥对，也可以创建一个新的密钥对。

   按照以下步骤在 EC2 中创建新的密钥对，并将它连接到新实例。

   a. 在下拉菜单中选择 **"Create a new key pair"**。

   b. 输入新密钥对名称。它生成一个新的密钥对。

   c. 点 **"下载密钥对"** 在您的本地系统中保存新密钥对。

9. 然后，您可以点 **Launch Instance** 来启动您的实例。
您可以检查实例的状态，它显示为 **Initializing**。

10. 实例状态变为 **running** 后，**连接**按钮将变为可用。

11. 点 **连接**。此时会出现一个窗口，其中包含有关如何使用 SSH 进行连接的说明。

    a. 选择 **A standalone SSH client** 作为首选连接方法并打开终端。

    b. 在您存储私钥的位置，确保您的密钥是公开可见的，以便 SSH 可以正常工作。要做到这一点，请运行以下命令：

    ```
    $ chmod 400 _your-instance-name.pem_>
    ```

    c. 使用其公共 DNS 连接到您的实例：

    ```
    $ ssh -i <_your-instance-name.pem_> ec2-user@<_your-instance-IP-address_>
    ```

    d. 键入 **yes** 以确认您要继续连接。
    因此，您通过 SSH 连接到您的实例。

验证

- 检查在使用 SSH 连接到您的实例的过程中是否能够执行任何操作。

其它资源

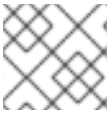
- 在红帽客户门户网站中创建一个问题单
- 使用 SSH 连接到您的 Linux 实例

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

要在 Amazon Web Services (AWS)上设置 RHEL 的高可用性(HA)部署，您可以将 RHEL 的 EC2 实例部署到 AWS 上的集群。

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 制作自定义 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

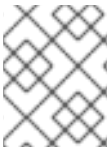
- 注册一个红帽客户门户网站（Red Hat Customer Portal） 帐户。
- 注册 AWS 并设置 AWS 资源。如需更多信息，请参阅使用 Amazon EC2 设置 。

3.1. AWS 上的 RED HAT ENTERPRISE LINUX 镜像选项

下表列出了镜像的不同选择并记录镜像选项的不同。

表 3.1. 镜像选项

镜像选项	订阅	示例情境	注意事项
部署红帽黄金镜像。	使用您现有的红帽订阅。	在 AWS 上选择红帽黄金镜像。有关黄金镜像以及如何在 Azure 上访问它们的详情，请查看 红帽云访问参考指南。	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。红帽直接为 Cloud Access 镜像提供支持。
部署已移至 AWS 的自定义镜像。	使用您现有的红帽订阅。	上传自定义镜像，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。红帽直接为自定义 RHEL 镜像提供支持。
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 市场 中选择。	根据 *随用随付* 模式按小时向 Amazon 付费。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

您可以使用红帽镜像构建器为 AWS 创建一个自定义镜像。如需更多信息，请参阅 制作自定义 RHEL 系统镜像。

重要

您不能将按需实例转换为自定义 RHEL 实例。从按需镜像改为自定义 RHEL *自带订阅* (BYOS)镜像：

1. 创建新的自定义 RHEL 实例，并从您的按需实例迁移数据。
2. 在迁移数据后取消您的 on-demand 实例以避免出现重复账单。

其他资源

- 编写自定义 RHEL 系统镜像
- AWS Management Console
- AWS Marketplace

3.2. 理解基础镜像

要从 ISO 镜像创建基础虚拟机，您可以使用预配置的基础镜像及其配置设置。

3.2.1. 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

3.2.2. 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。

3.3. 从 ISO 镜像创建基本虚拟机

要从 ISO 镜像创建 RHEL 9 基础镜像，请为虚拟化启用您的主机并创建 RHEL 虚拟机(VM)。

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。

3.3.1. 从 RHEL ISO 镜像创建虚拟机

步骤

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。具体步骤请参阅 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**。
   例如，以下命令使用 **/home/username/Downloads/rhel9.iso** 镜像创建一个 **kvmtest** 虚拟机：

```
# virt-install \
    --name kvmtest --memory 2048 --vcpus 2 \
    --cdrom /home/username/Downloads/rhel9.iso,bus=virtio \
    --os-variant=rhel9.0
```

   b. 如果使用 web 控制台创建虚拟机，请按照 使用 web 控制台创建虚拟机 中的流程操作，并注意以下几点：
      - 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。

3.3.2. 完成 RHEL 安装

要完成要在 Amazon Web Services (AWS)上部署的 RHEL 系统 **安装**，**自定义 安装概述** 视图，开始安装，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。

其他资源

- 订阅自动附加和更新
- cloud-init 简介

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

为了能够在 Amazon Web Services (AWS)上运行 RHEL 实例，您必须首先将 RHEL 镜像上传到 AWS。

3.4.1. 安装 AWS CLI

在 AWS 中使用 AWS CLI 管理 HA 集群需要许多流程。

先决条件

- 您已创建了 AWS Access Key ID 和 AWS Secret Access Key，并可以访问它们。具体说明和详情，请参考 快速配置 AWS CLI。

步骤

1. 使用 **dnf** 命令安装 AWS 命令行工具。

```
# dnf install awscli
```

2. 使用 **aws --version** 命令验证您是否安装了 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
```

3. 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

3.4.2. 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，您需要创建一个 S3 存储桶，然后将您的镜像移到存储桶。

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 **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

3.4.3. 创建 vmimport 角色

要使用 VM Import 服务将 RHEL 虚拟机(VM)导入到 Amazon Web Services (AWS)，您需要创建 **vmimport** 角色。

如需更多信息，请参阅 Amazon 文档中的 VM Import/Export 将虚拟机导入为镜像。

流程

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

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

2. 使用 **create role** 命令创建 **vmimport** 角色。指定 **trust-policy.json** 文件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

3.4.4. 将镜像转换为 S3

通过使用 **qemu-img** 命令，您可以转换您的镜像，以便您可以将其推送到 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 导入/导出是如何工作的。

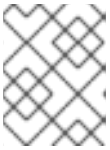
步骤

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

3.4.5. 将您的镜像导入为快照

要在 Amazon Elastic Cloud Compute (EC2)服务中启动 RHEL 实例，您需要 Amazon Machine Image (AMI)。要创建系统的 AMI，您必须首先将 RHEL 系统镜像的快照上传到 EC2。

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

其它资源

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

3.4.6. 从上传的快照创建 AMI

要在 Amazon Elastic Cloud Compute (EC2)服务中启动 RHEL 实例，您需要 Amazon Machine Image (AMI)。要创建系统的 AMI，您可以使用之前上传的 RHEL 系统快照。

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" --ena-support \
   --block-device-mappings "{\"DeviceName\": \"/dev/sda1\",\"Ebs\":
{\"SnapshotId\": \"snap-0ce7f009b69ab274d\"}}"
```

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。

3.4.7. 从 AMI 启动实例

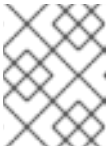
要启动并配置 Amazon Elastic Compute Cloud (EC2)实例，请使用 Amazon Machine Image (AMI)。

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**。

有关实例类型的信息，请参阅 Amazon EC2 实例类型。

4. 点 **Next: Configure Instance Details**。
   a. 输入您要创建的实**例数量**。
   b. 对于 **Network**，选择您在设置 AWS 环境时创建的 VPC。为实例选择子网或创建新子网。
   c. 为 Auto-assign Public IP 选择 **Enable**。



注意

这些是创建基本实例所需的最小配置选项。根据您的应用程序要求查看其他选项。

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

注意

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

7. 点 **Next: 配置安全组**。选择设置 AWS 环境时创建的安全组。
8. 点 **Review and Launch**。验证您的选择。
9. 点 **Launch**。此时会提示您选择现有密钥对或创建新密钥对。选择设置 AWS 环境时创建的密钥对。

注意

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

10. 点 **Launch Instances**。
11. 点 **View Instances**。您可以命名实例。
现在，您可以通过选择一个实例并单击 **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

3.4.8. 附加红帽订阅

使用 **subscription-manager** 命令，您可以注册并附加红帽订阅到 RHEL 实例。

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。
3. **可选：** 要在 Red Hat Hybrid Cloud Console 中收集有关实例的各种系统指标，您可以使用 Red Hat Insights 注册实例。

```
# insights-client register --display-name <display-name-value>
```

   有关 Red Hat Insights 进一步配置的详情，请参考 Red Hat Insights 的客户端配置指南 。

其它资源

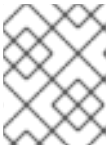
- 创建红帽客户门户网站激活码
- 通过命令行附加和删除订阅
- 使用并配置 Red Hat Subscription Manager
- Red Hat Insights 的客户端配置指南

3.4.9. 对 AWS 黄金镜像设置自动注册

要在 Amazon Web Services (AWS)上更快、更适地部署 RHEL 9 虚拟机，您可以将 RHEL 9 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 9 黄金镜像。具体步骤请参阅 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，通过执行 **subscription-manager identity** 命令来验证系统是否已注册到 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
- 向混合云控制台添加云源

3.5. 其他资源

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

第 4 章 在 AWS 上配置红帽高可用性集群

要创建集群，当节点出现故障时，RHEL 节点会自动重新分发其工作负载，请使用 Red Hat High Availability Add-On。此类高可用性(HA)集群也可以托管在公有云平台上，包括 AWS。在 AWS 上创建 RHEL HA 集群与在非云环境中创建 HA 集群类似。

要使用 EC2 实例作为集群节点在 Amazon Web Services (AWS)上配置 Red Hat HA 集群，请参阅以下部分。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 。

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。

4.1. 在公有云平台上使用高可用性集群的好处

高可用性(HA)集群是一组链接在一起的计算机（称为 *节点*），以运行特定的工作负载。HA 集群的目的是在出现硬件或软件故障时提供冗余。如果 HA 集群中的节点失败，Pacemaker 集群资源管理器会将工作负载分发到其他节点，且在集群中运行的服务中不会出现显著的停机时间。

您还可以在公有云平台上运行 HA 集群。在这种情况下，您要将云中的虚拟机(VM)实例用作单独的集群节点。在公有云平台上使用 HA 集群有以下优点：

- 改进了可用性：如果出现虚拟机故障，工作负载会快速地重新分发到其他节点，因此运行的服务不会中断。
- 可扩展性：在需求高时可以启动其他节点，在需求低时停止其它节点。
- 节约成本：采用现收现付定价时，您只需为正在运行的节点支付费用。
- 简化管理：有些公共云平台提供管理界面，以便更轻松地配置 HA 集群。

要在 Red Hat Enterprise Linux (RHEL)系统上启用 HA，红帽提供了一个高可用性附加组件。高可用性附加组件提供了在 RHEL 系统上创建 HA 集群的所有必要组件。这些组件包括高可用性服务管理和集群管理工具。

其它资源

- 高可用性附加组件概述

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。

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

在 AWS 中使用 AWS CLI 管理 HA 集群需要许多流程。

先决条件

- 您已创建了 AWS Access Key ID 和 AWS Secret Access Key，并可以访问它们。具体说明和详情，请参考 快速配置 AWS CLI。

步骤

1. 使用 **dnf** 命令安装 AWS 命令行工具。

```
# dnf install awscli
```

2. 使用 **aws --version** 命令验证您是否安装了 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
```

3. 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，也可以使用黄金镜像或按需镜像。

先决条件

- 您已设置了一个 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。
   有关实例类型的信息，请参阅 Amazon EC2 实例类型。
4. 点 **Next: Configure Instance Details**。
   a. 输入您要为集群创建的 **Number of instances**。这个示例流程使用三个集群节点。

   注意

   不要启动自动缩放组。

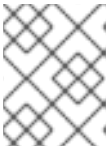
   b. 对于 **Network**，请选择您在 Set up AWS environment 中创建的 VPC。选择实例子网以创建新子网。
   c. 为 Auto-assign Public IP 选择 **Enable**。以下是 **Configure Instance Details** 所需的最小选择。根据您的特定 HA 应用，您可能需要进行其他选择。

   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。

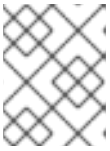
5. 单击 **Next: Add Storage**，并验证默认存储是否足够。除非您的 HA 应用程序需要其他存储选项，您不需要修改这些设置。
6. 点击 **Next: Add Tags**。

   注意

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

7. 点击 **Next: Configure Security Group**。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 **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. 更改密钥文件的权限，以便只有 root 用户可以读取它。

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

4.6. 连接到 EC2 实例

使用 **所有节点上的 AWS 控制台**，您可以连接到 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

在每个节点上，您需要安装高可用性软件包和代理，以便在 AWS 上配置红帽高可用性集群。

步骤

1. 删除 AWS Red Hat Update Infrastructure (RHUI)客户端。

```
$ 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，以及来自高可用性频道的 AWS 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。

流程

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>
```

Example:

```
[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>
```

Example:

```
[root@node01 clouduser]# pcs cluster setup new_cluster node01 node02 node03

[...]

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
node02: Cluster Enabled
node03: Cluster Enabled
node01: Cluster Enabled
```

2. 启动集群。

```
[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。

要在 AWS 集群上配置隔离，您可以使用多种方法：

- 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。

先决条件

- 您必须使用 **fence_aws** 隔离代理。要获取 **fence_aws**，请在集群中安装 **resource-agents** 软件包。

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=AKIAI123456MRMJA
   secret_key=a75EYIG4RVL3hdsdAslK7koQ8dzaDyn5yoIZ/ \
   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. 通过使用集群的 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，要在集群中的所有节点中配置隔离设备：

```
[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
```

有关创建隔离设备的具体参数的详情，请参考 **fence_aws** man page 或 配置和管理高可用性集群指南。

验证

1. 在节点上显示配置的隔离设备及其参数：

```
[root@nodea ~]# pcs stonith config fence${CLUSTER}

Resource: <clustername> (class=stonith type=fence_aws)
Attributes: access_key=XXXXXXXXXXXXXXXXXXX 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)
```

2. 测试其中一个集群节点的隔离代理。

```
# 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。

例如：

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

Node: ip-10-0-0-58 fenced
```

3. 检查状态以验证该节点是否已隔离。

```
# pcs status
```

例如：

```
[root@ip-10-0-0-48 ~]# pcs status

Cluster name: newcluster
Stack: corosync
Current DC: ip-10-0-0-46 (version 1.1.18-11.el7-2b07d5c5a9) - partition with quorum
Last updated: Fri Mar  2 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
```

4. 启动上一步中隔离的节点。

```
# pcs cluster start <awshostname>
```

5. 检查状态以验证节点已启动。

```
# pcs status
```

例如：

```
[root@ip-10-0-0-48 ~]# pcs status

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

3 nodes configured
1 resource configured

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

```
Full list of resources:

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

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

4.10. 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. 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：

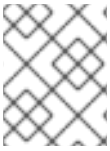
```
[root@ip-10-0-0-48 ~]# aws ec2 describe-instances --output text --query
'Reservations[*].Instances[*].[InstanceId,Tags[?Key==Name].Value]'
i-07f1ac63af0ec0ac6
ip-10-0-0-48
i-063fc5fe93b4167b2
ip-10-0-0-46
i-08bd39eb03a6fd2c7
ip-10-0-0-58
```

4.11. 在 AWS 中设置 IP 地址资源

为确保使用 IP 地址访问由集群管理的、在发生故障转移时通过网络访问资源的客户端，集群必须包含 *IP 地址资源*（使用特定网络资源代理）。

RHEL HA Add-On 提供了一组资源代理，它创建 IP 地址资源来管理 AWS 上的各种 IP 地址。要确定要配置哪些资源代理，请考虑您希望 HA 集群管理的 AWS IP 地址的类型：

- 如果您需要管理互联网公开的 IP 地址，请使用 **awseip** network 资源。
- 如果您需要管理仅限于单个 AWS 可用区(AZ)的专用 IP 地址，请使用 **awsvip** 和 **IPaddr2** 网络资源。
- 如果您需要管理在同一 AWS 区域中的多个 AWS AZ 移动的 IP 地址，请使用 **aws-vpc-move-ip** 网络资源。



注意

如果 HA 集群不管理任何 IP 地址，则不需要在 AWS 上管理虚拟 IP 地址的资源代理。如果您需要对特定部署进行进一步指导，请参阅 AWS 供应商。

4.11.1. 创建 IP 地址资源来管理互联网公开的 IP 地址

为确保高可用性(HA)客户端可以访问使用面向公共的互联网连接的 RHEL 9 节点，请配置 *AWS Secondary Elastic IP Address* (**awseip**)资源以使用弹性 IP 地址。

先决条件

- 已安装了之前配置的集群。
- 您的集群节点必须有权访问 RHEL HA 软件仓库。如需更多信息，请参阅 安装高可用性软件包和代理。
- 您已设置 AWS CLI。具体步骤请参阅安装 AWS CLI。

流程

1. 安装 **resource-agents-cloud** 软件包。

```
# dnf install resource-agents-cloud
```

2. 使用 AWS 命令行界面(CLI)，创建一个弹性 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. **可选：** 显示 **awseip** 的描述。这显示了这个代理的选项和默认操作。

```
# pcs resource describe awseip
```

4. 创建 Secondary Elastic IP 地址资源，它使用之前使用 AWS CLI 指定分配的 IP 地址。另外，创建二级 Elastic IP 地址将属于的资源组。

```
# pcs resource create <resource-id> 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

1. 显示集群的状态，以验证所需资源是否正在运行。

```
# pcs status
```

以下输出显示了运行一个运行集群的示例，其中 **vip** 和 **elastic** 资源已作为 **networking-group** 资源组的一部分启动：

```
[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
     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
```

2. 从本地工作站启动 SSH 会话到之前创建的弹性 IP 地址。

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

Example:

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

3. 验证您通过 SSH 连接的主机是否与创建的弹性资源关联。

4.11.2. 创建 IP 地址资源来管理私有 IP 地址，仅限于单个 AWS 可用区

为确保 AWS 上的高可用性(HA)客户端可以访问使用私有 IP 地址且只能在单个 AWS 可用区(AZ)中移动的 RHEL 9 节点，请配置 *AWS Secondary Private IP Address* (**awsvip**)资源以使用虚拟 IP 地址。

您可以在集群中的任何节点上完成以下步骤。

先决条件

- 已安装了之前配置的集群。
- 您的集群节点可以访问 RHEL HA 软件仓库。如需更多信息，请参阅 安装高可用性软件包和代理。
- 您已设置 AWS CLI。具体步骤请参阅安装 AWS CLI。

流程

1. 安装 **resource-agents-cloud** 软件包。

```
# dnf install resource-agents-cloud
```

2. **可选：** 查看 **awsvip** 描述。这显示了这个代理的选项和默认操作。

```
# pcs resource describe awsvip
```

3. 使用 **VPC CIDR** 块中未使用的私有 IP 地址创建二级私有 IP 地址。另外，创建二级私有 IP 地址所属的资源组。

```
# pcs resource create <resource-id> awsvip secondary_private_ip=<Unused-IP-Address> --group <group-name>
```

例如：

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

4. 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。确保虚拟 IP 属于与您在上一步中创建的二级私有 IP 地址相同的资源组。

```
# pcs resource create <resource-id> IPaddr2 ip=<secondary-private-IP> --group <group-name>
```

例如：

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

验证

- 显示集群的状态，以验证所需资源是否正在运行。

```
# pcs status
```

以下输出显示了运行一个运行集群的示例，其中 **vip** 和 **privip** 资源已作为 **networking-group** 资源组的一部分启动：

```
[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
```

4.11.3. 创建 IP 地址资源来管理可在多个 AWS 可用区间移动的 IP 地址

为确保 AWS 上的高可用性(HA)客户端可以访问可在同一 AWS 区域中的多个 AWS 可用区移动的 RHEL 9 节点，请配置一个 **aws-vpc-move-ip** 资源以使用弹性 IP 地址。

先决条件

- 已安装了之前配置的集群。
- 您的集群节点可以访问 RHEL HA 软件仓库。如需更多信息，请参阅 安装高可用性软件包和代理。
- 您已设置 AWS CLI。具体步骤请参阅安装 AWS CLI。
- 在集群中配置了 Identity and Access Management (IAM)用户，并具有以下权限：
  - 修改路由表
  - 创建安全组
  - 创建 IAM 策略和角色

步骤

1. 安装 **resource-agents-cloud** 软件包。

```
# dnf install resource-agents-cloud
```

2. **可选：** 查看 **aws-vpc-move-ip** 描述。这显示了这个代理的选项和默认操作。

```
# pcs resource describe aws-vpc-move-ip
```

3. 为 IAM 用户设置 **OverlayIPAgent** IAM 策略。
   a. 在 AWS 控制台中，导航到 **Services → IAM → Policies → Create OverlayIPAgent Policy**
   b. 输入以下配置，并更改 < *region*>、& lt *;account-id>* 和 & *lt;ClusterRouteTableID&* gt; 值，使其与您的集群对应。

```
{
   "Version": "2012-10-17",
   "Statement": [
```

```
        {
            "Sid": "Stmt1424870324000",
            "Effect": "Allow",
            "Action":  "ec2:DescribeRouteTables",
            "Resource": "*"
        },
        {
            "Sid": "Stmt1424860166260",
            "Action": [
                "ec2:CreateRoute",
                "ec2:ReplaceRoute"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:ec2:<region>:<account-id>:route-
table/<ClusterRouteTableID>"
        }
    ]
}
```

4. 在 AWS 控制台中，在集群中的所有节点上禁用 **Source/Destination Check** 功能。为此，请右键单击每个节点 → **Networking** → **Change Source/Destination Checks**。在出现的弹出消息中，单击 **Yes, Disable**。

5. 为集群创建路由表。要做到这一点，请在集群的一个节点中使用以下命令：

```
# aws ec2 create-route --route-table-id <ClusterRouteTableID> --destination-cidr-block
<NewCIDRblockIP/NetMask> --instance-id <ClusterNodeID>
```

   在命令中，按如下所示替换值：

   - **ClusterRouteTableID** ：现有集群 VPC 路由表 ID。
   - **NewCIDRblockIP/NetMask**: VPC 类间路由(CIDR)块之外的新 IP 地址和子网掩码。例如，如果 VPC CIDR 块是 **172.31.0.0/16**，新的 IP 地址/子网掩码可以是 **192.168.0.15/32**。
   - **ClusterNodeID** ：集群中另一节点的实例 ID。

6. 在集群的一个节点上，创建一个 **aws-vpc-move-ip** 资源，该资源使用客户端可访问的可用 IP 地址。以下示例创建一个名为 **vpcip** 的资源，该资源使用 IP **192.168.0.15**。

```
# pcs resource create vpcip aws-vpc-move-ip ip=192.168.0.15 interface=eth0
routing_table=<ClusterRouteTableID>
```

7. 在集群的所有节点上，编辑 **/etc/hosts/** 文件，并使用新创建的资源的 IP 地址添加一行。例如：

```
192.168.0.15 vpcip
```

验证

1. 测试新 **aws-vpc-move-ip** 资源的故障转移功能：

```
# pcs resource move vpcip
```

2. 如果故障转移成功，在 **vpcip** 资源移动后删除自动创建的约束：

```
# pcs resource clear vpcip
```

其他资源

- IAM 用户

4.11.4. 其他资源

- 高可用性附加组件概述
- 配置和管理高可用性集群
- **aws-vpc-move-ip**的配置示例
- 使用 **aws-vpc-route53**的高可用 AWS 服务的 DNS 名称故障切换

4.12. 配置共享块存储

要创建额外的存储资源，您可以使用 Amazon Elastic Block Storage (EBS) Multi-Attach 卷为红帽高可用性集群配置共享块存储。请注意，这个流程是可选的，以下步骤假设三个带有 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 创建一个共享块卷。

```
$ 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

4.13. 其他资源

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